doi:10.11835/j.issn.1000-582X.2021.102

基于带随机网络的多种群粒子群优化算法求解多资源受限柔性作业车间调度问题

崔航浩,张春江,李新宇
(华中科技大学 机械科学与工程学院,武汉 430070)

摘要:多资源受限柔性作业车间调度问题(MRC-FJSP,multi-resource constrained flexible job shop scheduling problem)是一类复杂的组合优化问题。针对以最小化最大完工时间为目标的MRC-FJSP,提出了一种带随机网络的多种群粒子群优化算法(MPSO-RDnet,multi-population particle swarm optimization algorithm with random network)。首先,设计了一种半主动解码和基于启发式规则解码相结合的新型解码方式,对原有解空间进行有效裁剪。其次,提出了基于关键路径的两种邻域结构,提高算法局部搜索能力;引入了基于随机网络的多种群策略,提高算法全局搜索能力;提出了面向算法搜索停滞问题的重新初始化策略,增强算法的鲁棒性。最后,采用 MRC-FJSP 基准算例 SFTSP 进行测试,验证了算法的可行性和有效性。

关键词:多资源受限;柔性作业车间;调度;粒子群优化算法;随机网络
中图分类号:TP301.6 **文献标志码**:A **文章编号**:1000-582X(2022)04-056-11

A multi-population particle swarm optimization algorithm with random network for solving multi-resource constrained flexible job shop scheduling problems

CUI Hanghao, *ZHANG Chunjiang*, *LI Xinyu*
(School of Mechanical Science & Engineering, Huazhong University of Science & Technology, Wuhan 430070, P. R. China)

Abstract: Multi-resource constrained flexible job shop scheduling problem (MRC-FJSP) is a kind of complex combinatorial optimization problem. A multi-population particle swarm optimization algorithm with random network (MPSO-RDnet) was proposed for solving MRC-FJSP with the objective to minimize makespan. First, a new decoding method which combines semi-active decoding and heuristic rule decoding was designed. The original solution space was cut out effectively. Second, two neighborhood structures based on the critical path were designed to improve the local search ability of the population, and a multi-population strategy based on the random network structure graph was added to improve the global search ability of the algorithm. A reinitialization strategy for the algorithm search stagnation was proposed to enhance the robustness of the algorithm. Numerical experiments verified the effectiveness and efficiency of

收稿日期:2020-10-09 **网络出版日期**:2021-01-28
基金项目:国家重点研发计划资助项目(2018AAA0101700)。
Supported by the National Key Research and Development Program of China (2018AAA0101700).
作者简介:崔航浩(1997—),男,硕士研究生,主要从事车间调度及其智能优化方法方向研究,(E-mail)15071434956@163.com。
通信作者:李新宇,男,教授,主要从事车间调度理论及应用方向研究,(E-mail)lixinyu@hust.edu.cn。

proposed algorithm.

Keywords: multi-resource constraints; flexible job shop; schedule; particle swarm optimization algorithm; random network

多资源受限柔性作业车间调度问题(MRC-FJSP, multi-resource constrained flexible job shop scheduling problems)是一种由经典柔性作业车间调度问题(FJSP,flexible job shop scheduling problems)扩展而来的 NP-hard 问题。在许多现实的生产环境中,不仅需要考虑加工机器的分配,还要考虑用于服务加工机器的各类受限附资源的分配[1]。MRC-FJSP 正是为了解决这一类生产调度问题而被提出的,自 2004 年提出以来就备受专家学者和工程技术人员的关注。

与传统 FJSP 相比,MRC-FJSP 增加了机器准备时间和附资源约束,是比传统 FJSP 更为复杂的 NP-hard 问题。近似方法可以在一个合理的时间范围内求得一个较优解,在 MRC-FJSP 的科学研究和工程实践中得到广泛的应用。在过去的十多年里,随着计算机科学与技术的发展,大量元启发式算法(如遗传算法[2-3]、果蝇算法[4]、候鸟优化算法[5]等)被用于求解 MRC-FJSP,取得了不错的效果。Wu 等[2]采用遗传算法(GA, genetic algorithm)求解以最小化最大完工时间为目标的 MRC-FJSP;Hao 等[6]采用合作分布估计算法(CEDA,cooperative estimation of distribution algorithm)求解 MRC-FJSP,通过引入分治策略扩展了合作进化的框架,充分考虑了群体决策过程中相互依赖关系的影响。

随着人工智能技术的突飞猛进,近几年涌现出许多更加高级的群智能算法,这些新算法为 MRC-FJSP 的求解提供了新的技术思路。Wang 等[7]提出采用基于知识库的多智能体进化算法(KMEA,knowledge-based multi-agent evolutionary algorithm)求解 MRC-FJSP,使用混合初始化机制来平衡初始智能体群的多样性和质量,并使用知识库来存储搜索过程中获取的有用信息;Cao 等[8]提出采用一种基于强化学习和代理模型的布谷鸟搜索算法(CSRS,cuckoo search algorithm with reinforcement learning and surrogate modeling)求解 MRC-FJSP。但是目前大多文献只是停留在算法层面的革新,并没有深入挖掘 MRC-FJSP 的特征信息。

粒子群优化算法(PSO,particle swarm optimization)是 Eberhart 等[9]在 1995 年首次提出的,受自然界中鸟群觅食行为的启发,被成功应用到各种组合优化问题上[10-11]。笔者针对 MRC-FJSP 问题特点,对 PSO 算法进行改进,设计基于关键路径的邻域结构,引入基于随机网络结构的多种群策略和重新初始化策略,成功将 PSO 算法应用于求解 MRC-FJSP,取得了较好的结果。

1　MRC-FJSP 问题描述

多资源柔性作业车间调度问题是传统 FJSP 的拓展[12],主要在 FJSP 的基础上增加了机器准备时间和附资源约束。MRC-FJSP 包含机器分配子问题、工序排列子问题和机器配置子问题。MRC-FJSP 可以被描述如下:

1)有 n 个工件($J_1,J_2,\cdots,J_n$)要在 m 台机器($M_1,M_2,\cdots,M_m$)上加工[13];

2)每个工件包含n_i道工序,工序的顺序是预先确定的;

3)每道工序O_{ij}都可以从可选加工机器集Ω_{ij}中任选一台机器进行加工;

4)工序O_{ij}在机器M_k上的加工时间是恒定且预知的;

5)在车间有 r 类附资源($R^1,R^2,\cdots,R^r$),每类附资源的数量是预先确定的;

6)同一台机器在同一时刻只能加工一道工序[14];

7)同一个附资源在同一时刻只能服务一台机器;

8)同一工件的同一道工序在同一时刻只能被一台机器加工;

9)每一道工序一旦开始加工就不允许中断[15];

10)同一工件的不同工序之间有优先级约束,不同工件的不同工序之间没有优先级约束[16];

11)所有工件在零时刻都可以被加工,所有机器和附资源在零时刻都可以被使用;

12)考虑机器准备时间[3]。

以最小化最大完工时间为目标函数，其公式表示形式如下：

$$\min C_{\max}=\min(\max(C_i)),1\leqslant i\leqslant n。\tag{1}$$

式中C_i为工件i的加工完成时间。

2 基于 MPSO-RDnet 求解 MRC-FJSP

2.1 编码与解码

2.1.1 编码

由于不考虑同一类附资源的单件之间的差异性，单独编写机器配置向量会导致搜索空间过大，使算法的寻优效率降低。因此，MRC-FJSP 只编写工序排列向量和机器分配向量，各加工机器的附资源配置在解码阶段动态决策。

对于工序排列向量，采用 FJSP 研究文献中常用的基于工件编号的表示方法。使用相同的工件编号来表示同一个工件的所有工序。从左到右，第k次出现的工件编号，表示对应工件的第k道工序。以图 1 为例，工序排列向量{1,3,2,1,2,4,1}表示的加工序列为{O_{11},O_{31}, O_{21},O_{12},O_{22},O_{41},O_{13}}。

	O_{11}	O_{31}	O_{21}	O_{12}	O_{22}	O_{41}	O_{13}
工序排列向量	1	3	2	1	2	4	1

	O_{11}	O_{12}	O_{13}	O_{21}	O_{22}	O_{31}	O_{41}
机器分配向量	1	3	1	1	2	2	3

图 1 粒子个体编码

Fig. 1 Encoding of individual particles

机器分配向量采用基于机器编号的表示方法，其长度与工序排列向量相同。从左到右，依次表示 1 号工件第 1 道工序至最后一道工序的所选机器、2 号工件第 1 道工序至最后一道工序的所选机器、…、n 号工件第 1 道工序至最后一道工序的所选机器。以图 2 为例，机器分配向量{1,3,1,1,2,2,3}表示的各工序与所选机器的对应关系为{(O_{11},1), (O_{12},3), (O_{13},1), (O_{21},1), (O_{22},2), (O_{31},2), (O_{41},3)}。

2.1.2 解码

解码的过程就是确定每道工序在所选机器上的开始加工时间。对于 MRC-FJSP 而言，如果要开始加工某一道工序，需要同时满足 3 个条件：

1)工件已到达所选机器；

2)所选机器处于空闲状态，且已完成加工前的准备工作；

3)所选机器需要配置的所有附资源均可以立刻投入使用。

基于上述约束条件，MRC-FJSP 中各工序的开始加工时间可以通过公式(2)计算得到：

$$s_{ij}=\max(s_{ij-1}+p_{ij-1k'},A_k+t_{k'k},A_{hk})。\tag{2}$$

式中：s_{ij}表示工序O_{ij}的开始加工时间；$p_{ij-1k'}$表示工序O_{ij-1}在机器k'上的加工时间，$s_{ij-1}+p_{ij-1k'}$表示工件i的最早可用时间；A_k表示机器k的释放时间，$t_{k'k}$表示机器k的准备时间，$A_k+t_{k'k}$表示机器k的最早可用时间；A_{hk}表示机器k所需的附资源h的释放时间。

机器配置子问题采用启发式规则进行处理：在s_{ij}时刻，如果同一类附资源有多个单件可供选择的话，选择释放时间最接近s_{ij}的单件。这一选择不会推迟当前工序的开始加工时刻，但为后续工序的尽早加工留出了余地。

2.2 种群初始化

种群初始化方式影响算法的收敛速度和求解质量。工序排列向量采用随机初始化方式，机器分配向量采用混合初始化方式，通过试错法设定针对机器分配向量的 3 种初始化策略的选择概率。其中选择随机初始化的概率为 60%，选择最早完工时间(ECT，earliest completion time)优先的概率为 20%，选择准备时间与加工时间之和最短(SSPT，shortest setup plus process time)优先的概率为 20%。

2.3　子种群“社会认知”阶段

“社会认知”阶段是子种群内部的信息交互，是子种群内较差粒子向较优粒子靠近的过程。子种群“社会认知”阶段的具体操作步骤如下。

第 1 步：对每个子种群中的粒子按照适应度值排序，找到每个子种群中最优的 2 个粒子和最差的 2 个粒子；

第 2 步：最优粒子与次劣粒子交互。交互时，工序排列向量采用改进优先工序交叉（IPOX，improved precedence operation crossover）算子，机器分配向量采用多点保留交叉（MPX，multi-point crossover）算子。从交叉得到的 2 个新粒子中随机选取一个，替换粒子对中的较差粒子；

第 3 步：次优粒子与最差粒子交互，交互和替换方法与第 2 步相同。

IPOX 算子和 MPX 算子的具体执行过程如图 2 和图 3。图中 $J1$ 和 $J2$ 表示工件集合，$P1$ 和 $P2$ 表示父代 1 和父代 2，$C1$ 和 $C2$ 表示交叉得到的子代 1 和子代 2。

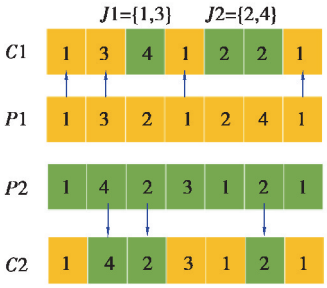


图 2　IPOX 算子操作示例

Fig. 2　Example of IPOX operator operation

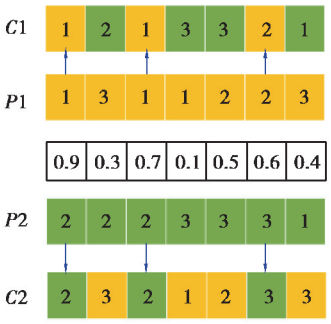


图 3　MPX 算子操作示例

Fig. 3　Example of MPX operator operation

2.4　子种群“自我认知”阶段

“自我认知”阶段是粒子的邻域搜索过程。如果在附资源约束下进行邻域搜索，可行的移动方案会非常少，在寻找可行邻域解的过程中会消耗大量的计算资源，得不偿失。因此 MRC-FJSP 的邻域移动分为两步，先进行带准备时间的 FJSP 的邻域移动，得到一个中间解；再使用 2.1.2 节中所述方法完成各机器的附资源配置，得到一个可行邻域解。

定理 1：当机器准备工作的完成时刻与工序的开始加工时刻相等时，机器时间利用率最大[17]。

证明：当准备工作完成后，当前机器已被某个工件锁定，不能再加工其他工件，只能等候该工件到达。如果延长这一等待时间，最大完工时间可能不会变大，但一定不会变小。当机器准备时间与工序加工时间无缝衔接时，时间利用率最高。

关键路径的变化是改变最大完工时间的关键，也是构造邻域结构的重要组成部分。关键路径的定义是从 0 时刻到终止时刻，顺次满足公式(3)的工序所组成的路径：

$$s_{O_{ij}} - (s_{\mathrm{KPJ}[O_{ij}]} + p_{\mathrm{KPJ}[O_{ij}]k[\mathrm{KPJ}[O_{ij}]]}) \leqslant t_{k'k}。\tag{3}$$

式中：$\mathrm{KPJ}[O_{ij}]$表示工序O_{ij}所在关键路径上的紧前工序；$s_{\mathrm{KPJ}[O_{ij}]} + p_{\mathrm{KPJ}[O_{ij}]k[\mathrm{KPJ}[O_{ij}]]}$表示 $\mathrm{KPJ}[O_{ij}]$的完工时间（开始时间＋加工时间）。

以 MRC-FJSP 的问题特征为导向，设计出 MRC-FJSP 的 2 个邻域结构：换机器的邻域结构（N-CM，neighborhood of changing machine）和同机器的邻域结构（N-SM，neighborhood of the same machine）。

2.4.1　换机器的邻域结构

基于 N-CM 的邻域移动可以归纳为以下步骤：

1）从当前解的某条关键路径中随机选取一道工序，判断该工序是否有 2 个以上（含 2 个）的可选机器。若有，则进行步骤 3；若没有，则重新选取工序，直至找到满足条件的工序O_{ij}。

2）把工序O_{ij}从它当前机器的加工序列中删除。

3)从工序O_{ij}的其他可选机器中随机选择一个,把O_{ij}插入到所选机器的加工序列中,并保证插入后的解依然为可行解。

可行插入

一个可行的插入,是要保证插入后的加工路线不存在闭合回路。

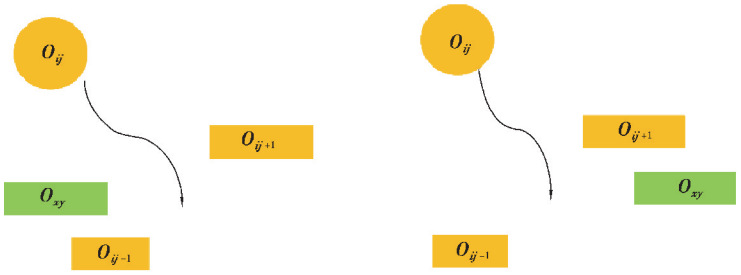
$$L_k:L_k=(O_{xy}\in Q_k \mid s_{xy}<s_{ij+1}); \tag{4}$$

$$R_k:R_k=(O_{xy}\in Q_k \mid s_{xy}+p_{xyk}>s_{ij-1}+p_{ij-1k'})。 \tag{5}$$

式中:Q_k表示在机器k上加工的所有工序的集合;对于集合L_k里的工序而言,不存在从O_{ij}到O_{xy}的加工路径;对于集合R_k里的工序而言,不存在从O_{xy}到O_{ij}的加工路径。

定理2:把O_{ij}插入到L_k-R_k所含工序之后和R_k-L_k所含工序之前,都是可行插入[18]。

用F_{ijk}表示把O_{ij}插入机器k的加工序列中得到的可行解集。定理2的可视化示例见图4。



(a)插入L_k-R_k所含工序之后　　(b)插入R_k-L_k所含工序之前

图4　可行插入的可视化示例图

Fig. 4　Visual examples of feasible insertion

最优插入

在所有可行插入中,可以获得最好解的插入就是最优插入。

示例1:当$L_k\cap R_k\neq\Phi$时,把O_{ij}插入到L_k-R_k所含工序之后和R_k-L_k所含工序之前,会有一个最优插入。

示例2:当$L_k\cap R_k=\Phi$时,把O_{ij}插入到L_k所含工序之后和R_k所含工序之前,都是最优插入。

2.4.2　同机器的邻域结构

N-CM在进行邻域移动之前,必须事先确定L_k和R_k,通过二进制搜索算法获得L_k和R_k的时间复杂度为$O(\log|Q_k|)$。为了提高算法的运行速度,降低时间开销,对于不换机器的邻域移动,采用一种更加简单高效的方法。

基于N-SM的邻域移动也是在关键路径上进行的。在介绍N-SM之前,首先说明关键块的含义。关键块指的是在同一台机器上的关键工序组成的最大序列。

基于N-SM的邻域移动,就是交换同一关键块的块首和块尾2道工序(图5)。但是在交换过程中,需要满足下列条件:

1)首关键块只交换块尾2道工序,尾关键块只交换块首2道工序;

2)如果把工序O_{ij}移动到工序$O_{i'j'}$之后,则必须保证O_{ij+1}在$O_{i'j'}$之后;

3)如果把工序O_{ij}移动到工序$O_{i'j'}$之前,则必须保证O_{ij-1}在$O_{i'j'}$之前;

4)同一个工件的工序之间不得交换。

子种群“自我认知”阶段的具体操作步骤如下:

第1步:找到当前解的一条关键路径;

第2步:执行基于N-CM的邻域移动;

第3步:执行N-SM的邻域移动2次,确保2次选择的关键块分别属于不同的加工机器;

第4步:存档邻域解1,调用解码算法计算其最大完工时间$C_{\max}$;

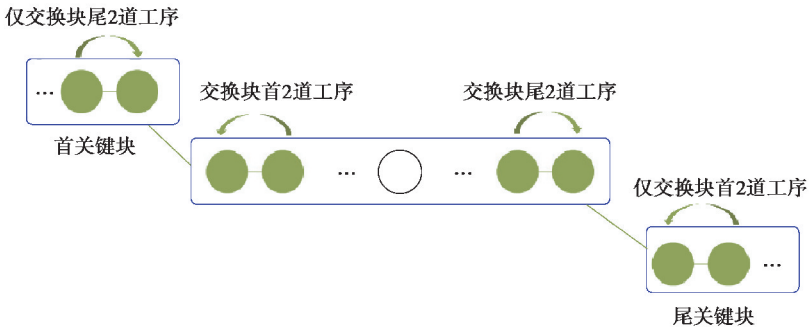


图 5　基于 N-SM 的邻域移动

Fig. 5　Neighborhood mobilizing based on N-SM

第 5 步：重复第 1 步到第 3 步 1 次；

第 6 步：存档邻域解 2，调用解码算法计算其$C_{\max}$；

第 7 步：比较邻域解 1 和邻域解 2 的$C_{\max}$，用较优的邻域解替换当前粒子。

2.5　子种群间信息交互

MRC-FJSP 的工件数量和机器数量都相对较多，导致解空间比较庞大，如果仅仅依靠单个种群进行迭代搜索，算法的效率比较低。多种群并行搜索机制既可以满足单次迭代搜索方向的多元化需求，也在客观上维护了种群的多样性，在复杂优化问题中被广泛应用。目前文献中常用的多种群策略是离散重组和基于网络结构通信的多种群策略。

离散重组大体操作如下：在每次迭代完成后，打破原有的子种群束缚，将所有个体重新归为一个大种群；根据适应度值高低，将种群划分为若干个层次；每个层次随机选取一定数量的个体填充入各子种群中；新组建的子种群再依据相应的规则进行下一轮迭代搜索。虽然离散重组起到了保持种群多样性的作用，但每次重组后，原有子种群个体之间建立的关系被打破，新子种群个体之间需要重新磨合，导致算法后期收敛的精度不高。

目前应用比较广泛的网络结构图有 3 种：无标度网络[19]、分块对角网络[19]和随机网络[20]。随机网络是指度分布满足均匀分布的网络结构，见图 6。各节点之间的连接关系通过连接概率（CP，connection probability）进行调节。当 CP 取值较大时，各节点的度数普遍较高；当 CP 取值较小时，各节点的度数普遍较低。随机网络由于连接概率的可变性，适应能力较强，能够应对许多具有不同性质的复杂问题。

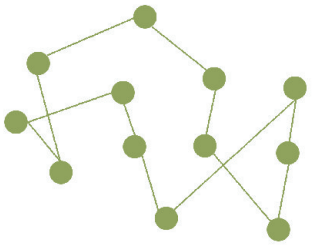


图 6　随机网络结构图

Fig. 6　Random network structure

本研究中采用基于随机网络结构的多种群策略，在数值实验环节会给出带有不同多种群策略的 PSO 算法的对比结果。

为避免算法早期精英片段在子种群之间传播过快，导致早熟收敛，特设计了自适应信息交互控制因子 P_c：

$$P_c=\left(\frac{I_{\text{now}}}{I_{\max}}\right)^R。\tag{6}$$

式中：I_{now}表示当前迭代次数；I_{max}表示最大迭代次数；R 为传播频率，在数值实验环节进行设置。

子种群间信息交互阶段的具体操作步骤如下。

第 1 步：生成一个位于区间(0,1)内的随机数。如果该随机数小于 P_c，则进行第 2 步到第 4 步；否则，本次迭代不进行子种群间信息交互；

第 2 步：根据连接概率 CP 获取用以指导子种群交互的关系矩阵 $\boldsymbol{A}$。$\boldsymbol{A}$ 是一个二维矩阵，里面存储着 0－1 变量。当变量取值为 1 时，表示矩阵横纵坐标编号代表的 2 个子种群存在交互关系；当变量取值为 0 时，表示矩阵横纵坐标编号代表的 2 个子种群不存在交互关系；

第 3 步：按照关系矩阵 $\boldsymbol{A}$，2 个需要交互的子种群里适应度值居中的粒子分别与另一个子种群的最优粒子进行交互，得到 2 个新位置。交互时，工序排列向量采用 IPOX 算子，机器分配向量采用 MPX 算子；

第 4 步：在 2 个新位置中随机选取 1 个替换中间粒子的当前位置。

2.6 子种群重新初始化

有些时候会出现某个子种群连续多次迭代中最优解未得到更新的情况，此时子种群已陷入局部最优。为了避免浪费计算资源、提高算法的鲁棒性，对连续 T 次都未更新最优解的子种群执行重新初始化操作。子种群重新初始化操作是对搜索停滞的子种群中适应度值靠后的 3 个个体初始化。

2.7 算法流程

结合上文对算法各组件的论述，MPSO-RDnet 的算法流程如图 7 所示。

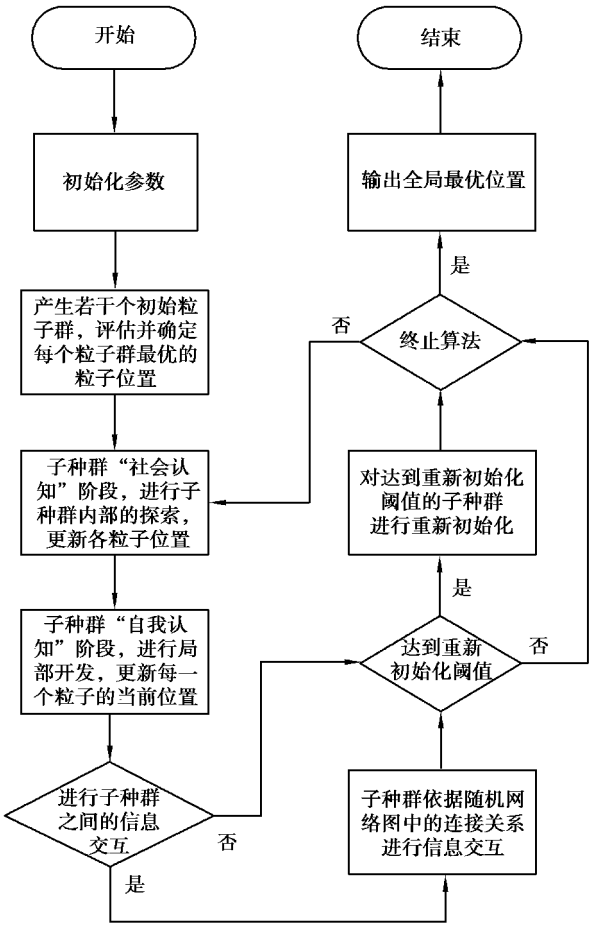


图 7 MPSO-RDnet 算法流程图

Fig. 7 Flow chart of improved MPSO-RDnet algorithm

3　数值实验与结果分析

3.1　标准测试算例

目前 MRC-FJSP 的研究者都采用 Wu 等[2]在 2008 年提出的 SFTSP 算例测试自己算法的性能，该算例可以在决策分析实验室门户网站的子页：https://dalab.ie.nthu.edu.tw/newsen_content.php?id=0 上找到。SFTSP 算例共包含 5 个 large-scale 算例（LS1、LS2、LS3、LS4 和 LS5）和 5 个 wide-range 算例（WR1、WR2、WR3、WR4 和 WR5）。large-scale 算例包含 100 个工件、36 台机器，每台机器需要配置 3 类附资源，加工时间位于区间 [1，15] 内，机器准备时间位于区间 [1，5] 内；wide-range 算例包含 60 个工件、36 台机器，每台机器需要配置 3 类附资源，加工时间位于区间 [1，50] 内，机器准备时间位于区间 [1，15] 内。

MPSO-RDnet 算法在 MATLAB R2016a 软件上编写，运行环境为 Core 4C＋6G 1.9GHz CPU 及 Windows10 系统。

3.2　性能评价指标与参数设置

3.2.1　性能评价指标

为了综合评估算法的收敛性和稳定性，提出了平均相对优胜偏差（ARSD，average relative success deviation）的概念。在介绍 ARSD 之前，首先给出相对优胜偏差（RSD，relative success deviation）的定义。

$$\mathrm{RSD}=\frac{1}{\mathrm{RT}}\sum_{\mathrm{rt}=1}^{\mathrm{RT}}\frac{C_{\max}^{\mathrm{rt}}-C_{\max}^{*}}{C_{\max}^{*}}\times 100。\tag{7}$$

式中：RT 为算法针对该算例的独立运行次数，$C_{\max}^{*}$ 表示现存文献中给出的该算例的当前最优值，$C_{\max}^{\mathrm{rt}}$ 为本次运行求得的该算例的最优值。

$$\mathrm{ARSD}=\frac{1}{\mathrm{BM}}\sum_{\mathrm{bm}=1}^{\mathrm{BM}}\mathrm{RSD}_{\mathrm{bm}}。\tag{8}$$

式中：$\mathrm{RSD}_{\mathrm{bm}}$ 为算法求解第 bm 个算例的相对优胜偏差；BM 表示 SFTSP 的总算例数，取值为 10。

ARSD 越小，说明算法的综合性能越好。

3.2.2　性能评价指标

为了后面算法对比的公平性，参考了文献中其他算法的参数设置，MPSO-RDnet 把解的最大评估次数 10 000 次作为迭代终止条件，种群规模设为 60。算法运行中，对子种群规模、连接概率、传播频率和子种群重新初始化阈值进行多次调整，得到使运行效果最佳的值分别为 5、0.1、0.6、10。

3.3　基于随机网络结构的多种群策略的有效性验证

多种群策略在维护种群多样性方面有显著的优势。为了找到最适合的多种群策略，采用实验设计法。在保证其他参数不变的前提下，PSO 在不同多种群策略下针对每一个算例独立运行 10 次，得到的 ARSD 如表 1 所示。

表 1　带有不同多种群策略的 PSO 求得的 ARSD

Table 1　ARSD obtained by PSO with multi-population strategies

多种群策略类型	ARSD 值
无多种群策略	1.87
离散重组	−0.25
无标度网络	1.12
随机网络	−1.53
分块对角网络	−0.96

从表中可以看出，带随机网络的多种群 PSO 算法在求解 MRC-FJSP 上效果最好。这是因为随机网络结

构中的连接概率可以根据问题需要进行调节，使算法的鲁棒性更高。

3.4 算法对比

为了验证MPSO-RDnet在求解MRC-FJSP上的适用性和有效性，特将MPSO-RDnet算法与目前求解MRC-FJSP最先进的5个算法（nFOA[4]、KMEA[7]、SM²-MBO[5]、CCIWO[21]、WSA[22]）进行比对。与这5个算法一样，MPSO-RDnet也针对每一个算例独立运行10次，得到的对比结果见表2。从表中可以看出MPSO-RDnet得到了所有算例的上界，均值刷新了除WR4以外的原有保持记录，标准差（SD）方面也表现不错。这说明了MPSO-RDnet的鲁棒性非常高。但在运行时间（t_{CPU}）上，MPSO-RDnet位于中等水平，这说明MPSO-RDnet的求解效率还有待提升。

表2 算法对比结果表

Table 2 Algorithm comparison results

算例	nFOA				KMEA				SM²-MBO			
	最优	均值	SD	t_{CPU}/s	最优	均值	SD	t_{CPU}/s	最优	均值	SD	t_{CPU}/s
LS1	104	108.4	3.51	4.13	98	101.4	**1.56**	1.56	90	92.9	1.79	0.75
LS2	113	117.5	3.14	3.71	107	113.3	2.33	2.33	99	100.8	1.16	0.64
LS3	102	107.8	2.89	3.63	99	103.2	1.51	1.51	93	94.9	**0.91**	0.58
LS4	119	123.6	2.10	4.00	117	121.1	1.80	1.80	110	112.4	**1.16**	0.59
LS5	107	115.6	4.00	3.76	107	112.5	1.65	1.65	101	103.3	**1.09**	0.62
WR1	245	250.4	3.57	3.46	229	238.0	3.83	3.83	217	224.5	3.25	0.43
WR2	191	204.8	3.29	3.00	186	198.0	4.34	4.34	173	179.1	2.66	0.42
WR3	220	226.8	3.43	3.10	210	217.8	2.74	2.74	198	201.9	1.70	0.42
WR4	**185**	189.8	4.04	2.73	**185**	185.6	1.74	1.74	**185**	**185.0**	**0.00**	0.42
WR5	222	227.3	3.13	3.20	204	214.8	3.66	3.66	191	196.3	3.10	0.38

算例	CCIWO				WSA				MPSO-RDnet			
	最优	均值	SD	t_{CPU}/s	最优	均值	SD	t_{CPU}/s	最优	均值	SD	t_{CPU}/s
LS1	89	93.2	2.04	3.05	86	88.5	2.74	7.23	**83**	**85.3**	2.24	2.81
LS2	99	101.9	1.68	3.00	97	98.8	2.02	6.27	**94**	**95.6**	**1.08**	2.62
LS3	93	96.5	1.74	2.89	90	92.6	2.74	5.83	**87**	**89.5**	2.03	2.04
LS4	109	112.8	2.05	2.96	108	109.3	1.57	7.11	**103**	**105.9**	2.86	3.17
LS5	99	103.3	1.62	3.06	98	100.9	3.15	6.73	**97**	**98.6**	1.12	2.74
WR1	219	227.1	4.43	1.81	216	221.3	6.27	6.20	**213**	**216.8**	**2.31**	3.11
WR2	174	187.5	6.81	1.62	173	177.8	5.62	5.57	**170**	**172.9**	**2.45**	1.98
WR3	196	203.4	3.48	1.77	196	199.4	3.85	6.23	**195**	**196.3**	**1.27**	2.28
WR4	**185**	**185.0**	**0.00**	1.69	**185**	**185.0**	**0.00**	5.33	**185**	**185.0**	**0.00**	1.25
WR5	192	202.5	5.77	1.72	189	193.8	5.31	6.63	**187**	**190.2**	**2.74**	1.62

注：1.加粗字体表示相应指标的最优值；

2.各算法的运行环境分别为：nFOA（Core i5 2.8 GHz CPU）、KMEA（Core i5 2.4 GHz CPU）、SM²-MBO（Core i5 2.3 GHz CPU）、CCIWO（Core i7 3.6 GHz CPU）、WSA（Core i7 2.4 GHz CPU）、MPSO-RDnet（Core 4C+6G 1.9 GHz CPU）。

4　结　语

研究了以最小化最大完工时间为目标的 MRC-FJSP，提出了一种 MPSO-RDnet 算法，设计了基于关键路径的邻域结构。最后通过数值实验验证了基于随机网络结构的多种群策略的有效性，并与其他求解 MRC-FJSP 的算法进行了对比，有效地验证了 MPSO-RDnet 算法的优越性。

参考文献：

[1] Gargeya V B, Deane R H. Scheduling research in multiple resource constrained job shops: a review and critique[J]. International Journal of Production Research, 1996, 34(8): 2077-2097.

[2] Wu J, Chien C. Modeling semiconductor testing job scheduling and dynamic testing machine configuration[J]. Expert Systems with Applications, 2008, 35(1/2): 485-496.

[3] Wu J Z, Hao X C, Chien C F, et al. A novel bi-vector encoding genetic algorithm for the simultaneous multiple resources scheduling problem[J]. Journal of Intelligent Manufacturing, 2012, 23(6): 2255-2270.

[4] Zheng X L, Wang L, Wang S Y. A novel fruit fly optimization algorithm for the semiconductor final testing scheduling problem[J]. Knowledge-Based Systems, 2014, 57: 95-103.

[5] Gao L, Pan Q K. A shuffled multi-swarm micro-migrating birds optimizer for a multi-resource-constrained flexible job shop scheduling problem[J]. Information Sciences, 2016, 372: 655-676.

[6] Hao X C, Wu J Z, Chien C F, et al. The cooperative estimation of distribution algorithm: a novel approach for semiconductor final test scheduling problems[J]. Journal of Intelligent Manufacturing, 2014, 25(5): 867-879.

[7] Wang S Y, Wang L. A knowledge-based multi-agent evolutionary algorithm for semiconductor final testing scheduling problem[J]. Knowledge-Based Systems, 2015, 84: 1-9.

[8] Cao Z C, Lin C R, Zhou M C, et al. Scheduling semiconductor testing facility by using cuckoo search algorithm with reinforcement learning and surrogate modeling[J]. IEEE Transactions on Automation Science and Engineering, 2019, 16(2): 825-837.

[9] Shi Y, Eberhart R. A modified particle swarm optimizer[C]// 1998 IEEE International Conference on Evolutionary Computation Proceedings, IEEE World Congress on Computational Intelligence (Cat. No.98TH8360), May 4-9, 1998, Anchorage, AK, USA. IEEE, 1998: 69-73.

[10] 郭文忠，陈国龙，陈振. 离散粒子群优化算法研究综述[J]. 福州大学学报(自然科学版)，2011，39(5)：631-638.
Guo W Z, Chen G L, Chen Z. Survey on discrete particle swarm optimization algorithm[J]. Journal of Fuzhou University (Natural Science Edition), 2011, 39(5): 631-638. (in Chinese)

[11] 罗辞勇，卢斌，陈民铀，等. 组合粒子群优化和分布估计的多目标优化算法[J].重庆大学学报，2010，33(4)：31-36.
Luo C Y, Lu B, Chen M Y, et al. Hybrid multiobjective particle swarm optimization and estimation of distribution algorithm[J]. Journal of Chongqing University, 2010, 33(4): 31-36. (in Chinese)

[12] 张智聪，郑力，张涛. 半导体测试调度研究[J].半导体技术，2008(1)：46-50.
Zhang Z C, Zheng L, Zhang T. Investigation on semiconductor test scheduling[J]. Semiconductor Technology, 2008, 33(1): 46-50. (in Chinese)

[13] 王思涵，黎阳，李新宇. 基于鲸鱼群算法的柔性作业车间调度方法[J].重庆大学学报，2020，43(1)：1-11.
Wang S H, Li Y, Li X Y. An improved whale swarm algorithm for flexible job-shop scheduling problem[J]. Journal of Chongqing University, 2020, 43(1): 1-11. (in Chinese)

[14] 马庆吉. 基于改进灰狼算法的柔性作业车间调度方法研究[D]. 武汉：华中科技大学，2019.
Ma Q J. Research on flexible job shop scheduling problem based on an improved grey wolf optimization [D]. Wuhan:

Huazhong University of Science and Technology, 2019. (in Chinese)

[15] 曾强，杨育，王小磊，等. 应用需求时间窗的柔性作业车间调度优化模型[J].重庆大学学报,2011,34(2)：86-94.
Zeng Q, Yang Y, Wang X L et al. Optimal model and algorithm for flexible job-shop scheduling problem based on demand time window[J]. Journal of Chongqing University, 2011, 34(2): 86-94. (in Chinese)

[16] 陈魁,毕利.改进粒子群算法在考虑运输时间下的 FJSP 研究[J/OL].系统仿真学报，2020-07-11[2020-10-02].https://doi.org/10.16182/j.issn1004731x.joss.19-0672.
Chen K, Bi L. Research on FJSP of improved particle swarm optimization algorithm considering transportation time [J/OL]. Journal of System Simulation, 2020-07-11[2020-10-02].https://doi.org/10.16182/j.issn1004731x.joss.19-0672. (in Chinese)

[17] 徐新黎. 生产调度问题的智能优化方法研究及应用[D].杭州:浙江工业大学,2008.
Xu X L. Research on intelligent production scheduling optimization methods and its applications [D]. Hangzhou: Zhejiang University of Technology, 2008. (in Chinese)

[18] Mastrolilli M, Gambardella L M. Effective neighbourhood functions for the flexible job shop problem[J]. Journal of Scheduling, 2000, 3(1): 3-20.

[19] Shi X Q, Long W, Li Y Y, et al. Research on the performance of multi-population genetic algorithms with different complex network structures[J]. Soft Computing, 2020, 24(17): 13441-13459.

[20] Shi X, Long W, Li Y, et al. Multi-population genetic algorithm with ER network for solving flexible job shop scheduling problems[J/OL]. PloS One, 2020,15(5): e0233759 [2020-10-02].https://doi.org/ 10.1371/journal.pone.0233759.

[21] Sang H, Duan P, Li J. An effective invasive weed optimization algorithm for scheduling semiconductor final testing problem[J]. Swarm and Evolutionary Computation,2018,38: 42-53.

[22] 付坤. 基于鲸鱼群算法的柔性作业车间调度方法研究[D]. 武汉:华中科技大学，2019.
Fu K. Research on flexible job shop scheduling methods based on whale swarm algorithm [D]. Wuhan: Huazhong University of Science and Technology, 2019. (in Chinese)

（编辑 罗 敏）