doi:10.3772/j.issn.1002-0470.2019.05.011

基于 Petri 网的物流仓库多 AGV 调度方法的研究①

李圣男② 邢科新③ 林叶贵 张贵军

(浙江工业大学信息工程学院 杭州 310023)

摘 要 针对多自动导引车(AGV)在大规模物流仓储中存在的路径规划问题,对基于时间 Petri 网的多 AGV 调度优化算法进行了研究。该算法利用时间 Petri 网对大规模双向车道环境下多 AGV 的仓库调度过程进行建模,并在分解后对 AGV 进行单独分析,减少了算法的时间复杂度;引入传统外点惩罚函数法构建以 AGV 调度时间为指标的目标函数,通过对 AGV 运行路径信息的依次迭代和更新解决了其在调度过程中的碰撞问题;在此基础上增加碰撞类型分析,以目标函数最优为原则对路径进行局部规划,实现调度方案最优。实验结果表明在大规模调度环境中该算法能快速收敛出无碰撞死锁的最优路径方案,并能保证多 AGV 在动态仓库物流调度中具有良好的实时适应性。

关键词 自动导引车(AGV),物流调度,时间 Petri 网,外点惩罚函数,碰撞分析

0 引 言

近年来,阿里、京东等电商的飞速发展,促使物流行业规模的不断升级。现实中物流仓库系统是一个复杂的动态多目标的离散系统,本文针对物流仓库的调度[1]提出 3 点问题:(1)如何构建合理的仓库运行地图;(2)针对不同的调度任务如何获取自动导引车(automated guide vehicle,AGV)的最优调度路径;(3)如何解决 AGV 运行过程中存在的死锁和冲突。针对这些问题,目前国内外学者做了很多研究。贺丽娜等人[2]结合 A* 算法与时间窗算法,通过记录对比仓库节点的空闲和占有时间来防止 AGV 之间的冲突,实现无冲突调度。Luka 等人[3]提出了改进的银行家算法,通过动态申请节点资源来确定仓库中存在的不安全状态并加以处理,消除碰撞环节。还有学者[4-6]提出的一些优化算法,对不同的场景有较好的实际应用价值。

然而随着仓库规模的不断增大,上述方法导致调度系统效率降低,生产成本和计算时间增加,因此引入 Petri 网[7]来研究仓库调度问题。国内外对基于 Petri 网的调度算法的研究有着较好的发展。Wu[8]采用有色 Petri 网,动态分析网中的基元,设置约束,预防并消除碰撞与死锁,但是该算法在效率上有明显的不足。Nishi 和 Maeno[9]提出了基于 Petri 网的拉格朗日松弛法来优化路径规划问题,提高了算法的运行效率,但未考虑实际冲突的类型,使调度方案在某些情况下无法达到最优,造成资源的浪费。周卫东等人[10]提出扩展 Petri 网,结合遗传算法,能收敛出最优的调度方案,但是遗传算法较为复杂,不适用于动态的调度环境。

本文针对上述基于 Petri 网调度中仍存在的一系列问题,提出了改进的多 AGV 调度优化算法。在双向车道的调度环境下,基于时间 Petri 网并结合外点惩罚函数法,整合 AGV 运行路径中存在的碰撞和死锁,提高了算法的效率;对碰撞类型进行分析,综

① 国家自然科学基金(61773346)资助项目。
② 男,1993 年生,硕士生;研究方向:多移动机器人调度;E-mail:526669242@qq.com
③ 通信作者,E-mail:xkx@zjut.edu.cn
(收稿日期:2018-07-25)

合考虑总的时间代价和效率值选择相应的局部路径规划方案,保证了运行路径最优。

1 仓库调度问题描述

基于大规模的物流仓库作为多 AGV 的调度环境,以节点集 $V = \{v_1, v_2, \cdots, v_n\}$ 和双向道路集 $E = \{e_1, e_2, \cdots, e_n\}$ 来构建仓库的二维无向交通运输系统模型图。4 节点的仓库二维无向图如图 1 所示。假设物流调度仓库中存在 r 个节点, m 辆 AGV,则调度过程如下。

(1) AGV $u_j(1 \leqslant j \leqslant m)$ 空闲时处于停靠站 $S_i(1 \leqslant i \leqslant m)$, 部分停靠站设置充电所 $e_k(k < i)$。对于低电量的 AGVu_j 进行电源的补充。

(2) 上位机下达 AGVu_j 的运输任务。AGVu_j 一次只能接受一个任务,并且预先了解仓库布局模型的映射。

(3) AGVu_j 进行 3 阶段调度。第 1 阶段为 AGV 从停靠站前往物料装载站。第 2 阶段 AGVu_j 接收物料,前往物料卸载站。第 3 阶段 AGVu_j 完成物料卸载,回到停靠站,完成调度任务。

(4) 仓库中双向车道可供 AGVu_j 在两节点间以任意方向运行,且仅在节点处停靠或转向。两辆或以上的 AGV 不能同时在同一条车道上相向运行,且不能同时停靠在同一节点上。

(5) AGVu_j 在运行过程中匀速运行,并且可以在运行过程中因突发情况改变车辆的运动状态。

根据上述过程本文以 AGV 的运输时间(包括 AGV 运行时的转向时间,遇到避障时所需等待时间)为目标函数来规划多 AGV 的调度路径。

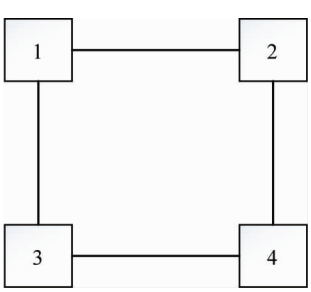

图 1 4 节点的二维无向仓库交通系统图

2 基于时间 Petri 网的建模与分解

2.1 时间 Petri 网建模

Petri 网作为一种图形建模工具,可以对各种离散时间模型进行数学化描述。本文引入时间 Petri 网对 AGV 调度过程进行建模。定义 5 元组 $PN = \{P, T, \omega, M_k, \varnothing_t\}$ 来映射仓库调度的调度过程。其中, $P = \{p_1, \cdots, p_n\}$ 为库所集,表示仓库中的的节点; $T = \{t_1, \cdots, t_n\}$ 为变迁集, $(P \cap T = \varnothing)$ 表示 AGV 发生状态改变的事件; $\omega: (P \times T) \cup (T \times P) \rightarrow Z^+$ 为库所与变迁之间的关联程度,表示仓库的分部结构; $M_k: p \rightarrow N$ 表示 k 时刻系统库所集的状态量;触发序列 $r_k: (t_1, \cdots, t_n) \rightarrow t_i \in \{0, 1\}$ 是系统变迁集,用于描述 k 时刻时间 Petri 网的运行情况; $\varnothing_t: t \rightarrow N$ 表示变迁 t_i 的持续触发时间,表示 AGV 在两相邻节点间的运行时间。

在图 1 中加入时间 Petri 网的基本元素后,4 节点的 Petri 网仓库模型如图 2 所示。

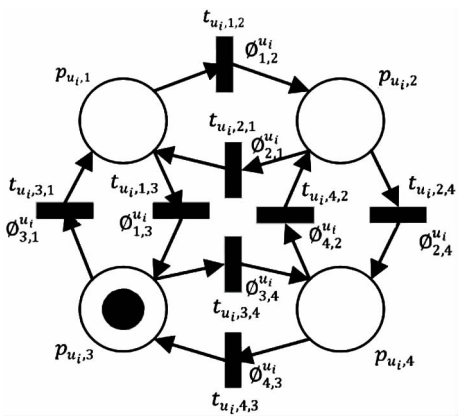

图 2 4 节点的无向车道仓库 Petri 网模型

其中库所 $p_{u_i,x}$ 表示 AGV $u_j(1 \leqslant j \leqslant m)$ 位于节点 $x(1 \leqslant x \leqslant 9)$ 上,变迁 $t_{u_i,a,b}$ 表示 AGV u_i 从节点 $a(1 \leqslant a \leqslant 9)$ 运行到相邻节点 $b(1 \leqslant b \leqslant 9, a \neq b)$ 这一事件, $\varnothing_t = 1$ 表示 AGV 在两节点间运行时间为 1(为方便下文分析实验,默认 $\varnothing_t = 1$)。为预防碰撞和死锁的产生,本文引入了资源库所 $P_{R,x}$, 使资源不会在同一时刻分配给 2 辆或以上的 AGV。为了区别不同类型库所集,本文定义节点库所类型

为 0,资源库所的类型为 1,定义关联矩阵 $\boldsymbol{A}^+$, $\boldsymbol{A}^-$:

$$(\boldsymbol{A}^+)_{ji} = \begin{cases} 0 & (Type(p_i) = 1) \\ \omega(t_j, p_i) & (Type(p_i) = 0) \end{cases} \tag{1}$$

$$(\boldsymbol{A}^-)_{ji} = \begin{cases} \omega(p_i, t_j) - \omega(t_j, p_i) & (Type(p_i) = 1) \\ \omega(p_i, t_j) & (Type(p_i) = 0) \end{cases} \tag{2}$$

其中 $p \in P, t \in T$, $\omega(p, t) > 0(\omega(t, p) > 0)$ 表示库所 p (变迁 t)到变迁 t(库所 p)之间的有向弧,否则 $\omega(p, t) = 0(\omega(t, p) = 0)$。

在 k 时刻如果 M_k 满足触发式:

$$M_k - (\boldsymbol{A}^-)^T \geqslant 0 \tag{3}$$

那么系统状态量 M_k 将转化为 M_{k+1} 通过如下等式:

$$M_{k+1} = M_k + (\boldsymbol{A}^+ - \boldsymbol{A}^-)^T r_k \tag{4}$$

AGV 的调度可以由带时间约束 $\varnothing_t$ 的触发序列 r_k 进行描述,因此可转化为整数规划问题进行分析[11,12]。

2.2 基于资源节点的时间 Petri 网分解

为防止库所与变迁数量过多而导致 Petri 网状态爆炸,在不改变原网的结构和性质的情况下,以变迁为指标将时间 Petri 网分解资源库所,将原网分解成以 AGV 为单位的独立子网。子网之间互不干扰,很大程度上减少了计算的复杂性,对满足 AGV 完成运输任务的总时间等于单 AGV 完成各自任务所需时间的代数累加和这一条件的时间 Petri 网模型均可进行分解。

则分解后子网中节点库所与变迁间的触发关系为

$$M_k^{u_j} - (\boldsymbol{A}_{u_j}^-)^{\mathrm{T}} r_k^{u_j} \geqslant 0 (1 \leqslant i \leqslant m) \tag{5}$$

其中,$\boldsymbol{A}_{u_j}^-, \boldsymbol{A}_{u_j}^+ \in Z^{\mathrm{T}_{u_j} \times P_{u_j}}$, 为 AGV$u_j$ 子网的关联矩阵。

对于 k 时刻原网中的资源库所,变迁的触发条件为

$$M_k^R - (B^-)^{\mathrm{T}} r_k \geqslant 0 \tag{6}$$

$$B^- = [(\boldsymbol{A}_{u_1}^-)^{\mathrm{T}}, \cdots, \boldsymbol{A}_{u_m}^-)^{\mathrm{T}}]^{\mathrm{T}} (\boldsymbol{A}_{u_m}^- \in Z^{\mathrm{T}_{u_i} \times P_R}) \tag{7}$$

$$B^+ = [(\boldsymbol{A}_{u_1}^+)^{\mathrm{T}}, \cdots, \boldsymbol{A}_{u_m}^+)^{\mathrm{T}}]^{\mathrm{T}} (\boldsymbol{A}_{u_m}^+ \in Z^{\mathrm{T}_{u_i} \times P_R}) \tag{8}$$

$(Type(p_i) = 1)$

3 外点惩罚函数法和碰撞分析

由于资源库所的存在,AGV 在各自子网中获得的触发序列在原网中一般是不可行的。本文引入外点惩罚函数法搜索产生碰撞和死锁的 AGV 路径触发序列,进行碰撞类型分析,准确快速地收敛出最优的调度方案。

步骤 1 初次优化

初始化迭代次数 N 为 1。在接收运输任务后,AGVu_j 在不考虑其余 AGV$u_i (i \neq j)$ 的情况下进行最短调度路径的计算。用子网模型中的触发序列 $r_k^{u_j}$ 来描述 AGVu_j 的路径序列,以时间为指标的目标函数定义如下:

$$J_{\text{total}}(r_k^{u_j}) = \sum_{j=1}^{m} \min_{\{r_k^{u_j}\}} J_{u_j} \tag{9}$$

$$J_{u_j} = \sum_{k=1}^{N_t} \delta_{u_j,k} \tag{10}$$

$$\delta_{u_j,k} = \begin{cases} 0 & (M_k^{u_j} = M_f^{u_j}) \cup (\delta_{u_j,\,k-1}) = 1 \\ 1 & (M_k^{u_j} \neq M_f^{u_j}) \cup (\delta_{u_j,\,k-1}) = 0 \end{cases} \tag{11}$$

其中,变量 $\delta_{u_i,\,k} \in \{0,1\}$, 设 $M_k^{u_j}$ 表示子网中 AGVu_j 第 k 时刻的库所的状态量,如果任务完成,即 $M_k^{u_j} = M_f^{u_j}(M_f^{u_j}$ 为最终目标状态量),变量 $\delta_{u_j,\,k}$ 取 0,反之 $M_k^{u_j} \neq M_f^{u_j}$, 变量 $\delta_{u_j,\,k}$ 取 1。时间函数即为各子网 AGVu_j 时间函数 J_{u_j} 的代数累加和。

步骤 2 收敛性判断

若路径收敛结果 $r_k^{u_j}$ 满足如下任意条件:

(1) $r_k^{u_1}, \cdots, r_k^{u_m}$ 满足限制条件(6);(2) $R_{k,N}^{u_j} = R_{k,N+1}^{u_j} (j = 1,2,\cdots,m)$, 其中 $R_{k,N}^{u_j}, R_{k,N+1}^{u_j}$ 分别为 AGV u_j 在第 N 次和 $N+1$ 次迭代的路径。则将第 N 迭代获得的触发序列 $r_k^{u_j}$ 作为最终结果,算法终止。否则算法至步骤 3。

步骤 3 再优化

时间函数按照 AGV$u_1, \cdots,$ AGVu_m 的顺序执行再优化。当 AGVu_j 执行再优化的时候,需要保持其余子网的 AGVu_i 的运行触发序列不变。其中 AGVu_j 再优化的目标函数公式如下:

$$J_{u_j} = \min_{\{r_k^{u_j}\}} (J_{u_j} + \sum_{k=1}^{N_t} \sum_{p \in P_{Ru_j}} \varepsilon_{p,u_j}^{(N)} \times \max\{0, -\partial_{u_j,\,p,\,k(r_{kj}^{u_j})}\}$$

(12)

式中,$-\partial_{u_j,p,k(r_k^{u_j})}$ 表示 AGVu_j 在 k 时刻的触发变迁 $r_k^{u_j}$ 作用下库所集 P 的拥堵情况,再优化重构目标函数,在步骤 1 中构建的目标函数加入原网资源节点处变迁的触发条件(6)作为惩罚项组成新的目标函数。$\varepsilon_{p,v_j}^{(N)}$ 作为子网在第 N 次迭代中惩罚项中的惩罚系数。由实验获得 $\Delta\varepsilon$ 取 0.5,算法运行效率最高。

完成目标函数的计算,更新惩罚系数:

$$\varepsilon_{p,u_j}^{(N+1)} = \varepsilon_{p,u_j}^{(N)} + \Delta\varepsilon \sum_{k=1}^{N_t} \partial_{u_j,p,k(r_k^{u_j})} \quad (i=j) \tag{13}$$

$$\varepsilon_{p,u_i}^{(N+1)} = \varepsilon_{p,u_i}^{(N)} \quad (i \neq j) \tag{14}$$

步骤 4 分析路径情况更新触发序列

获取在步骤 3 中完成第 N 次迭代的 AGV 和其余子网中 AGV 所发生的碰撞情况,进行分析。其中 $\hat{r}_k^{u_j}$, $r_k^{u_j}$ 分别为第 N 次及第 $N+1$ 次迭代获得的子网 u_j 的触发序列。

类型 1 节点资源争夺。如图 3 所示,k 时刻 AGVu_i 和 AGVu_j 分别位于节点 1 和节点 5 处,于 $k+1$ 时刻在节点 2 处发生碰撞。此时对装载优先级低的 AGVu_j 进行运输路径的调整,让 AGVu_i 优先通过。

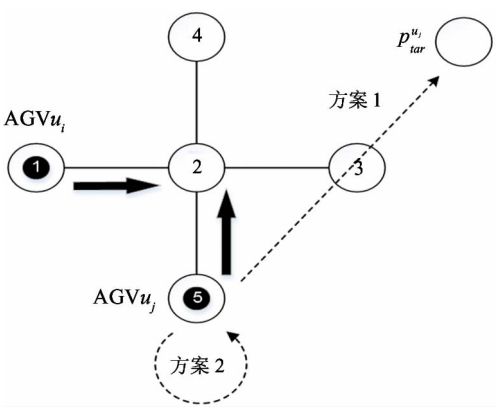

图 3 资源节点争夺示意图

$$\text{Reroute} = \begin{cases} keep(\hat{p}_k^{u_j} = p_{k+1}^{u_j}) J_{keep}^{u_j} < J_{Dijk}^{u_j} \\ Dijstra((p_k^{u_j}, p_{tar}^{u_j}) \setminus \{p_{k+1}^{u_j} = p_{col}^{u_j}\} J_{keep}^{u_j} > J_{Dijk}^{u_j} \end{cases} \tag{15}$$

方案 $keep(\hat{p}_k^{u_j} = p_{k+1}^{u_j})$ 表示 AGVu_j 在 $k+1$ 时刻保持与 k 时刻的位置不变,即原地停止运动等待 AGVu_i 优先通过,再进行自身的运输任务;方案 $Dijstra((p_k^{u_j}, p_{tar}^{u_j}) \setminus \{p_{k+1}^{u_j} = p_{col}^{u_j}\}$ 则重新规划出 AGVu_j 在 k 时刻节点 $p_k^{u_j}$ 到目标节点 $p_{tar}^{u_j}$ 的运输路径($k+1$ 时刻 AGVu_j 运行节点不包括冲突节点 $p_{col}^{u_j}$)。比较两个方案运行路径的目标函数 $J_{keep}^{u_j}$, $J_{Dijk}^{u_j}$, 保证路径最优。

类型 2 双向车道资源抢夺。如图 4 所示,k 时刻 AGVu_i、AGVu_j 分别位于图中节点 1、节点 2 处,并且在 $k+1$ 时刻前往节点 2、节点 1,此时将抢夺车道资源。对此采用两种方案对 AGVu_j 的路径采取调整:

$$\text{Reroute} = \begin{cases} Search(p_{k+1}^{u_j} = \hat{p}_{k+1,f}^{u_j}, p_{k+2}^{u_j} = p_k^{u_j}) J_{cha}^{u_j} < J_{Dijk}^{u_i} \\ Dijstgra((p_k^{u_j}, p_{tar}^{u_j}) \setminus \{\hat{p}_{k+1}^{u_j}\} J_{cha}^{u_j} > J_{Dijk}^{u_i} \end{cases} \tag{16}$$

$Search(p_{k+1}^{u_j} = \hat{p}_{k+1,f}^{u_j})$ 即 AGVu_j 开始搜索 $k+1$ 时刻 $\hat{p}_{k+1}^{u_j}$ 附近的空闲节点 $\hat{p}_{k+1,f}^{u_j}$, 并在 $k+1$ 时刻移动到 $\hat{p}_{k+1,f}^{u_j}$ 上,让 AGVu_i 先行通过 $p_{u_i,2}$, 避免路径上的冲突; $Dijstgra((p_k^{u_j}, p_{tar}^{u_j}) \setminus \{\hat{p}_{k+1}^{u_j}\}$ 如类型 1 所示。同样通过比较两个方案所规划路径的时间目标函数选择最优方案。

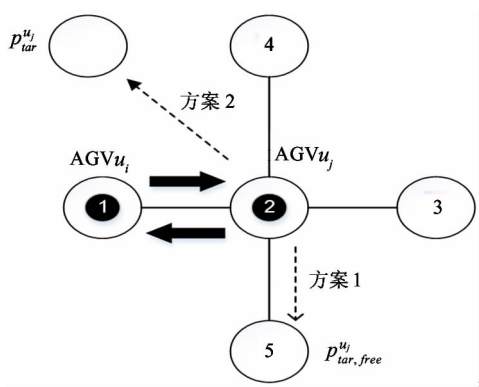

图 4 道路资源争夺示意图

类型 3 目标节点资源抢夺。如图 5 所示,k 时刻 AGVu_i 位于途中节点 1 处,并向节点 2 进行移动,AGVu_j 完成运输任务停留在节点 2 处。在 $k+1$ 时刻,正在卸载物料的 AGVu_j 阻碍了 AGVu_i 的运行,产生了目标节点冲突。对此采取如下方案:

$$\text{Reroute} = \begin{cases} Search(p_{k+1}^{u_j} = \hat{p}_{k+1,f}^{u_j}, p_{k+2}^{u_j} = p_{tar}^{u_j}) J_{cha}^{u_i} < J_{Dijk}^{u_i} \\ Dijstgra((p_k^{u_i}, p_{tar}^{u_i}) \setminus \{\hat{p}_{k+1}^{u_i}\} J_{cha}^{u_i} > J_{Dijk}^{u_i} \end{cases} \tag{17}$$

$Search(p_{k+1}^{u_j} = \hat{p}_{k+1,f}^{u_j}, p_{k+2}^{u_j} = p_{tar}^{u_j})$ 针对位于目标

节点 $p_{tar}^{u_j}$ 的 AGVu_j 在 $k+1$ 时刻搜索并到达 $p_{tar}^{u_j}$ 附近的空闲节点 $p_{tar,f}^{u_j}$。AGVu_i 通过 $p_{tar}^{u_j}$ 后在 $k+2$ 时刻 AGVu_j 重新回到 $p_{tar}^{u_j}$。避免在 $p_{u_i,2}$ 处的冲撞。$Dijstgra((p_k^{u_i}, p_{tar}^{u_i}) \setminus \{\hat{p}_{k+1}^{u_i}\})$ 为重新规划路径。

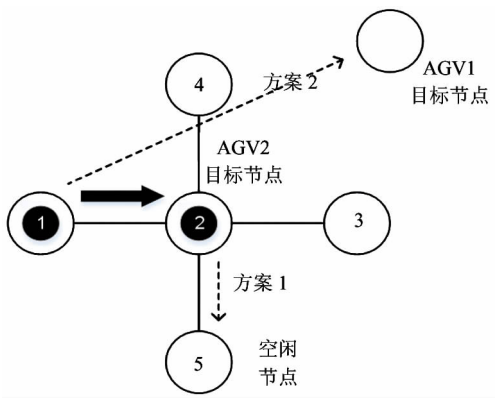

图 5　目标节点阻碍 AGV 运行示意图

最后更新 AGVu_j 的运行路径 $\hat{r}_k^{u_j} = r_k^{u_j}$，$N = N+1$，算法回到第二步，进行新一轮的迭代。

4　实验仿真和数据比较

本文简化物流仓库结构，设计如图 6 所示的 10×10 的二维无向物流仓库交通网络。该仓库模型包括 100 个节点以及 180 条双向车道。根据上述 Petri 网理论，该网络将转化为 900 个库所，2 880 个变迁的时间 Petri 网络模型，引入上述算法进行路径优化，完成运输调度。对于如下 3 个实验均在 8G，2.3GHz 的英特尔 Core-i7-4712 处理器环境下的 Matlab2013a/GUI 软件中进行仿真，为了验证上述算法在大规模的物流环境中能快速收敛，以及在动态的调度环境中有良好的适应性，本文做出以下实验。

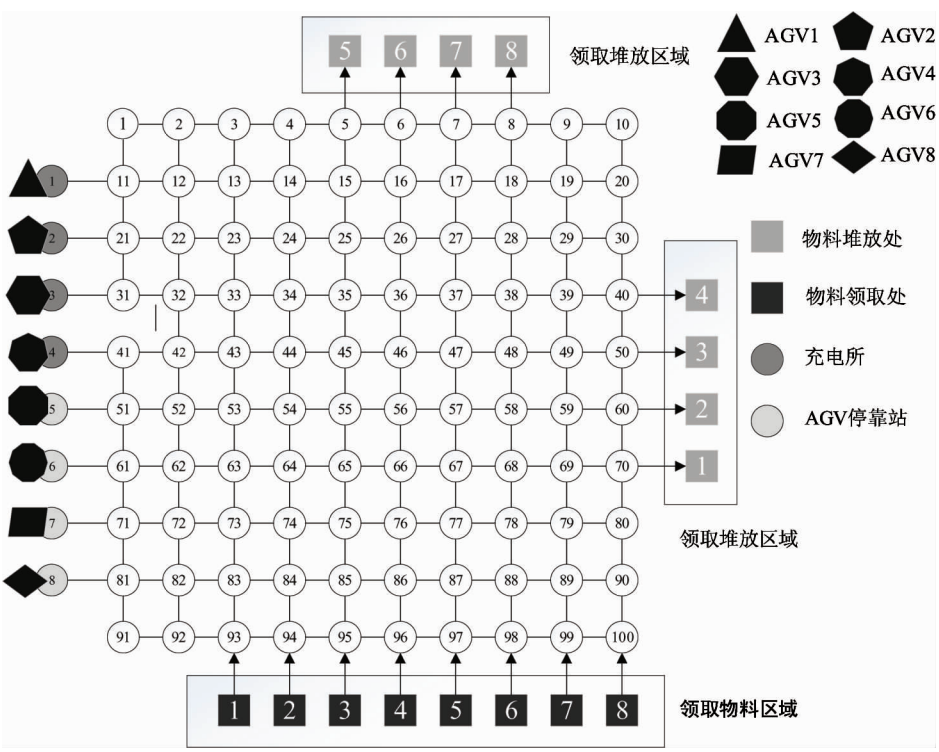

图 6　物流仓库示意图

实验 1　验证本文算法的可行性。对仓库中空闲 AGV，逐一随机分配运输任务。运输任务初始化如表 1 所示。

首先不采取上述路径优化算法对 AGV 所获得最短调度路径的优化，8 辆 AGV 调度结果如图 7 所示。

AGV 运输过程中各阶段具体情况如表 2 所示。由表中数据可知，在 11 时刻，处于卸货的 AGVu_3 停滞于节点 94 阶段，阻碍了同一时刻将经过同一节点的AGVu_4，造成目标节点的冲突类型；在15时

表 1　AGV 运输任务分配表

任务编号	装载点 S	卸载点 E	优先级	车辆 ID
任务 1	98	60	1	AGV1
任务 2	100	8	2	AGV2
任务 3	94	7	3	AGV3
任务 4	93	70	4	AGV4
任务 5	96	5	5	AGV5
任务 6	95	50	6	AGV6
任务 7	99	40	7	AGV7
任务 8	97	6	8	AGV8

刻,分别位于节点 80 和节点 90 的 AGVu_2 和 AGVu_7 在下一时刻产生相向冲突;在 25 时刻,位于节点 47 的 AGVu_6 与位于节点 36 的 AGVu_8 在下一时刻同时前往节点 46,造成了节点冲突。此外,在 16,17,21,24,33,34 时刻分别产生了上述类型的冲突。因此未经算法优化的调度方案是不可行的,因此引入上文提出的优化算法。实验结果如图 8 所示。

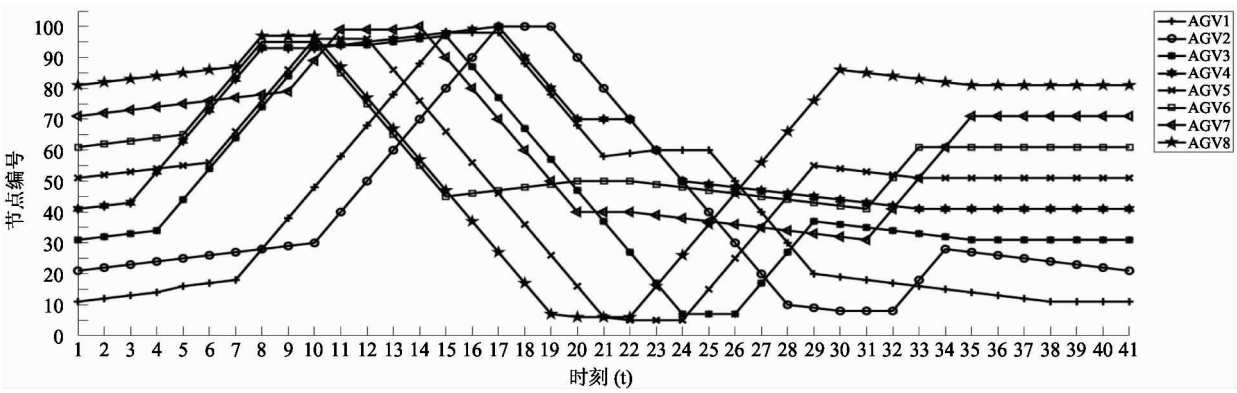

图 7　未优化前 AGV 运输路径

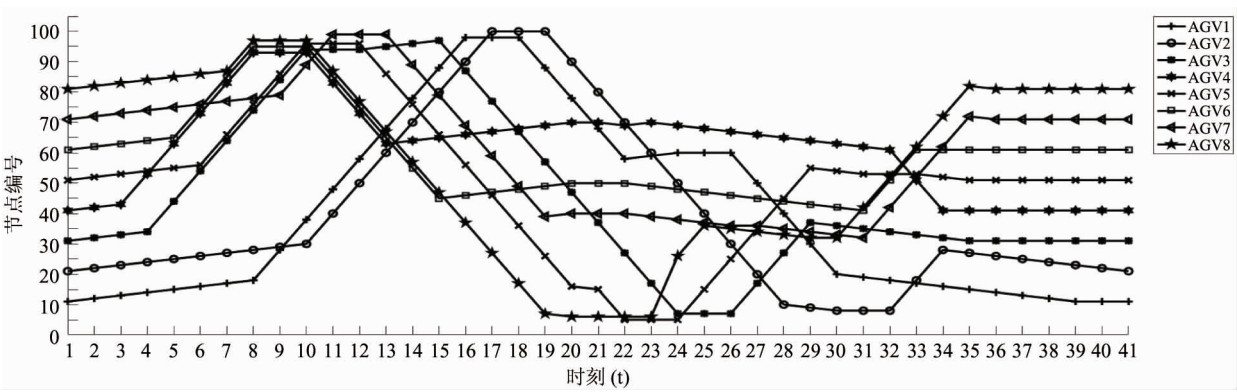

图 8　优化后 AGV 运输路径图

在上述实验环境下,对于第 11 时刻产生的目标节点冲突,低优先级 AGVu_4 在第 10 时刻改变运行路径,从节点 93 运行至节点 83,并经过节点 63 转弯至卸货点 70,避免了节点 94 上的冲突;在第 13 时刻 AGVu_7 重新规划从装载点 99 至卸载点 40 的运行路径,避免在第 15 时刻与处于节点 80 的 AGVu_2 产生冲突;第 25 时刻的节点冲突,低优先级 AGVu_8 同样重新改变下一时刻运行节点,往节点 35 运行。其余产生的冲突和死锁情况,均在算法的迭代运算中消除。本实验验证了上述算法的有效性。

实验 2　本实验提供 $n \times n(4 \leqslant n \leqslant 10$, n 为整数)节点的物流仓库与 l 辆 AGV,为验证本文所提出算法的快速收敛性和调度方案的准确性,对比经典的时间窗路径优化算法,通过改变物流仓库节点个数和 AGV 数量作出 16 组对比实验。为比较路径方案的准确性和高效性,本实验再增加一个效率指标,

定义如下：

$$E = Average\left(\sum_{k=1}^{l} \frac{t_{u_i}^{r}}{n_{u_i} \times t_{u_i}^{s} + t_{u_i}^{w} + t_{u_i}^{r}}\right) \quad (18)$$

其中，$t_{u_i}^{r}$ 表示 AGVu_i 完成运输任务的实际运输时间，n_{u_i} 表示 AGVu_i 运行过程中的转弯次数，$t_{u_i}^{s}$ 表示 AGVu_i 在转弯时所消耗的时间，$t_{u_i}^{w}$ 为 AGVu_i 在行驶过程中的等待时间(AGV 在运输过程中转弯次数过多或等待时间过长均会降低调度系统的运行效率)。两种算法的各组实验结果如表 3 所示。

表 2 实验 1 AGV 未优化前路径调度信息

		AGV1	AGV2	AGV3	AGV4	AGV5	AGV6	AGV7	AGV8
阶段 1	节点编号(n)	11-18-98	21-30-100	31-34-94	41-43-93	51-56-96	61-65-95	71-79-99	81-87-97
	起始时刻(t)	1	1	1	1	1	1	1	1
	结束时刻(t)	16	17	10	8	10	8	11	8
	里程(m)	15	16	9	7	9	7	10	7
阶段 2	节点编号(n)	98-58-60	100-10-8	94-97-7	93-100-70	96-6-5	95-55-50	99-100-40	97-7-6
	起始时刻(t)	18	19	12	10	12	10	13	10
	结束时刻(t)	24	30	24	20	22	20	20	20
	里程(m)	6	11	12	10	10	10	7	10
阶段 3	节点编号(n)	60-20-11	8-28-21	7-37-31	70-50-41	5-55-51	50-41-61	40-31-71	6-86-81
	起始时刻(t)	26	32	26	22	24	22	22	22
	结束时刻(t)	39	41	35	33	33	33	35	35
	里程(m)	13	9	9	11	9	11	13	13

注：因 AGV 运行过程中节点数过多，仅在节点编号中记录 AGV 拐弯的节点。

表 3 实验 2 中 16 组实验结果信息

编号	仓库节点数量($n \times n$)	AGV 数量(n)	基于 Petri 网的外点惩罚函数算法(P)			基于时间窗的路径规划算法(T)			算法性能比(%)(P/T)
			CPU 时间(s)	累计完成时间(s)	平均运输效率(%)	CPU 时间(s)	累计完成时间(s)	平均运输效率(%)	
1	6	3	0.227	24.5	89.9	0.213	24.5	89.9	100
2	6	4	0.245	34	91.1	0.439	36.5	89.4	94.4
3	7	4	0.267	40	92.7	0.528	41	90.4	97.6
4	7	5	0.357	48	91.6	1.828	51	88.7	94.1
5	7	6	0.401	60	90.1	1.243	60.5	89.1	99.2
6	8	4	0.621	120	79.2	4.276	126.5	77.4	94.7
7	8	5	0.685	143	77.1	7.432	150.5	75.0	95.0
8	8	6	0.877	176	76.7	9.200	184	74.9	95.7
9	9	5	0.671	172	83	7.826	181.5	79.5	94.8
10	9	6	0.892	206.5	80.7	8.230	215	78.8	96.4
11	9	7	1.327	232	79.9	10.298	244	78.1	95
12	10	6	0.927	231	82.7	10.378	239	81.1	96.7
13	10	7	1.201	269	82.5	14.307	280.5	80.9	96.0
14	10	8	1.479	311	81.9	19.218	324	80.3	96

注：实验 1 ~ 7 中因地图较小，仅对 AGV 分配单阶段的调度任务

显而易见,随着物流仓库规模的不断增大,AGV 数量的不断增多,本文提出算法的收敛速度呈缓慢的线性增长,且收敛时间均在 1.5 s 以内;而基于时间窗的调度算法的收敛时间在实验 8 ~ 16 中远大于本文提出的算法,并且两种算法在调度方案的准确性和效率值来看,相差在 5% 以内,足以验证本文所提出算法的快速收敛性,相较于时间窗算法,更适合应用于大规模的调度环境中。

实验 3 由实验 2 知,本文提出的调度算法收敛速度均可限制在 1.5 s 以内,因此在静态路径规划的基础上,本文在 AGVu_i 到达仓库节点的同时,以当前节点为初始节点进行调度路径的重新规划来排除刻仓库中发生的突发情况。该实验在实验 1 的基础上,分别在第 5 时刻的节点 65,第 25 时刻的节点 36 以及第 30 时刻的节点 63 处加入 3 个障碍物,其中节点 63 与节点 65 的障碍物停留 1 个时间单位后消除,节点 36 的障碍物存在 2 个时间单位。面对此突发情况,AGV 的调度路径规划结果如图 9 所示(因图 9 较复杂,仅显示重新路径规划的 AGV)。

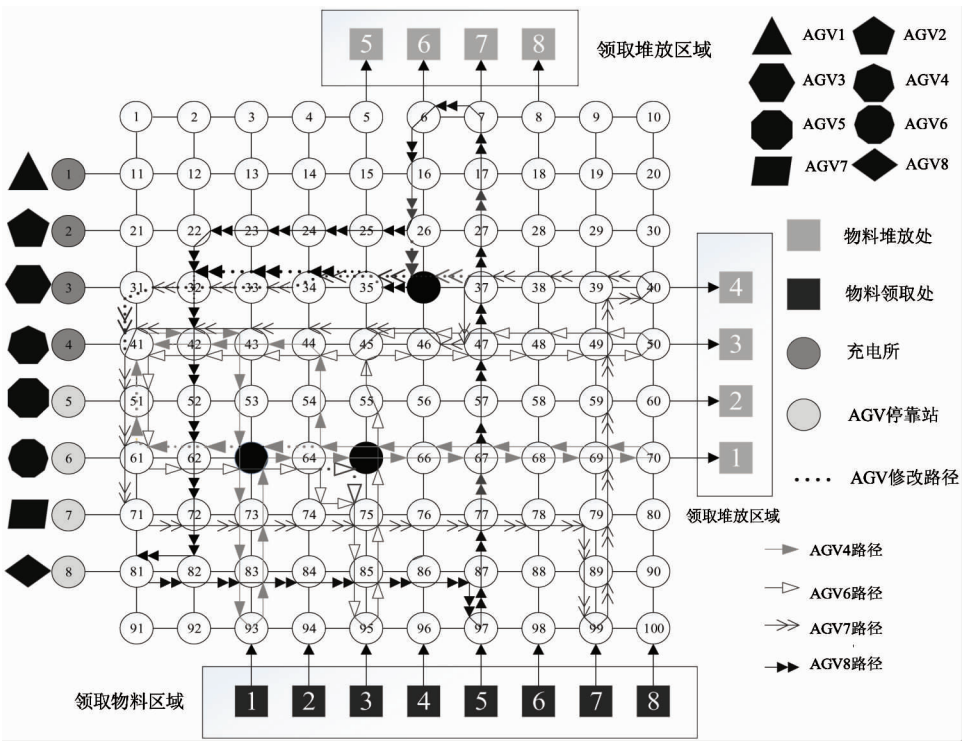

图 9 动态性验证 AGV 路径规划图

在上述情况下,AGVu_4、u_6、u_7、u_8 分别调整了运行路线,成功地避开了障碍物节点,完成运输任务,验证了该算法在实际调度中具有良好的鲁棒性和动态适应性。

5 结 论

本文针对于多 AGV 的物流仓库调度问题,采用时间 Petri 网对双向车道的物流仓库进行建模,面对大规模的物流仓库,通过时间 Petri 网中对资源库所的分解方式将原网以单辆 AGV 为单位逐一分析并整合,降低了 AGV 无冲突路径方案的计算时间。针对 AGV 在路径规划时遇到的碰撞和死锁问题,本文采用改进的外点惩罚函数法对各子网中 AGV 的运行路径进行迭代运算,对算法搜索到的碰撞节点进行节点资源、道路资源、目标节点资源三方面冲突类型进行分析,以时间最优为原则选择相应的解决方案,对 AGV 的路径进行局部优化、重新规划出一套准确无冲突的运行路径。并且由于算法的快速收敛性,使其在实际的动态仓库环境中有很好的适应性。仿真结果验证了算法的可行性和有效性。

在今后面对愈加智能的物流仓库应用时,大规

模仓库调度存在一些特殊的运行情况,比如 AGV 在运输过程中电量不足,无法完成任务;在 AGV 运行的路径中加入加速节点和减速节点;AGV 在运输过程中接收到更高优先级的调度任务。面对上述情况,AGV 调度如何规划做到最优是本文在今后研究中需要考虑的问题。

参考文献

[1] 李继明, 徐震浩, 顾幸生. 基于混合离散粒子群优化的多时间因素作业车间调度研究[J]. 高技术通讯, 2015, 25(11-12):980-989

[2] 贺丽娜, 楼佩煌, 钱晓明,等. 基于时间窗的自动导引车无碰撞路径规划 2010 全国现代制造集成技术[J]. 2010, 16(12):2630-2634

[3] Kalinovcic L, Petrovic T, Bogdan S, et al. Modified Banker's algorithm for scheduling in multi-AGV systems [C]. In: Proceedings of the Automation Science and Engineering, Trieste, Italy, 2011. 351-356

[4] Li J, Meng X, Zhou M C, et al. A two-stage approach to path planning and collision avoidance of multibridge machining systems[J]. *IEEE Transactions on Systems Man & Cybernetics Systems*, 2017, 47(7):1039-1049

[5] 刘志峰, 廖凌浩, 杨文通,等. 基于自适应遗传算法的模具生产调度研究[J]. 高技术通讯,2011,21(9):962-966

[6] 夏田, 王娜. 改进蚁群算法在多 AGV 作业调度中的应用[J]. 物流技术, 2015, 34(23):87-89

[7] 蒋昌俊. Petri 网理论与方法研究综述[J]. 控制与决策, 1997(6):631-636

[8] Wu N Q. Necessary and sufficient conditions for deadlock-free operation in flexible manufacturing systems using a colored Petri net model [J]. *IEEE Transactions on Systems Man & Cybernetics Part C*, 1999, 29(2):192-204

[9] Nishi T, Maeno R. Petri net decomposition approach to optimization of route planning problems for AGV systems [J]. *IEEE Transactions on Automation Science & Engineering*, 2010, 7(3):523-537

[10] 周卫东, 杨加敏, 贾磊,等. 一种 Petri 网结合遗传算法的优化方法及应用[J]. 山东大学学报(工学版), 2005, 35(4):59-63

[11] 李明, 李歧强, 郭庆强,等. 集成启发式规则的混合整数规划调度模型[J]. 高技术通讯, 2010, 20(9):971-977

[12] Kodama A, Nishi T. General conversion of integer programming problems into optimal firing sequence problem of Petri nets [C]. In: Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management, Bali, Indonesia, 2016. 395-399

Research on multi-AGV scheduling method of logistics warehouse based on Petri net

Li Shengnan, Xing Kexin, Lin Yegui, Zhang Guijun

(College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023)

Abstract

A scheduling optimization algorithm based on the timed Petri net is presented for solving the multi-automated guide vehicle (AGV) route planning problem in the logistics warehouse. At first, the timed Petri net is used for modeling the scheduling process of multi-AGV in the warehouses with large-scale bidirectional lanes, which greatly decreases the calculative complexity by analyzing the AGV separately; Secondly, the research structures an objective function with AGV scheduling-time as index by applying the traditional external penalty function methods, which solves collision problems by the successive iteration and update of AGV routine; Moreover, it increases the amount of analyses of collision types and local planning of paths under the principle of the best objective function, which optimizes the scheduling scheme. Finally, the results from the experiment indicate that the algorithm has the advantage of rapid convergence about gaining the optimum collision-free scheduling scheme, which could ensure that the multi-AGVs have a good timely adaptation under the dynamic logistics warehouse dispatch.

Key words: automated guide vehicle (AGV), logistics distribution, Petri net, extend penalty algorithm, collision analysis