基于模拟退火和粒子群算法的流水车间调度问题求解

郭忠滨 1120220508

摘要

本文主要研究流水车间调度问题（FSP），并应用了模拟退火算法（SA）和粒子群优化算法（PSO）进行求解。流水车间调度问题是一个经典的组合优化问题，旨在确定一组工件在多台机器上的加工顺序，以最小化总完工时间。模拟退火算法通过逐步降低温度，模拟物理退火过程来搜索最优解；而粒子群优化算法则基于群体智能理论，利用粒子群体的协作与竞争来进行全局优化。

本实验分别实现了这两种算法，并对比了它们在不同测试用例中的性能。结果表明，模拟退火算法在解决此类问题上表现出较好的稳定性和优化能力，实验结果展示了两算法在求解流水车间调度问题中的有效性和各自的优势。

关键词：流水车间调度问题；模拟退火算法；粒子群优化算法；参数微调；性能比较

1 引言

1.1 问题背景介绍

流水车间调度问题是一个在制造业、物流和服务业中广泛应用的经典组合优化问题[1]。其主要目标是确定一组工件在多台机器上的加工顺序，使得总完工时间最小化。每个工件需要按照固定的顺序依次经过所有机器，每台机器在同一时间只能加工一个工件。由于调度的解空间非常大，这使得问题求解的复杂性和难度显著增加。

在实际应用中，流水车间调度问题涉及到资源的有效利用和生产效率的提升。优化调度不仅可以减少生产周期，降低成本，还能提高设备的利用率和生产系统的柔性。因此，开发高效的求解算法对于实际生产具有重要意义。

1.2 优化调度问题的数学表示

在置换流水车间调度问题（Permutation Flow Shop Scheduling Problem，PFSP）中，我们考虑有 n 个工件和 m 台机器。每个工件需要按照固定的顺序依次经过所有机器，每台机器在同一时间只能加工一个工件。优化的目标是找到一个工件的排列，使得总完工时间（即完成最后一个工件加工的时间）最小化。

1. 决策变量：

   令 $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ 表示工件的排列顺序，其中 π_i 表示第 i 个位置上的工件。

2. 参数：

   p_{ij}：工件 j 在机器 i 上的加工时间。

3. 目标函数：

   最小化总完工时间 $C_{\max}$，即：

$$\min C_{\max} = \min \max_{1 \le j \le n} \{C_{mj}\}$$

   其中 C_{ij} 表示工件 j 在机器 i 上的完成时间。

4. 约束条件：

   (a) 每个工件必须按照给定的顺序依次经过所有机器：

$$C_{ij} \ge C_{i-1,j} + p_{ij}, \quad \forall i > 1, \forall j$$

   其中 C_{ij} 表示工件 j 在机器 i 上的完成时间，且 $C_{0j} = 0$。

   (b) 每台机器在同一时间只能加工一个工件：

$$C_{ij} \ge C_{i,j-1} + p_{i,j-1}, \quad \forall i, \forall j > 1$$

   其中 $C_{i,j-1}$ 表示工件 $j-1$ 在机器 i 上的完成时间，且 $C_{i,0} = 0$。

上述模型中，目标函数表示最小化总完工时间，约束条件保证了工件的加工顺序和每台机器的加工能力。

1.3 整体解决方案

为了解决上述优化问题，本文采用了模拟退火算法和粒子群优化算法两种方法。模拟退火算法通过逐步降低温度，模拟物理退火过程来搜索最优解，而粒子群优化算法则基于群体智能理论，利用粒子群体的协作与竞争进行全局优化。结果表明，模拟退火算法在解决此类问题上表现出较好的稳定性和优化能力。

2 算法设计

2.1 模拟退火算法

模拟退火算法（Simulated Annealing，SA）是一种随机优化算法，其灵感来自于金属退火过程。通过逐渐降低系统温度，模拟退火算法在搜索解空间时能够跳出局部最优，从而有更高的概率找到全局最优解。算法的关键在于接受劣解的概率随着温度的降低逐渐减小，使得算法在前期具有较强的探索能力，而在后期则更注重解的精细化。

其算法流程伪代码如下所示：

Algorithm 1: 模拟退火算法

Input: 初始调度 π，初始温度 T_0，降温速率 α，停止温度 T_{min}，加工时间 p_{ij}

Output: 最优调度 π^*

初始化 π^* 和 $T \leftarrow T_0$;

while $T > T_{min}$ **do**

  通过交换 π 中的两个位置生成邻域调度 π';

  计算 π' 的最大完工时间 $C_{\max}(\pi')$;

  if $C_{max}(\pi') < C_{max}(\pi)$ **then**

    $\pi \leftarrow \pi'$;

  else

    生成一个随机数 $r \in [0, 1]$;

    if $r < \exp\left(\frac{C_{max}(\pi)-C_{max}(\pi')}{T}\right)$ **then**

      $\pi \leftarrow \pi'$;

  if $C_{max}(\pi) < C_{max}(\pi^*)$ **then**

    $\pi^* \leftarrow \pi$;

  更新 $T \leftarrow \alpha T$;

return π^*

2.2 粒子群优化算法

粒子群优化算法（Particle Swarm Optimization，PSO）是一种基于群体智能的全局优化算法[2]。算法通过模拟粒子群体在搜索空间中的协作与竞争来寻找最优解。每个粒子代表一个候选解，粒子通过更新速度和位置来迭代搜索最优解。

1. 粒子位置和速度更新：

   每个粒子的速度和位置根据自身的最优位置和群体的最优位置进行更新。具体公式如下：

$$v_i^{t+1} = w \cdot v_i^t + c_1 \cdot r_1 \cdot (p_i^t - x_i^t) + c_2 \cdot r_2 \cdot (g^t - x_i^t)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1}$$

   其中v_i^t 是粒子 i 在第 t 代的速度，x_i^t 是粒子 i 在第 t 代的位置，p_i^t 是粒子 i 的历史最优位置，g^t 是群体全局最优位置，w 是惯性权重，c_1 和 c_2 是加速常数，r_1 和 r_2 是随机数。

2. 适应度计算：

   在此问题中，适应度即为工件在当前调度顺序下的总完工时间[3]。

Algorithm 2: 粒子群优化算法

Input: 粒子数量 N，最大迭代次数 M，惯性权重 w，认知社会系数 c_1, c_2，加工时间 p_{ij}

Output: 最优调度 π^*

初始化粒子位置 x_i 和速度 v_i 评估每个粒子的适应度 初始化个体最优位置 p_i 和全局最优位置 g;

for $t \leftarrow 1$ **to** M **do**

  for 每个粒子 i **do**

    更新速度 v_i：

$$v_i^{t+1} = w \cdot v_i^t + c_1 \cdot r_1 \cdot (p_i^t - x_i^t) + c_2 \cdot r_2 \cdot (g^t - x_i^t)$$

    更新位置 x_i：

$$x_i^{t+1} = x_i^t + v_i^{t+1}$$

    评估新位置的适应度;

    if 新适应度优于个体最优 **then**

      更新个体最优位置 p_i;

  更新全局最优位置 g;

return g

3 实验

3.1 实验设置

3.1.1 计算机配置

- 操作系统：Windows 11
- 处理器：Intel Core i7-14700KF
- 内存：48GB DDR5
- 编程语言：Python 3.11.5
- 主要库：random, numpy, matplotlib, colour, time

实验数据集包含11组不同规模的流　车间调度实例，将其命名为数据文件data.txt。每个实例包含工件数量、机器数量及每个工件在具体机器上的加工时间。

实验为了避免随机数对测试结果评估的影响，同一选取了2024作为随机数种子进行试验

3.1.2 模拟退火实验参数

- 初始温度T_0：1000
- 降温速率α：0.999
- 退火临界温度T_{min}：0.0001
- 初始解：以2024为随机种子生成的工作排列
- 迭代次数：每个实例运行20次

3.1.3 粒子群优化算法参数

- 粒子数量N：50
- 最大迭代次数M：1000
- 惯性权重w：线性递减，从0.9递减到0.4
- 认知系数c_1：1.49445
- 认知系数c_2：1.49445
- 迭代次数：每个实例运行10次

3.2 模拟退火算法参数实验

为评估不同参数对模拟退火算法结果的影响，我们选择实例6和实例7进行测试。测试参数包括初始温度T_0、降温速率α和退火临界温度T_{min}对加工完成时间的影响。具体测试参数和结果如下表所示：

表 1: 模拟退火算法参数实例6效果对比

初始温度T_0	降温速率α	退火临界温度T_{min}	加工完成时间
500	0.995	0.0001	1375
1000	0.999	0.0001	1359
1000	**0.995**	**0.0001**	**1336**
1000	0.995	0.01	1368
1500	0.995	0.0001	1359
1500	0.995	0.01	1377
1500	0.999	0.0001	1364

表 2: 模拟退火算法参数实例7效果对比

初始温度T_0	降温速率α	退火临界温度T_{min}	加工完成时间
500	0.995	0.0001	1849
1000	0.999	0.0001	1846
1000	0.995	0.0001	1849
1000	0.995	0.01	1846
1500	0.995	0.0001	1852
1500	0.995	0.01	1858
1500	**0.999**	**0.0001**	**1842**

从表中可以看出，较高的初始温度和较慢的降温速率（即较大的α一般能获得更好的结果。这是因为较高的初始温度使得算法在早期具有较强的探索能力，而较慢的降温速率可以使得算法有更多时间在低温阶段进行局部搜索，从而找到更优解。

3.3 粒子群优化算法参数实验

为了评估不同参数对粒子群优化算法结果的影响，我们选择实例6和实例7进行测试。测试参数包括粒子数量N、最大迭代次数M、认知系数c_1和社会系数c_2对加工完成时间的影响。具体测试参数和结果如下表所示：

表 3: 粒子群优化算法参数实例6效果对比

粒子数量N	最大迭代次数M	认知系数c_1	社会系数c_2	加工完成时间
30	1000	2	2	1393
30	500	1.49445	1.49445	1392
30	1000	1.49445	1.49445	1394
50	500	1.49445	1.49445	1386
50	**1000**	**1.49445**	**1.49445**	**1382**

表 4: 粒子群优化算法参数实例7效果对比

粒子数量N	最大迭代次数M	认知系数c_1	社会系数c_2	加工完成时间
30	1000	2	2	1880
30	500	1.49445	1.49445	1890
30	1000	1.49445	1.49445	1864
50	**500**	**1.49445**	**1.49445**	**1862**
50	1000	1.49445	1.49445	1863

从表中可以看出，增加粒子数量和迭代次数一般可以获得更好的结果。这是因为更多的粒子和更多的迭代次数可以增加算法的搜索范围和搜索时间，从而提高找到最优解的概率。

3.4 算法改进

考虑到PSO算法在计算效率上具有较大缺陷，实验同样测试了利用CUDA加速计算的过程，以实现更高的拟合效率。然而，此尝试在题目所给的样例上效果并不明显，对于运算时间上来说并无显著提升，预计在数量级更大的置换流水车间调度问题上才有效果。

3.5 算法对比

3.5.1 两算法效果对比

表 5: SA与PSO算法效率和结果对比

实例编号	应用算法	最优运行结果	（单轮）运行时间(s)
0	SA	7038	0.106
	PSO	7038	3.382
1	SA	6269	0.077
	PSO	6269	2.662
2	SA	5066	0.062
	PSO	5066	2.532
3	SA	6655	0.091
	PSA	6655	3.168
4	SA	9431	0.106
	PSO	9431	3.632
5	SA	6961	0.091
	PSO	6961	2.946
6	SA	1336	0.327
	PSO	1382	7.152
7	SA	1839	0.488
	PSO	1857	9.715
8	SA	932	0.143
	PSO	940	4.261
9	SA	1758	0.441
	PSO	1787	8.813
10	SA	2776	0.702
	PSO	2880	21.250

可以看出，在当前选取的训练参数下，无论是在结果优化精度还是运行时间效率上，模拟退火算法的结果都明显优于粒子群算法。

3.5.2 最优调度方案

表 6: 各实例最优调度方案展示

实例编号	最优调度时间	最优调度序列
0	7038	[7, 2, 0, 10, 6, 8, 1, 4, 9, 3, 5]
1	6269	[3, 1, 0, 4, 2]
2	5066	[3, 7, 5, 4, 0, 1, 2, 6]
3	6655	[6, 7, 5, 9, 4, 8, 3, 0, 2, 1]
4	9431	[2, 5, 12, 11, 7, 10, 4, 13, 9, 14, 0, 1, 8, 6, 3]
5	6961	[1, 0, 2, 5, 7, 6, 4, 3]
6	1336	[12, 1, 14, 7, 10, 9, 11, 8, 17, 13, 0, 16, 2, 3, 6, 5, 15, 4]
7	1839	[9, 12, 3, 16, 10, 4, 13, 6, 11, 8, 17, 14, 1, 15, 0, 7, 2, 5]
8	932	[14, 3, 4, 7, 9, 12, 1, 0, 11, 6, 2, 13, 8, 5, 10, 15]
9	1758	[11, 10, 14, 13, 1, 9, 8, 0, 7, 4, 15, 6, 12, 5, 3, 2]
10	2776	[7, 8, 34, 39, 38, 19, 33, 29, 32, 36, 27, 20, 1, 3, 15, 11, 2, 23, 26, 28, 21, 9, 0, 31, 6, 30, 35, 4, 14, 37, 13, 24, 5, 12, 16, 10, 17, 22, 18, 25]

4 总结

4.1 工作总结

本文研究了流水车间调度问题，并应用模拟退火算法（SA）和粒子群优化算法（PSO）进行求解。通过实验对比了两种算法在不同实例上的性能表现。实验结果表明，模拟退火算法在解决流水车间调度问题上表现出较好的稳定性和优化能力。本文还尝试了自适应惯性权重和局部搜索策略，进一步提升了PSO算法的性能。

4.2 方法不足与改进

尽管本文的工作取得了一定的成果，但仍有一些不足之处和改进空间，通过进一步的研究和改进，本文的方法有望在实际应用中取得更好的效果：

1. **参数调优**：本文采用的参数设置相对简单，未进行系统的参数调优。进一步研究可以采用自动化参数调优方法，例如网格搜索或贝叶斯优化，以找到更优的参数组合。

2. **算法效率**：本文的PSO算法在没有CUDA加速的情况下运行速度较慢。未来可以尝试将算法移植到GPU上，利用CUDA或其他并行计算技术加速计算过程。

3. **局部搜索策略**：尽管本文引入了局部搜索策略，但其效果可能受限于具体实现方式。可以进一步研究更为高效的局部搜索算法，例如混合遗传算法或蚁群算法，结合PSO以提升整体性能。

4. **多目标优化**：本文仅关注总完工时间这一单一目标，未来可以扩展到多目标优化，考虑其他优化指标，如机器负载平衡、工件等待时间等，从而提升调度方案的综合性能。

5. **实例多样性**：实验仅在有限数量的实例上进行，缺乏对更大规模和复杂实例的测试。未来可以扩展实验数据集，测试算法在不同规模和复杂性实例上的性能，以验证其通用性和鲁棒性。

参考文献

[1] RUIZ R,VZQUEZ-RODR GUEZ J A.The Hybrid Flow Shop Scheduling Problem[J].European Journal of Operational Research,2010,205(1):1-18.

[2] 杨维,李歧强.粒子群优化算法综述[J].中国工程科学,2004,(05):87-94.

[3] 田野. 粒子群优化算法及其应用研究[D].吉林大学,2011.