doi:10.11835/j.issn.1000-582X.2021.03

# 基于离散型鲸鱼优化算法的 AGV 与机器集成调度方法

邹裕吉,宋豫川,王　毅,王馨坤
(重庆大学 机械传动国家重点实验室,重庆 400044)

**摘要**:针对制造系统中考虑路径冲突的 AGV(automated guided vehicles)与机器集成调度问题,提出一种基于时间窗和 Dijkstra 算法的离散型鲸鱼优化算法。首先,以最小化最大完工时间为目标,建立 AGV 与机器集成调度的数学模型,并采用一种三段式编码实现 AGV 和机器的集成编码,建立连续空间与离散空间之间的映射关系;然后,为了保证初始种群的质量和多样性,设计一种结合混沌映射和对立学习的扩展型 GLR(global, local, random)种群初始化方法;运用 Levy 飞行算子和阈值重启操作进一步提高算法的全局搜索能力;最后,为了提高算法的局部搜索能力,引入结合问题特点的变邻域搜索算法。标准算例仿真实验和柔性仿真实验证明了该算法解决 AGV 和机器集成调度问题的可行性和优越性。

**关键词**:AGV 与机器集成调度;鲸鱼优化算法;Levy 步长;变邻域搜索;Dijkstra 算法
**中图分类号**:TP273;TH187　　**文献标志码**:A　　**文章编号**:1000-582X(2022)06-055-20

# AGV and machine integrated scheduling method based on discrete whale optimization algorithm

*ZOU Yuji*, *SONG Yuchuan*, *WANG Yi*, *WANG Xinkun*
(State Key Laboratory of Mechanical Transmissions, Chongqing University, Chongqing 400044, P. R. China)

**Abstract**: In order to address the integrated scheduling problem of AGVs (automated guided vehicles) and machines with considering path conlict in manufacturing system, an improved discrete whale optimization algorithm based on time window and Dijkstra algorithm was proposed. First, with the goal of minimizing the maximum completion time, a mathematical model of AGV-and-machine integrated scheduling was established. Then, a three-stage coding was used to realize the integrated coding of AGVs and machine, and a continuous space and discrete space were established. Second, in order to ensure the quality and

**收稿日期**:2020-11-25　　**网络出版日期**:2021-04-02
**基金项目**:国家自然科学基金资助项目(51205429);重庆市科技局重庆市技术创新与应用示范专项项目(cstc2018jszx-cyzdX0150)。
Supported by National Natural Science Foundation of China (51205429) and Chongqing Technology Innovation and Application Demonstration Project of Chongqing Science and Technology Bureau (cstc2018jszx-cyzdX0150).
**作者简介**:邹裕吉(1996—),男,重庆大学硕士研究生,主要从事车间调度、智能优化算法研究,(E-mail)zouyuji@cqu.edu.cn。
**通信作者**:宋豫川,男,重庆大学教授,博士生导师,主要从事智能制造和绿色制造等研究,(E-mail)syc@cqu.edu.cn。

diversity of the initial population, an extended GLR population initialization method combining chaotic mapping and opposition learning was designed. Then, the Levy flight operator and threshold restart operation were used to further improve the algorithm's global search capability. Finally, in order to improve the local search ability of the algorithm, a variable neighborhood search algorithm combined with the features of the problem was introduced. Standard simulation experiments and flexible simulation experiments have proved the feasibility and superiority of the proposed algorithm to solve the problem of AGV-and-machine integrated scheduling.

**Keywords**: AGV-and-machine and machine integrated scheduling; whale optimization algorithm; Levy step size; variable neighborhood search algorithm; Dijkstra algorithm

随着"中国制造2025"的提出,以及云计算、互联网和大数据等信息技术的发展,大规模定制化和多品种小批量的生产模式逐渐成为主流。在这种以个性化为主的生产模式下,产品种类繁多,工艺流程复杂,合理的车间调度方案对于提高生产效率至关重要。在传统的柔性作业车间调度问题研究中通常不考虑工件的转移时间,导致这种调度结果并不是理论最优调度结果,对于实际生产的指导存在不足。AGV(automated guided vehicles)是一种高柔性、高可靠性及高效率的先进物流装备,在数字化车间中负责实现工件的转移[1]。在这种情况下,机器的调度和AGV的调度会相互影响,所以AGV和机器集成调度问题越来越受关注。

土耳其学者Ulusoy等[2-4]于1993年首先将AGV和机器集成调度作为研究对象,在不考虑AGV路径冲突的前提下,应用遗传算法求解调度问题,并建立了标准算例库。Abdelmaguid等[5]在Ulusoy的基础上提出一种精英保留策略,研究了6种交叉操作和3种变异操作的最佳组合以及交叉和变异最优概率范围。Zheng等[6]针对该问题,建立了混合整数线性规划模型,以最大完工时间为优化目标,提出禁忌搜索算法结合12种邻域结构求解该模型。Deroussi等[7]提出一种混合粒子群遗传算法用以解决柔性制造系统中AGV和机器集成调度问题,包含了一种基于AGV的解决方案。刘二辉等[8]提出一种改进花授粉算法求解AGV和机器集成调度问题,并研究了AGV数量对调度结果的影响。

以上研究为了将问题简化,都聚焦于AGV和机器的调度,着力于解决AGV和机器的任务指派、任务执行时序问题,将AGV的行驶道路设定为单道单向,并且不考虑潜在的路径冲突。当AGV的行驶道路为单道双向时,系统的运行效率将会得到提高,但路径冲突也会随之增加。路径冲突如果不解决,就会出现AGV碰撞以及路径被死锁等问题,打乱调度计划甚至造成生产系统瘫痪。因此,在AGV和机器集成调度问题中,考虑AGV路径冲突具有重要的现实意义。Said-imehrabad等[9]建立了一个由车间调度和无冲突路径组成的数学模型,提出一种二阶段蚁群算法,首先解决AGV的分配问题,然后再解决AGV的路径冲突问题。Shi等[10]考虑AGV运行时间、停车和转弯的影响,提出一种两阶段调度策略,首先通过Dijkstra算法规划路径,然后考虑AGV之间的约束关系并通过遗传算法对结果进一步优化。Lyu等[11]提出一种改进遗传算法求解柔性作业车间AGV和机器的集成调度问题,通过Dijkstra算法为AGV规划无冲突的最短路径,并把AGV数量当做变量研究最大完工时间的变化,若在算法中采用主动解码,其结果将会更优。

AGV和机器集成调度问题已经被证明是NP难问题的叠加[7],这类问题的解空间庞大且复杂,精确算法难以在可接受的时间内求得最优解,启发式算法更适合求解此类问题。鲸鱼优化算法是Mirjalili等[12]于2016年模仿座头鲸独特捕食行为提出的仿生算法,和其他智能优化算法相比,具有结构简单、参数少、搜索能力强且易于实现等优点,在航路规划[13]、港口拖船调度[14]及选址问题[15]等离散优化问题中的应用越来越多。

综上所述,传统的作业车间调度研究不考虑工件转移时间,无法满足当前生产模式转变所带来的新需求。同时,目前对于AGV和机器集成调度问题的研究大多不考虑路径冲突,与实际生产情况不符,而考虑路径冲突的研究大多采用分阶段的策略进行求解,并非真正意义上的集成调度。鲸鱼优化算法作为一种启发式算法在离散优化领域取得了一定的成功,但在调度领域的应用不多。为了实现无冲突路径的AGV和机器集成调度,笔者首先以最大完工时间为优化目标,建立数学模型;然后,为了求解该模型,将鲸鱼优化算法应

用在集成调度领域，对于鲸鱼优化算法存在的不足，提出针对性的改进方法，即基于时间窗和 Dijkstra 的离散型鲸鱼优化算法；鲸鱼优化算法作为算法迭代的整体框架，基于时间窗的 Dijkstra 算法用于算法解码过程中 AGV 的路径规划；最后，通过标准算例对比实验和柔性算例实验验证了所提算法的性能。

# 1　问题描述及数学模型

## 1.1　问题描述

假设有 $n$ 个需要加工的工件，$m$ 台用于加工的机器，$k$ 台用于运输工件的 AGV。每一个工件有 $n_i$ 道工序，每道工序至少可由一台机器加工，选择不同的机器加工时间一般不同。每台 AGV 可以在任意两台机器以及仓库之间运输工件，运输的路线一般是起点和终点之间可行的最短路径，运输时间取决于运输路线的长度以及路途中的冲突状况。AGV 完成一个运输任务分为空载和负载两个阶段，空载阶段 AGV 需要从当前位置行驶到工件当前所在的位置；负载阶段 AGV 将工件从当前位置运输到加工机器所在位置。

已知条件有：AGV 数量、各机器的位置、所有工序可选择进行加工的机器及对应的加工时间、任意两节点之间的距离（当不可通行时为无穷大）。要求在满足一定的约束条件的情况下达到优化目标的目的，具体可分为以下几个子问题：为每一道工序分配机器和 AGV；为各机器安排加工任务序列和各加工任务开工时间；为各 AGV 安排运输任务序列和各运输任务开始时间。此外，为了简化问题，存在以下假设：

1）在开始时刻，所有 AGV 和工件都在仓库，AGV 随机出发，有先后顺序。

2）每台 AGV 均可运输所有工件且一次只能运输一个工件。

3）AGV 的行驶速度恒定不变。

4）每台加工机器的工件缓冲区无限大。

5）AGV 完成运输任务后可以停靠在机床旁边，不需要返回仓库。

6）每台机器一次只能加工一个工件且一旦开始加工就不会中断。

7）加工准备时间以及装卸工件的时间算在加工时间中。

8）所有路段同一时刻只允许一台 AGV 通过，且任意路段可容纳 AGV 的车身，不存在 AGV 占用两个车道的情况。

9）同一个工件的工序有先后约束，不同工件之间的工序无约束。

10）不考虑 AGV 故障、电量等因素，也不考虑机器故障。

11）所有工件都增加一道虚拟工序，加工时间为 0，加工机器的位置为仓库位置，便于处理将工件运送回仓库。

## 1.2　数学模型

根据以上说明，在建立模型之前，对引入的符号和变量做如下说明：$C_i$ 为工件 $i$ 的完工时间；$m$ 为机器数量；$n$ 为工件数量；$v$ 为 AGV 数量；$p$ 为电子地图中节点数量；$H$ 为调度周期时长；$n_i$ 为工件 $i$ 的工序数；$O_{ij}$ 为工件 $i$ 的第 $j$ 道工序，$i\in\{1,2,\cdots,n\}$，$j\in\{1,2,\cdots,n_i\}$；$R_w$ 为编号为 $w$ 的 AGV，$w\in\{1,2,\cdots,v\}$；$M_k$ 为编号为 $k$ 的机床，$k\in\{1,2,\cdots,m\}$；$p^{\mathrm{s}}_{ijk}$ 为工序 $O_{ij}$ 在 $M_k$ 上开始加工的时间；$p^{\mathrm{e}}_{ijk}$ 为工序 $O_{ij}$ 在 $M_k$ 上的完工时间；$p_{ijk}$ 为工序 $O_{ij}$ 在 $M_k$ 上加工所需时间；$e^{\mathrm{s}}_{ijw}$ 为工序 $O_{ij}$ 由 $R_w$ 运输，小车空载开始时间；$e_{ijw}$ 为工序 $O_{ij}$ 由 $R_w$ 运输，小车空载运行时间；$e^{\mathrm{e}}_{ijw}$ 为工序 $O_{ij}$ 由 $R_w$ 运输，小车空载结束时间；$l^{\mathrm{s}}_{ijw}$ 为工序 $O_{ij}$ 由 $R_w$ 运输，小车负载开始时间；$l_{ijw}$ 为工序 $O_{ij}$ 由 $R_w$ 运输，小车负载运行时间；$l^{\mathrm{e}}_{ijw}$ 为工序 $O_{ij}$ 由 $R_w$ 运输，小车负载结束时间；$\mathrm{s}^{\mathrm{in}}_{wxr}$ 为编号 $w$ 的 AGV 执行其第 $x$ 个任务驶入第 $r$ 个路段的节点；$\mathrm{s}^{\mathrm{out}}_{wxr}$ 为编号 $w$ 的 AGV 执行其第 $x$ 个任务驶出第 $r$ 个路段的节点；$t^{\mathrm{in}}_{wxr}$ 为编号 $w$ 的 AGV 执行其第 $x$ 个任务驶入第 $r$ 个路段的节点的时间；$t^{\mathrm{out}}_{wxr}$ 为编号 $w$ 的 AGV 执行其第 $x$ 个任务驶出第 $r$ 个路段的节点的时间。

$$x_{ijk}=\begin{cases}1 & \text{若 } O_{ij} \text{ 在机器 } k \text{ 上加工，}\\ 0 & \text{其他。}\end{cases}\qquad z_{ijw}=\begin{cases}1 & \text{若 } O_{ij} \text{ 由 } R_w \text{ 负责运输，}\\ 0 & \text{其他。}\end{cases}$$

$$\alpha_{pqijk}=\begin{cases}1 & O_{pq} \text{先于} O_{ij} \text{在机器 } k \text{ 上加工，}\\ 0 & \text{其他。}\end{cases}\qquad \beta_{pqijw}=\begin{cases}1 & O_{pq} \text{先于} O_{ij} \text{由 } R_w \text{ 运输，}\\ 0 & \text{其他。}\end{cases}$$

$\delta_{wst}=\begin{cases}1 & \text{若 } R_w \text{ 在 } t \text{ 时刻到达节点 } z,\\ 0 & \text{其他。}\end{cases}$ $\varepsilon_{rt}^{z}=\begin{cases}1 & t \text{ 时刻不允许从节点 } z \text{ 进入路段 } r,\\ 0 & \text{其他。}\end{cases}$

目标函数如下：

$$f=\min(\max(C_1,\cdots,C_i,\cdots,C_n))。\tag{1}$$

约束条件如下：

1)任意工序都只能由一台机器完成加工。

$$\sum_{k=1}^{m}x_{ijk}=1。\tag{2}$$

2)任意一道工序最多只由一台 AGV 负责运输。

$$\sum_{w=1}^{v}z_{ijw}\leqslant 1。\tag{3}$$

3)AGV 空载出发时间不早于 AGV 上一次运输任务结束时间和工件开工时间。

$$e_{ijw}^{s}\geqslant p_{ijk}^{s},\tag{4}$$

$$t_{ijw}^{s}\geqslant t_{pqw}^{e}\cdot(1-\beta_{pqijw}),\forall p\in(1,2,\cdots,n),\forall q\in(1,2,\cdots,n_p)。\tag{5}$$

4)AGV 空载结束时间为空载出发时间与空载运行时间之和。

$$e_{ijw}^{e}=e_{ijw}^{s}+e_{ijw}。\tag{6}$$

5)AGV 负载出发时间不早于 AGV 空载结束时间和工件完工时间。

$$l_{ijw}^{s}\geqslant e_{ijw}^{e},\tag{7}$$

$$l_{ijw}^{s}\geqslant p_{ijk}^{e}。\tag{8}$$

6)AGV 负载结束时间为负载出发时间与负载运行时间之和。

$$l_{ijw}^{e}=l_{ijw}^{s}+l_{ijw}。\tag{9}$$

7)某工序的开工时间不早于负载结束时间和机器前序工序的完工时间。

$$p_{ijk}^{s}\geqslant l_{ijw}^{e},\tag{10}$$

$$p_{ijk}^{s}\geqslant p_{pqk}^{e}\cdot(1-\alpha_{pqijk})。\tag{11}$$

8)工序的完工时间为开工时间和加工时间之和。

$$p_{ijk}^{e}=p_{ijk}^{s}+p_{ijk}。\tag{12}$$

9)工件的完工时间为 AGV 将其运送到仓库的时间。

$$C_i=\sum_{w=1}^{v}l_{in_iw}^{e}\cdot z_{ijw}。\tag{13}$$

10)某个路段有 AGV 进入之后，在该 AGV 驶出该路段之前，不允许其他 AGV 从该路段出口驶入。

$$\varepsilon_{rt}^{z}=1,z=z_{v'x'r}^{\text{out}},t\in[t_{v'x'r}^{\text{in}},t_{v'x'r}^{\text{out}}]。\tag{14}$$

11)同一时刻任意一个节点只能容下一台 AGV。

$$\sum_{w=1}^{v}\sum_{z=1}^{p}\delta_{wst}\leqslant 1,t\in(1,H)。\tag{15}$$

## 2 算法描述

### 2.1 基本鲸鱼优化算法

鲸鱼优化算法是 Mirjalili 通过座头鲸独特的泡泡网狩猎方式抽象而来，种群中鲸鱼的最优位置就是算法所获得的最优解，迭代过程主要分为两个阶段：探索阶段和开发阶段，通过 $|\boldsymbol{A}|$ 和 1 的大小关系来区分[12]。

$$A=2a\cdot r_1-a,\tag{16}$$

$$a=2-2\cdot\frac{f}{f_{\max}}。\tag{17}$$

式中：$r_1$ 是 0 到 1 之间的随机数；$f$ 为当前迭代次数；$f_{\max}$ 为最大迭代次数。

当 $|\boldsymbol{A}|\leqslant 1$ 时，算法侧重于局部开发，鲸鱼个体向当前种群最优鲸鱼个体靠近，有收缩包围和螺旋更新两种靠近方式，为了实现这两种方式的同步，引入概率 $p$ 去判断应该执行哪种更新方式。

$$\boldsymbol{X}_{t+1}=\begin{cases}\boldsymbol{X}_{\text{best}}(t)-\boldsymbol{A}\cdot\boldsymbol{D} & p<0.5,\\ \boldsymbol{X}_{\text{best}}+\boldsymbol{D}\cdot e^{bl}\cdot\cos(2\pi l) & p\geqslant 0.5。\end{cases}\tag{18}$$

$$\boldsymbol{D}=\left|\boldsymbol{C}\,\boldsymbol{X}_{\text{best}}-\boldsymbol{X}_{t}\right|,\tag{19}$$

$$\boldsymbol{C}=2\cdot\boldsymbol{r}_{2}。\tag{20}$$

式中：$\boldsymbol{D}$ 为更新步长；$b$ 为控制螺旋线形状的常量系数；$l$ 为[0,1]范围服从均匀分布的随机数；$\boldsymbol{r}_2$ 为 0 到 1 之间的随机数。

当 $|\boldsymbol{A}|>1$ 时，算法侧重于进行全局搜索，鲸鱼个体向当前种群中一个随机选择的鲸鱼个体靠近。

$$\boldsymbol{X}_{t+1}=\boldsymbol{X}_{\text{rand}}(t)-\boldsymbol{A}\cdot\boldsymbol{D},\tag{21}$$

$$\boldsymbol{D}=\left|\boldsymbol{C}\cdot\boldsymbol{X}_{\text{rand}}-\boldsymbol{X}_{t}\right|。\tag{22}$$

鲸鱼优化算法作为新型群智能优化算法，存在结构简单、参数少、搜索能力强且易于实现等优点，但也有群智能优化算法的通病，如收敛速度慢、容易陷入局部最优及收敛精度较低等。针对以上缺点，笔者结合车间调度问题的特点提出一种改进型鲸鱼优化算法。

## 2.2　离散化改造

为了更加清晰地描述问题，此处引入一个实例加以说明，各工件的信息如表 1 所示。

**表 1　加工信息**

**Table 1　Processing information**

| 工件 | 工序 | 可选择机器的加工时间/min | | | |
|---|---|---|---|---|---|
| | | $M1$ | $M2$ | $M3$ | $M4$ |
| $J_1$ | $O_{11}$ | 2 | 4 | — | 3 |
| | $O_{12}$ | 4 | — | 5 | 2 |
| | $O_{13}$ | 6 | — | 4 | 8 |
| $J_2$ | $O_{21}$ | — | 3 | 2 | 6 |
| | $O_{22}$ | 5 | 7 | — | — |
| $J_3$ | $O_{31}$ | 2 | 3 | 5 | — |
| | $O_{32}$ | 4 | 5 | — | 8 |
| | $O_{33}$ | 6 | — | — | 4 |
| | $O_{34}$ | 5 | 4 | — | — |

### 2.2.1　问题编码

作业车间机器和 AGV 共同调度问题中的调度子问题包含 3 个部分：工序排序、机器分配和 AGV 分配。笔者采用三段式编码，由 3 条基因串组成：第 1 条为基于工序的基因串，第 2 条为基于机器的基因串，第 3 条为基于 AGV 的基因串。基因串中每个位置值的取值范围为[$-\delta$,$\delta$]。由于引入了 Levy 步长更新鲸鱼个体，如果 $\delta$ 取得过小，更新后的个体工序序列基因值相对大小变化较大，导致个体被破坏；如果 $\delta$ 取得过大，则会导致 Levy 步长策略效果不明显，一般设为 2～5 之间的数。图 1 所示为一个合法的编码。

基于工序的基因串

| 1.46 | −0.12 | 0.23 | 1.67 | 2.74 | −2.62 | −1.35 | 2.81 | 1.31 |
|---|---|---|---|---|---|---|---|---|

基于机器的基因串

| −2.72 | 2.82 | 0.12 | −2.52 | −1.91 | 0.48 | 1.63 | −0.32 | 0.31 |
|---|---|---|---|---|---|---|---|---|

基于AGV的基因串

| −1.63 | 1.62 | −0.13 | 0.23 | 2.57 | −2.86 | 1.53 | 1.94 | 0.22 |
|---|---|---|---|---|---|---|---|---|

**图 1　个体连续编码示意图**

**Fig. 1　Schematic diagram of individual continuous coding**

2.2.2 编码转换

鲸鱼优化算法的解空间是连续空间，然而调度问题是离散组合优化问题。为了使得鲸鱼优化算法适用于调度问题，需要将连续的解空间映射到离散的解空间。由于工序排序、机器选择和AGV选择的约束不同，需要采取不同的转换机制。

采用LPV(largest position value)规则[16]对工序序列进行解空间的转换。具体的操作过程为，首先给所有基因从左至右安排一个从1开始的固定ID与之对应，然后将他们按从大到小的顺序排序，最后根据每个基因对应的ID得到一个新的基因串，按照每个工件的工序数转换为可以用以解码的序列，过程见图2。

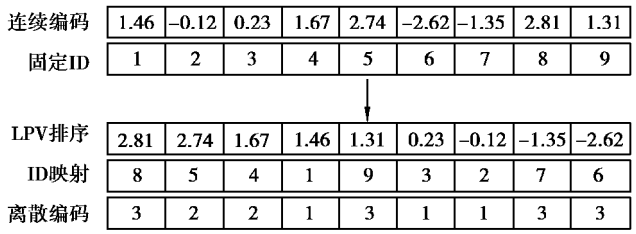


**图2 LPV转换示意图**

**Fig. 2 LPV conversion diagram**

图2中，离散编码中的值表示的是工件号，从左至右出现的次数表示的是工序，第1个3表示的是工序$O_{31}$，第2个3表示的是工序$O_{32}$。

对于机器和AGV的解空间转换，采用文献[17]的方法。首先根据式(16)将连续的编码转换为离散编码，然后根据工序可选机器和AGV序列，转换为具体的机器和AGV编号。同时，通过逆运算实现从离散空间映射到连续空间，具体见式(23)。在AGV和机器基因串中，从左至右依次为从第一个工件到最后一个工件每一道工序对应的机器和AGV。具体见图3和图4。

$$u(i)=\mathrm{round}\left(\frac{m(i)+\delta}{2\delta}(z(i)-1)+1\right), \tag{23}$$

式中：$m(i)$为连续编码的基因值；$z(i)$为该工序可选的加工机器数或者AGV数；$u(i)$为得到的离散基因值。

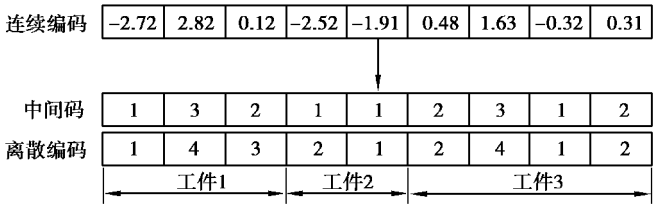


**图3 机器编码转换示意图**

**Fig. 3 Schematic diagram of machine coding conversion**

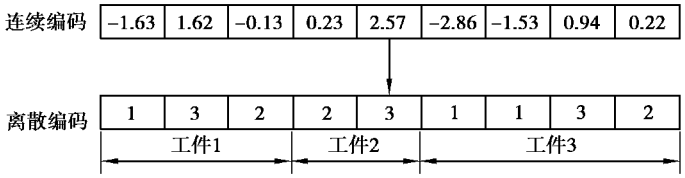


**图4 AGV编码转换示意图**

**Fig. 4 Schematic diagram of AGV encoding conversion**

基于机器的基因串前3个基因值为工件1对应的3道工序对应的机器，图3和4表达的意思为：工件1的3道工序分别选择1,4,3号机器进行加工；基于AGV的基因串与基于机器的基因串类似。AGV的基因

串前 3 个数字的意思为：工件 1 3 道工序分别选择 1，3，2 号 AGV 进行运输。针对机器和 AGV 共同调度问题，上述编码及其转换方式不会产生非法解。

2.2.3　规则调整

算法迭代过程中，可能会导致鲸鱼个体的某些维度越界。在原始鲸鱼优化算法中，每一轮迭代完成以后才会对越界个体进行调整。然而这种处理方式会导致一些问题，如在鲸鱼个体迭代过程中可能会选中越界的个体作为位置更新的基准，进而造成更多个体越界，不仅会增加算法的运行时间而且还会导致一些非越界个体被破坏。因此，选择在个体位置更新以后就进行个体越界的调整。原始鲸鱼优化算法中，将越界的维度设置为最大值或者最小值，当越界情况较多时，导致个体之间相似度越来越大。因此，可以将越界的维度设置为接近边界的一个值。具体调整规则见式(24)。

$$X(i)=\begin{cases}\delta-0.05\delta\cdot r_1 & X(i)>\delta,\\ -\delta+0.05\delta\cdot r_1 & X(i)<\delta。\end{cases}\tag{24}$$

式中 $X(i)$表示个体 $X$ 的第 $i$ 个维度。

### 2.3　种群初始化

初始化种群对于进化算法来说十分重要[18]，初始解的质量和多样性对于算法的求解精度和收敛速度有非常大的影响。这里提出一种扩展 GLR 选择方法结合混沌映射和对立学习的种群初始化方法。

如果在生成初始解的时候，考虑各机器和 AGV 的工作时间与负载均衡可以大大提高种群的质量。在 FJSP(flexible job-shop scheduling problem)问题上 GLR(global，local，random)机器选择方法[19]被证明是一个比较有效的方法，该方法在随机生成 50%个体的基础上，考虑工作时间与负载均衡，使用全局选择方法生成 20%的个体。在该方法中，首先随机选择一道工序，将所有可加工机器已工作时间加上该工序的加工时间，然后选择目前加工时间最小的那个机器作为本道工序的加工机器并更新其加工时间。局部选择方法生成 30%的个体，在该方法中，首先随机选择一个工件，为该工件的所有工序选择机器，按照全局选择的方法进行选择机器，不同的是，当一个工件所有工序都选择了对应的机器后，将所有的机器运行时间置零。该方法也应用在 AGV 的选择上，在 AGV 序列初始化时，为了减少计算量，在计算 AGV 的运行时间时不考虑冲突。

由于对于优化问题的全局最优解没有任何先验知识，初始种群应该尽可能地均匀分布在解空间中。所以增强初始种群的多样性以奠定算法全局搜索的基础一直是算法优化的一个方向。目前，大多数研究主要采用混沌映射策略和对立学习策略来增强智能优化算法初始种群的多样性。Ewees 等[20]提出一种基于混沌映射的多元宇宙优化算法用以提高算法的全局搜索性能；Shen 等[21]提出一种基于 PSO 和混沌映射的多种群优化算法；Alamri 等[22]提出一种对立学习策略对鲸鱼优化算法进行改进。混沌映射和对立学习在改进算法上取得了许多成功，但是由于有很多混沌映射和对立学习方法可供选择，所以选择什么方法可以最大程度上提高算法性能是一个值得研究的问题；Elaziz 等[23]使用基于差分进化算法的超启发式算法选择混沌映射方法和对立学习方法及其比例以得出最优组合，通过对 35 组标准函数的测试证明了其结论的可靠性。在经过 GLR 方法生成个体的基础上使用高斯映射和标准对立学习且比例为 25%的策略生成初始化种群。高斯映射和标准对立学习表达式见式(25)(26)。

$$x_d^{i+1}\begin{cases}0 & x_d=0,\\ \dfrac{1}{x_d^i}-\left[\dfrac{1}{x_d^i}\right] & 其他。\end{cases}\tag{25}$$

$$x'_d=x^i_{d\mathrm{L}}+x^i_{d\mathrm{U}}-x^i_d。\tag{26}$$

式中：$x_d^{i+1}$ 表示经过高斯映射后的大小；[]表示取整；$x'_d$ 表示经过对立学习后的大小；$x^i_{d\mathrm{L}}$ 和 $x^i_{d\mathrm{U}}$ 分别为 $x^i_d$ 取值的上下限。

### 2.4　Levy 飞行策略和阈值

Levy 飞行是一种服从 Levy 分布的随机搜索路径，Levy 分布是法国数学家 Levy 提出的一种概率分布。

由于Levy分布是一种重尾分布，所以Levy飞行是一种短距离和偶尔长距离相间的搜索方式。随着Levy飞行在连续优化领域取得大量成功，近年来逐渐有学者将其应用到组合优化问题的研究中。王学武等[24]提出一种结合Levy飞行的粒子群算法以解决点焊机器人的路径优化问题；徐坤等[25]提出一种Levy飞行模式与蚁群算法的信息素更新方式相结合的算法用来解决TSP问题；Liu等[26]将levy飞行结合差分进化算法用来解决JSP问题。以上表明Levy策略在解决组合优化问题上也有较优异的表现。鲸鱼优化算法在迭代后期$a$逐渐减小，算法容易陷入局部最优。因此，将Levy飞行策略引入到算法迭代后期，增强其跳出局部最优的能力。同时，在算法的探索阶段引入Levy飞行策略可以增强算法的全局搜索能力。

$$\boldsymbol{X}(t+1)=\boldsymbol{X}(t+1)+[r-1/2]\oplus L。\tag{27}$$

式中：$\boldsymbol{X}(t)$表示的是经过鲸鱼优化算法迭代以后的个体；$[r-1/2]$有3种取值，$r$是服从范围在[0,1]的均匀分布函数，当$r<1/2$时，$[r-1/2]$取$-1$；当$r=1/2$时，$[r-1/2]$取0；当$r>1/2$时，$[r-1/2]$取1；$\oplus$表示的是Levy飞行操作。所采用的工序编码转换方式，本质上取决于各个维度的相对值大小，所以对于工序向量要对每个维度单独进行Levy飞行操作。Levy飞行具体的数学表达式见式(28)～(31)。

$$L(s)\sim|s|^{-1-\beta},0<\beta\leqslant 2,\tag{28}$$

$$s=\frac{u}{|v|^{\frac{1}{\beta}}},\tag{29}$$

$$\mu\sim N(0,\sigma_u^2),v\sim N(0,\sigma_v^2),\tag{30}$$

$$\sigma_u=\left\{\frac{\Gamma(1+\beta)\sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left[\frac{(1+\beta)}{2}\right]\beta\,2^{\frac{\beta-1}{2}}}\right\},\sigma_v=1。\tag{31}$$

式中：$\Gamma$是标准伽马函数；$\beta$一般取1.5。

阈值操作也是防止算法陷入局部最优常见做法之一。具体的做法为：在算法开始时设置一个全局最优保持代数$g$并令其等于零，然后在算法迭代过程中记录当前最优值保持代数，当$g$达到设置的阈值$l$时，用随机生成的个体代替50%较劣个体。阈值的设置对算法有较大的影响，当阈值设置过小时，算法不易收敛且导致算法复杂度增加；当阈值设置过大时，会造成算法收敛速度变慢。一般设为最大迭代次数的10%～15%。

### 2.5 局部搜索

在开发阶段，种群个体以当前最优个体为基准来更新自己的位置，对较优个体进行局部搜索可以较大提升算法求解精度和收敛速度。由于AGV设备比较昂贵，在车间中AGV数量往往较少。对于机器和AGV共同调度的问题，经常出现工件等待AGV运输的现象，合理地分配AGV对于减小最大完工时间至关重要。提出的局部搜索分为两部分：基于工序和机器的变邻域搜索和基于约束AGV的邻域搜索，把最迟完成搬运任务的AGV定义为约束AGV。

变邻域搜索算法由Mladenović等[27]于1997年提出，通过不同的邻域递进式排查使得搜索空间更加广阔深入，具有较强局部搜索能力。对较优的3个个体进行变邻域搜索，相对于对最优个体进行变邻域搜索可以增大获得最优解的可能性。设计VNS(variable neighborhood search)算法首先要设计邻域结构，采用以下3种邻域结构。

1)邻域结构N1：在其工序的基因串中，随机选择两个位置，保证这两个位置必须对应不同工件的工序，互换它们的位置。

2)邻域结构N2：在其工序的基因串中，随机选择两个位置，将后一个基因插入到前一个基因的前一个位置。

3)邻域结构N3：在其机器的基因串中，随机选择一个位置，保证该位置对应工序的可加工机器不唯一，从该工序的可加工机器中随机选择一个替换该位置的机器。

在以上 3 种邻域结构的基础上，基于工序和机器的变邻域搜索算法步骤如下。

步骤 1　设置初始参数，将要进行变邻域搜索的个体设为初始个体 $X$；当前迭代次数 $n$ 设为 1，最大迭代次数 $n_{\max}$ 设为 5，$p$ 设为 1，$p_{\max}$ 设为 3。

步骤 2　判断是否达到循环终止条件 $n \geqslant n_{\max}$，如满足，输出当前个体 $X$；否则，转步骤 3。

步骤 3　在个体 $X$ 的基础上随机选择一个邻域结构，得到一个扰动个体。

步骤 4　在扰动个体 $X'$ 的基础上进行变邻域搜索，其具体步骤为：

1）判断是否达到终止条件 $p \geqslant p_{\max}$，如满足，输出当前解 $X'$。

2）在 $X'$ 的基础上选择编号和 $p$ 对应的邻域结构，得到新个体 $X''$，如果有 $f(X'') \leqslant f(X')$，则 $X' \leftarrow X''$，$p \leftarrow 1$；如果 $f(X'') = f(X')$，则以 0.5 的概率更新个体，$p \leftarrow 1$；否则个体不变，$p \leftarrow p+1$，转 1）。

步骤 5　令 $X \leftarrow X'$，$n \leftarrow n+1$，转步骤 2。

基于约束 AGV 的邻域搜索以基于工序和机器的变邻域搜索输出的个体为初始解，步骤如下。

步骤①　初始化参数，将初始个体设为 $X$；当前操作的 AGV 任务编号 $n \leftarrow 1$，$n_{\max}$ 为总任务数。

步骤②　判断是否满足终止条件 $n \geqslant n_{\max}$，如满足，输出个体 $X$；否则，转步骤③。

步骤③　将当前任务分配给当前运行时间最小的 AGV 得到新个体 $X'$；如果 $f(X') < f(X)$，$X \leftarrow X'$；如果 $f(X') = f(X)$，则以 0.5 的概率接受当前个体；否则，不接受当前个体更新。$n \leftarrow n+1$，转步骤 2。

### 2.6　多 AGV 路径规划

在算法对个体进行解码的过程中，工序确定加工机器以后，还需要根据工件的最早可开工时间以及机器的加工时间确定工序的实际开完工时间。在传统的车间调度方案中，工序的最早可开工时间为上道工序的完工时间，而 AGV 和机器集成调度问题要考虑工件的转移时间以及路途中可能遇到的路径冲突，因此要想确定工序的最早可开工时间，需要先给 AGV 规划无冲突路径，笔者采用一种基于时间窗的 Dijkstra 算法为 AGV 规划无冲突路径。

#### 2.6.1　Dijkstra 算法

在最短路径规划中 Dijkstra 算法能够从全局出发，具有较强的稳定性且算法简单[28]。Dijkstra 算法要访问所有节点及其连通的节点，肯定可以求出最短路径，但是在节点和路段较多的地图中，求解时间会急剧增大。工厂环境中，AGV 沿着固定的轨道行走，AGV 也只需往返于各机器之间。因此，工厂环境下的地图节点和路段都比较少，Dijstra 算法可以较好地发挥性能。另外，优化目标是最小化最大完工时间，AGV 只有尽早完成任务工件才能尽早开工，基于贪心策略选择最短路径是全局最优解。因此，应用 Dijkstra 算法对 AGV 进行路径规划是一个较好的选择。

#### 2.6.2　基于时间窗的 Dijkstra 算法

在多 AGV 参与的制造系统中，路径冲突是路径规划中常见的问题。只有解决了路径冲突才能实现真正意义上的集成调度。在工厂环境下的 AGV 冲突主要有：相向冲突、节点占用冲突和路口冲突。如图 5 所示。

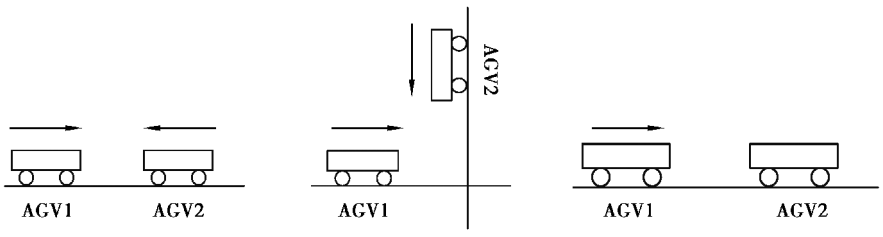


**图 5　AGV 冲突示意图**

**Fig. 5　AGV conflict diagram**

AGV 路径冲突本质上为 AGV 在时间或者空间上出现了重叠，时间窗方法是一种冲突预测方法，将地图以路段为单位记录锁定的时间。时间窗被证明是一个可以有效避免 AGV 冲突的方法。对于时间窗的表示

方式为：$[t_{in},t_{out},n_{in},n_{out}]$，$n_{in}$为进入该路段的节点，$n_{out}$为离开该路段的节点，$t_{in}$和$t_{out}$分别为进入和离开的时间。这种表示方式可以包含两部分信息，节点的时间窗和路段的时间窗。由于 AGV 车身通过节点需要时间且要保持安全距离，因此会在一段时间内占用该节点。当两个节点时间窗有重叠时，说明这两个 AGV 有节点冲突。当两个路段时间窗有重叠并且两个 AGV 进入该路段的节点不同时，说明这两个 AGV 有相向冲突。

解决 AGV 冲突主要有两种方法：等待法和二次规划法。对于相向冲突，只有二次规划才能解决冲突。对于节点占用冲突和路口冲突，等待法肯定可以解决冲突，在时间窗上表现为将要等待的 AGV 的时间窗往后平移至没有重叠，二次规划也可以解决，但是可能会导致新的冲突甚至死锁。假设等待法多花费的时间为 $\Delta t$，二次规划的行驶时间为 $T_2$。如果满足式(32)，则采用二次规划法。

$$T_1+\Delta t_1>T_n+\Delta t_n。\tag{32}$$

将时间窗嵌入到 Dijkstra 算法中，在对 AGV 进行路径规划的同时考虑避碰。为了减少计算量，对已经规划好路径的 AGV 不再更改路径。算法流程图图 6。

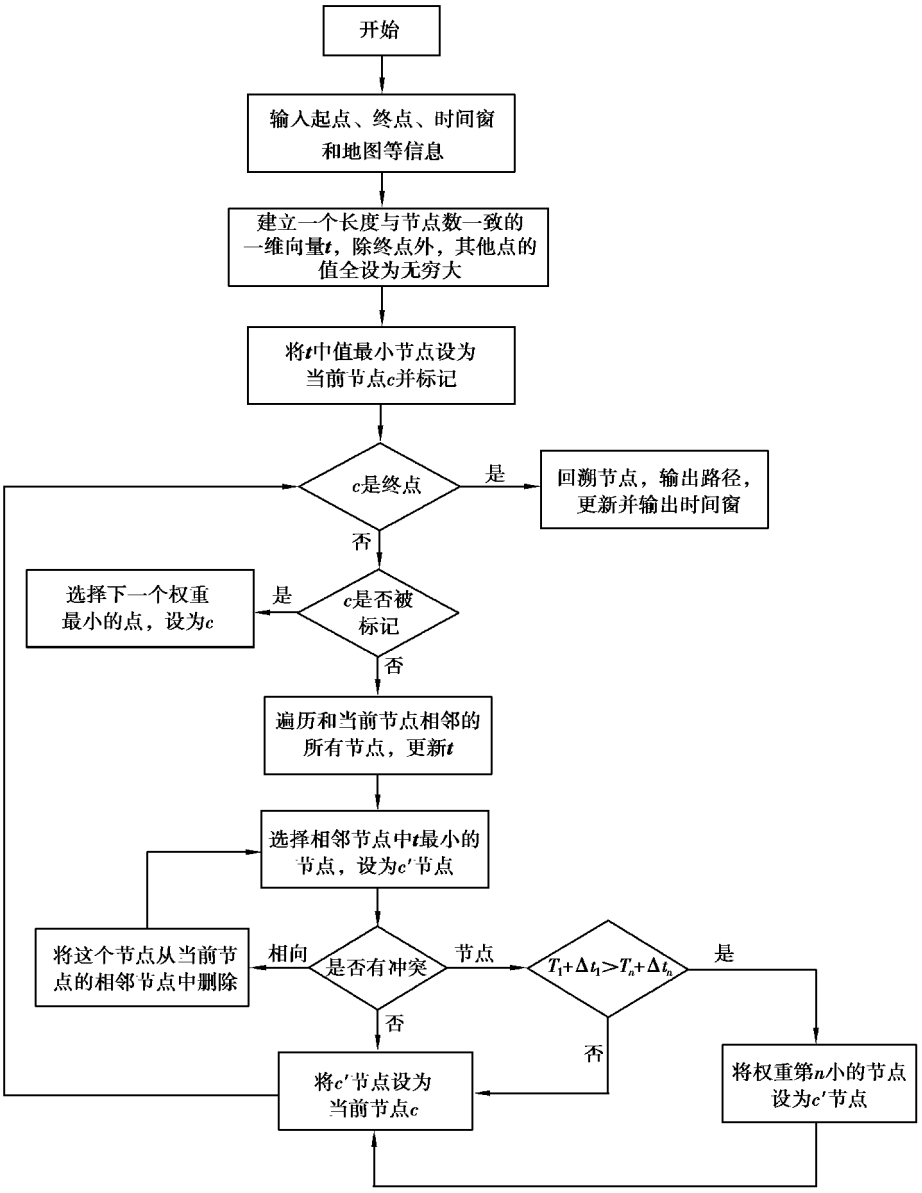


**图 6 基于时间窗的 Dijkstra 算法流程图**

**Fig. 6 Dijkstra algorithm flow chart based on time window**

### 2.7 解　码

以最大完工时间最小为优化目标时，最优解必然在主动调度集中[29]。故采用主动解码，在不推迟已经被安排的工序开工时间的前提下，根据工件的最早可开工时间查找机器能够完成本道工序的空闲时间段进行插入式解码。传统的作业车间调度中，工件的最早可开工时间为前序工序的完工时间，而 AGV 和机器集成问题需要考虑工件的转移时间，所以工件的最早可开工时间为 AGV 负载结束时间。

### 2.8 时间复杂度分析

时间复杂度是算法的一个重要性能，可以体现算法的运行效率。假设原始鲸鱼优化算法的种群规模为 $N$，鲸鱼个体的维度为 $n$，则初始化阶段的时间复杂度 $T_1=O(n)$。假设进入迭代以后，$|\boldsymbol{A}|$ 的计算时间为 $t_1$，时间复杂度为 $T_2=O(N\cdot t_1)=O(N)$，个体解码的计算时间为 $f(n)$，处理鲸鱼个体维度越界的时间为 $t_2$，这两部分时间复杂度 $T_3=O(Nf(n)+t_2\cdot n))$，假设 3 种更新方式的个体数量分别为 $m_1$，$m_2$，$m_3$，每一维度更新时间分别为 $t_3$，$t_4$，$t_5$，则迭代过程中时间复杂度 $T_4=O(m_1(n\cdot t_3)+m_2(n\cdot t_4)+m_3(n\cdot t_5))$。原始鲸鱼优化算法的时间复杂度为这 3 部分相加，具体见式(33)。

$$T=2O(n)+O(N)+O(N\cdot(n+f(n)))=O(N\cdot(n+f(n)))。\tag{33}$$

在所提算法中，假设机器数为 $m$，计算每个机器负载的时间为 $t_6$，GLR 选择策略的时间复杂度 $T_1=O(0.5N\cdot n+0.5N\cdot n\cdot(m\cdot t_6))=O(N\cdot n)$，假设混沌映射和对立学习的计算时间为 $t_7$ 和 $t_8$，该过程时间复杂度 $T_2=O(N\cdot n\cdot t_7+N\cdot t_8)=O(N\cdot n)$。进入迭代以后，$|\boldsymbol{A}|$ 的计算时间为 $t_9$，时间复杂度 $T_3=O(N\cdot t_9)=O(N)$；设Levy 步长的计算时间为 $t_{10}$，位置更新复杂度 $T_4=O(m_1(n(t_2+t_8))+m_2(n\cdot t_3)+m_3(n(t_4+t_8))+N\cdot t_{10})=O(N\cdot n)$。假设编码转换的时候，LPV 公式计算时间为 $t_{11}$，编码转换时间复杂度为 $T_5=O(N\cdot n\cdot t_{11})=O(N\cdot n)$；解码和处理越界的时间复杂度和原始鲸鱼优化算法一样为 $O(Nf(n)+t_1\cdot n))$。阈值重启操作的时间复杂度 $T_6=O(0.5N\cdot n)=O(N\cdot n)$。假设构造邻域的时间为 $t_{12}$，计算 AGV 运行的时间为 $t_{13}$，局部搜索的时间复杂度为 $T_6=O(n_{\max}\cdot q_{\max}\cdot f(n)\cdot(t_{12}+t_{13}))$。所提算法的时间复杂度见式(34)。

$$T=5O(N\cdot n)+O(N)+O(N\cdot(n+f(n)))+O(f(n))=O(N\cdot(n+f(n)))。\tag{34}$$

综上所述，所提算法和原始鲸鱼优化算法的时间复杂度是一样的，在同样的运行环境下的运行时间在同一数量级。

### 2.9 算法流程

算法流程见图 7，具体流程如下。

步骤 1　读入初始信息，包括工件的加工信息和工厂的地图信息。设置算法参数：种群规模 NIND、最大迭代次数 $i_{\text{itmax}}$。

步骤 2　根据种群规模和编码方式初始化种群。

步骤 3　判断是否达到最大种群最大迭代次数，若达到，输出最优解；否则，计算所有个体最大完工时间并按照基于时间窗的 Dijkstra 算法进行路径规划，记录当前全局最优解，转步骤 4。

步骤 4　判断当前全局最优解保持代数是否达到阈值，如达到，转步骤 5，否则转步骤 6。

步骤 5　随机生成种群规模 50%的个体以替代种群中 50%较劣的个体，转步骤 6。

步骤 6　更新鲸鱼种群。

步骤 7　对当前种群较优个体进行局部搜索。

步骤 8　保持种群规模，选择较优个体进入下一次迭代，转步骤 3。

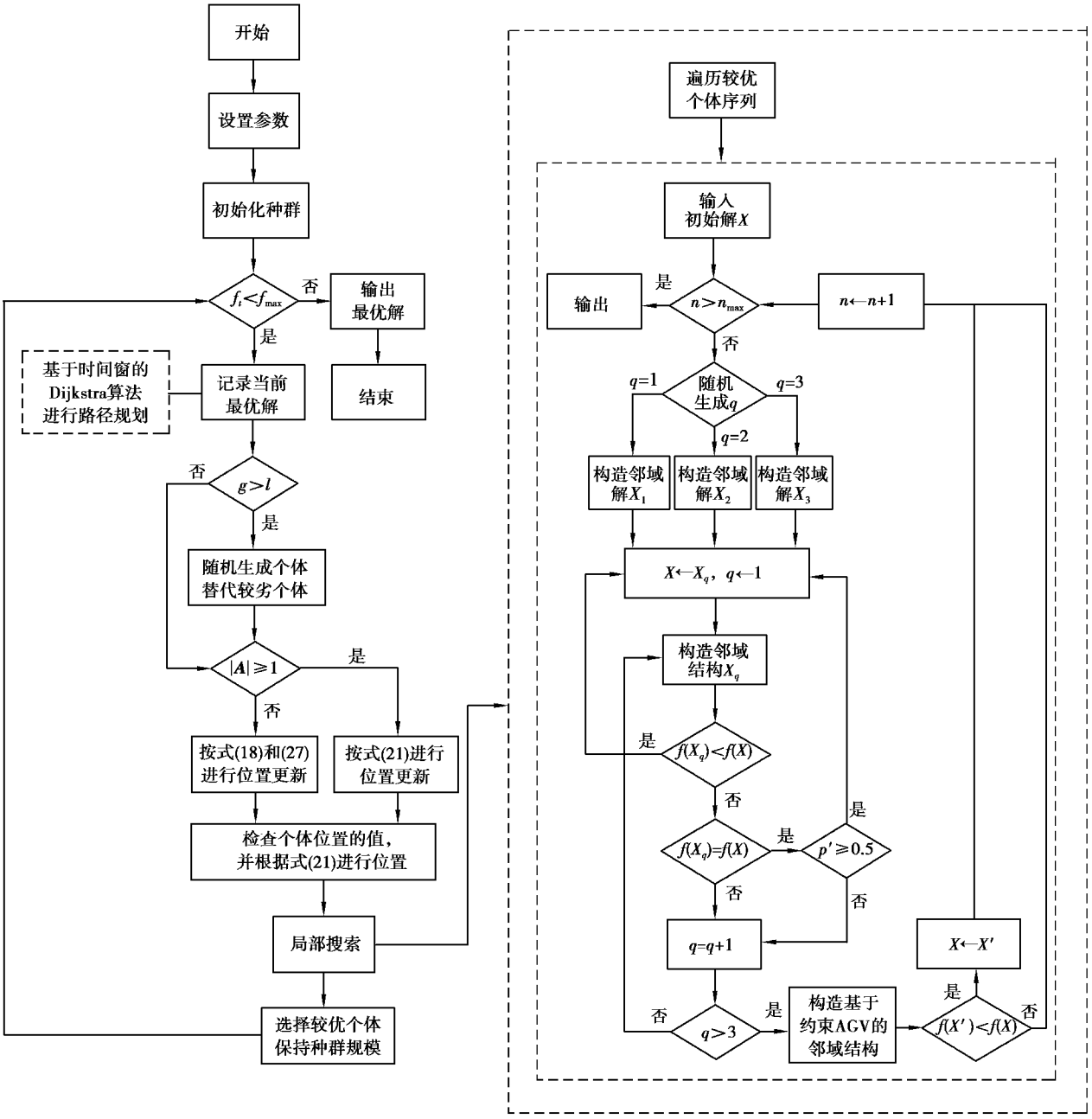


**图 7 算法流程图**

**Fig. 7 Algorithm flowchart**

## 3 实验分析

为了确定和检验离散型鲸鱼优化算法在求解 AGV 和机器集成调度时的性能，笔者在 MATLAB 2019a 的编程环境，Inter(R) Core(TM) i5-8500 CPU，3.0 GHz 主频，8.0 Gb 内存的运行环境下开展以下实验：

1)标准算例对比实验。对 Ulusoy 提出的标准算例库进行实验，并与其他学者已有的研究成果进行对比。

2)柔性算例实验。对文献[11]提出的柔性作业车间 AGV 和机器集成调度算例进行实验。

### 3.1 标准算例实验

表 2 中，LB by Ulusoy 是由 Ulusoy 等[2]提出的下限法得到的理论最优值；LB by Zheng 是由 Zheng

等[6]提出的改进下限估算法得到的理论最优值；STW 是 Bilge 等[4]提出的滑动时间窗算法；AGA 是 Abdelmaguid 等[5]提出的混合启发式遗传算法；MTS 是 Montane 等[30]提出的禁忌搜索算法；PGA 是 Lyu 等[11]提出的改进遗传算法；WOA 为原始鲸鱼优化算法；IWOA 是笔者提出的离散型鲸鱼优化算法。其中 STW，AGA，MTS，RGA 以及 PGA 均来自于文献中的原始数据。WOA 和 IWOA 部分公共参数设置如下：种群规模 80，最大迭代次数 100。WOA 的个体边界设为 1，IWOA 的个体边界设为 3，最优个体保持代数阈值设为 20。WOA 和 IWOA 对每一个算例独立运行 5 次取最优解，和其他文献结果对比如表 2 和表 3 所示。

**表 2　$t/p>0.25$ 的算法结果比较**

**Table 2　Comparison of algorithm results with $t/p>0.25$**

| 算例 | 最优解 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | LB by Ulusoy | LB by Zheng | STW | AGA | MTS | RGA | PGA | WOA | IWOA |
| EX11 | 72 | 72 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |
| EX21 | 86 | 86 | 105 | 102 | 100 | 100 | 100 | 104 | 100 |
| EX31 | 81 | 81 | 105 | 99 | 99 | 99 | 99 | 108 | 99 |
| EX41 | 62 | 76 | 118 | 112 | 112 | 112 | 112 | 122 | 112 |
| EX51 | 60 | 60 | 89 | 87 | 87 | 87 | 87 | 90 | 87 |
| EX61 | 96 | 96 | 120 | 118 | 118 | 118 | 118 | 129 | 118 |
| EX71 | 76 | 76 | 119 | 115 | 111 | 111 | 111 | 124 | 111 |
| EX81 | 146 | 146 | 169 | 161 | 161 | 161 | 161 | 161 | 161 |
| EX91 | 93 | 93 | 120 | 118 | 116 | 116 | 116 | 121 | 116 |
| EX101 | 124 | 124 | 153 | 147 | 147 | 147 | 150 | 153 | **146** |
| EX12 | 66 | 68 | 82 | 82 | 82 | 82 | 82 | 82 | 82 |
| EX22 | 76 | 76 | 80 | 76 | 76 | 76 | 76 | 80 | 76 |
| EX32 | 75 | 75 | 88 | 85 | 85 | 85 | 85 | 89 | 85 |
| EX42 | 60 | 64 | 93 | 88 | 87 | 87 | 87 | 97 | 87 |
| EX52 | 54 | 59 | 69 | 69 | 69 | 69 | 69 | 71 | 69 |
| EX62 | 86 | 86 | 100 | 98 | 98 | 98 | 98 | 105 | 98 |
| EX72 | 74 | 74 | 90 | 79 | 79 | 79 | 79 | 92 | 79 |
| EX82 | 140 | 140 | 151 | 151 | 151 | 151 | 151 | 151 | 151 |
| EX92 | 91 | 91 | 104 | 104 | 102 | 102 | 102 | 107 | 102 |
| EX102 | 114 | 114 | 139 | 136 | 135 | 135 | 135 | 146 | 135 |
| EX13 | 64 | 66 | 84 | 84 | 84 | 84 | 84 | 84 | 84 |
| EX23 | 82 | 82 | 86 | 86 | 86 | 86 | 86 | 90 | 86 |
| EX33 | 77 | 77 | 86 | 86 | 86 | 86 | 86 | 88 | 86 |
| EX43 | 58 | 66 | 95 | 89 | 89 | 89 | 89 | 100 | 89 |

续表 2

| 算例 | 最优解 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | LB by Ulusoy | LB by Zheng | STW | AGA | MTS | RGA | PGA | WOA | IWOA |
| EX53 | 52 | 57 | 76 | 74 | 74 | 74 | 74 | 76 | 74 |
| EX63 | 88 | 88 | 104 | 104 | 103 | 103 | 103 | 109 | 103 |
| EX73 | 76 | 76 | 91 | 86 | 83 | 83 | 83 | 96 | 83 |
| EX83 | 142 | 142 | 153 | 153 | 153 | 153 | 153 | 153 | 153 |
| EX93 | 93 | 93 | 110 | 106 | 105 | 105 | 105 | 107 | 105 |
| EX103 | 116 | 116 | 143 | 141 | 139 | 139 | 139 | 150 | **137** |
| EX14 | 68 | 68 | 108 | 103 | 103 | 103 | 103 | 108 | 103 |
| EX24 | 84 | 84 | 116 | 108 | 108 | 108 | 108 | 116 | 108 |
| EX34 | 81 | 84 | 116 | 111 | 111 | 111 | 111 | 122 | 111 |
| EX44 | 62 | 76 | 126 | 126 | 126 | 126 | 126 | 139 | **121** |
| EX54 | 56 | 56 | 99 | 96 | 96 | 96 | 96 | 99 | 96 |
| EX64 | 90 | 90 | 120 | 120 | 120 | 120 | 120 | 131 | 120 |
| EX74 | 76 | 76 | 136 | 127 | **126** | 126 | 127 | 145 | 127 |
| EX84 | 148 | 148 | 163 | 163 | 163 | 163 | 163 | 166 | 163 |
| EX94 | 91 | 91 | 125 | 122 | 122 | 122 | 122 | 125 | **120** |
| EX104 | 120 | 120 | 171 | 159 | 158 | 158 | 158 | 177 | **157** |

表 3 $t/p<0.25$ 的算法结果比较

Table 3 Comparison of algorithm results with $t/p<0.25$

| 算例 | 最优解 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | LB by Ulusoy | LB by Zheng | STW | AGA | MTS | RGA | PGA | WOA | IWOA |
| EX110 | 126 | 126 | 126 | 126 | 126 | 126 | 126 | 126 | 126 |
| EX210 | 148 | 148 | 148 | 148 | 148 | 148 | 148 | 148 | 148 |
| EX310 | 138 | 138 | 150 | 150 | 150 | 150 | 150 | 150 | 150 |
| EX410 | 112 | 112 | 121 | 119 | 119 | 119 | 119 | 121 | 119 |
| EX510 | 102 | 102 | 102 | 102 | 102 | 102 | 102 | 102 | 102 |
| EX610 | 163 | 163 | 186 | 186 | 186 | 186 | 196 | 186 | 186 |
| EX710 | 137 | 137 | 137 | 137 | 137 | 137 | 137 | 137 | 137 |
| EX810 | 271 | 271 | 292 | 292 | 292 | 292 | 292 | 292 | 292 |
| EX910 | 150 | 150 | 176 | 176 | 176 | 176 | 176 | 176 | 176 |
| EX1010 | 218 | 218 | 238 | 238 | 238 | 238 | 242 | 246 | 238 |

**续表 3**

| 算例 | 最优解 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | LB by Ulusoy | LB by Zheng | STW | AGA | MTS | RGA | PGA | WOA | IWOA |
| EX120 | 123 | 123 | 123 | 123 | 123 | 123 | 123 | 123 | 123 |
| EX220 | 143 | 143 | 143 | 143 | 143 | 143 | 143 | 143 | 143 |
| EX320 | 135 | 135 | 148 | 145 | 145 | 145 | 145 | 145 | 145 |
| EX420 | 111 | 111 | 116 | 114 | 114 | 114 | 116 | 115 | 114 |
| EX520 | 99 | 99 | 100 | 100 | 100 | 100 | 100 | 100 | 100 |
| EX620 | 160 | 60 | 183 | 181 | 181 | 181 | 187 | 181 | 181 |
| EX720 | 136 | 136 | 136 | 136 | 136 | 136 | 136 | 136 | 136 |
| EX820 | 268 | 268 | 287 | 287 | 287 | 287 | 287 | 287 | 287 |
| EX920 | 150 | 150 | 174 | 173 | 173 | 173 | 179 | 173 | 173 |
| EX1020 | 216 | 213 | 236 | 236 | 236 | 236 | 236 | 240 | 236 |
| EX130 | 122 | 122 | 122 | 122 | 122 | 122 | 122 | 122 | 122 |
| EX230 | 146 | 146 | 146 | 146 | 146 | 146 | 146 | 146 | 146 |
| EX330 | 136 | 136 | 149 | 146 | 146 | 146 | 146 | 146 | 146 |
| EX430 | 110 | 110 | 116 | 114 | 114 | 114 | 114 | 114 | 114 |
| EX530 | 98 | 98 | 99 | 99 | 99 | 99 | 99 | 99 | 99 |
| EX630 | 161 | 161 | 184 | 182 | 182 | 182 | 182 | 182 | 182 |
| EX730 | 137 | 137 | 137 | 137 | 137 | 137 | 137 | 137 | 137 |
| EX830 | 269 | 269 | 288 | 288 | 288 | 288 | 288 | 288 | 288 |
| EX930 | 151 | 151 | 176 | 174 | 174 | 174 | 177 | 174 | 174 |
| EX1030 | 217 | 214 | 237 | 237 | 237 | 237 | 237 | 237 | 237 |
| EX140 | 124 | 124 | 124 | 124 | 124 | 124 | 124 | 124 | 124 |
| EX241 | 217 | 217 | 217 | 217 | 217 | 217 | 217 | 217 | 217 |
| EX340 | 138 | 138 | 151 | 151 | 151 | 151 | 151 | 151 | 151 |
| EX341 | 203 | 203 | 222 | 221 | 221 | 221 | 221 | 221 | 221 |
| EX441 | 166 | 166 | 179 | 172 | 172 | 172 | 172 | 173 | **171** |
| EX541 | 148 | 148 | 154 | 148 | 148 | 148 | 148 | 148 | 148 |
| EX640 | 161 | 161 | 185 | 184 | 184 | 184 | 192 | 184 | 184 |
| EX740 | 137 | 137 | 138 | 137 | 137 | 137 | 137 | 137 | 137 |
| EX741 | 203 | 203 | 203 | 203 | 203 | 203 | 203 | 203 | 203 |
| EX840 | 272 | 272 | 293 | 293 | 293 | 293 | **292** | 293 | 293 |
| EX940 | 149 | 149 | 177 | 175 | 175 | 175 | 175 | 179 | 175 |
| EX1040 | 219 | 216 | 240 | 240 | 240 | 240 | 240 | 242 | 240 |

表2和表3中加粗的数据为各算法中最优结果。从表2和表3中的82个标准算例的对比中可以发现原始鲸鱼优化算法在这种解空间较大的情形下,算法很难收敛到全局最优解,性能表现较差。对其进行改进后,所有算例结果均不劣于原始鲸鱼优化算法,并有41个算例结果更优;相较于原始鲸鱼优化算法的结果,IWOA有19个算例结果提升了5%以上,其中有7个算例结果提升了10%以上。在表2的40个标准算例中,IWOA所有算例结果均不劣于其他学者采用的方法;其中算例EX101、EX103、EX44、EX94、EX104结果优于其他算法结果;特别的,算例EX22达到了理论最优解。在表3中的42个标准算例中,IWOA有15个算例达到了理论最优值;除算例EX840外,IWOA剩下所有算例的结果均不劣于其他学者提出的方法,其中算例EX441的结果优于其他算法结果。由此可以看出,在解决AGV和机器集成调度的问题时,IWOA和其他算法相比具有一定的优越性。

### 3.2 柔性算例实验

为了验证IWOA在柔性生产系统中的性能,对文献[11]中的算例进行实验,该算例中包含4个工件、6台机器,AGV数量从1增加到6。文献[11]中PGA运行参数为:种群规模设为80,最大迭代次数设为80,交叉率设为0.9,变异率设为0.1,算法独立运行10次取最好结果。为了更好地对比,IWOA和WOA的种群规模、最大迭代次数、独立运行次数和文献[11]设为相同的值。WOA的个体边界设为1,IWOA的个体边界设为3,最优个体保持代数阈值设为10。表4中$v$表示AGV数量,$C_{\max}$表示最大完工时间,单位为min,$t_{\text{cpu}}$表示运行时间,单位为s,$M_{\text{T}}$为平均最大完工时间,单位为min。

**表4 柔性算例结果**

**Table 4 Flexible study results**

| 算例 | AGV数量 | $C_{\max}$ | | | $t_{\text{cpu}}$ | | | $M_{\text{T}}$ | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | PGA | WOA | IWOA | PGA | WOA | IWOA | PGA | WOA | IWOA |
| 1 | 1 | 129.0 | 141.0 | 131.0 | 264.0 | 241.0 | 272.2 | 130.5 | 143.6 | 135.7 |
| | 2 | 92.5 | 110.0 | 90.0 | 282.2 | 257.6 | 295 | 93.8 | 112.4 | 93.5 |
| | 3 | 91.0 | 101.0 | 89.0 | 287.4 | 262.3 | 297.4 | 92.6 | 103.3 | 92.9 |
| | 4 | 84.0 | 95.0 | 82.0 | 293.6 | 272.5 | 303.7 | 85.6 | 97.1 | 84.6 |
| | 5 | 82.0 | 90.0 | 80.0 | 318.2 | 294.6 | 330.1 | 83.6 | 92.3 | 83.9 |
| | 6 | 82.0 | 87.0 | 80.0 | 321.6 | 302.6 | 335.6 | 83.1 | 88.9 | 83.4 |
| 2 | 1 | 100.0 | 106.0 | 98.0 | 161.9 | 142.6 | 201.3 | 100.5 | 108.6 | 106.3 |
| | 2 | 63.0 | 65.0 | 60.0 | 186.5 | 159.8 | 210.3 | 63.8 | 66.4 | 62.9 |
| | 3 | 54.0 | 58.0 | 49.0 | 169.2 | 158.3 | 215.4 | 55.51 | 60.3 | 56.1 |
| | 4 | 54.0 | 56.0 | 49.0 | 176 | 160.8 | 223.1 | 55.0 | 58.4 | 54.0 |
| 3 | 3 | 105.0 | 108.0 | 101.0 | 559 | 489.7 | 620.6 | 109.7 | 109.6 | 105.5 |
| | 4 | 100.5 | 105.0 | 100.0 | 550.9 | 493.5 | 623.5 | 105.5 | 107.4 | 106.5 |
| | 5 | 97.0 | 102.0 | 96.0 | 589.6 | 512.6 | 624.7 | 100.2 | 103.6 | 102.9 |
| | 6 | 96.0 | 100.0 | 96.0 | 599.5 | 521.4 | 642.3 | 98.9 | 101.9 | 101.9 |
| | 7 | 96.0 | 97.0 | 96.0 | 610.3 | 539.4 | 654.2 | 97.0 | 98.7 | 100.6 |

从表4中可以看出,除了算例1中AGV数量为1时,IWOA的最大完工时间大于PGA,在其他算例中,IWOA的最大完工时间均不劣于其他两种算法;并且在各算例中可以看出,最大完工时间随着AGV数量的增加而减少,但增加到一定数量后不再变化,这是由于在AGV数量较少时,AGV数量为制约最大完工时间的主要因素,较多工件加工完成后需要等待AGV的运输,当AGV增加到一定数量后,加工设备成为主要制约因素,较多工件运输完成后需要等待加工机器。从表4中看出在加工机器数量相同的情况下,在算例1和算例2中,随着AGV数量的增加,IWOA的最大完工时间小于PGA,这是由于IWOA使用了贪婪解码,可

以在较大程度上利用加工机器的空闲时间。在运行时间方面，WOA 机制简单，运行时间最短，IWOA 的运行时间最长。从最大完工时间可以看出，在 IWOA 中平均值和最小值相差较大，这是因为 IWOA 中加入了 Levy 飞行的策略，可能导致出现一些较差解，但同时也可以使算法拥有跳出局部最优的能力。

图 8 为算例 1 中 AGV 数量为 3 时的甘特图，ET 表示 AGV 的空载行程。可以看出，所有工序开工时间在 AGV 负载结束时间、工件前序工序完工时间及机器前序工序完工时间之后，满足约束条件。

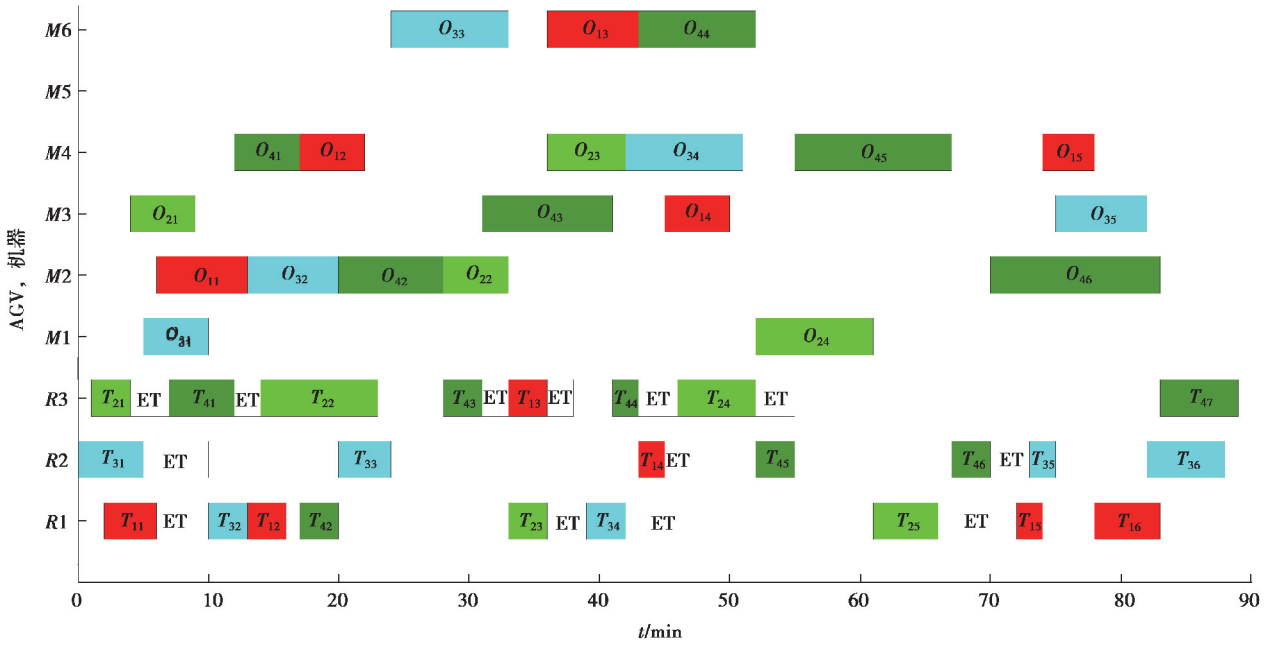


**图 8　最优调度结果甘特图**

**Fig. 8　Optimal scheduling result Gantt chart**

图 9 为算例 1 中 AGV 数量为 3 时各路段的时间窗图，用不同颜色区分 AGV，可以看出，在任意时刻的纵坐标上不会出现同一辆 AGV，说明 AGV 分配方案是可行的；在任意路段，不会出现不同 AGV 时间窗的重叠，说明 AGV 之间不会发生碰撞，IWOA 可以为 AGV 规划无冲突的路径。

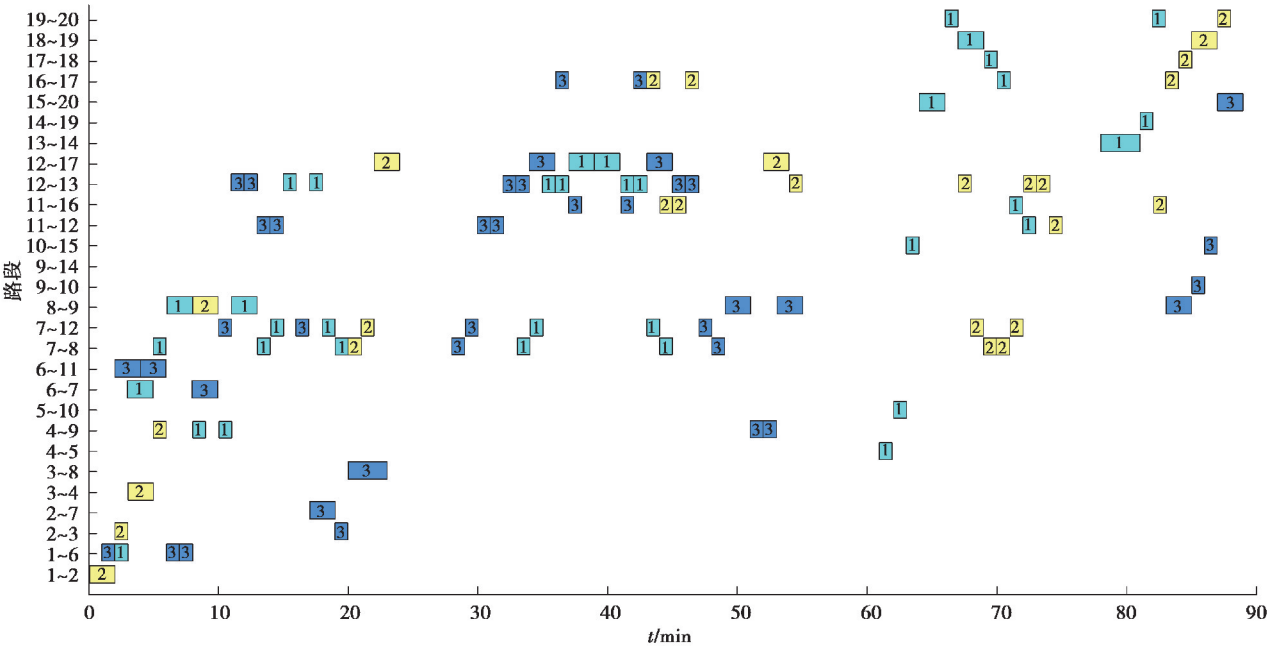


**图 9　各路段时间窗**

**Fig. 9　Time window of each road section**

图10是算例1中WOA和IWOA在AGV数量为3时的收敛曲线。WOA在迭代中后期目标值出现最大完工时间较大幅度减少，体现了WOA在后期具有较强局部开发能力的特点；但在这之后，最大完工时间不再改变，说明WOA陷入了局部最优。相对WOA来说，IWOA的初始种群的质量较高，验证了IWOA的种群初始化方法的有效性；IWOA在第25代就开始收敛到了89 min，说明算法有较好的收敛速度和收敛精度。

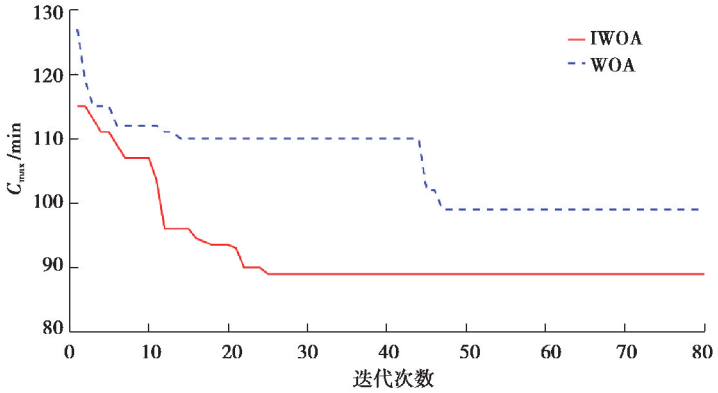


**图10 算法收敛曲线**

**Fig. 10 Algorithm convergence curve**

## 4 结 论

针对柔性作业车间AGV和机器集成调度的问题，以最小化最大完工时间为优化目标建立了相应的数学模型，并提出一种离散型鲸鱼优化算法进行求解。在离散型鲸鱼优化算法中，提出一种结合GLR选择方法和混沌映射及对立学习的初始化策略，可以提高初始种群的多样性；对原始鲸鱼优化算法中的调整规则进行了改进，节省算法运行时间的同时不仅可以避免个体被破坏，还可以在出现较多越界情况时防止个体相似度增大；引入levy飞行策略和阈值重启操作，增强算法的全局搜索能力的同时增强算法跳出局部最优的能力；设计了一种局部搜索策略，增强算法的收敛精度。通过和82个标准算例以及柔性算法的仿真对比实验，证明了所提出算法的可行性和优越性。下一步将研究考虑完工时间、AGV行驶时间、AGV等待时间等AGV和机器集成调度多目标优化问题。

**参考文献：**

[1] 付建林，张恒志，张剑，等. 自动导引车调度优化研究综述[J]. 系统仿真学报，2020，32(9)：1664-1675.
Fu J L，Zhang H Z，Zhang J，et al. Review on AGV scheduling optimization[J]. Journal of System Simulation，2020，32(9)：1664-1675. (in Chinese)

[2] Ulusoy G，Bilge Ü. Simultaneous scheduling of machines and automated guided vehicles[J]. International Journal of Production Research，1993，31(12)：2857-2873.

[3] Ulusoy G，Sivrikaya-Serifoğlu F，Bilge Ü. A genetic algorithm approach to the simultaneous scheduling of machines and automated guided vehicles[J]. Computers & Operations Research，1997，24(4)：335-351.

[4] Bilge Ü，Ulusoy G. A time window approach to simultaneous scheduling of machines and material handling system in an FMS[J]. Operations Research，1995，43(6)：1058-1070.

[5] Abdelmaguid T F，Nassef A O，Kamal B A，et al. A hybrid GA/heuristic approach to the simultaneous scheduling of

machines and automated guided vehicles[J]. International Journal of Production Research, 2004, 42(2): 267-281.

[6] Zheng Y, Xiao Y J, Seo Y. A tabu search algorithm for simultaneous machine/AGV scheduling problem[J]. International Journal of Production Research, 2014, 52(19): 5748-5763.

[7] Deroussi L, Gourgand M, Tchernev N. A simple metaheuristic approach to the simultaneous scheduling of machines and automated guided vehicles[J]. International Journal of Production Research, 2008, 46(8): 2143-2164.

[8] 刘二辉，姚锡凡，陶韬，等. 基于改进花授粉算法的共融 AGV 作业车间调度[J]. 计算机集成制造系统，2019，25(9)：2219-2236.

Liu E H, Yao X F, Tao T, et al. Improved flower pollinaton algorithm for job shop scheduling problems integrated with AGVs[J]. Computer Integrated Manufacturing Systems, 2019, 25(9): 2219-2236. (in Chinese)

[9] Saidi-Mehrabad M, Dehnavi-Arani S, Evazabadian F, et al. An Ant Colony Algorithm (ACA) for solving the new integrated model of job shop scheduling and conflict-free routing of AGVs[J]. Computers & Industrial Engineering, 2015, 86: 2-13.

[10] Shi Y J, Wang X C, Sun X Y, et al. A two-phase strategy with micro genetic algorithm for scheduling Multiple AGVs[C]// 2016 IEEE International Conference on Systems, Man, and Cybernetics. October 9-12, 2016, Budapest, Hungary. IEEE, 2016: 3101-3106.

[11] Lyu X F, Song Y C, He C Z, et al. Approach to integrated scheduling problems considering optimal number of automated guided vehicles and conflict-free routing in flexible manufacturing systems[J]. IEEE Access, 2019, 7: 74909-74924.

[12] Mirjalili S, Lewis A. The whale optimization algorithm[J]. Advances in Engineering Software, 2016, 95: 51-67.

[13] 吴坤，谭劭昌. 基于改进鲸鱼优化算法的无人机航路规划[J]. 航空学报，2020，41(S2)：724286.

Wu K, Tan S C. Path planning of UAVs based on improved whale optimization algorithm[J]. Acta Aeronautica et Astronautica Sinica, 2020, 41(S2): 724286. (in Chinese)

[14] 李伯棠，王智利，周海英，等. 港口拖轮调度模糊规划优化模型及算法[J]. 计算机集成制造系统，2021，27(5)：1518-1530.

Li B T, Wang Z L, Zhou H Y, et al. Fuzzy programming model and algorithm of port tugboat scheduling[J]. Computer Integrated Manufacturing Systems, 2021, 27(5): 1518-1530. (in Chinese)

[15] Zhang H, Tang L, Yang C, et al. Locating electric vehicle charging stations with service capacity using the improved whale optimization algorithm[J]. Advanced Engineering Informatics, 2019, 41: 100901.

[16] Yuan Y, Xu H. An integrated search heuristic for large-scale flexible job shop scheduling problems[J]. Computers & Operations Research, 2013, 40(12): 2864-2877.

[17] Yuan Y, Xu H. Flexible job shop scheduling using hybrid differential evolution algorithms[J]. Computers & Industrial Engineering, 2013, 65(2): 246-260.

[18] Haupt R L, Haupt S E. Practical genetic algorithms[M]. New York: Wiley, 2003.

[19] 高亮，张国辉，王晓娟. 柔性作业车间调度智能算法及其应用[M]. 武汉：华中科技大学出版社，2012：35-38.

Gao L, Zhang G H, Wang X J. Intelligent algorithm for flexible job shop scheduling and its application [M]. Wuhan: Huazhong University of Science and Technology Press, 2012: 35-38. (in Chinese)

[20] Ewees A A, El Aziz M A, Hassanien A E. Chaotic multi-verse optimizer-based feature selection[J]. Neural Computing and Applications, 2019, 31(4): 991-1006.

[21] Shen L Y, Xu L H, Wei R H, et al. Multi-swarm optimization with chaotic mapping for dynamic optimization problems[C]// 2015 8th International Symposium on Computational Intelligence and Design (ISCID). December 12-13, 2015, Hangzhou, China. IEEE, 2015: 132-137.

[22] Alamri H S, Alsariera Y A, Zamli K Z. Opposition-based whale optimization algorithm[J]. Advanced Science Letters, 2018, 24(10): 7461-7464.

[23] Elaziz M A, Mirjalili S. A hyper-heuristic for improving the initial population of whale optimization algorithm[J]. Knowledge-Based Systems, 2019, 172: 42-63.

[24] 王学武，严益鑫，顾幸生. 基于莱维飞行粒子群算法的焊接机器人路径规划[J]. 控制与决策，2017，32(2)：373-377.
Wang X W, Yan Y X, Gu X S. Welding robot path planning based on Levy-PSO[J]. Control and Decision, 2017, 32(2): 373-377. (in Chinese)

[25] 徐坤，陈志军，闫学勤. 基于莱维飞行的改进蚁群算法求解 TSP 问题[J]. 计算机工程与设计，2019，40(1)：245-249.
Xu K, Chen Z J, Yan X Q. Improved ant colony algorithm based on Levy flight to solve TSP problem[J]. Computer Engineering and Design, 2019, 40(1): 245-249. (in Chinese)

[26] Liu M, Yao X F, Li Y X. Hybrid whale optimization algorithm enhanced with Lévy flight and differential evolution for job shop scheduling problems[J]. Applied Soft Computing, 2020, 87: 105954.

[27] Mladenović N, Hansen P. Variable neighborhood search[J]. Computers & Operations Research, 1997, 24(11): 1097-1100.

[28] 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[J]. 计算机集成制造系统，2010，16(12)：2630-2634.
He L N, Lou P H, Qian X M, et al. Conflict-free automated guided vehicles routing based on time window[J]. Computer Integrated Manufacturing Systems, 2010, 16(12): 2630-2634. (in Chinese)

[29] 赵诗奎，方水良. 基于工序编码和邻域搜索策略的遗传算法优化作业车间调度[J]. 机械工程学报，2013，49(16)：160-169.
Zhao S K, Fang S L. Operation-based encoding and neighborhood search genetic algorithm for job shop scheduling optimization[J]. Journal of Mechanical Engineering, 2013, 49(16): 160-169. (in Chinese)

[30] Alfredo Tang Montané F, Galvão R D. A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service[J]. Computers & Operations Research, 2006, 33(3): 595-619.

（编辑 张 苹）