论文题目 基于迭代贪婪算法的阻塞混合流水车间调度问题研究

分类号 TP181 单位代码 10447
密级 无 学号 2010170107

聊城大学

硕士学位论文

基于迭代贪婪算法的阻塞混合流水车间调度问题研究

Blocking hybrid flow shop scheduling problems based on iterative greedy algorithms

作者姓名 秦浩翔
专业名称 软件工程
指导教师姓名 韩玉艳 副教授
学院 计算机学院
论文提交日期 2023 年 5 月

分 类 号 TP181 单 位 代 码 10447
密 级 无 学 号 2010170107

聊城大学

硕 士 学 位 论 文

基于迭代贪婪算法的阻塞混合流水车间调度问题研究

Blocking hybrid flow shop scheduling problems based on iterative greedy algorithms

作 者 姓 名 秦浩翔
专 业 名 称 软件工程
指导教师姓名 韩玉艳 副教授
学 院 计算机学院
论文提交日期 2023 年 5 月

原 创 性 声 明

本人郑重声明：所提交的学位论文是本人在导师的指导下，独立进行研究取得的成果。除文中已经注明引用的内容外，论文中不含其他人已经发表或撰写过的研究成果，也不包含为获得聊城大学或其他教育机构的学位证书而使用过的材料。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式标明。本人承担本声明的法律责任。

学位论文作者签名：秦浩翔 日期 2023.6.5

导 师 签 名：[illegible] 日期 2023.6.5

学位论文使用授权声明

本学位论文作者完全了解聊城大学有关保留、使用学位论文的规定，即：聊城大学有权保留并向国家有关部门或机构送交学位论文的复印件和磁盘，允许论文被查阅和借阅。本人授权聊城大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或其它手段保存、汇编学位论文。

学位论文作者签名：秦浩翔 日期 2023.6.5

导 师 签 名：[illegible] 日期 2023.6.5

摘 要

二十大报告中指出推进新型工业化，加快建设制造强国，明确将“智能制造”作为发展的主要方向，依托优势企业，紧扣关键工序智能化，建设重点领域智能数字化车间。作为智能制造中极为重要的一环，智能优化调度直接影响了企业的加工效率和经济效益。因此，如何使用合理的生产模式和设计高效的调度方法优化车间工艺流程，提高其生产效率，成为学术界研究的热点。

随着经济全球化的发展，并行机调度的生产模式应运而生。阻塞混合流水车间调度问题（Blocking Hybrid Flow shop Scheduling Problem, BHFSP）是混合流水车间与阻塞约束相结合的一类典型问题，其普遍存在于化学化工、纺织、炼钢、半导体材料等工业生产中，该问题包含着众多的加工工序和复杂的约束条件。然而，在实际生产环境中，除了考虑已有的工艺流程外，还需考虑其他约束条件，如工件成组、分布式工厂和机器设定时间等。本文针对以上约束，分别构建了以最大完工时间（makespan）为优化目标的阻塞混合流水车间调度、阻塞混合成组流水车间调度和分布式阻塞混合流水车间调度的数学模型，并根据问题特性设计了高效的迭代贪婪算法来求解，其具体研究内容如下：

首先，阐述并总结了本文的研究背景、研究内容及论文结构，并对流水车间调度的相关问题及迭代贪婪算法的研究现状进行综合分析与总结。

其次，针对以最小化 makespan 为优化目标的 BHFSP，建立了其数学模型，并设计了双层变异迭代贪婪算法来求解该问题。在所提算法中，使用了双层变异策略来替代原始的贪婪局部搜索策略，降低了原始迭代贪婪算法算法的复杂性，进一步提升了其全局搜索能力。

紧接着，针对阻塞混合成组流水车间调度问题（Blocking Hybrid Flow Shop Group Scheduling Problem, BHFGSP），建立了该问题的数学模型，并设计了一种新的编码和解码方式和提出了组邻域搜索迭代算法以有效地解决工件和工件组的排序问题。

最后，针对分布式阻塞混合流水车间调度问题（Distributed Blocking Hybrid Flow Shop Scheduling Problems，DBHFSP），建立了该问题的数学模型，并提出了一种协同进化迭代贪婪算法来求解 DBHFSP。在所提算法中，采用基于两种交换算子（跨工

厂和工厂内部）的邻域搜索策略减少工件的阻塞时间，并进一步通过局部强化策略来调整每个工厂的工件排序。

关键词：阻塞混合流水车间；成组调度；分布式；迭代贪婪算法；完工时间

ABSTRACT

In the report of the 20th National Congress of China, it was pointed out to promote the new industrialization and accelerate the construction of a powerful manufacturing country, explicitly take "intelligent manufacturing" as the main direction of development. Relying on advantageous enterprises, it closely adheres to the intelligent key processes and build intelligent digital workshops in key fields. As an important part of intelligent manufacturing, intelligent optimal scheduling directly affects the processing efficiency and economic benefits of enterprises. Therefore, how to use reasonable production modes and design efficient scheduling methods to optimize shop process and improve its production efficiency has become a hot topic in academic research.

With the development of economic globalization, the production mode of parallel machine scheduling has emerged as the times require. Blocking Hybrid Flow Shop Scheduling Problem (BHFSP) is a typical problem that uses parallel machine production mode for processing. It commonly exists in industrial production such as chemical engineering, textile, steel making, semiconductor materials, and other industries. This problem includes numerous processing procedures and complex constraints. However, in actual production environments, in addition to considering existing process flows, other constraints need to be considered, such as job grouping, distributed factories, and machine setup times. Based on the above-mentioned constraints, this paper constructs mathematical models of the BHFSP, Blocking Hybrid Flow Shop Group Scheduling (BHFGSP) and Distributed Blocking Hybrid Flow Shop Scheduling (DBHFSP) for minimizing the makespan, and designs efficient Iterative Greedy（IG）algorithms to solve the problem. The specific research is shown as follows:

First, the research background, content, and structure of this paper are described and summarized. Related literature reviews of the flow shop scheduling, and research status of IG algorithms are summarized and analyzed.

Then, a mathematical model of the BHFSP is established for minimizing the

makespan, and a double level mutation IG algorithm is designed to solve the problem. In the proposed algorithm, a double level mutation strategy is used to replace the original greedy local search strategy, reducing the complexity of the original IG algorithm, and further improving the global search ability of the algorithm.

Next, a mathematical model of the BHFGSP is established. A new encoding and decoding method are designed and an iterative algorithm of group neighborhood search is proposed to solve the ordering problem of job and job group effectively.

Finally, a mathematical model of the DBHFSP is established. Then, this paper proposes a collaborative iterative greedy (CIG) algorithm to solve the DBHFSP. In CIG, a neighborhood search strategy based on two swap operators (cross-factory and inner-factory) is designed to reduce the blocking time of jobs, and a local reinforcement strategy is proposed to further adjust the job ordering of each factory.

Key Words：Blocking hybrid flow shop；Group scheduling；Distribution；Iterative greedy algorithm；Makespan

目 录

第一章 绪论

1.1 研究背景及意义

制造业是我国国民经济的主体，是立国之本, 强国之基，打造具有国际竞争力的制造业，是提高综合国力、保障国家安全、建设世界强国的必由之路。作为制造业生产中极为重要的一环，调度始终影响着其生产效率。随着我国经济的不断增长，企业的生产规模也在随之扩大，调度对企业的生产与管理的影响日益突出，针对这一复杂问题建立合理的数学模型并设计高效的智能优化算法是提高企业加工效率、降低其生产成本的重要途径与措施[1]。从目前许多研究成果来看，一个好的调度方案可以更高效的利用现有资源，减少不必要的时间与能源浪费，帮助企业提高社会竞争力[2]。

流水车间调度问题(Flow Shop Scheduling Problem, FSP)作为调度领域的一个重要分支，是众多企业中普遍存在的一类组合优化问题[3]，其包含着复杂的约束条件，为了尽可能提高作业的生产效率，减少其生产成本，在调度的过程中要考虑不同工件的排列组合。随着工件规模的不断扩大，其问题复杂程度开始呈指数上升，因此，FSP是一类复杂的组合优化问题。

作为传统 FSP 的一类扩展问题，混合流水车间调度问题 (Hybrid Flow Shop Scheduling Problem, HFSP) 比 FSP 更为复杂，它不仅涵盖了 FSP 的所有特征，还突破了加工机器的唯一性约束，即，在任意加工阶段都可能存在数量不等的并行机，每个工件都可以在给定机器集中任意选择某台机器进行加工。相比传统的 FSP，HFSP 在现实中的应用更加普遍，常见于化工[4]、炼钢-连铸[5-6]、微电子[7]等生产过程。因此，研究 HFSP 可以减少工件序列的完工时间，进而提高企业的生产效率。

在实际生产过程中，由于加工产品属性、生产成本等限制，使得加工阶段之间没有缓冲区来存放已加工完成的产品。例如，在制药过程中[8]，某些药品对温度的要求比较高，为了一直维持高温的状态，药品一旦被加工完毕就要尽可能被运往下一阶段进行加工，此时，若下游机器都是忙状态，加工好的材料只能被阻塞在当前机器上直至下游机器空闲为止，这就有可能导致药品质量的下降。因此，针对这一类特殊的产品加工设计高效的优化算法来减少前序机器上的阻塞是非常重要的。

除上述问题外，在实际的生产过程中还存在众多限制因素，如机器空闲与设定时间，分布式工厂等。因此，针对不同的问题特性建立符合条件要求的数学模型及设计高效的智能优化算法不仅具有学术科研意义，同时还具备现实的应用价值。

鉴于此，本文针对不同约束分别建立了以最大完工时间为优化目标的 BHFSP 的数学模型。此外，考虑到分布式工厂以及工件成组的约束限制，本文将问题进一步扩展，构建了以最大完工时间为优化目标的 DBHFSP 及 BHFGSP 的数学模型, 并针对问题特性设计了求解上述问题的迭代贪婪（Iterative Greedy）IG 算法。

本文研究意义主要体现在以下几个方面：

（1）根据优化目标的不同建立了 BHFSP 的数学模型。阻塞约束普遍存在于实际生产中，不同的优化目标也导致了问题性质发生变化，已有的数学模型难以反应实际的生产情况。因此，建立满足上述调度需求的数学模型就变得极为重要。

（2）将分布式生产模式和工件成组技术分别考虑到 BHFSP 中，建立了以最大完工时间为优化目标的 DBHFSP 和 BHFGSP 的数学模型，弥补和完善了现有调度理论。

（3）基于以上数学模型，分别设计了针对问题特性的智能优化算法，进一步弥补和完善了现有智能算法在求解 BHFSP 及相关问题上的不足。本文对传统 IG 算法进行了改进，使其成为可以有效求解大规模、多约束和非线性问题的优化算法。

总之，本文为求解 BHFSP 及其扩展问题，设计了一系列高效的智能优化算法，是控制与科学、自动化、计算机、及工业工程与管理等众多交叉学科的研究方向，具有重要学术研究和应用价值。

1.2 研究内容与方法

BHFSP 比传统的 FSP 更复杂，现存的文献研究在求解该问题时很难取得非常好的优化效果，虽然群智能优化算法在迭代过程中能在一定程度上保持算法的多样性，但过多的解会提高算法的计算复杂程度，这就导致了每个解的局部搜索邻域都无法被充分挖掘。 鉴于此，本文的所有策略均是基于 IG 算法框架来设计的，但由于所求解问题的约束不同，导致算法设计的侧重点也不尽相同。针对上一节研究内容，本节主要介绍其研究内容与方法。

1.2.1 求解阻塞混合流水车间调度问题的双层变异迭代贪婪算法

使用 NEH 算法来产生初始解，首先按照工件的加工时间长短降序排列，取出第

一个工件单独放置，随后取出第二个工件并将其插入到第一个工件的相邻两个位置进行测试，选择完工时间最短的序列作为当前序列。接着，依次取出第三个、第四个直至最后一个工件，每次取出新工件时都将其插入到新生成序列的所有位置中，并最终选择完工时间最小的那个位置插入。

在破坏重构阶段，随机从初始解序列中选择几个不同的工件，并将取出的工件依次插入到剩余序列的所有位置中进行测试，最终选择完工时间最小的位置插入。当所有提取的工件被测试完毕后，该阶段结束。

设计双层变异迭代贪婪策略代替传统 IG 算法的贪婪局部搜索策略，降低了算法的计算复杂度，在保证算法局部搜索能力的基础上提高了解的多样性。在双层变异策略中，第一层采用简单的工件交换策略，第二层采用折半交换变异策略。

最后，为保持解的多样性，使用传统的模拟退火接受准则来更新当前解。

1.2.2 求解阻塞混合成组流水车间调度问题的组邻域搜索迭代贪婪算法

以最大完工时间为优化目标，针对工件的分组及阻塞约束设计了一种新型的编码与解码方案，提出了一种新的 IG 算法。在算法初始化阶段，使用了 NEH 初始化策略及破坏重构策略来改变不同的组的排列位置。

随后，设计了基于工件组的邻域搜索策略和基于工件的邻域搜索策略来减少调度序列的阻塞时间。在两种邻域搜索策略中均包括 4 种交换算子，在每一次迭代过程中，都会分别从所提的两种邻域搜索策略中随机选择一个搜索策略来扰动当前的工件序列，当以上过程执行完毕后，使用模拟退火接受准则来提高算法的多样性。

1.2.3 求解分布式阻塞混合流水车间调度问题的协同进化迭代贪婪算法

研究带分布式工厂的 BHFSP，应考虑一些子问题，如工件排序、工厂分配、机器选择。由于这些问题是高度耦合的，基于此，本文继续对 IG 算法进行改进，提出了一种协同进化 IG 算法来求解上述问题，在该算法中，将解的个体数由一个增加到两个。

首先，设计了一种新型的 NEH 初始化策略，该方法可以合理的为工厂分配工件。所提策略与原 NEH 初始化方法的不同之处在于：最初的工件加工顺序是按升序进行排列的，而本文不只采用了按升序排列的工件序列作为初始序列，还将降序排列的工件序列作为了初始序列。该章的实验测试证明，对两个工件序列分别使用升序和降序的 NEH 初始化策略时有着更好的性能。

在得到两个初始化方案后，首先，分别对其工件序列执行随机关键工厂扰动和随机任意工厂扰动策略。其次，对两个解的所有工厂中的工件均执行邻域搜索策略。随后，对每个工厂中的工件序列执行破坏重构策略。紧接着，对两个解执行局部强化策略来进一步改进每个工厂中的调度方案。最后，使用最简单的精英选择策略来更新当前解。

1.3 论文结构

第一章 绪论 综合论述了本文研究背景、研究动机、研究目标和研究内容，采取的研究方法以及本文的研究成果及意义。

第二章 相关工作综述 简要论述与本文课题相关的已公布的研究工作。其内容主要包括：混合流水车间调度问题的研究现状、阻塞混合流水车间调度问题的研究现状、分布式流水车间调度问题的研究现状、成组流水车间调度问题的研究现状和迭代贪婪算法的研究现状，并指出现有研究成果存在的不足，为后续章节的研究做好铺垫。

第三章 求解阻塞混合流水车间调度问题的双层变异迭代贪婪算法。主要包括以下内容：（1）介绍本章问题的研究背景；（2）建立以最大完工时间为优化目标的 BHFSP 的数学模型；（3）提出带有双层变异策略的 IG 算法；（4）将本章所提 IGDLM 算法与目前性能较优的智能优化算法进行对比，验证了所提算法的有效性。

第四章 求解阻塞混合成组流水车间调度问题的组邻域搜索迭代贪婪算法。本章基于第三章研究内容，将工件的成组约束考虑进 BHFSP 中。具体包括以下内容：（1）简要介绍所研究的问题；（2）建立了以最大完工时间为优化目标的 BHFGSP 数学模型；（3）为求解该问题，提出了基于组邻域搜索的 IG 算法；（4）验证模型的正确性，通过仿真测试验证了所提算法的性能。

第五章 求解分布式阻塞混合流水车间调度问题的协同进化迭代贪婪算法。本章基于第三章所研究的内容，考虑分布式环境下的 BHFSP。其主要内容包括：（1）简要介绍所研究的问题；（2）建立了以最大完工时间为优化目标的 DBHFSP 数学模型；（3）为求解该问题，提出基于协同进化机制的 IG 算法；（4）验证模型的有效性，将所提算法于其他目前性能较强的元启发式算法进行对比，验证了所提算法的高效性。

第六章 总结与展望。对论文的研究成果进行总结，并展望在未来需要进一步研究的问题。

本文所有章节在逻辑上相互独立但又环环相扣，根据问题的复杂程度，研究内容由浅入，难度逐步加大，其论文结构如图 1-1 所示。

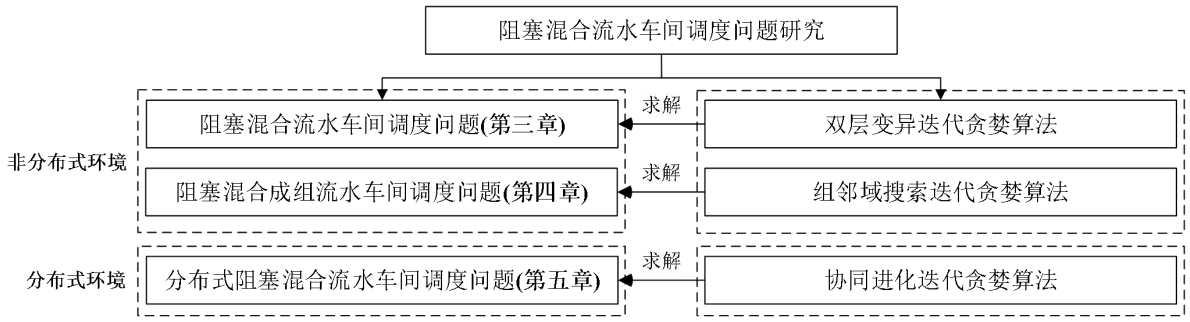


图 1-1 论文结构

第二章 相关工作综述

2.1 混合流水车间调度问题研究现状

HFSP 是传统 PFSP 的一种扩展问题[9-10]，与 PFSP 不同，HFSP 中的一个工艺阶段可能存在多台并行机，且必须保证在整个生产车间中至少有一个阶段的机器数大于 1。在加工开始前，有一系列的代加工工件需要被放置在机器上进行处理，待加工工件必须经过所有的加工工序，直至最后一个工件被处理完毕，加工过程才算完成。在现代制造系统中，HFSP 与并联相同机器的应用非常普遍[11-12]。采用并行机的生产模式可以提高企业的吞吐量，减少瓶颈阶段的影响。

自 1973 年 Salvador[13]提出 HFSP 的概念以来，并行机与流水车间调度相结合的问题开始受到广大学者的关注和研究。为求解更为复杂的 HFSP，研究人员分别提出了不同的求解算法，其主流算法主要包括精确算法、启发式算法和元启发式算法 3 个类别[14]。精确算法可以在小规模且简单的问题上求得精确解，如已提出的分支定界法[15]和动态规划算法[16]，但由于 HFSP 属于非确定性多项式难题，在最优解的搜索过程中涉及到解空间的组合爆炸，因此难以用传统的精确算法求解这类大规模的复杂问题。启发式算法如 NEH（Nawaz, Enscore and Ham）算法[17]和 MME（MinMax-NEH）算法[18] ，在求解复杂的组合优化问题时其优化效果不如元启发式算法，但它们常作为初始化策略被嵌入到元启发式算法中。

作为元启发式算法的一种，遗传算法（Genetic Algorithm，GA）是求解复杂组合有问题的有效算法[19-20]。为求解 HFSP，许多学者提出了高效的优化算法，文献[21]考虑到能源消耗, 提出了一种新型蛙跳算法来优化能源消耗和工件序列的总延迟时间。文献[22]构建了描述空间问题的数学概率模型，并以该模型为基础，通过采样方法产生新个体，进而更新所提模型的参数。文献[23]考虑到 HFSP 的 NP-hard 特性，提出了一种变邻域改进遗传算法来求解该问题。文献[24]为求解以总流经时间为优化目标的 lot-streaming HFSP，提出了两种竞争机制，其分别用于提高在群体中产生更好解的概率与增强两条线之间的相互关联。文献[25]以最小化 makespan 为优化目标，提出了一个新的控制参数来平衡算法的全局与局部搜索，并提出了一种前向与后向译码相结合

的混合表示算法来求解该问题。文献[26]通过引入构造性启发式算法来优化粒子群优化（Particle Swarm Optimization，PSO）算法的初始化方案，随后又嵌入变邻域搜索算法，进而减少计算量。文献[27]提出了两阶段蚁群算法来求解批量和日历约束的 HFSP 问题。随后，提出了一种混合迭代局部搜索算法来求解经济批量和序列的 HFSP[28]。Kurdi[29]为 HFSP 中的多处理器任务调度设计了一个带有新的非守护进程操作程序的 AC 系统。文献[30]设计了一种混合算法，用于解决带有并行机器的专用两级 HFSP。文献[31]提出了 IG 算法的四种变体，及针对 HFSP 可变块 heuristic 算法，优化工件序列的总流动时间。为进一步优化 makespan，文献[32]提出了新的 GA 算法来求解具有机器合格性和不相关机器的 HFSP 问题。

此外，为了优化生产车间的能源消耗，文献[33]首先考虑了能源消耗和资源约束，并提出了离散帝国主义竞争算法来求解具有最大完工时间目标的 HFSP。同样，采用非支配排序遗传算法 II（Non Dominated Sorting Genetic Algorithm II，NSGA-II）来求解具有能耗目标的 HFSP[34]，该算法可以有效的为工件选择合适的机器，同时进行合理的操作排序，而不会受主观因素的干扰。之后，文献[35]提出了一种离散的融入帝国主义竞争和变邻域搜索算法的两阶段元启发式算法来求解具有能耗阈值的 HFSP。文献[36]考虑到 HFSP 中存在的机器设置时间和机器运输操作，提出了一种基于分解的三阶段多目标算法，以最小化能耗和最大完工时间。除此之外，HFSP 不仅仅只作为学术研究方向，其也在普遍存在于现实生活中，如炼钢连铸与精炼过程[37]，薄膜晶体管液晶显示器的生产过程[38]，半导体晶圆制造设备的加工[39]等。

由以上研究可知，已有许多学者针对 HFSP 提出了高效的求解算法，如遗传算法，粒子群优化算法，迭代贪婪算法等，并在此基础上做了扩展研究，如考虑到工件的批量、日历约束、机器设定时间和运输时间、资源约束等，亦或是针对不同的优化目标，如总流经时间，最大完工时间，能源消耗。但上述列举的求解算法并没有考虑到存在于 HFSP 中的阻塞状况，且本论文的所有研究均是以 BHFSP 为基础展开的，因此，作者又查阅了 BHFSP 的相关文献，并将其内容单独放在 2.2 章中。

2.2 阻塞混合流水车间问题研究现状

在上述针对 HFSP 的研究中，在某一阶段加工完成的工件需要将其放入缓冲区中等待下一阶段的加工，但该操作并未考虑实际的产品属性、企业生产成本以及机器之

间无缓冲区的约束限制，为此，本章考虑到不同机器之间存在无缓冲区的限制, 针对其约束做了相关的调查。经作者查询相关资料发现，由于无缓冲区限制，导致加工完毕的产品被阻塞在当前机器上,延长了加工过程中等待时间及整个序列的完工时间，进而降低了企业的生产效率。由此可知, 针对不同规模的调度序列，工件排序上的差异会导致不同程度的阻塞，找到一个最优的工件调度序列来减少因无缓冲区造成的阻塞问题就变得尤为迫切，设计有效的求解算法可以降低企业的生产成本，从而提高其生产效率。

相比 HFSP，BHFSP 的研究要少得多。为求解该问题，一些学者设计了高效的智能优化算法。文献[40]提出了一种新型 GA 来研究实际金属加工企业中的两阶段 BHFSP，该研究对调度序列的最大完工时间目标进行了优化。Nakkaew 等人[41]分别提出了 GA 和离散人工蜂群（Discrete Artificial Bee Colony，DABC）算法，以求解 BHFSP 中的序列依赖性设置时间问题，并使总生产时间最小化。为求解机器人中的 BHFSP，Elmi 等人[42]考虑了机器合格性、多个机器人和多个零件类型等因素，设计了该问题的混合整数线性规划（Mixed Integer Liner Programming，MILP）模型，并提出了基于块属性的 SA 算法来求解该问题。Missaoui 等人[43]提出了一种以 IG 方法为中心的元启发式方法，通过优化延迟和提前的总和来研究 BHFSP。Moccellin 等人[44]研究了具有机器阻塞、序列相关和序列无关设置时间的 HFSP，并提出了一种基于传统 SPT 和 LPT 规则的算法，以最小化最大完工时间。文献[45]开发了 BHFSP 的四个 MILP 模型，并提出了一种改进的回溯搜索优化算法，以缩短序列的完成时间，将变异思想集成到 IG 算法中，随后针对 BHFSP 建立了 MILP 模型，该模型考虑了机器有无设置时间，在论文最后验证了模型的正确性[46]。Trabelsi 等人[47]提出了一种新的 BHFSP 的数学模型，并给出了一个下限，随后优化了该问题的最大完工时间。考虑到到期日窗口的目标，Aqil 等人[48]提出了水波优化和候鸟优化算法，以研究具有序列依赖性设置时间的 BHFSP。

上述文献考虑了 HFSP 的阻塞约束，分别提出了优化该问题的有效算法。然而，很少有策略根据阻塞特性来调整序列排序，已有算法大多在工件排序的层次上来优化目标，而针对问题特性，即，工作阻塞因素的考虑却少之又少。本文弥补了以往策略设计中的不足，文中所提出的很多调度策略都是针对工件阻塞这一因素来进行设计的。它们可以有效地减少因工件阻塞导致的完工时间延长问题。

2.3 分布式流水车间调度问题研究现状

2.3.1 分布式流水车间调度问题

随着经济全球化的发展以及市场竞争的加剧，不同企业之间的联系越来越紧密。传统的集中制造模式已难以灵活满足当前的市场需求[49-51]，为了应对快速变化的全球市场，分布式或多品种制造模式被用于提高企业的资源利用率和生产效率[52]。自从分布式流水车间调度问题（Distributed Permutation Flow Shop Scheduling Problem，DPFSP）[53]被提出以来，许多学者开始对 DPFSP 进行了广泛的研究[54]。作为传统流水车间调度问题的一种扩展，DPFSP 比 PFSP 更为复杂[55]。在 DPFSP 中，公司设立了多个工厂来并行处理同一批产品，这样可以更有效地分配资源，分散公司的生产压力，减少产品生产周期和风险[56-58]。

为了满足现代市场的需求，许多企业都建立了多个工厂来提高产品的生产效率，这已经成为一个新的研究热点。为求解该类问题，不少研究人员同样针对此类问题设计了有效的求解算法。如：文献[59]提出了一种新的多目标脑风暴优化算法，用于求解具有总延误约束的 DPFSP 问题。考虑到工作的组装，文献[60]使用回溯搜索超启发式来求解 DPFSP。在文献[61]中，提出了一种多目标鲸鱼群算法来优化 DPFSP 的节能问题。考虑到序列相关的设置时间，文献[62]提出了一种有效的 IG 算法来求解该问题。在文献[63-64]中，为优化 DPFSP 的 total flow time 和 makespan，提出了一种迭代改进算法和一种化学反应优化算法。为求解 DPFSP 问题，文献[65]设计了新的禁忌搜索算法。Rifai 等人[66] 为求解三个目标的分布式可重入流水车间调度（DRPFS）问题，提出了一种多目标自适应大邻域搜索算法，即总成本、最大完工时间和平均延迟。文献[58, 67]提出了一系列基于构造启发式和元启发式框架的算法来求解 DPFSP 和分布式装配流水车间调度问题（Distributed Assembly Permutation Flow Shop Scheduling Problem，DAPFSP）。Ruiz 等人[68]使用 IG 算法解决了这个问题，并证明了其有效性。 Ochi 等人[69]设计了有界搜索迭代贪婪算法 BSIG 来求解 DAPFSP 问题。在同样的问题上，文献[70]提出了一种基于群体思维的 IG 算法来优化总流量时间。最近，文献[71]为优化带阻塞约束的 DPFSP 的 makespan，提出了一种高效的 IG 算法。类似地，Chen 等人[72]也使用 IG 算法来求解具有阻塞约束的 DPFSP。

以上均是针对 DPFSP 开展的研究，其中也存在不少新的约束条件，如总延误，序列相关，可重入，装配，阻塞等。但这些研究并没有考虑 BHFSP 在分布式环境下

的情况，在下一节，将会阐述目前带有分布式环境下的相关研究。

2.3.2 分布式混合流水车间调度问题

为了进一步提高产品的生产效率，结合 DPFSP 和 HFS 各自的优点，企业开始在各个工厂中使用一种更高效、更具有实际应用价值的加工模式，即将两种生产模式相结合。于是，就出现了 DHFSP。显然，这种带有并行生产线的分布式生产方式相较于传统的流水车间调度问题更为有效。

到目前为止，关于分布式工厂结合并行机调度的研究仍然相当有限，其已发表的有：文献[73]提出了一种自调整迭代贪婪（SIG）算法来优化作业序列的最大完工时间。文献[74]为求解具有机器设定时间的两阶段 DHFSP，提出了一种带 Memeplex 分组的洗牌蛙跳算法。文献[52]提出了一种双种群协同模因算法来求解异构 DHFSP。文献[75]提出了具有 shar 介质规则的 DNEH 和求解 DHFSP 的多邻域迭代贪婪算法。文献[55]提出了一种混合离散人工蜂群算法，用于求解作业恶化的并行分批 DPFSP。文献[76]提出了一种协同进化算法，用于求解加工时间和交货时间均为模糊的多目标模糊 DHFSP。考虑到多处理器任务，文献[77]设计了一种动态洗牌蛙跳算法来求解 DHFSP。文献[78]考虑到分布式异构工厂，提出了一种改进的人工蜂群算法，以解决序列依赖设置时间的 DHFSP 问题。

以上研究综合考虑了并行机床和多工厂（同构或异构）的情况。所发表的论文都是非常新颖的，此外，其所设计的算法性能也十分突出，但是，它们均没有考虑到调度序列的阻塞状况。如前文所述，阻塞约束又是普遍存在于现实生活中的（如药品制造），而目前此方面的研究又相对匮乏，因此，研究带阻塞约束的 DHFSP 就变得十分有价值。

本文提出的基于协同进化的 IG 算法成功求解了上述问题，在阻塞约束的限制下，优化了工件序列在 DHFSP 中的最大完工时间。此外，在现实生活中，许多制造厂开始使用成组的加工模式，在下一节，将阐述成组约束在流水车间调度问题中的研究现状。

2.4 成组流水车间调度问题研究现状

为了满足当今市场的需求并提高企业的生产效率，一些车间的机器被划分为有组织的制造单元。在成组加工模式中，由有组织的制造单元加工的一组产品被视为工作

组。在一些制造厂，如软垫家具[79]、印刷电路板[80]和汽车喷漆车间[81]都采用了成组调度模式，以提高生产过程的灵活性。因为每组中的任务均是具有相似属性的工件，所以，该加工模式可以一定程度上提高产品的加工效率，减少机器的安装、调整、换洗、设定等时间[82]。此外，成组调度的应用已成功扩展到分布式的加工环境，进一步提高了企业的生产效率和灵活性[83]。

近年来，流水成组调度问题（Flow Group Scheduling Problem，FGSP）得到了越来越多的关注，其应用也在不断增加[84-85]，但研究成果相比 HFSP 和 DPFSP 要少得多。从文献[86]的描述中可以看出，成组调度可以有效减少机器安装时间、产品完成时间、原材料库存和生产提前期，同时，FGSP 已被证明是一个 NP-hard 问题[87]。针对 FGSP，Neufeld 等人[85]改进了构造性启发式算法以最小化完工时间。Costa 等人[88]通过使用随机抽样搜索方法和混合元启发式遗传算法解决了 FGSP 问题。随后，Costa 等人[89]提出了一种原始的元启发式方法来解决带有阻塞约束的 FGSP。在文献[90]中，为了最小化总完工时间，提出了一种混合遗传和模拟退火算法来求解 FGSP。文献[91]考虑了机器之间具有作业传输时间的多阶段 FGSP，同时考虑了组之间的顺序相关设置时间，随后提出了一种快速且易于实现的混合算法，以在最小化完工时间的情况下求解 FGSP。Keshavarz 等人[92]提出了一种基于粒子群优化的混合元启发式算法来求解 FGSP 问题，目标是总延迟和总加权时间。

上述文献解决了具有不同目标的 FGSP，并取得了良好的效果。然而，目前还没有求解带阻塞约束的 HFGSP 的研究，更没有针对该问题特性设计的策略。本文提出的基于邻域搜索的 IG 算法首次解决了上述问题。

2.5 迭代贪婪算法研究现状

IG 算法是一种简单高效的智能优化算法。由于其结构简单、可嵌入性强等优势受到了制造业领域研究人员的广泛关注。IG 算法具有的一些明显优势是：（1）该算法结构简单，参数少，可以将构造性启发式和元启发式算法集成到其框架中。（2）与现有的群体智能算法不同，IG 算法在每次迭代中只生成一个解，因此它可以专注于对解的深入探索。（3） IG 算法通过对工件序列的局部扰动，每次操作以当前最优的方式安排工件加工，有效地减少了由于工件阻塞而造成的资源浪费。如图 2-1 所示为传统 IG 算法的流程图，与传统的群智能算法相比，改进的 IG 算法能够更深入地探索

单个解，具有较强的局部搜索性能。

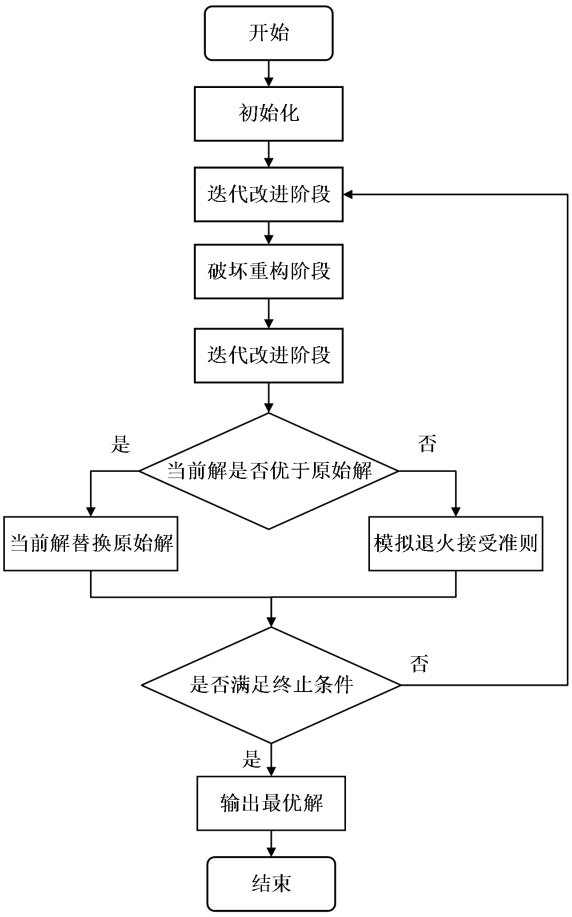


图 2-1 传统 IG 算法流程图

在 Ruben 和 Thomas[93]首次使用该算法求解 FSP 之后，一系列改进的 IG 算法被用于求解 FSP。为优化无等待 FSP 的最大完工时间，Ding 等人[94]提出了一种基于禁忌的重建策略，以增强 IG 算法的搜索能力。紧接着，文献[62]修改了 IG 算法，使用了六种不同的算子来求解分布式排列 FSP。Fernandez Viagas 等人[95]采用了基于 IG 的算法和波束搜索初始化，以最小化 FSP 的总体延迟。Ochi 和 Driss[69]提出了有界搜索 IG 算法来求解 DAPFSP。类似地，文献[70]提出了一种基于群体思维的改进 IG 算法，作为求解 DAPFSP 的一种手段。

针对 HFSP，文献[96]提出了一种 IG 算法来求解具有多处理器任务的 HFSP。Rodriguez 等人使用 IG 算法来求解大规模无关并行机调度问题[97]。文献[75]使用 IG 算法以最大完工时间为优化目标来求解 DHFSP。文献[98]提出了一种 IG 搜索元启发式算法，该算法可以最大限度地缩短在具有顺序相关设置时间的制造工厂中遇到的 HFSP 的制造时间。Ztop 等人[31]设计了一系列策略和四种 IG 变体，即 IGRS、IGT、IGTALL

和 VBIH 算法，来求解 HFSP 问题。这些变体在文献中表现出良好的性能。从实验结果来看，IG 算法在求解 HFSP 的各种研究中表现出良好的性能。

因此，本文同样使用 IG 算法来求解 BHFSP 及其相关问题。与以往研究中的 IG 算法不同，本文所提出的 IG 算法是针对阻塞、分布式工厂和成组约束等问题特性设计的搜索策略。通过实验证明，相比其他 IG 算法以及元启发式算法，所提策略可以取得更好的实验结果。

2.6 本章小结

本章综合论述了目前流水车间调度领域的研究成果，此外，还阐述了用于求解相关问题的迭代贪婪算法的研究现状。基于以上调查和分析，本章得出以下结论：

以最大完工时间为优化目标的阻塞混合流水车间调度问题数学模型的相关文献较少，且很多研究均没有阐述问题详细的编码、解码过程，或是提供一些算例来验证其有效性。将分布式的调度环境以及成组约束加入到 BHFSP 中也是十分具有学术研究价值的。因此，针对上述问题特性，本文构建了相对应的数学模型，并提出了一系列改进的 IG 算法来优化该问题。通过综合分析，虽然目前存在多种高效的智能优化算法来求解这些问题。但它们存在的策略还有待改进，其性能有待提高。需要针对问题的特性来设计相应的优化策略。

以下 4 种策略设计具有非常大的潜力：（1）IG 算法是局部搜索能力较强的智能优化算法，因此可以结合提高多样性的策略，从而更加有效的减少单次迭代的计算量。（2）可以针对工件的阻塞约束来设计相应的求解算法，如着重调整阻塞程度较深的工件位置进而减少整个调度序列的能耗或是完工时间。（3）在分布式的调度环境下，可以着重调整关键工厂中的阻塞工件来减少整体的完工时间，应顾忌到每个工厂的阻塞状况。（4）阻塞混合成组调度问题的研究目前还处于几乎空白的状态，不仅缺少相应的数学建模，还缺少相应的求解算法，以该领域为基准进行展开研究可以取得不少具有原创性的研究成果。针对成组约束，可以将算法主体分为两大部分，一类是针对组的排列位置来设计相应的搜索策略，另一类是根据每组中的工件阻塞状况来设计相应的策略。

接来下的章节将重点研究上述所提到的问题，以问题复杂程度由浅入深，层层递进展开。从不同的问题特性和优化目标角度出发设计了不同的迭代贪婪算法。

第三章 求解阻塞混合流水车间调度问题的双层变异迭代贪婪算法

3.1 研究背景

针对 BHFSP, 调整调度序列中部分工件的排列顺序，有可能缩短工件被阻塞在加工机器上的时间，进而降低完工时间。如图 3-1 所示，两个不同的加工序列对比图，其工件调度顺序分别为 1→2→3→4→5 和 1→2→4→3→5，最后的完工时间分别为 24 和 22，由于工件 3 和 4 的加工顺序不同，最终所造成的完工时间也不同。故在较优解的基础上提高算法的局部搜索能力有助于找到更好的解，进而减少因阻塞约束所延长的最大完工时间，提高产品的生产效率。传统的 IG 算法使用 NEH 启发式算法获得较好解，在此基础上，通过破坏和重构策略加强局部邻域的搜索能力，且已被成功用于求解流水车间调度及其相关问题。

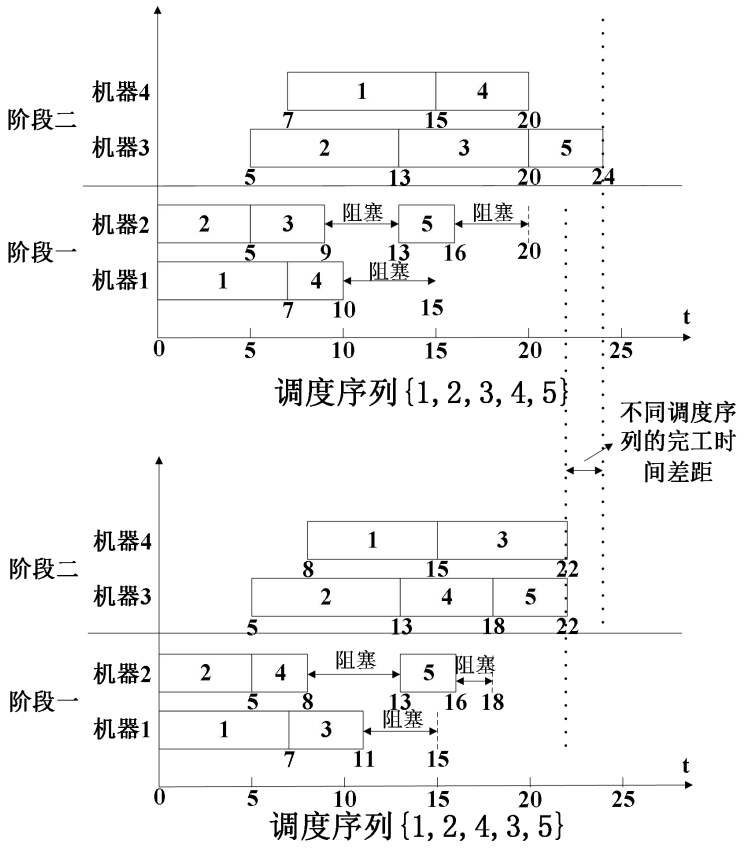


图 3-1 不同调度序列对比图

然而，仅仅采用局部搜索算法有时候难以使解在迭代过程中跳出某一搜索范围，降低了解的多样性，传统 IG 算法使用 SA 接受准则来提高解的多样性，但根据[62]实验得知其效果不是特别明显。传统 IG 算法使用的迭代贪婪改进策略在前期探索中表现出了较优的搜索性能。然而，当迭代达到一定次数后，继续使用该方法已难以再改善工件序列的阻塞状况。因此，就需要重新设计一种应对策略来对工件序列执行更大范围的探索。由文献[99]可知，GA 的变异策略具有较强的全局搜索能力，有利于解跳出局部的搜索范围。鉴于此，本文重新设计了一种策略来代替传统 IG 算法中的迭代贪婪改进方法，即，基于双层变异策略的迭代贪婪算法（Iterative Greedy-Double Level Mutation, IGDLM）来提高 IG 算法的全局搜索性能。

根据以上研究方案，为求解无缓冲区的 BHFSP，本文基于文献[45]建立了其数学模型，随后结合 BHFSP 的问题特性设计了一种基于交换算子的双层变异策略，所设计的方法弥补了传统 IG 算法在迭代后期全局探索上的不足，进一步减少阻塞造成的完工时间延后问题。 通过与 5 种经典的智能优化算法的在 100 个算例下的数字统计比较，证明了所提策略与 IG 算法结合能够获得更优的目标值，并为更大规模的 BHFSP 提供更好的调度排序方案。

本章结构安排如下：3.2 节建立了以最大完工时间为优化目标的阻塞混合流水车间调度问题的数学模型；3.3 节设计了本章所设计的 IGDLM 算法；3.4 节验证所提算法的性能并分析实验结果。最后，3.5 节为本章小结。

3.2 问题描述与数学模型

BHFSP与企业调度、产品产量、品质要求以及机器处理能力息息相关。在规定时间内产生的调度序列能够满足生产要求、阻塞时长尽可能少以及最大完工时间尽可能短的情况下，合理安排排序序列，调整加工工件的先后顺序，确保其生产可以顺利进行。本节首先从数学符号定义，约束条件，决策变量，优化目标来阐述BHFSP的数学模型，随后给出一个具体案例来说明阻塞约束对调度序列最大完工时间造成的影响。

BHFSP可以简单描述为：n 个代加工工件要经过数量为 L 的加工阶段，每个加工阶段存在数目不一定相等的 M_s 台并行机器，机器数量为 $M_s \geq 1$，且至少存在一个阶段，其机器的数量满足约束条件：$M_s > 1$。在已知条件下，问题输入的是待加工的调度序列，问题的输出是最大完工时间。针对该问题，数学模型需要满足的假设有：（1）

所有机器在0时刻均是可用的状态，所有工件在0时刻均可被加工；（2）每个工件均有 L 个工序，并且必须满足在每个阶段都进行加工；（3）所有工件在各个工序的加工时间是已知的；（4）每台机器在同一时刻只能处理一个工件；（5）每个工件在同一时刻只能被同一台机器加工；（6）不同阶段之间不存在缓冲区，若某一工件在当前阶段被处理完后，下一阶段的机器均是忙碌状态，那么该工件就会被阻塞在当前阶段的机器上，直至下个阶段机器处于空闲状态为止；（7）工件一旦被加工，就不能被中断或者被其他工件抢占。为方便描述该问题，定义如下数学符号：

J：工件集合 $J=\{1,2,...,n\}$，n 为工件总数.

j：工件序号，$j \in J$.

S：阶段集合 $S=\{1,2,...,L\}$，L 为加工阶段总数.

s：加工阶段编号，$s \in S$.

m：机器编号.

M_s：阶段 s 存在的并行机数.

K_s：阶段 s 的并行机集合，$K_s = \{1,...,m,...,M_s\}$.

$t_{j,m}$：工件 j 在机器 m 上的加工时间.

M：一个极大正数.

决策变量：

$B_{s,j}$：工件 j 在加工阶段 s 的开始时间.

$E_{s,j}$：工件 j 在加工阶段 s 的完工时间.

$D_{s,j}$：工件 j 在加工阶段 s 的离开时间.

$C_{\max}$：工件序列的最大完工时间.

$$X_{j,m}=\begin{cases} 1 \text{ 工件 } j \text{ 安排在机器 } m \text{ 上加工} \\ 0 \text{ 其他} \end{cases}$$

$$Y_{j,j',m}=\begin{cases} 1 \text{ 在机器 } m \text{ 上, 工件 } j \text{ 先于工件 } j' \text{ 加工, } j<j' \\ 0 \text{ 其他} \end{cases}$$

优化目标：

最小化 $C_{\max}$ （3-1）

约束条件：

$$\sum_{m \in K_s} X_{j,m} = 1 \ \forall j \in J, s \in S \quad (3\text{-}2)$$

$$E_{s,j} = B_{s,j} + \sum_{m \in K_s} (t_{j,m} \cdot X_{j,m}) \ \forall j \in J, s \in S \quad (3\text{-}3)$$

$$D_{s,j} = B_{s+1,j} \ \forall j \in J, s \in \{1,2,...,S-1\} \quad (3\text{-}4)$$

$$D_{s,j} \le B_{s,j'} + M(3 - Y_{j,j',m} - X_{j,m} - X_{j',m}) \quad (3\text{-}5)$$
$$\forall j \in J,\ j' \in J,\ j < j',\ s \in S,\ m \in K_s$$

$$D_{s,j'} \le B_{s,j} + M(2 + Y_{j,j',m} - X_{j,m} - X_{j',m}) \quad (3\text{-}6)$$
$$\forall j \in J,\ j' \in J,\ j < j',\ s \in S,\ m \in K_s$$

$$E_{s,j} \le D_{s,j} \ \forall j \in J, s \in S \quad (3\text{-}7)$$

$$D_{L,j} \le C_{\max} \ \forall j \in J \quad (3\text{-}8)$$

$$B_{s,j} \ge 0 \ \forall j \in J, s \in S \quad (3\text{-}9)$$

上式中，（3-1）表示调度序列的最大完工时间。约束（3-2）表示所有工件在任意阶段只能被一台机器加工。约束（3-3）表示：工件的完工时间等于工件开始时间与其加工时间之和。（3-4）-（3-6）表示阻塞约束，其中，约束（3-4）表示工件离开时间等于其在下一阶段的开始时间。约束（3-5）和（3-6）表示同一机器的非重叠性，即，在同一台机器上，前一个工件的离开时间大于等于后一个工件的开始时间。约束（3-7）表示工件只有在加工完成后才可以被释放。约束（3-8）表示最大完工时间要大于等于所有工件在最后一个阶段的离开时间。约束（3-9）表示所有工件的开始时间不可小于 0。

本文采用了基于整数的编码方式，为了更直观阐述该数学模型的思想，下面将给出一个简单的甘特图。

图 3-2 展示了不带阻塞和带阻塞约束的 HFSP 的甘特图。在带阻塞约束的生产过程中，工件 3 和 4 均发生了不同程度的阻塞，其中，工件 3 在时刻 9 被阻塞，时刻 14，阻塞状态解除，阻塞总时长为 5，随后被传到下一阶段的机器 3 上继续处理。由于调度序列的最大完工时间是由最后一个加工完毕的工件来决定的，对比甘特图中不带阻塞约束的 HFSP 加工流程，工件 3 因在阶段一的 9-14 时间段里被阻塞在机器上，从而导致了工件 5 在时刻 23 才开始第二阶段的处理，导致加工时间延长了 23-20=3。

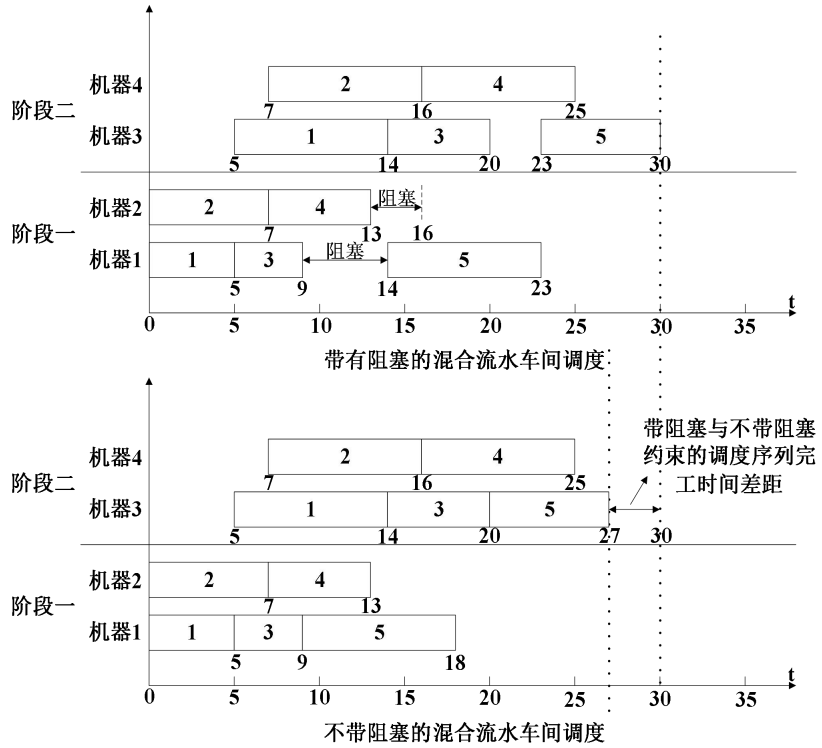


图3-2 带阻塞与不带阻塞的混合流水车间调度对比图

由以上实例对比可知，同一调度序列在面对不同约束的条件下，其完工时间也不同，带阻塞约束的调度序列，其完工时间大于不带阻塞约束调度序列的完工时间。阻塞的存在确实会延长调度时间，此外，随着工件规模的不断扩大，其阻塞对序列完工时间造成的影响也会随之扩大。因此，就需要专门设计一些高校的优化算法对序列进行调整，从而减少阻塞约束对完工时间的影响。

3.3 基于双层变异的迭代贪婪算法

由图3-2可知，阻塞约束会导致后续工件的开始时间延后，拖慢了整体序列的完工时间，进而降低了工厂的生产效率。在传统IG算法中，迭代贪婪改进策略与破坏-重构策略都是基于插入算子实现的，每次调整只是更改了部分工件序列的相对位置。为此，本章设计了基于交换算子的双层变异策略来弥补IG算法在全局搜索上的不充分，进而降低了阻塞约束对完工时间的影响。考虑到局部最优解的位置随调度排序的变化发生无规则运动，本章针对BHFSP，提出了一种双层变异策略来改善IG算法在全局探索上的不足。其算法流程图如下所示：

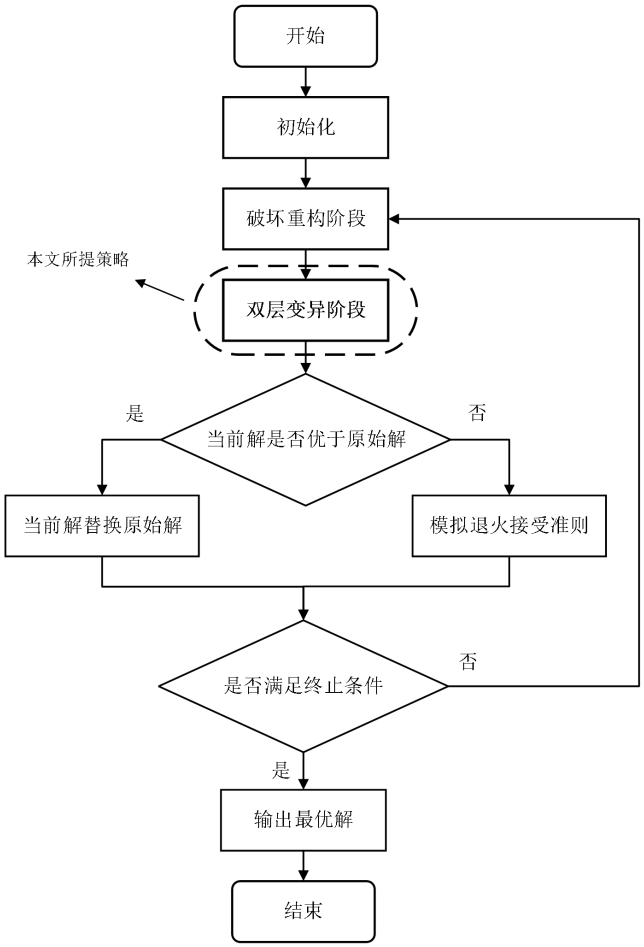


图3-3 算法流程图

3.3.1 初始化

针对BHFSP的优化，调度序列可能因排列顺序和阻塞程度的不同而导致最大完工时间延长。为了尽可能减少阻塞对完工时间造成的影响，采用有效的初始化策略就变得十分重要。NEH启发式算法在元启发式算法中常被用来产生初始解，并且已被证明是非常有效的方法[17]。因此，本章同样使用NEH初始化策略来产生初始的调度序列，从而减少阻塞约束的影响。NEH启发式策略的步骤如下所示：

步骤1 计算每个工件在所有阶段的加工时间之和 $TP_j=\sum^{S}p_{s,j}$，$j\in\{1,2,3,...,J\}$；并按照 TP_j 值对各工件进行降序排列，由此获得初始序列 $\pi^{origin}=\{\pi_1,\pi_2,...,\pi_J\}$。

步骤2 取出 π^{origin} 的前两个工件 π_1 和 π_2，可得到两种不同的排序序列 $\{\pi_1,\pi_2\}$ 和 $\{\pi_2,\pi_1\}$，将其中目标值较小的序列作为当前的调度序列，记为 π^{new}，随后，令 $j=3$。

步骤3 取出 π^{origin} 中的第 j 个工件 π_j，并将其插入到 π^{new} 中所有的位置，一共得到 j 个不同的新序列，选择目标值最小的序列来替换 π^{new}。

步骤 4 令 $j=j+1$。若 $j\le J$，则转到步骤 3；否则，得到序列 π^{new}，初始化阶段结束。

3.3.2 破坏和重构策略

小范围调整部分序列可以有效减少阻塞的时间，缩短最大完工时间。该操作需要使用局部搜索能力强的策略来完成，而已知的破坏和重构策略通过调整少部分工件的相对位置来获得更优质的解，具有较强的局部搜索能力。因此，本文引入了破坏和重构策略来改变部分序列的调度顺序，达到提升解性能的目的。该策略操作步骤如下：

步骤1 针对初始化阶段产生的调度方案 $\pi^{new}=\{\pi_1,\pi_2...\pi_J\}$，随机从 π^{new} 中选则 d 个不同的工件组成序列 π^{remove}。

步骤2 令 $j=1$，将 π_1^{remove} 插入到 π^{new} 中所有的位置，共有 $J-d+1$ 种不同的插入方式。将目标值最小的调度排序作为当前序列 π^{new}'，并令 $j=j+1$。

步骤3 取出工件 π_j^{remove}，将其插入到 π^{new}'的所有位置，共有 $J-d+j$ 种不同的插入方式。将目标值最小的调度排序作为当前序列 π^{new}'。

步骤4 令 $j=j+1$。若 $j\le d$，则转到**步骤3**；否则，得到序列 π^{new}'，破坏和重构策略结束。

下面通过一个具体实例来说明上述步骤：

（1） 假设由初始化产生的调度序列为 $\pi^{new}=\{5,3,2,4,1\}$，d=2，则从 π^{new} 中随机提取2个不同的工件，假设所提工件为3和4，将工件3和4从 π^{new} 中取出，组成新序列 $\pi^{remove}=\{3,4\}$，π^{new} 变为 $\pi^{new}=\{5,2,1\}$。

（2） 取出 π^{remove} 中的工件3，将其插入到 π^{new} 的所有位置，共得到4个新排列 {3,5,2,1}，{5,3,2,1}，{5,2,3,1}，{5,2,1,3}，计算所有序列的最大完工时间，将最大完工时间最小的序列作为当前调度序列 π^{new}'。假设序列{5,2,1,3}的目标值最小，则有 π^{new}'={5,2,1,3}。

（3） 取出工件4，将其插入到 π 中所有的位置，共得到5种不同的新序列，分别为 {4,5,2,1,3}，{5,4,2,1,3}，{5,2,4,1,3}，{5,2,1,4,3}，{5,2,1,3,4}，选择目标值最小的序列作为当前序列 π^{new}'。

（4） 此时，π^{remove} 为空集，破坏和重构策略结束。

3.3.3 双层变异策略

在BHFSP中，n 个工件存在 $n!$ 种不同的排列组合方式，随着工件数目的不断增多，其解空间也会急剧增大。此外，阻塞时间也会随着工件排序的不同而发生无规律变化，从而导致工件序列的最大完工时间发生改变。此外，排列顺序、工件阻塞与完工时间存在复杂的非线性关系，导致很难找到全局最优解的所在位置。因此，就需要设计跳跃性能更强的全局搜索策略，通过更大幅度的改变不同工件的位置来减少序列的阻塞时间。

在传统IG算法中，破坏重构策略已经使用大量的插入操作来调整工件的排列顺序，若使用破坏重构策略之后继续使用迭代贪婪改进策略，改变的依旧是部分工件的相对位置，所得解的多样性难以得到提升。

因此，设计基于交换算子的双层变异策略。一方面可以避免算法陷入局部最优，提高解的多样性。另一方面，选取搜索效率高且时间复杂度低的变异策略可以增加算法的迭代次数，使进化过程得以充分的进行。此外，插入算子的时间复杂度为$O(n^3)$，若提前确定好算法的终止时间，则会耗费过多的计算时间，算法整体迭代的次数必然会减少，进而降低了搜索到更优解的可能性。

为弥补传统IG算法全局探索能力的不足，新设计的搜索策略就需要具有较强的全局搜索能力。由文献[99]可知，GA算法中的变异算子具有较强的全局搜索能力，且使用交换算子的时间复杂度更低（$O(n^2)$）。因此，本文设计了基于交换算子的双层变异策略来替换传统IG算法中的迭代贪婪改进方案。一方面可以避免算法陷入局部最优，提高解的多样性；另一方面，可以增加迭代次数，使进化过程更加充分的进行。具体操作如下：

提前设定变异概率 P_m 、定义随机数 $r1$ 和 $r2$ ， $r1,r2 \in (0,1)$ 。当 $r1 < P_m$ 时，采用简单的交换算子执行第一层变异过程；当 $r2 < P_m$ 时，采用折半交换算子执行第二层的变异过程。

第一层变异过程：对破坏重构策略产生的序列 π^{new}'，随机从中选择两个工件 a 和 b，且 $a \neq b$，交换两个工件所在的位置，如果得到的新序列 π^{new}"的最大完工时间小于 π^{new}'的最大完工时间，则替换 π^{new}'并继续进行同样的交换操作，直至交换的次数达到 J ，第一层变异过程结束。

为方便理解，下面将继续通过一个实例来说明该策略的实施过程，具体步骤如下：

（1） 假设由上一步得到的序列为π^{new}'={5,3,2,4,1}。

（2） 随机选取两个工件a和b，假设，$a=1$，$b=2$，交换a和b的位置，得到新序列π^{new}"={5,3,1,4,2}。

（3） 对比π^{new}"和π^{new}'的最大完工时间，如果π^{new}"的最大完工时间小于π^{new}'的最大完工时间，则令π^{new}'=π^{new}"；否则令π^{new}"=π^{new}'。

（4） 重复步骤（1）和（2），直至交换次数达到J。

第二层变异过程：采用折半交换变异算子，针对第一层变异策略得到的调度序列，令π=π^{new}"，并将π记为π=$\{\pi_1,\pi_2...\pi_{J-1},\pi_J\}$，有以下3个步骤：

步骤1 令下标p^{Front}指向π中的第一个位置上的工件π_1，下标p^{Back}指向π中的最后一个位置上的工件π_J，交换π_1和π_J的位置，得到π'=$\{\pi_J,\pi_2...\pi_{J-1},\pi_1\}$，如果$\pi'$的目标值小于$\pi$，则令$\pi=\pi'$；否则$\pi$保持不变。假设$\pi'$的目标值小于$\pi$，$\pi$的序列则变为$\pi$=$\{\pi_J,\pi_2...\pi_{J-1},\pi_1\}$。

步骤2 令p^{Front}向后移一个位置，p^{Back}向前移一个位置，交换π=$\{\pi_J,\pi_2...\pi_{J-1},\pi_1\}$中两个下标所指向的工件$\pi_2$和$\pi_{J-1}$，得到$\pi'$=$\{\pi_J,\pi_{J-1}...\pi_2,\pi_1\}$，如果$\pi'$的目标值小于$\pi$，则令$\pi=\pi'$；否则$\pi$保持不变。

步骤3 重复步骤2，直至满足$p^{Front}\geq p^{Back}$，第二层变异策略结束。

同样，本章将给出一个实例来帮助理解此过程：

（1） 假设由第一层变异策略产生的调度序列为π^{new}"={5,3,2,4,1}，令π=π^{new}"。

（2） 令p^{Front}指向工件5，p^{Back}指向工件1，交换工件5和1的位置，得到π'={1,3,2,4,5}，若π'最大完工时间小于π的最大完工时间，则令π=π'，否则π保持不变。假设此次操作后得到的π'目标函数值大于π，则π保持不变。

（3） 令p^{Front}后移，p^{Back}前移，交换p^{Front}和p^{Back}所指向工件3和4的位置，得到π'={1,4,2,3,5}，比较π'和π的最大完工时间，如果π'的目标函数值小于π，则令$\pi=\pi'$；否则π保持不变。

（4） 令p^{Front}后移，p^{Back}前移，此时p^{Front}等于p^{Back}，该策略结束。

3.3.4 SA 接受准则

执行完以上策略后，需要判断是否要使当前产生的解参与到下一次的迭代中，一般而言，最简单的接受准则是：若当前解的最大完工时间比初始解的完工时间更短，则用当前解替换初始解。否则，舍弃当前解；该操作虽然简单易行，但在解的多样性

保持上并不显著。因此，本章使用了SA接受准则来提升解的多样性[100]，其策略核心思想为：如果新生成的解的最大完工时间多于当前解的最大完工时间，将依旧选择以一定概率 $random \le \exp\{-(C_{\max}(\pi^{current}) - C_{\max}(\pi^{origin}))/Temperature\}$，$random \in (0,1)$，$T \in (0,1)$ $Temperature = T \cdot (\sum_{s=1}^{S}\sum_{i=1}^{J} p_{s,j})/(S \cdot J \cdot 10)$，接受当前解，而非直接用新解替换当前解。

3.4 实验结果与分析

本章所提出的IGDLM算法是由C++编程，并基于Visual Studio 2019环境实现的，PC机使用的为Windows10.0系统，16GB RAM，奔腾处理器 Core i7，2.60 GHZ。

在所测实例中，工件数的取值范围为 $J \in \{20,40,60,80,100\}$，加工阶段数量的取值范围为 $S \in \{5,10\}$，每个阶段中机器数的取值范围是 $M_s \in \{1,5\}$，工件加工时间取值范围为 $p_{s,j} \in \{1,99\}$。综上，共有5×2=10种不同的组合方式（J×S），每种组合分别包含随机产生的10个不同的算例，共5×2×10=100个算例，经过多次预实验测试，最终，决定将参数 P_m，温度 T 设置为 $P_m = 0.1$，$T = 0.5$。

为了验证本文所提算法的性能，将其与传统IGA[93]和同样用于求解HFSP的GA算法[101]、有效候鸟优化（Effective Migrating Birds Optimization，EMBO）算法[24]、离散人工蜂群（Discrete Artificial Bee Colony，DABC）算法[25]和改进DPSO算法[26]进行了对比测试。IGA、GA、EMBO、DABC、DPSO算法的参数设置与策略如下所示（表3-1）：

IGA: 使用NEH策略产生初始解，随后采纳工件的破坏重构策略、基于插入算子的迭代改进方案和SA接受准则。

GA: 随机产生初始种群，在算法迭代过程中使用交叉算子、变异算子、精英保留方法和SA策略。

DABC: 随机产生初始种群，在算法迭代过程中使用一个新的增强战略和一种针对问题特性的局部优化方法，搜索蜜蜂未探索的领域。

EMBO: 随机产生初始种群，在算法迭代过程中使用两种竞争机制、联合搜索策略和动态验收准则等方案，并将局部搜索策略和Glover算子应用在陷入局部最优的个体中。

DPSO: 采用NEH和SPT初始化方案分别产生两个初始解，其他解随机产生，随后，调用变邻域搜索策略对所得到的初始解进行改进，紧接着，在迭代过程中采用相邻成对交换策略来提高解的质量。

表 3-1 算法的参数设置

算法	种群大小	破坏工件数	交叉率	变异率	恒定因子	降温因子
	Psize	*d*	*Pc*	*Pm*	*a*	*T*
IGDLM	1	4	/	0.3	/	0.5
IGA	1	3	/	/	/	0.5
GA	100	/	0.7	0.1	4	0.85
DABC	20	/	/	/	30	/
EMBO	25	/	/	/	10	/
DPSO	100	/	/	/	200	/

为公平起见，将所有对比算法设定在同一个终止时间 $TimeLimit = J \times S \times CPU$ ，$CPU \in \{5,10\}$ 的范围内，且算法一旦到达 $TimeLimit$ 规定的时间点，将终止运行。实验测试共有100个算例，所有算例重复执行5次。

3.4.1 参数的敏感度测试

破坏重构策略中的参数 d 是影响IGDLM算法整体性能极为关键的一个参数，其中，所提取的工件数 d 对平衡算法的全局搜索和局部探索均起着十分重要的作用。针对本章所使用的破坏重构策略，d 值过高会导致算法性能下降，计算时间增多，但是，d 值过小则会导致对解的开发力度不够，搜索范围过小。为此，本章对 d 参数展开了敏感度测试，其测试集为 $d \in \{2,3,4,5,6\}$ ，用所有测试算例中最大完工时间的均值作为实验结果，其效应图如图3-4所示。

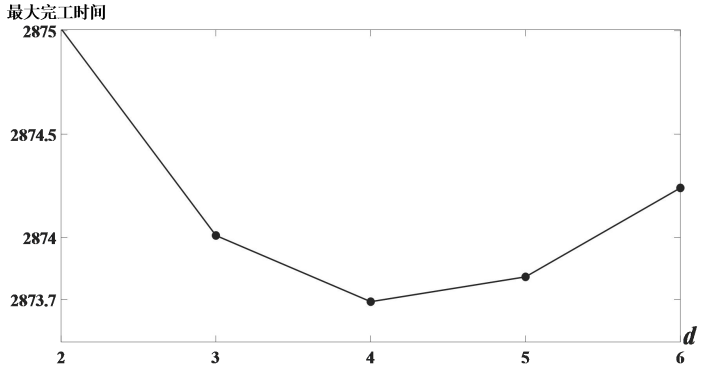


图3-4 不同*d*值的效应图

由图3-4可知，当$d=4$时，最大完工时间的均值最小，当d大于4时，算法虽然具有较强的局部探索能力，但对工件探索需要较长的时间，从而降低了IGDLM的收敛速度。当d小于4时，算法虽然具有较快的收敛速度，但其局部搜索能力较差。因此，本章最终选择$d=4$作为设定的参数，可以平衡IGDLM算法的局部搜索能力和收敛速度。

在所提双层变异策略中，变异概率P_m的设定值对解性能的影响程度也不同，为了尽可能快的找到较优的调度序列，本节对P_m进行了敏感度测试，将测试范围取为$P_m \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$，同样使用算例的最大完工时间平均值作为最终的测试结果，其结果如图3-5所示。由图可知，当$P_m=0.3$时，序列最大完工时间最小，当$P_m<0.3$时，算法变异概率较低，其所提策略对解多样性的改进并不明显。当$P_m>0.3$时，变异概率较高，对工件序列改变的次数增多，导致了算法强化性和多样性的不平衡。因此，本文最终选择$P_m=0.3$。

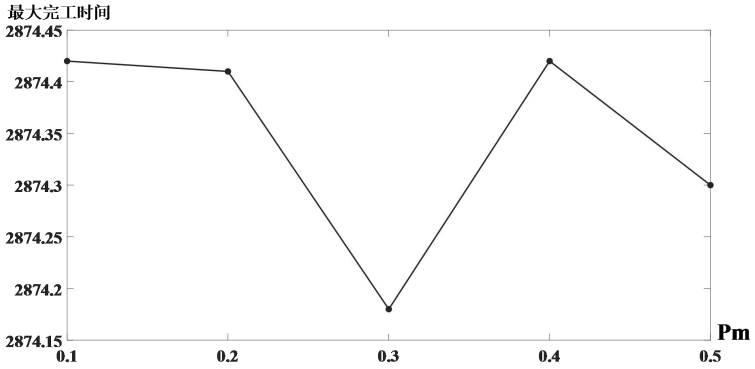


图3-5 不同P_m的效应图

3.4.2 实验结果分析

为验证所提策略的性能，表3-2给出了不带双层变异策略（$IGDLM_{N\_DLM}$）和带双层变异策略（IGDLM）得到的最大完工时间的最小值（MIN）以及均值（AVG），此外，表3-2还测试了Wilcoxon Two-Sided Rank Sum（显著性水平为0.05的），“†”和“‡”分别表示所提算法的AVG和MIN值与$IGDLM_{N_DLM}$存在差异性。在表3-2中，加粗字体表示在同一规模问题中所获得的最好值。

由表3-2可知，$IGDLM_{N_DLM}$获得了5组最小的MIN值，IGDLM获得9个最小的MIN值。由表3-2可知，除去80×5规模，IGDLM在其他规模的问题上均获得了最小的MIN值。针对AVG值，IGDLM除在40×10规模的AGV值不如$IGDLM_{N_DLM}$外，在其他规模

的问题上均获得了最小的AVG值。总之，由表3-2可知，IGDLM算法所获得的AVG与MIN的AVG值均好于$IGDLM_{N_DLM}$，其原因可能为：带双层变异策略的IGDLM算法提升了在迭代后期的全局搜索能力，使解的多样性得到进一步的提升。

在表3-3和表3-4中，每一行数据均表示在求解同一规模中的10个算例时，IGDLM算法与对比算法所获得的最大完工时间（AVG）以及最小值（MIN）的均值。此外，表3-3和表3-4给出了Wilcoxon Two-Sided Rank Sum检验（显著性水平为0.05），“†”和“‡”分别表示对比算法与IGDLM算法存在差异性。

表 3-2 CPU=5 有无双层变异策略的最大完工时间

	$IGDLM_{N\_DLM}$		IGDLM	
J×S	**AVG**	**MIN**	**AVG**	**MIN**
20×5	794.5	546	**794.1**	**545**
20×10	**1350.2**	**1218**	**1350.2**	**1218**
40×5	1371.1†	904‡	**1367.3**	**889**
40×10	**2217.1**	**1565**	2217.7	**1565**
60×5	**2761.1**	**1586**	**2761.1**	**1586**
60×10	3184.2†	2101‡	**3180.3**	**2076**
80×5	**3404.5**	**1484**	**3404.5**	1487
80×10	4273.9	**3758**	**4272.9**	**3758**
100×5	3908.6†	1532‡	**3906.1**	**1506**
100×10	5502.5†	5193‡	**5487.6**	**5163**
AVG	2876.77	1988.7	**2874.18**	**1979.3**

由表3-3和表3-4可知，IGDLM算法在所有规模中取得的AVG值均优于其他对比算法。通过分析表3-3实验数据可知，在规模为20×10的问题中，IGDLM算法求解最大完工时间得到的MIN值比DABC多9，但在其他规模上均取得了最小值。除此之外，IGA可以在20×5, 40×10，60×5和80×10规模的问题上取得最小值，DABC可以在20×10规模的算例上取得最小目标值，GA, EMBO和DPSP均没有取得最好值。由表3-4可知，IGDLM算法除在规模为60×10的算例中没有取得最好的MIN值外，在其他规模上均取得了最好的MIN值。此外，其他对比算法，IGA分别在20×5, 40×10，60×5和80×10的算例上取得了最好的MIN值，DABC在规模为20×10, 60×5的算例上取得了最好的MIN值，而EMBO只在60×10规模的算例中取得了1个最小的MIN值。GA和DPSO没有取得最好的MIN值。最后，从表3-3和表3-4的实验结果来看，IGDLM算法的AVG均好于其他对比算法。

表 3-3 CPU=5 时所有算法获得的 AVG 和 MIN 值

	IGDLM		IGA		GA		DABC		EMBO		DPSO	
J×S	**AVG**	**MIN**	**AVG**	**MIN**	**AVG**	**MIN**	**AVG**	**MIN**	**AVG**	**MIN**	**AVG**	**MIN**
20×5	**794.1**	**545**	794.2	**545**	799.3†	559‡	796.4	554‡	795.7	546	832.8†	589‡
20×10	1350.2	**1218**	**1349.9**	1218	1360.1†	**1218**	1356.6†	**1209**	1358.6	**1218**	1477.7†	1297‡
40×5	**1367.3**	**889**	1368.6	894	1387.3†	923‡	1399.4†	931‡	1372.8†	903‡	1482†	1017‡
40×10	**2217.7**	**1565**	2219.5	**1565**	2241.5†	1585‡	2263.7†	1582‡	2232.1†	1581‡	2403.7†	1657‡
60×5	**2761.1**	**1586**	**2761.1**	**1586**	2762.5	1600‡	2763.9	1599‡	2762.1	1730‡	2806.2†	2340‡
60×10	**3180.3**	**2076**	3186.5†	2094‡	3202.8†	2114‡	3244.7†	2190‡	3189.5†	2071‡	3331.7†	2340‡
80×5	**3404.5**	**1487**	3405.9†	1492‡	3417.6†	1560‡	3442.9†	1627‡	3407.9†	1500‡	3471.2†	1564‡
80×10	**4272.9**	**3758**	4277.2†	**3758**	4293†	3767‡	4380.8†	3767‡	4280.5†	3767‡	4414.7†	3819‡
100×5	**3906.1**	**1506**	3909.1†	1527‡	3919.2†	1599‡	3939.3†	1660‡	3913†	1529‡	3953.3†	1666‡
100×10	**5487.6**	**5163**	5507.6†	5188‡	5523.7†	5201‡	5760.7†	5194‡	5507.3†	5194‡	5800.1†	5245‡
AVG	**2874.2**	**1979.3**	2877.9	1986.7	2890.7	2012.6	2934.8	2031.3	2882	2003.9	2997.3	2153.4

表 3-4 CPU=10 时所有算法获得的 AVG 和 MIN 值

	IGDLM		IGA		GA		DABC		EMBO		DPSO	
J×S	**AVG**	**MIN**	**AVG**	**MIN**	**AVG**	**MIN**	**AVG**	**MIN**	**AVG**	**MIN**	**AVG**	**MIN**
20×5	**794**	**545**	**794**	**545**	799.3	559‡	795	551‡	795.7	546	825.6†	589‡
20×10	**1350**	**1218**	**1350**	**1218**	1360.1†	1218	1354†	**1209**	1358.6†	**1218**	1448.9†	1297‡
40×5	**1366**	**888**	1367	893	1387.3†	923‡	1391.8†	923‡	1372.8†	903‡	1475.9†	1017‡
40×10	**2217**	**1565**	2218	**1565**	2241.5†	1585‡	2251.9†	1582‡	2232.1†	1581‡	2392.8†	1610‡
60×5	**2761**	**1586**	**2761**	**1586**	2762.5	1600‡	**2761.1**	**1586**	2762.1	1593‡	2801.3†	1657‡
60×10	**3178**	**2074**	3184.4†	2094‡	3202.8†	2114‡	3228.7†	2168‡	3189†	**2067**	3331.7†	2340‡
80×5	**3403**	**1483**	3404.2†	1491‡	3416.7†	1560‡	3433†	1614‡	3407†	1492‡	3471.2†	1564‡
80×10	**4273**	**3758**	4275.5†	**3758**	4291.4†	3767‡	4357.2†	3764‡	4280.5†	3767‡	4414.7†	3819‡
100×5	**3906**	**1503**	3908.3†	1525‡	3917.6†	1584‡	3934.2†	1631‡	3911.4†	1514‡	3952†	1666‡
100×10	**5485**	**5163**	5503.3†	5188‡	5519.4†	5194‡	5722.3†	5194‡	5502.4†	5194‡	5800.1†	5245‡
AVG	**2873**	**1978**	2877	1986	2889.9	2010	2922.9	2022.2	2881.2	1988	2991.4	2080.4

由以上实验结果可知，本章所提出的双层变异策略在求解BHFSP优化问题上是十分有效的。其有效的原因可能是阻塞约束的影响，延长了调度序列的最大完工时间，而所提策略可以及时有效的改变工件的位置，进而探索更为广阔的解空间。

3.5 本章小结

针对传统HFSP, 通常有足够多的缓冲区来存放已加工完毕的工件，但是，在实际的生产过程中，由于产品属性，技术条件等因素的限制，导致相邻阶段之间不存在缓冲区，从而导致了阻塞现象的发生，该问题演就变成了BHFSP, 与传统HFSP相比，

该问题更加复杂，是制造业亟需解决的重要问题。目前存在的智能优化算法不能很好的平衡其全局搜索和局部搜索能力，本章根据其问题设计了相应的求解算法。

鉴于此，本章针对BHFSP展开了研究。首先，构建了以最大完工时间为优化目标的BHFSP数学模型；随后，针对BHFSP的问题特性设计了基于双层变异策略的IGDLM算法进行求解；最后，通过与其他求解类似的问题的经典智能优化算法进行实验对比，验证了IGDLM的高效性。

本章仅考虑了以最大完工时间为优化目标的BHFSP，并没有考虑其在分布式的调度环境下的情况。因此，在下一章将研究以最大完工时间为优化目标的DBHFSP，并根据问题特性建立相应的数学模型，阐述其计算过程，最终提出相对应的IG算法来优化该问题。

第四章 求解阻塞混合成组流水车间调度问题的组邻域搜索迭代贪婪算法

4.1 研究背景

在之前的章节中，BHFSP 已被证明是 NP 难问题，若在 BHFSP 中考虑成组技术，即 BHFGSP，问题的复杂性将进一步增加。在目前已经发表的研究成果中，可以找到不少求解 HFSP，BHFSP 和 FSGSP 的相关算法。然而，在 HFSP 的相关研究中，还没有文献提出一个针对阻塞和成组约束的数学模型。但根据以往调查发现，这些约束在现实世界中却又是普遍存在的，因此，设计相应的数学模型，开发可以求解该问题的算法具有一定的学术意义和应用价值。

本章首先设计了以最大完工时间为目标的 BHFGSP 数学模型，随后，提出了一种新的针对特定问题的 IG 算法来优化该问题。与现有的智能优化算法相比，IG 算法具有参数少、结构简单的特点，并表现出较强的局部搜索能力[22]。因此，本章继续使用该算法，并将其设计成为一种能够有效求解 BHFGSP 的优化算法。

本章的主要贡献如下。

1）首次提出了以最大完工时间为目标的 BHFGSP 的数学模型，随后利用 CPLEX 验证了模型的正确性。

2）开发了新的编码和解码方式，并列出了解码和解码过程。通过该过程，可以得知工件序列的最大完工时间。

3）同时针对成组约束和阻塞约束设计相应的策略。

4）在新的 IG 算法中，分别设计了基于工件组和阻塞工件的邻域搜索策略，以优化解决方案的最大完工时间。

5）提出了八种交换算子并将其嵌入到邻域搜索策略中，所提算子可以提高新 IG 算法的全局和局部搜索能力。

本章的其余部分组织如下。在第 2 节中建立了 BHFGSP 的数学模型。在第 3 节中提出了基于工件组和阻塞工件的邻域搜索策略的 IG 算法。第 4 节对模型进行了验证、对算法进行了仿真实验和分析。第 5 节对本章内容进行了总结。

4.2 问题描述与数学模型

4.2.1 BHFGSP 的数学模型

BHFGSP 描述如下：工厂包含S个不同的加工阶段。在所有阶段设置了$M_s(M_s = M_1, M_2, ..., M_S)$个并行机器来处理工件序列，在任何相邻机器之间没有中间缓冲区。关于待调度的工件组和工件，有以下定义和规则：需要在机器上处理一系列组F，其中组f中的工件集被表示为ω_f，组f中的工件j必须经历所有阶段，同一系列中的工件需要在同一台机器上处理。在处理之前，应在组f_1和组f_2之间考虑序列依赖的设定时间（Sequence-Dependent Setup Time, SDST），set_{s,f_1,f_2}，$set_{s,0,f_2}$表示阶段s中第一个组的 SDST。阶段s工件的加工时间设置为$p_{j,s}$。本章的优化目标是最小化序列的C_{max}（makespan）。为了进一步描述该问题，给出以下假设：

- 所有工件在时刻 0 均可用。
- 同一组中的工件必须在同一台机器上连续处理，一旦开始处理，则不能被其他组中断。
- 在任何时刻，一个工件只能被一台机器所处理，且每台机器一次只能处理一个工件。
- 运输时间默认包含在加工时间中。
- 所有工件必须在所有阶段连续处理，不允许跳过某个阶段或提前结束。

该问题的目标、符号、决策变量和约束总结如下：

符号定义：

C_{max}　　工件序列的最大完工时间.

S　　阶段数量.

s　　阶段编号，$s \in \{1, 2, \cdots, S\}$.

M_s　　在加工阶段s的并行机数量.

k　　记录阶段s的机器编号，$k \in \{1, 2, \cdots, M_s\}$

F　　组的数量.

f, f_1, f_2　　组的编号，$f, f_1, f_2 \in \{0, 1, \cdots, F\}$，$0$是一个空组，其表示在某台机器上组序列的开始时间.

ω_f　　组f中的工件序列.

N　　工件数量.

j, j_1, j_2　　工件编号，$j, j_1, j_2 \in \{0, 1, \cdots, N\}$．$0$是一个空的工件序列，$\omega_0$表示了工件序列的开始.

$p_{j,s}$　　工件j在加工阶段s的加工时间.

set_{s,f_1,f_2}　　在加工阶段s上f_1与f_2之间的设定时间，$set_{s,f,f}=0$.

决策变量：

$c_{j,s}$　　工件j在加工阶段s的完工时间.

$d_{j,s}$　　工件j在加工阶段s的离开时间.

$w_{s,f,k}$　　二进制决策变量，若组f在加工阶段s的机器k上加工，则为 1，否则为 0.

x_{s,f_1,f_2}　　二进制决策变量，若组f_1和组f_2在加工阶段s的相同机器上加工，并且f_1先于f_2加工，则为 1，否则，为 0.

z_{s,j_1,j_2}　　二进制决策变量，若工件j_1和工件j_2在加工阶段s的相同机器上加工，并且j_1先于j_2加工，则为 1，否则，为 0.

优化目标：

$$\text{最小化 } C_{max}=\left(\max_{j=1,2,\cdots,N} d_{j,S}\right)$$

约束条件：

$$\sum_{k=1}^{M_s} w_{s,f,k}=1,\ \forall s\in\{1,2,\cdots,S\},\ \forall f\in\{1,2,\cdots,F\} \tag{4-1}$$

$$x_{s,f_1,f_2}+x_{s,f_2,f_1}=\sum_{k=1}^{M_s}(w_{s,f_1,k}\times w_{s,f_2,k}),\ \forall s\in\{1,2,\cdots,S\},\ \forall f_1,f_2\in\{1,2,\cdots,F\}\wedge f_1>f_2 \tag{4-2}$$

$$z_{s,0,j}=1,\ \forall s\in\{1,2,\cdots,S\},\ \forall j\in\{1,2,\cdots,N\} \tag{4-3}$$

$$z_{s,j,j}=0,\ \forall s\in\{1,2,\cdots,S\},\ \forall j\in\{1,2,\cdots,N\} \tag{4-4}$$

$$z_{s,j_1,j_2}+z_{s,j_2,j_1}=1,\ \forall s\in\{1,2,\cdots,S\},\ \forall f\in\{1,2,\cdots,F\},\ \forall j_1,j_2\in\omega_f\wedge j_1\neq j_2 \tag{4-5}$$

$$z_{s,j_1,j_2}=x_{s,f_1,f_2},\ \forall s\in\{1,2,\cdots,S\},\ \forall f_1,f_2\in\{1,2,\cdots,F\}\wedge f_1\neq f_2,\ \forall j_1\in\omega_{f_1},j_2\in\omega_{f_2} \tag{4-6}$$

$$c_{0,s}=0,\ \forall s\in\{1,2,\cdots,S\} \tag{4-7}$$

$$d_{0,s}=0,\ \forall s\in\{1,2,\cdots,S\} \tag{4-8}$$

$$c_{j_2,s}\geq d_{j_1,s}+set_{s,f_1,f_2}+p_{j_2,s}+(z_{s,j_1,j_2}-1)\times inf,\ \forall s\in\{1,2,\cdots,S\},\ \forall f_1\in\{0,1,\cdots,F\},\ \forall f_2\in\{1,2,\cdots,F\},\ \forall j_1\in\omega_{f_1},\ \forall j_2\in\omega_{f_2} \tag{4-9}$$

$$d_{j,s} \geq c_{j,s},\ \forall s \in \{1,2,\cdots,S\},\ \forall j \in \{1,2,\cdots,N\} \quad (4\text{-}10)$$

$$c_{j,s} = d_{j,s-1} + p_{j,s},\ \forall s \in \{2,3,\cdots,S\},\ \forall j \in \{1,2,\cdots,N\} \quad (4\text{-}11)$$

由约束（4-1）可知，每个组在特定阶段只能分配给一台机器加工。约束（4-2）确保将两个不同的组指定给同一台机器，一个组不能同时是另一个组的前序和后继组。约束（4-3）表示当工件是第一个要在机器上处理的工件时，其前驱必须是空工件。约束（4-4）表明工件本身没有前驱和后继关系。约束（4-5）确保若两个不同的工件属于同一个组，则这两个工件之间必须有明确的排列顺序。约束（4-6）确保属于同一组的工件不能与属于该组的其他工件混合。此外，组的顺序关系决定了工件安排的前后关系。约束（4-7）-约束（4-8）分别表示各个阶段空工件的完工时间和离开时间。约束条件（4-9）确保在同一阶段，来自不同组的两个具有顺序关系的工件的完工时间和离开时间。组的完工时间不少于其前驱加上其加工时间、离开时间和组之间的 SDST。约束（4-10）表示工件的离开时间不小于其在同一阶段的完工时间。约束（4-11）表示工件的完工时间等于在当前阶段的加工时间与在最终阶段的离开时间之和。

在数学模型中，空组和工件表示处理序列的开始，属于同一个组的工件不会与其他工件分离和混合。

4.2.2 编码和解码过程

高效的编码和解码过程可以降低计算复杂度，提高解的质量，特别是在求解大规模复杂组合优化问题上亦是如此[102]。为了描述编码的执行过程，本章使用了在数学模型符号中确定的变量$c_{j,s}$、$d_{j,s}$和set_{s,f_1,f_2}。此外，本章还提供了中间变量$MIdle_{s,m_s}$来表示每个阶段不同组的机器选择方案：在加工阶段s，处理当前组机器的第一次可用时间m_s，BHFGSP 的目标通过方程$C_{max} = \max d_{j,S}(j = 1,2,\cdots,N)$计算。该问题考虑了不同组的 SDST 和阻塞约束。若$d_{j,s} = c_{j,s}$，则表示工件 j 在加工阶段 s 上没有发生阻塞，否则，若$d_{j,s} > c_{j,s}$则表示工件 j 在加工阶段 s 上被阻塞。在算法 4-1 中给出了解码的详细信息。类似地，本章根据参考文献[103]中 HFSP 的置换编码方法来进行编码操作，即，解决方案编码被设置为（π,τ），其中$\pi = \{\pi_1,\pi_2,...,\pi_l,...,\pi_F\}$包含 F 个组，$\tau = \{\tau_1,...,\tau_l,...,\tau_F\}$表示每个工件组中的工件序列，其中$\tau_l = \{\tau_{l,1},...,\tau_{l,n_l}\}$包含工件组$\pi_l$中的工件序列$n_l$。

由算法 4-1 可知，在步骤 1 中，如果机器m_s处于空闲状态，将选择机器来处理属

于同一组的工件序列，若处于同一阶段的所有机器均开始处理工件组，则将选择一台具有最小$MIdle_{s,m_s}+Set_{s,f_1,f_2}$值的机器来处理该组。在步骤 2 中，选择适当的机器后，此组中的工件将从阶段 1 开始直到最后一个阶段被安排连续加工。在步骤 3 中，当所有组在最后阶段被处理完毕时，将计算其序列的目标值。基于解码过程，本章提出了一种新的 IG 算法，通过减少不同组之间的阻塞条件和 SDST 来最小化工件序列的最大完工时间。

算法 4-1 BHFGSP 的解码过程

Input: 解决方案 （π,τ）

Output: C_{max}

Begin:

For f=1 **to** F **do**

步骤 1: 在每个阶段选择合适的加工机器

For s=1 **to** S **do**

If m_s 没有加工工件.

$MIdle_{s,m_s}=MIdle_{s,m_s}+Set_{s,0,f_2}$

Else

找到含有最小值$MIdle_{s,m_s}+Set_{s,f_1,f_2}$的机器$m_s$

EndIf

EndFor

步骤 2: 在所选择的机器上安排工件加工

For j=$\tau_{f,1}$ **to**τ_{f,n_f} **do**

$c_{j,1}=MIdle_{1,m_1}+p_{j,1}$

$d_{j,1}=c_{j,1}$

$MIdle_{1,m_1}=d_{j,1}$

For s=2 **to** S **do**

$c_{j,s}=\max\{MIdle_{s,m_s},c_{j,s-1}\}+p_{j,s}$

$d_{j,s}=c_{j,s}$

$MIdle_{s,m_s}=d_{j,s}$

$d_{j,s-1}=c_{j,s}-p_{j,s}$

$MIdle_{s-1,m_{s-1}}=d_{j,s-1}$

EndFor

EndFor

EndFor

步骤 3: 获得工件序列的最大完工时间

$C_{max}=\max\limits_{j=1,2,\cdots,N} d_{j,S}$

End

4.2.3 实例说明

为了更好地描述 BHFGSP 的解码过程，图 4-1 展示了 F = 4, S = 3, N = 8,

$\tau_1=\{1,2\}$，$\tau_2=\{3,4,5\}$，$\tau_3=\{6\}$和$\tau_4=\{7,8\}$的调度甘特图，所有工件都按照预先安排的顺序进行处理。表 4-1 到 4-2 分别列出了各阶段工件的加工时间和组 SDST。具体如下：

表 4-1 工件在每一阶段的加工时间

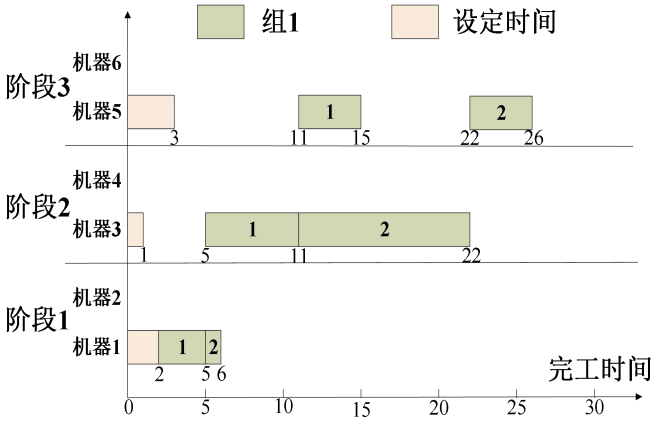
Family	Job	Stage1	Stage2	Stage3
1	1	3	6	4
	2	1	11	4
2	3	1	3	5
	4	1	3	4
	5	3	7	9
3	6	3	3	3
4	7	5	4	3
	8	5	3	4

表 4-2 在每一阶段组之间的设定时间

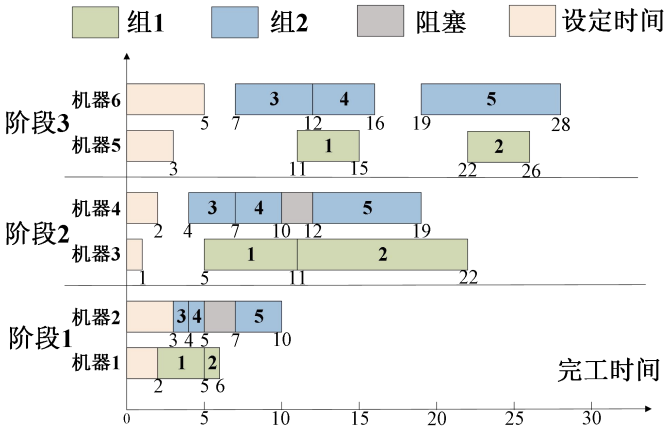
Stage1						Stage2						Stage3					
Family	0	1	2	3	4	Family	0	1	2	3	4	Family	0	1	2	3	4
0	0	2	3	4	4	0	0	1	2	3	4	0	0	3	5	6	5
1	0	0	2	1	3	1	0	0	4	5	4	1	0	0	4	6	7
2	0	5	0	7	6	2	0	3	0	6	3	2	0	7	0	6	5
3	0	3	4	0	2	3	0	3	2	0	3	3	0	4	3	0	2
4	0	5	3	4	0	4	0	2	4	3	0	4	0	2	2	2	0

工件组 1 和工件组 2 分别分配给未处理工件的空闲机器。当工件组 3 在第一阶段选择机器时，根据公式$MIdle_{s,m_s}+Set_{s,f_1,f_2}$，可以得到：$MIdle_{1,1}+Set_{1,1,3}=11+1=12$，$MIdle_{1,2}+Set_{1,2,3}=12+7=19$。因此，$m_1=1$被选为组 3 的加工机器。随后，$MIdle_{1,1}=12$，$c_{6,1}=MIdle_{1,1}+p_{6,1}=12+3=15$，$d_{6,1}=c_{6,1}=15$，$MIdle_{1,1}=d_{6,1}$. 在第二阶段，根据方程式$MIdle_{2,1}+Set_{2,1,3}=22+5=27$，$MIdle_{2,2}+Set_{2,2,3}=19+6=25$，选择机器$m_2=2$。之后，令$MIdle_{2,2}=25$，$c_{6,2}=\max\{MIdle_{2,2},\ c_{6,1}\}+p_{6,2}=\max\{25,\ 15\}+3=28$，$d_{6,2}=c_{6,2}=28$，$MIdle_{2,2}=c_{6,2}=28$，$d_{6,1}=c_{6,2}-p_{6,2}=28-3=25$，$MIdle_{1,1}=d_{6,1}=25$。在第三阶段，计算以下方程式：$MIdle_{3,1}+Set_{3,1,4}=35+7=32$，$MIdle_{3,2}+Set_{3,2,4}=28+5=32$。根据结果，选择机器$m_3=1$来处理该组。随后，计算$c_{6,3}=\max\{MIdle_{3,1},\ c_{6,2}\}+p_{6,3}=\max\{32,\ 28\}+3=35$，$d_{6,3}=c_{6,3}=35$，$MIdle_{3,1}=d_{6,3}=35$，$d_{6,2}=c_{6,3}-p_{6,3}=32$，$MIdle_{2,2}=d_{6,2}=32$。组 4 执行与组 3 类似的步骤，组 4 的具体调度过程如图 4-1（d）所示，最后，总完工时间C_{max}为 40。

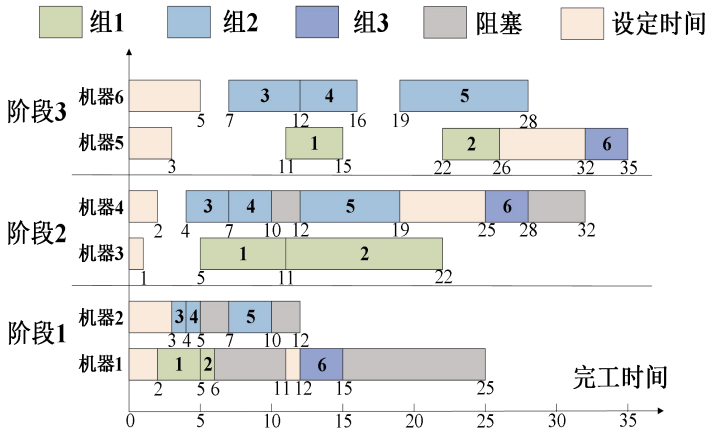
为了更直观地展示阻塞约束和组之间的设定时间对最大完工时间的影响，本节添加了组 5 来比较两个序列的最大完工时间，即$\pi_1=\{1,2,3,4,5\}$和$\pi_2=\{1,2,3,5,4\}$。如图 4-2 所示，与组序列π_1相比，在第一阶段，π_2阻塞时间减少了 1，组之间的设定时间减少了 9，完成时间减少 2。在第二阶段，π_2阻塞时间减少 1，组之间的设定时间减少 8，完工时间减少 4。在最终阶段，与π_1相比，π_2组之间的设定时间和C_{max}分别减少了 10 和 5。为了减少阻塞约束和组之间的设定时间的影响，基于解码过程，本章开发了基于工件组和阻塞工件约束的邻域搜索策略。通过仿真结果表明，所提出的策略可以有效地提高解的质量，此外，为了便于描述，将后面部分中使用的解码过程简写为 *DP* ()。



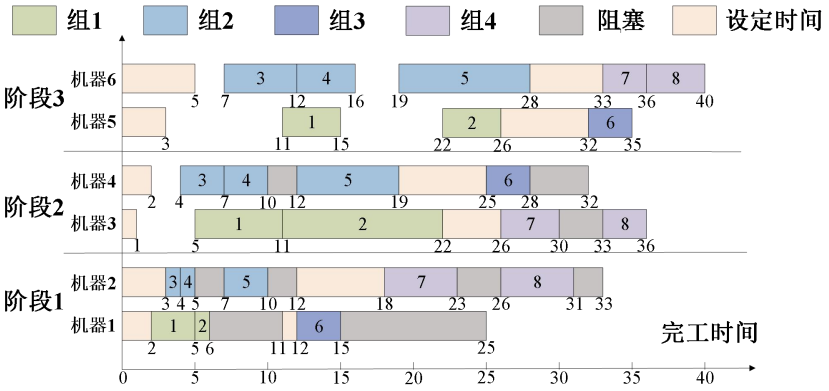
（a）组 1



（b）组 2



（c）组 3



（d）组 4

图 4-1 调度实例的甘特图

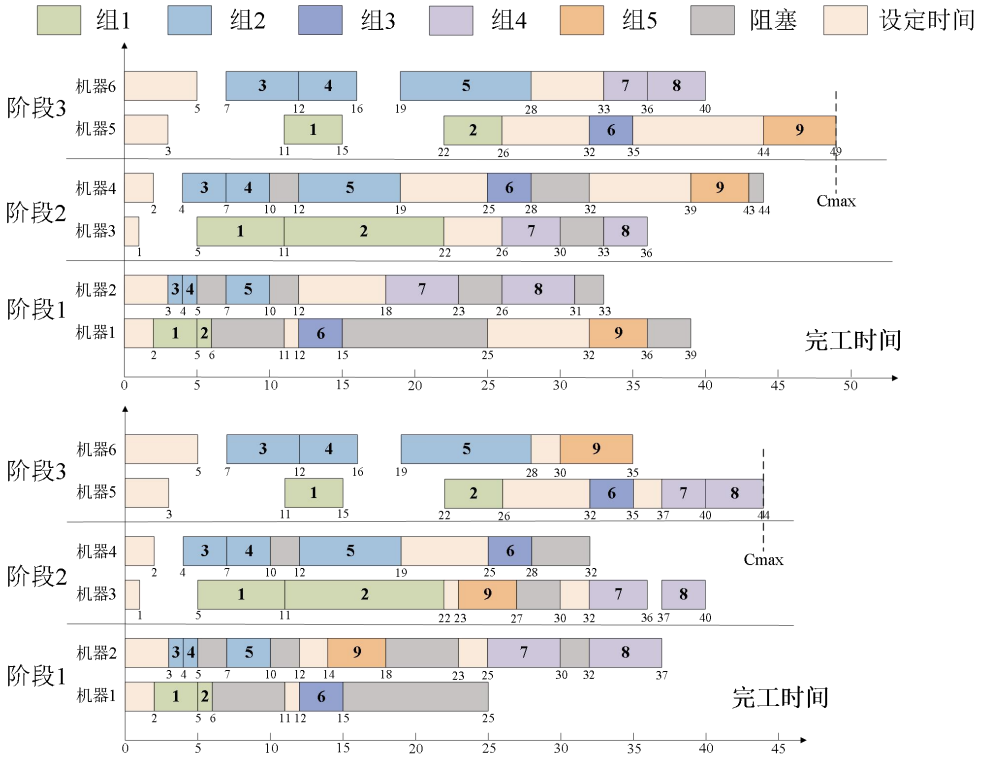


图 4-2 不同调度序列的甘特图

4.3 基于组邻域搜索策略的迭代贪婪算法

在处理 BHFGSP 时，如果工件在前一阶段的机器上被阻塞，则工件的离开时间可能会延长，因此，无法提前预测工件序列的实际完工时间。为求解复杂的组合优化问题，除了使用解码程序来解决机器选择、组调度和最大完工时间计算的问题外，还需要设计一种有效的智能优化算法来对工件和组进行排序。

Ruben 设计的传统 IG 算法只解决了工件的排序问题[93]，并没有考虑工件成组的概念，因此，它不适合本章所研究的问题。本章进一步考虑和设计了基于成组约束的策略，并提出了一种基于组邻域搜索策略的 IG（Neighborhood search Iterative Greedy，NIG）算法来求解 BHFGSP。该算法可以有效地解决工件和工件组的排序问题。其核心思想是：将 BHFGSP 分为两个子问题：一个是组序列排序，另一个是工件序列排序。

该算法保留了 IG 算法的主要结构，其组成如下：1）使用基于工件组的 NEH 初始化策略生成初始调度序列。2）采用破坏构造策略作为局部强化策略，探索有希望的解区域，提高局部搜索能力。3）组邻域搜索策略旨在改变工件组的排列顺序。4）

提出了基于阻塞工件的邻域搜索策略，对每个组的工件序列进行扰动。3）和 4）有效解决上述提到的两个子问题。5）模拟退火验收准则将依据概率更新当前解，并记录最佳解。该准则在一定程度上保持了算法的全局和局部搜索能力。NIG 算法的框架如图 4-3 所示。

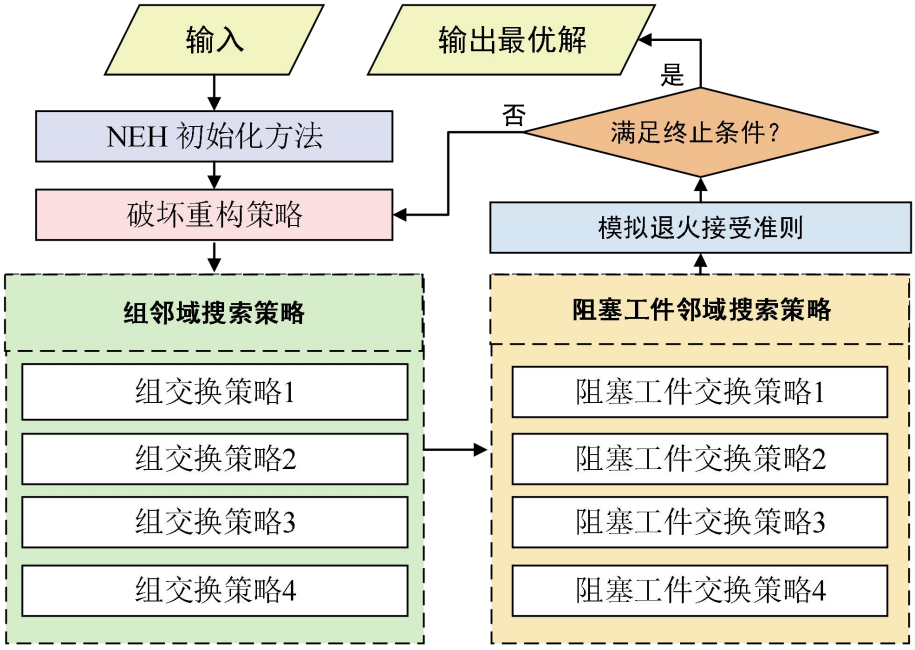


图 4-3 NIG 算法框架

4.3.1 NEH 初始化策略

当机器的 SDST 降低时，整个调度序列的最大完工时间也可能缩短。为了获得高质量的初始解，本节提出了基于工件组的 NEH 初始化策略来安排初始化过程中的工件组排序。NEH 方法作为一种有效的启发式策略，已被广泛应用并嵌入到 IG 算法的初始化策略中。由于传统的 NEH 策略是基于工件执行的，每个工件可以被插入到序列的任何位置，然而，由于在 BHFGSP 中组之间没有交叉工件分配的概念，该约束条件将被打破。因此，本节重新设计了 NEH 初始化策略，即，将对于工件的操作替换为对工件组的操作，解决了组内工件不能交叉的问题。

为了更方便地描述初始化过程，本节引入了一个新的符号定义 $p_{j,s,l}$，其含义为：在加工阶段 s 上，属于工件组 l 的工件 j 的加工时间。加工开始时，所有工件都随机分配给已建好的组，随后，按加工时间长段降序排列工件组的位置，从降序中逐个提取组，并将其插入到另一个组序列的所有位置以比较最大完工时间的值，此操作结束后，将保留最大完工时间最小的组序列，并参与新工件的插入测试，最后，将得到一

个完整的组序列。值得注意的是，当使用 NEH 初始化策略时，它不影响内部工件的排序，而只影响工件组与工件组之间的排序。

算法 4-1 NEH 初始化策略

Input: $(\pi^{origin},\tau^{origin})$, $\pi^{sub}=\{\}$

Output:(π^{N},τ^{N})

Begin:

$P_l \leftarrow \sum_{s=1}^{S}\sum_{j=1}^{n_l} p_{j,s,l},\ l=1,...,F$

$\pi^{\Delta} \leftarrow$ 根据降序序列P_l对组序列$\{\pi_1^{origin},...,\pi_l^{origin},...,\pi_F^{origin}\}$进行排序

For *l*=1 **to** *F*

For *i*=1 **to** $|\pi^{sub}|+1$

$\pi_i^{sub} \leftarrow$ 从π^{Δ}中提取组π_l^{origin}, 并将其插入到序列π^{sub}的第 i^{th} 个位置上

End

$(\pi^{sub},\tau^{sub}) \leftarrow arg\min_{i=1}^{|\pi^{sub}|+1} DP(\pi_i^{sub},\tau_i^{sub})$

End

$(\pi^{N},\tau^{N}) \leftarrow (\pi^{sub},\tau^{sub})$

End

4.3.2 破坏重构策略

使用初始化策略后，执行基于工件组的破坏重构策略，通过扰乱工件组序列来减少最大完工时间。与 NEH 初始化策略类似，破坏重构策略具有很强的局部搜索能力，它使用贪婪的插入操作在搜索区域中找到更好的解决方案，在最近提出的一些 IG 算法中，这种策略已被广泛使用[62,70]。此外，在本文中还使用了破坏重构策略来改变工件组的安排顺序，算法 4-2 提供了该策略的过程。

算法 4-2 破坏重构策略

Input: (π^{N},τ^{N}), $\pi^{remove}=\{\}$

Output:(π^{d},τ^{d})

Begin:

For *g*=1 **to** *d*

随即从$\pi^{initial}$中提取出一个工件, 将其插入到工件序列π^{remove}中

End

For *g*=1 **to** *d*

For *i*=1 **to** $|\pi^{initial}|+1$

$\pi_i^{sub} \leftarrow$ 随机提取一个组π^{remove}, 将其插入到π^{sub}的第 i^{th} 个位置

End

$\pi^{sub} \leftarrow arg\min_{i=1}^{|\pi^{sub}|+1} DP(\pi_i^{sub},\tau_i^{sub})$

End

$(\pi^{d},\tau^{d}) \leftarrow (\pi^{sub},\tau^{sub})$

End

在破坏重构策略中，主要包括以下三个部分：1）从原始组序列π中随机选择并删除 d 个组，并将其放入空序列π^{remove}中；2）将π^{remove}中的 d 个组逐个提取并插入到当前组的所有位置中测试；3）计算当前序列的最大完工时间，并记录目标值最低的序列。

4.3.3 组邻域搜索策略

在 BHFGSP 的调度程序中，工件的加工时间是预先确定的，一旦确定则不能更改，机器选择由解码过程决定，因此，减少完工时间的主要途径是改变不同组之间的 SDST 和工作的阻塞状况，本节设计了组邻域搜索策略来减少 SDST，进一步缩短优化的目标值，此外，所有策略都是基于交换算子设计的，以取代传统 IG 算法中使用的迭代改进策略，并将算法的时间复杂度从 $O(F^3)$降低到 $O(F^2)$（假设循环的迭代次数为 F），实验结果表明，该策略确实提高了所提算法的性能。

在 NIG 算法框架中，组邻域搜索策略有助于提高全局搜索能力。在所提出的组邻域搜索策略中，设计了四种基于工件组的交换算子来优化解决方案，其整体框架如下所示。

算法 4-3 组邻域搜索策略

Input: (π^d,τ^d)

Output: (π^g,τ^g)

Begin:

$rnd \leftarrow randi(1,4)$ % 整数在区间[1, 4]中随机生成

If $rnd=1$

$(\pi^g,\tau^g) \leftarrow$ 执行组交换算子 1 $((\pi^d,\tau^d))$

Else If $rnd=2$

$(\pi^g,\tau^g) \leftarrow$ 执行组交换算子 2 $((\pi^d,\tau^d))$

Else If $rnd=3$

$(\pi^g,\tau^g) \leftarrow$ 执行组交换算子 3 $((\pi^d,\tau^d))$

Else

$(\pi^g,\tau^g) \leftarrow$ 执行组交换算子 4 $((\pi^d,\tau^d))$

End If

End

- **组交换算子 1:** $(\pi^{inter},\tau^{inter}) \leftarrow (\pi^d,\tau^d)$。 从$\pi^{inter}$中随机选择两个组 π_a^{inter}，π_b^{inter}并交换它们所在的位置。若$DP(\pi^{inter},\tau^{inter}) <= DP(\pi^d,\tau^d)$， $(\pi^d,\tau^d) \leftarrow (\pi^{inter},\tau^{inter})$; 否则, $(\pi^{inter},\tau^{inter}) \leftarrow (\pi^d,\tau^d)$；当迭代完$R$次时，$(\pi^g,\tau^g) \leftarrow (\pi^{inter},\tau^{inter})$。
- **组交换算子 2:** $(\pi^{inter},\tau^{inter}) \leftarrow (\pi^d,\tau^d)$。 $count=1$， $count2=count+1$。 当

$count2<=F$ 时，交换 π_{count}^{inter} 和 π_{count2}^{inter} 的位置，若 $DP(\pi^{inter},\tau^{inter})<=DP(\pi^{d},\tau^{d})$，$(\pi^{d},\tau^{d})\leftarrow(\pi^{inter},\tau^{inter})$；否则，$(\pi^{inter},\tau^{inter})\leftarrow(\pi^{d},\tau^{d})$；$count2++$；继续交换$\pi_{count}^{inter}$和$\pi_{count2}^{inter}$的位置直到满足终止条件$count2=F$。当$count2$的数量达到$F$时，$count++$，$count2=count+1$；继续执行与上述相同的操作直到$count=F-1$。当所有的操作都完成时，$(\pi^{g},\tau^{g})\leftarrow(\pi^{inter},\tau^{inter})$。

- **组交换算子 3:** $(\pi^{inter},\tau^{inter})\leftarrow(\pi^{d},\tau^{d})$。$count=1$，$count2=count+1$。当$count2<=F$时，交换$\pi_{count}^{inter}$和$\pi_{count2}^{inter}$的位置，$(\pi^{inter},\tau^{inter})\leftarrow(\pi^{d},\tau^{d})$，$count2++$，持续执行该步骤直到满足$count2=F$。在经过所有交换操作后，计算工件序列的目标值，随后记录下具有最小完工时间的工件组的位置为pos。交换π_{count}^{inter}和π_{pos}^{inter}的位置，如果$DP(\pi^{inter},\tau^{inter})<=DP(\pi^{d},\tau^{d})$，$(\pi^{d},\tau^{d})\leftarrow(\pi^{inter},\tau^{inter})$；否则，$(\pi^{inter},\tau^{inter})\leftarrow(\pi^{d},\tau^{d})$；$count++$，$count2=count+1$；继续执行如上所示的相同步骤，直到$count=F-1$。当所有的操作完成时，$(\pi^{g},\tau^{g})\leftarrow(\pi^{inter},\tau^{inter})$。
- **组交换算子 4:** $(\pi^{inter},\tau^{inter})\leftarrow(\pi^{d},\tau^{d})$。$count=1$，$count2=count$。当$count2<=F-1$时，交换$\pi_{count2}^{inter}$和$\pi_{count2+1}^{inter}$的位置，$count2++$，重复该步骤，直到达到终止标准$count2=F-1$。当满足终止条件时，最佳序列记录为$(\pi^{inter},\tau^{inter})$。若$DP(\pi^{inter},\tau^{inter})<=DP(\pi^{d},\tau^{d})$，$(\pi^{d},\tau^{d})\leftarrow(\pi^{inter},\tau^{inter})$；否则，$(\pi^{inter},\tau^{inter})\leftarrow(\pi^{d},\tau^{d})$；$count++$，$count2=count$；继续执行与上述相同的步骤，直到满足$count=F-1$。当所有的操作都完成时，$(\pi^{g},\tau^{g})\leftarrow(\pi^{inter},\tau^{inter})$。

4.3.4 阻塞工件邻域搜索策略

由于无缓冲区的影响，工件可能会在加工过程中被阻塞在机器上，这种情况会阻碍整个进程的执行，并延长工件序列的完工时间。因此，在调整组之间的设定时间后，有必要进一步对组内的工件序列进行排序来减少阻塞约束对最大完工时间造成的影响，据作者所知，目前还没有相应的策略来求解 BHFGSP，本小节设计了基于阻塞工件的邻域搜索策略，以有效减少阻塞对优化目标的影响。

在该策略中，首先计算每个阶段所有工件的阻塞时间，然后根据其时间长度对包含工件序号的序列进行降序排序，如果某些工件的阻塞时间为 0，则将它们排列在阻塞工件之后，于是，被阻止时间最长的工件将最先受到关注。随后，根据工件阻塞时间按降序逐一安排。若一个组中仅包含一个工件，则针对工件的操作将自动转换为对组的操作，即执行阻塞工件交换策略 1。此外，阻塞工件交换策略 2，阻塞工件交换

策略 3 和阻塞工件交换策略 4 仅针对同一组中的工件序列执行。与组邻域搜索策略的思想类似，阻塞工件邻域搜索策略也是基于交换算子设计的。以上操作可以有效减少传统迭代改进策略中工件序列的计算时间，弥补 IG 算法全局搜索能力的不足。

接下来，本节将给出该策略的框架以及每个算子的具体执行过程。

算法 8-4 阻塞工件邻域搜索策略

Input: (π^g,τ^g), $JobInFam[j]$ % 记录工件 j 所属的工件组。

Output: (π^B,τ^B)

Begin:

$BlockingTime_j \leftarrow \sum_{s=1}^{S}(d_{j,s}-c_{j,s})$, $j=\{1,2,...,N\}$ % 记录所有阶段每个工件的阻塞时长。

$Blocking \leftarrow$ 对$\{BlockingTime_1,...,BlockingTime_j,...,BlockingTime_N\}$ 降序排序。若一些工件的阻塞时长为 0，则将它们按序安排到阻塞时长大于 0 的序列后面加工。$Blocking_j$ 表示在序列$Blocking$位置 j 的工件。

For j = 1 **to** N % 在序列$Blocking$的工件序号。

If $|JobInFam[Blocking_j]|=1$ % 只包括一个工件的组

$(\pi^B,\tau^B) \leftarrow$ 执行工件交换算子 1 $((\pi^g,\tau^g),\ JobInFam[Blocking_j])$

Else

$rnd \leftarrow randi(1,3)$ % 在区间[1, 3]内的一个随机整数

If $rnd=1$

$(\pi^B,\tau^B) \leftarrow$ 执行工件交换算子 2 $((\pi^g,\tau^g),JobInFam[Blocking_j])$

Else If $rnd=2$

$(\pi^B,\tau^B) \leftarrow$ 执行工件交换算子 3 $((\pi^g,\tau^g),JobInFam[Blocking_j])$

Else $rnd=3$

$(\pi^B,\tau^B) \leftarrow$ 执行工件交换算子 4 $((\pi^g,\tau^g),JobInFam[Blocking_j])$

End If

End If

End For

End

- **阻塞工件交换算子 1：**该策略将对工件的操作转为对组的操作。首先，找到按阻塞时长排序的组$JobInFam[Blocking_j]$，并记为 π^g_{select}。随后，将π^g_{select} 与π^g中的所有的组进行交换，随后，目标值最小的组序列记为 π^{inter}，若 $DP(\pi^{inter},\tau^{inter})<=DP(\pi^g,\tau^g)$，$(\pi^B,\tau^B)\leftarrow(\pi^{inter},\tau^{inter})$；否则，$(\pi^B,\tau^B)\leftarrow(\pi^g,\tau^g)$；
- **阻塞工件交换算子 2：**令 $count=1$，找到 $JobInFam[Blocking_j]$ 中的工件 $\tau^g_{select}=Blocking_j$，$Pos=Select$。$(\pi^B,\tau^B)=(\pi^{inter},\tau^{inter})=(\pi^g,\tau^g)$。当$count<=C$时，随机选择一个$\tau^{inter}_{random}$与$\tau^{inter}_{Pos}$属于相同组的工件，且 $\tau^{inter}_{random}!=\tau^{inter}_{Pos}$。交换$\tau^{inter}_{random}$和$\tau^{inter}_{Pos}$

的位置。随后获得新序列$(\pi^{inter},\tau^{inter})$，若$DP(\pi^{inter},\tau^{inter})<=DP(\pi^{B},\tau^{B})$，$(\pi^{B},\tau^{B})\leftarrow(\pi^{inter},\tau^{inter})$，$Pos\leftarrow random$；否则，$(\pi^{inter},\tau^{inter})\leftarrow(\pi^{B},\tau^{B})$；$count++$，重复上述步骤，直到满足终止标准。

- **阻塞工件交换算子 3：**设置$Pos=1$，找到$JobInFam[Blocking_j]$中满足条件的工件$\tau^{g}_{select}=Blocking_j$，$(\pi^{B},\tau^{B})=(\pi^{inter},\tau^{inter})=(\pi^{g},\tau^{g})$。当$Pos<=|JobInFam[Blocking_j]|$时，交换$\tau^{inter}_{select}$和$\tau^{inter}_{Pos}$的位置。若$DP(\pi^{inter},\tau^{inter})<=DP(\pi^{B},\tau^{B})$，$(\pi^{B},\tau^{B})\leftarrow(\pi^{inter},\tau^{inter})$，$select\leftarrow Pos$；否则，$(\pi^{inter},\tau^{inter})\leftarrow(\pi^{B},\tau^{B})$；$Pos++$，重复上述步骤，直到满足终止标准。
- **阻塞工件交换算子 4：**设置$Pos=1$，$\min Pos=-1$，找到属于组$JobInFam[Blocking_j]$的工件$\tau^{g}_{select}=Blocking_j$。$(\pi^{B},\tau^{B})=(\pi^{inter},\tau^{inter})=(\pi^{g},\tau^{g})$。当$Pos<=|JobInFam[Blocking_j]|$，交换$\tau^{inter}_{select}$和$\tau^{inter}_{Pos}$的位置。若$DP(\pi^{inter},\tau^{inter})<=DP(\pi^{B},\tau^{B})$，$(\pi^{inter},\tau^{inter})\leftarrow(\pi^{B},\tau^{B})$，$\min Pos\leftarrow Pos$；否则，$(\pi^{inter},\tau^{inter})\leftarrow(\pi^{B},\tau^{B})$；$Pos++$，重复上述步骤直到满足$Pos=JobInFam[Blocking_j].size()$。当操作结束时，如果$\min Pos!=-1$，交换$\tau^{B}_{select}$和$\tau^{B}_{\min Pos}$的位置。

4.4 实验结果与分析

4.4.1 测试数据和评价指标

为了测试 NIG 算法的性能，本节参考 DFGSP[83]、HFSP[24]和 BHFSP[48]中使用的测试算例，将工件规模，工件组规模及加工阶段设置为：$N=\{80,100,120,140,160,180,200,220,240,260,280,300\}$，$F=\{20,40,60\}$和$S=\{3,5,8\}$，总计 12×3×3=108 种不同的组合。针对不同组合，分别在[50，99]和[10，20]的均匀分布范围内随机生成工件的加工时间$p_{j,s}$和工件组之间的设定时间set_{s,f_1,f_2}。此外，所有策略都使用了与第三章相同的实验环境与编程语言。

参考文献[24, 62]，本节设置最大运行时间为算法运行的终止条件，该值为$\rho\times F\times S$毫秒，参数ρ有两个值：100 和 200。根据不同的时间范围，可以更直观、更全面地观察不同时长下的算法性能。若算法的运行时间达到设定的终止条件，则测试结束，否则，算法将在下一次迭代中继续执行。值得注意的是，在一些大规模实例中，算法的运行时间可能略大于本文设置的终止时间，此操作属于正常现象。在所有算例中均将

相对百分比偏差（RPD）作为响应变量（RV）进行计算，它测量单个测量结果与平均值的偏差，并已在许多文献中广泛使用[78, 104]。

4.4.2 参数设置和敏感度分析

该算法包括 3 个待测试参数：1）通过破坏重构策略提取的工件数（d）；2）组随机交换算子中的执行次数（R）。3）阻塞工件随机交换算子中的执行次数（C）。F 表示工件组的数量，c 表示所选组中的工件数，为了校准不同参数对所提的 NIG 算法的影响，本章依旧使用 Taguchi 进行实验测试。

三个参数设置了四个不同的级别，即 $d=\{2,3,4,5\}$、$R=\{F,2F,3F,4F\}$ 和 $C=\{c,2c,3c,4c\}$。随后，如表 4-3 所示，列出了一个带有 16 个 (d,R,C) 组合的正交数组 $L_{16}(4^3)$ 进行测试。为减少结果的差异性，本节使用了跨度较大的实例，即 80×3×20、120×3×20、160×5×40 和 200×5×40 四个规模来研究参数对算法性能的影响。针对不同 (d,R,C) 组合的实例，重复执行 30 次，并计算相应的 RV 值。根据表 4-3 中的 RV 值，表 4-4 显示了每个参数的平均 RV 值和显著性等级，在表 4-4 中，Delta 表示不同级别参数之间的 RV 差距，rank 表示参数对算法性能的影响程度。

表 4-3 正交数组和 RV 值

Case Number	Factor level			RV
	d	R	C	
1	2	F	c	0.64
2	2	2F	2c	0.58
3	2	3F	3c	0.52
4	2	4F	4c	0.50
5	3	F	2c	0.53
6	3	2F	c	0.52
7	3	3F	4c	0.53
8	3	4F	3c	0.51
9	4	F	3c	0.56
10	4	2F	4c	0.61
11	4	3F	c	0.59
12	4	4F	2c	0.59
13	5	F	4c	0.65
14	5	2F	3c	0.56
15	5	3F	2c	0.61
16	5	4F	c	0.60

由表 4-4 可知，参数 d 是最有影响力和最重要的参数，而 C 和 R 分别排在第二和第三。根据表 4-4 的结果，当 d=3 时，可以获得最小 RV 值，其原因可能是 d 值太小（d<3）会降低算法的局部搜索能力，导致较弱的强化性。然而，过大的 d 值（d>3）可能会对当前解造成过多干扰，并花费更多的时间，从而导致 NIG 算法的性能下降。

针对参数 C 和 R，它们对 NIG 算法的性能影响不大，其原因可能是：它们仅仅是针对组和阻塞工件的邻域搜索策略中执行的次数，在算法的每次迭代过程中，它们只根据概率进行选择，因此参数值的影响很小。

根据表 4-4 中的实验结果，本节最终将参数值设置为：$d = 3$, $C = 3c$, $R = 4F$。

表 4-4 每个参数的平均 RV 值和 rank 值

Level	d	R	C
1	0.56	0.59	0.59
2	**0.52**	0.57	0.58
3	0.59	0.56	**0.54**
4	0.60	**0.55**	0.57
Delta	0.08	0.04	0.05
Rank	1	3	2

4.4.3 MILP 模型的评估及与 NIG 算法的比较

为了验证数学模型和 NIG 算法的正确性与有效性，本节选取了 10 个不同规模的算例进行测试。数学模型使用数学编程求解器 CPLEX Studio IDE 进行编码。此外，将数学模型的最大运行时间设置为 1000s，针对所提出的 NIG 算法，本小节依旧使用在 4.4.1 节中提到的终止条件“$\rho \times F \times S$ 毫秒”，NIG 算法的参数设置参考 4.4.2 小节：$d = 3$, $C = 3c$, $R = 4F$。针对每个实例，NIG 算法独立重复运行 30 次，测试结果的平均值记为“Makespan”。表 4-5 列出了数学模型和 NIG 算法的相关运算结果，针对数学模型，本节统计并计算了其约束数、Gap 值、最大完工时间、RPD 值和实际运行时间，对于 NIG 算法，本节计算了最大完工时间、RPD 值和实际运行时间。

表 4-5 数学模型和 NIG 算法评估

	Mathematical Model					NIG		
J/F/S/M	Constraints	Gap	Makespan	RPD	Times	Makespan	RPD	Times
8/4/2/3	346	37.84%	37	0	0.59s	37	0	0.80s
10/5/2/3	516	39.58%	48	0	1.10s	48	0	1.00s
12/6/2/3	720	21.74%	55	0	1.86s	55	0	1.20s
14/7/2/3	958	14.45%	71	0	21.17s	71	0	1.40s
16/8/2/3	1230	11.49%	87	0	70.23s	87	0	1.60s
18/9/3/8	1895	2.22%	45	0	137.31s	53	17.18	2.70s
20/10/3/8	2357	12.50%	48	0	1000.00s	57	18.75	3.00s
22/11/3/8	2870	22.64%	53	0	1000.00s	62	16.98	3.30s
24/12/3/8	3434	44.78%	58	0	1000.00s	66	13.79	3.60s
26/13/3/8	4049	50.00%	64	0	1000.00s	65	1.56	3.90s

由表 4-5 可知，针对小规模实例，随着问题规模的不断扩大，数学模型约束的数量也随之不断增加，数学模型在所有实例中都取得了最佳的最大完工时间和 RPD 值，

而 NIG 算法仅在前五个实例中取得了最佳值。然而，在前五种情况下，除第一个实例外，NIG 算法比数学模型花费的时间更少。此外，数学模型得到的结果优于 NIG 算法得到的结果。然而，数学模型的运行时间很长，由此可知，随着问题规模的不断增大，数学模型的时间成本也将急剧增加。对于大规模问题，它甚至可能无法在有限的时间范围内得到可行的结果，此时，使用本文所提算法可以有效地解决这一问题。

4.4.4 NIG 算法策略性能评估

接下来，本章将对所提出的基于组和阻塞工件邻域搜索策略进行性能评估。如表 4-6 所示，没有组和阻塞工件的邻域搜索策略分别用 *NIG_N_F* 和 *NIG_N_J* 表示。NIG 算法包括两种邻域搜索策略，如表 4-6 所示，针对每个实例，三种算法分别进行了 30 次独立重复实验，随后，计算了 30 个数值的平均值和最佳值。同时，RPD 值被用于测量单个测试值与平均值的偏差，此外，本节对实验结果进行了统计测试与分析，以查看在每个实例中，对比算法和 NIG 算法是否存在显著性差异。

在表 4-6 中，*p* 值通过 T 检验获得，T 检验分别比较了 *NIG_N_F/NIG* 和 *NIG_N_J/NIG* 中 30 次重复的结果，若 *p* 值小于显著性水平 0.05，则拒绝原始假设，表明两种比较算法之间存在显著性差异；否则，接受假设，两种算法之间的差异不明显或没有差异，此外，本节用黑体标记测试结果中的最佳值，或标记有显著性差异的 *p* 值。

由表 4-6 可知，*NIG_N_J* 在平均值、最佳值和 RPD 值没有取得最好的实验结果，并且在 108 个实例中，结果与 NIG 算法有显著不同。由此可以看出，针对阻塞约束设计的阻塞工件邻域搜索策略对算法的性能有很大的影响。该策略有效地提高了算法的全局搜索能力，进一步增加了解的多样性，减少了阻塞约束对调度序列完成时间的影响。*NIG_N_F* 和 NIG，*NIG_N_F* 分别获得 24/108、20/108 和 22/108 最小平均值、最佳 RPD 值，NIG 分别获得 84/108、88/108、86/108 最小平均值、最佳 RPD 值。此外，两种算法有 49 个实验结果存在显著性差异，由此可以看出，NIG 算法优于 *NIG_N_F* 算法。

在一定程度上，组邻域搜索策略减少了不同组之间的机器设定时间，使组的排列更加合理，因此，进一步优化了调度序列的目标值。然而，通过对比实验可知，针对阻塞约束设计的阻塞工件邻域搜索策略对最大完工时间的影响大于针对组设计的策略，该操作说明针对阻塞工件设计的策略对算法性能影响更大，在下一节不同算法的

对比实验中，将进一步比较和分析基于阻塞工件的邻域搜索策略，以验证该方法的性能以及对 NIG 算法的影响。

表 4-6 NIG_N_J, NIG_N_F 和 NIG 的对比结果

		NIG_N_J				NIG_N_F				NIG		
	N×S	MEAN	BEST	RPD	p-value	MEAN	BEST	RPD	p-value	MEAN	BEST	RPD
F=20	80×3	3583	3576	7.52	**0.00**	3363	3346	0.92	**0.00**	**3338**	**3332**	**0.18**
	80×5	7342	7325	5.34	**0.00**	7098	7052	1.83	**0.00**	**7006**	**6970**	**0.52**
	80×8	11431	11420	6.38	**0.00**	10992	10893	2.3	**0.00**	**10825**	**10745**	**0.74**
	100×3	15175	15153	7.01	**0.00**	14815	14643	4.47	**0.00**	**14377**	**14181**	**1.39**
	100×5	57186	57182	6.18	**0.00**	53914	53888	0.11	0.13	**53894**	**53855**	**0.07**
	100×8	66775	66771	6.37	**0.00**	63007	62963	0.37	**0.00**	**62847**	**62776**	**0.11**
	120×3	31755	31742	4.69	**0.00**	31111	30692	2.56	**0.00**	**30580**	**30333**	**0.81**
	120×5	109270	109265	7.19	**0.00**	102307	102173	0.36	**0.00**	**102095**	**101937**	**0.16**
	120×8	61090	61051	6.47	**0.00**	58457	58169	1.89	**0.00**	**57622**	**57375**	**0.43**
	140×3	77293	77245	6.63	**0.00**	73721	72614	1.71	**0.00**	**72892**	**72484**	**0.56**
	140×5	84749	84709	6.18	**0.00**	80698	**79817**	1.1	0.23	**80506**	79899	**0.86**
	140×8	93690	93663	5.31	**0.00**	90399	89684	1.61	**0.00**	**89595**	**88968**	**0.7**
	160×3	222129	222123	7.86	**0.00**	206364	206080	0.21	0.93	**206351**	**205942**	**0.2**
	160×5	121554	121528	6.19	**0.00**	116352	114902	1.65	**0.00**	**115241**	**114466**	**0.68**
	160×8	265912	265899	6.52	**0.00**	250539	249807	0.36	0.51	**250365**	**249634**	**0.29**
	180×3	300394	300387	8.2	**0.00**	278762	277791	0.41	0.63	**278574**	**277620**	**0.34**
	180×5	162536	162508	5.61	**0.00**	157548	156625	2.37	**0.00**	**155817**	**153908**	**1.24**
	180×8	346908	346902	6.35	**0.00**	327758	**326184**	0.48	0.84	**327669**	326239	**0.46**
	200×3	388560	388549	8.98	**0.00**	**357765**	356620	**0.34**	0.87	357842	**356548**	0.36
	200×5	418192	418186	8.39	**0.00**	387976	386117	0.56	0.86	**387855**	**385805**	**0.53**
	200×8	446965	446962	6.84	**0.00**	**420265**	**418353**	**0.46**	0.88	420363	418358	0.48
	220×3	165021	164951	4.4	**0.00**	162330	160341	2.69	0.00	**161076**	**158070**	**1.9**
	220×5	256401	256363	5.61	**0.00**	**246604**	244386	**1.58**	0.83	246747	**242771**	1.64
	220×8	545756	545749	6.63	**0.00**	**514269**	**511827**	**0.48**	0.63	514636	512097	0.55
	240×3	587244	587237	9.15	**0.00**	**539952**	538244	**0.36**	0.96	539991	**538036**	**0.36**
	240×5	311582	311503	5.34	**0.00**	**299358**	**295778**	**1.21**	0.17	300398	296560	1.56
	240×8	660115	660107	7.41	**0.00**	**617737**	**614600**	**0.51**	0.78	618015	614721	0.56
	260×3	704782	704777	8.74	**0.00**	650702	648349	0.39	0.93	**650624**	**648162**	**0.38**
	260×5	751055	751048	8.59	**0.00**	695580	692008	0.57	0.74	**695183**	**691637**	**0.51**
	260×8	786828	786817	7.09	**0.00**	739041	735014	0.58	0.90	**738897**	**734750**	**0.56**
	280×3	834999	834994	9.01	**0.00**	769024	765989	0.4	0.81	**768771**	**765957**	**0.37**
	280×5	872658	872650	8.5	**0.00**	808675	804494	0.54	0.93	**808554**	**804314**	**0.53**
	280×8	919273	919265	7.01	**0.00**	863493	859107	0.52	0.88	**863300**	**859016**	**0.5**
	300×3	484274	484005	5.97	**0.00**	**462426**	**456989**	**1.19**	0.19	464181	458368	1.57
	300×5	1026763	1026756	8.98	**0.00**	**947301**	**942146**	**0.55**	0.93	947445	942254	0.56
	300×8	1066697	1066685	7.37	**0.00**	999793	994118	0.64	0.65	**999037**	**993470**	**0.56**
F=40	80×3	7034	7004	6.59	**0.00**	6764	6690	2.49	**0.00**	**6676**	**6599**	**1.17**
	80×5	17870	17830	6.29	**0.00**	17119	17049	1.83	**0.00**	**16920**	**16812**	**0.65**
	80×8	43187	43180	5.14	**0.00**	41287	41277	0.52	**0.00**	**41097**	**41075**	**0.05**

	100×3	24547	24496	7.63	**0.00**	23533	23299	3.19	**0.00**	**23094**	**22806**	**1.26**
	100×5	84779	84765	6.43	**0.00**	79877	79779	0.28	**0.00**	**79712**	**79658**	**0.07**
	100×8	94945	94930	5.38	**0.00**	90658	90522	0.62	**0.00**	**90240**	**90096**	**0.16**
	120×3	63286	63246	6.26	**0.00**	60218	59952	1.11	**0.00**	**59748**	**59556**	**0.32**
	120×5	139509	139496	6.44	**0.00**	131480	131370	0.31	**0.00**	**131224**	**131070**	**0.12**
	120×8	154494	154471	5.19	**0.00**	147442	147375	0.39	**0.00**	**147063**	**146868**	**0.13**
	140×3	94719	94648	6.57	**0.00**	91229	90184	2.64	**0.00**	**89447**	**88883**	**0.63**
	140×5	211116	211109	6.95	**0.00**	198289	198072	0.45	**0.00**	**197675**	**197392**	**0.14**
	140×8	224800	224791	5.87	**0.00**	213193	213040	0.4	**0.00**	**212712**	**212341**	**0.17**
	160×3	265976	265964	7.55	**0.00**	248132	247765	0.33	0.08	**247832**	**247306**	**0.21**
	160×5	287120	287106	6.94	**0.00**	**269089**	268617	**0.23**	0.89	269118	**268481**	0.24
	160×8	153306	153242	5.85	**0.00**	148400	147636	2.46	**0.00**	**146542**	**144833**	**1.18**
	180×3	347672	347662	8.12	**0.00**	322861	322111	0.41	0.17	**322422**	**321555**	**0.27**
	180×5	374189	374182	7.01	**0.00**	350945	350021	0.36	0.89	**350894**	**349674**	**0.35**
	180×8	394788	394778	5.79	**0.00**	375001	373657	0.48	0.24	**374484**	**373197**	**0.34**
	200×3	220300	220252	6.33	**0.00**	**209849**	**207176**	**1.29**	0.37	210361	207953	1.54
	200×5	156288	156199	2.6	**0.00**	156881	154069	2.99	**0.00**	**153469**	**152330**	**0.75**
	200×8	495140	495130	6.38	**0.00**	467331	465612	0.4	0.54	**466933**	**465464**	**0.32**
	220×3	135426	135275	2.38	**0.00**	135184	133199	2.19	**0.00**	**133813**	**132284**	**1.16**
	220×5	192551	192417	2.33	**0.00**	192258	191012	2.18	**0.01**	**191007**	**188164**	**1.51**
	220×8	608278	608267	6.12	**0.00**	576461	573483	0.57	0.61	**576001**	**573190**	**0.49**
	240×3	328153	328072	5.21	**0.00**	316598	313184	1.5	0.05	**315201**	**311917**	**1.05**
	240×5	693686	693677	7.61	**0.00**	648679	644847	0.62	0.81	**648382**	**644652**	**0.58**
	240×8	722626	722611	6.16	**0.00**	684586	680883	0.57	0.94	**684505**	**680705**	**0.56**
	260×3	260450	260286	3.01	**0.00**	263024	259131	4.03	**0.00**	**256400**	**252838**	**1.41**
	260×5	822004	821995	7.46	**0.00**	769222	**764916**	0.56	0.99	**769208**	764924	**0.56**
	260×8	856628	856618	6.91	**0.00**	806436	801574	0.64	0.93	**806305**	**801279**	**0.63**
	280×3	457123	457046	5.1	**0.00**	**440141**	437149	**1.19**	0.67	440524	**434945**	1.28
	280×5	959708	959694	8.46	**0.00**	**890928**	**884889**	**0.68**	0.91	891133	885490	0.71
	280×8	993217	993208	6.41	**0.00**	**938758**	**933348**	**0.58**	0.93	938892	933489	0.59
	300×3	515708	515674	5.08	**0.00**	**496316**	**490772**	**1.13**	0.33	497458	492160	1.36
	300×5	273676	273462	1.43	**0.00**	272660	270804	1.05	**0.00**	**271030**	**269819**	**0.45**
	300×8	1145416	1145405	7.12	**0.00**	**1075620**	1069316	**0.59**	0.93	1075773	**1069291**	0.61
	80×3	48172	48166	4.67	**0.00**	**46034**	**46024**	**0.02**	**0.00**	46073	46054	0.11
	80×5	56388	56379	5.19	**0.00**	**53680**	53679	0.14	**0.00**	53659	**53604**	**0.1**
	80×8	64473	64456	3.98	**0.00**	62139	62124	0.22	**0.00**	**62051**	**62006**	**0.07**
	100×3	100245	100240	6.17	**0.00**	94504	94485	0.09	**0.00**	**94460**	**94423**	**0.04**
	100×5	110900	110877	5.62	**0.00**	105199	105138	0.19	**0.00**	**105086**	**104997**	**0.08**
	100×8	121332	121308	4.43	**0.00**	116641	116614	0.4	**0.00**	**116310**	**116181**	**0.11**
	120×3	78339	78269	6.93	**0.00**	74086	73694	1.13	**0.00**	**73602**	**73261**	**0.47**
F=60	120×5	86048	85993	7.27	**0.00**	82017	81666	2.24	**0.00**	**80566**	**80217**	**0.43**
	120×8	185090	185084	5.62	**0.00**	175899	175694	0.38	**0.00**	**175447**	**175240**	**0.12**
	140×3	226706	226692	7.18	**0.00**	212296	212077	0.36	**0.00**	**211809**	**211528**	**0.13**
	140×5	245851	245827	6.08	**0.00**	232944	232682	0.51	**0.00**	**232257**	**231759**	**0.21**
	140×8	261318	261300	5.22	**0.00**	249426	249134	0.43	**0.00**	**248895**	**248356**	**0.22**
	160×3	306191	306183	7.82	**0.00**	284839	284479	0.3	0.05	**284507**	**283996**	**0.18**
	160×5	331205	331196	6.76	**0.00**	311142	310852	0.29	0.02	**310727**	**310242**	**0.16**
	160×8	349372	349353	5.72	**0.00**	331902	331498	0.43	0.01	**331240**	**330469**	**0.23**

180×3	397746	397740	8.08	**0.00**	369268	368609	0.34	0.30	**368941**	**367999**	**0.26**
180×5	209917	209713	5.67	**0.00**	201735	200024	1.55	0.19	**201062**	**198653**	**1.21**
180×8	223048	222883	3.98	**0.00**	217346	216287	1.32	**0.00**	**216062**	**214516**	**0.72**
200×3	491703	491699	7.5	**0.00**	458930	457635	0.33	0.65	**458689**	**457408**	**0.28**
200×5	520254	520226	6.91	**0.00**	488763	487094	0.43	0.43	**488296**	**486647**	**0.34**
200×8	549905	549895	6.14	**0.00**	520073	518467	0.38	0.99	**520065**	**518079**	**0.38**
220×3	603385	603376	8.25	**0.00**	559901	557876	0.45	0.55	**559406**	**557398**	**0.36**
220×5	637609	637587	7.05	**0.00**	**598452**	595886	0.48	0.91	598560	**595614**	0.49
220×8	665867	665841	5.75	**0.00**	**632336**	**629662**	**0.42**	0.91	632432	629924	0.44
240×3	354620	354488	5.92	**0.00**	345569	341423	3.22	**0.00**	**339599**	**334797**	**1.43**
240×5	375334	375230	5.68	**0.00**	**359181**	**355165**	**1.13**	0.05	361141	357828	1.68
240×8	397080	396975	3.41	**0.00**	393074	386326	2.37	**0.00**	**389238**	**383985**	**1.37**
260×3	834976	834967	8.58	**0.00**	774809	770032	0.75	0.10	**772400**	**769009**	**0.44**
260×5	893735	893720	7.32	**0.00**	839259	834056	0.78	0.18	**837262**	**832782**	**0.54**
260×8	934767	934763	6.41	**0.00**	884250	880050	0.66	0.12	**882335**	**878460**	**0.44**
280×3	486013	485933	4.87	**0.00**	469312	**463462**	1.26	0.47	**468501**	464161	**1.09**
280×5	1036616	1036594	7.91	**0.00**	**967587**	**960641**	**0.72**	0.94	967763	961375	0.74
280×8	1073307	1073295	6.03	**0.00**	**1018358**	**1012225**	**0.61**	0.66	1019143	1012769	0.68
300×3	555522	555384	5.07	**0.00**	540440	533969	2.22	**0.01**	**536813**	**528726**	**1.53**
300×5	1181436	1181418	7.87	**0.00**	**1102406**	1095572	**0.65**	0.87	1102769	**1095260**	0.69
300×8	609999	609808	3.01	**0.00**	604965	**592183**	2.16	**0.00**	**597482**	593013	**0.89**

4.4.5 NIG 算法的有效性

据作者所知，目前还没有针对 BHFGSP 设计的算法。因此，本章改进了用于求解 HFSP 的 IGA 算法[105]，以及用于求解 BHFSP 的 IGDLM 算法[106]，用作本章的对比算法。为了方便比较，所有算法均采用相同的编码和解码方法，然而，原始文献中的 IGA 和 IGDLM 算法都是针对工件序列设计的，如果将其直接应用于求解本章所研究的问题，将不可避免地打破工件不能跨组操作的限制。为此，本章将原文献中针对工件设计的所有策略都改为了针对组设计的策略，这样既保证了原算法的还原度，又很好地满足了成组的约束条件。随后，为了进一步验证上述基于阻塞工件的邻域搜索策略的性能并确保其算法的公平性，本节还将本章设计的阻塞工件邻域搜索策略嵌入到 IGA 和 IGDLM 中进行额外的实验对比。

通过该方式，本章对 IGA 和 IGDLM 进行了两组对比实验：1）保留了原始文献中的算法执行过程，并将工件策略转换为组策略。2）将基于阻塞工件的邻域搜索策略嵌入到 IGA 和 IGDLM 中，验证其性能，因为两种算法具有了针对阻塞约束设计的工件扰动策略，所以实验变得更加公平。为了获得最佳的算法性能，IGA 和 IGDLM 中的所有参数都是根据原文献设置的。 此外，为了观察算法在不同时间范围内的性能，本章还将终止条件 $\rho \times F \times S$ 设置为两个级别： ρ=100，200。最终得到四个统计

表，表 4-7 到 4-10，它们结合了不同的策略以及在不同的终止条件下执行，在表 4-7 到 4-10 中，使用了与表 4-6 相同的评价指标：平均值、最优值、RPD 和 *p* 值。

由表 4-7 可知，针对平均值、最优值和 RPD 值，IGA 分别获得了 17/108、11/108 和 22/108 的最小值， IGDLM 分别获得了 32/108、23/108、22/108 的最小值，NIG 获得了 59/108、75/108、54/108 的最小值，在 *p*-value 结果中，IGA、IGDLM 分别与 NIG 有 51 和 35 个具有显著性差异的结果。由表 4-9 可知，IGA 得到了 14/108、10/108、16/108 最小的平均值、最优值和 RPD 值，IGDLM 获得了 37/108、33/108、30/108 的最小值。NIG 获得了 67/108、67/108、67/108 的最小值，IGA 和 IGDLM 分别与 NIG 有 52 和 38 个显著性差异的结果。

从以上实验结果可知，NIG 在平均值、最优值、RPD 值的数量均多于其他两种比较算法。此外，NIG 具有最多的最好解，并且随着终止条件时间的延长，NIG 算法的整体性能也在提高。事实上，从 *p* 值可以看出，虽然 NIG 算法的性能优于 IGA 和 IGDLM，但解的差异性并不大，这不仅得益于 IG 系列算法强大的局部搜索能力，还得益于嵌入了基于阻塞工件的邻域搜索策略，从而有效地减少了工件序列的阻塞时间。

为了进一步验证所提出的阻塞工件邻域搜索策略的性能，本章在不同的终止条件下进行了对比实验，即，在 IGA 和 IGDLM 的代码实现中不使用本策略，如表 4-8 和表 4-10 所示，当 IGA 和 IGDLM 不使用该策略时，其性能将急剧下降。NIG 在所有指标中都达到了最佳值，并且在所有情况下的结果都与 IGA 和 IGDLM 有着显著性不同。由此可知，根据问题特性，即阻塞约束制定的策略，在算法的设计中起着非常重要的作用，该策略有效地提高了解决方案的多样性，其优点如下：一方面弥补了传统 IG 算法在全局搜索方面的不足，另一方面平衡了算法的整体局部和全局搜索能力。

为了更好地观察在不同情况下所有算法的收敛性，本节在不同规模的算例中，随机选取了 100×3×20、200×5×40 和 280×8×60 三个实例，并绘制了相应的收敛曲线图。在每次循环开始时，均记录一次在当前迭代过程中目前解的 makespan，并根据时间节点进行计数，例如在实例 100×3×20 中，将时间平均划分为 30 个子部分，并将它们作为 x 轴的刻度，y 轴表示目标函数值的范围，不同的算法由不同颜色的曲线表示，同样，为了检验有无邻域搜索策略算法的收敛性，本节绘制了两组不同的收敛曲线图。

如图 4-4 所示，（a-c）表示所有算法都使用本文所提出的邻域搜索策略，（d-f）表示 IGA 和 IGDLM 算法不使用该策略的情况。由（a-f）可知，所有算法都是收敛的，NIG 算法的收敛速度和寻找最优解的能力都优于 IGA 和 IGDLM 算法，然而，一旦 IGA 和 IGDLM 删除了阻塞工件邻域搜索策略，其搜索性能将会显著性下降，该实验同样证明了所提选择策略对算法的性能的影响，它通过改变同一组中阻塞工件的排列位置来缩短整个调度序列的完工时间，总之，从实验结果来看，针对阻塞约束设计的思想是合理有效的。

表 4-7 当ρ=100 时带有基于阻塞工件邻域搜索策略的对比结果

		IGA				**IGDLM**				**NIG**		
	N×S	MEAN	BEST	RPD	p-value	MEAN	BEST	RPD	p-value	MEAN	BEST	RPD
F=20	80×3	3361	3351	0.88	**0.00**	3348	3339	0.49	**0.00**	**3338**	**3332**	**0.18**
	80×5	7094	7039	1.79	**0.00**	7035	7005	0.93	**0.00**	**7006**	**6970**	**0.52**
	80×8	10933	10871	1.75	**0.00**	10869	**10813**	1.15	**0.00**	**10825**	**10745**	**0.74**
	100×3	14736	14570	3.91	**0.00**	14499	**14300**	2.24	**0.00**	**14377**	**14181**	**1.39**
	100×5	**53844**	**53827**	**0.03**	**0.00**	53923	53901	0.18	**0.01**	53894	53855	0.13
	100×8	63028	62976	0.47	**0.00**	**62810**	**62732**	**0.12**	0.17	62847	62776	0.18
	120×3	31262	30799	3.06	**0.00**	30897	30512	1.86	**0.00**	**30580**	**30333**	**0.81**
	120×5	102267	102192	0.32	**0.00**	102303	102247	0.36	**0.00**	**102095**	**101937**	**0.16**
	120×8	58658	58269	2.24	**0.00**	58185	57829	1.41	**0.00**	**57622**	**57375**	**0.43**
	140×3	73700	73062	1.77	**0.00**	**72889**	**72416**	**0.65**	0.97	72892	72484	0.66
	140×5	81145	80494	1.56	**0.00**	**80443**	**79896**	**0.68**	0.61	80506	79899	0.76
	140×8	90657	89639	1.9	**0.00**	89766	89095	0.9	0.18	**89595**	**88968**	**0.7**
	160×3	206400	206166	0.22	0.74	**206302**	205971	**0.18**	0.75	206351	**205942**	0.2
	160×5	116122	114812	1.45	**0.00**	115744	114636	1.12	0.05	**115241**	**114466**	**0.68**
	160×8	250596	249852	0.39	0.35	250505	249754	0.35	0.60	**250365**	**249634**	**0.29**
	180×3	278463	277646	0.3	0.78	**278264**	277651	**0.23**	0.32	278574	**277620**	0.34
	180×5	156875	155585	2.11	**0.01**	**155471**	**153635**	**1.2**	0.46	155817	153908	1.42
	180×8	**327195**	**325972**	**0.38**	0.26	327249	326197	0.39	0.31	327669	326239	0.52
	200×3	357992	**356693**	0.43	0.76	**357692**	356474	**0.34**	0.76	357842	356548	0.38
	200×5	387987	386211	0.57	0.85	**387846**	385836	**0.53**	0.99	387855	**385805**	**0.53**
	200×8	420383	418379	**0.48**	0.97	420406	418561	0.49	0.94	**420363**	**418358**	**0.48**
	220×3	162878	160883	3.04	**0.00**	162606	159779	2.87	**0.00**	**161076**	**158070**	**1.9**
	220×5	246430	244363	1.51	0.62	**246357**	243067	**1.48**	0.62	246747	**242771**	1.64
	220×8	514739	512242	0.57	0.90	**514384**	**511802**	**0.5**	0.75	514636	512097	0.55
	240×3	**539792**	**537989**	**0.34**	0.79	540151	538294	0.4	0.83	539991	538036	0.37
	240×5	300286	298308	1.55	0.87	**299173**	**295714**	**1.17**	0.16	300398	296560	1.58
	240×8	618051	**614534**	0.57	0.97	**617769**	614620	**0.53**	0.81	618015	614721	0.57
	260×3	**650542**	648208	**0.37**	0.92	650794	648528	0.41	0.85	650624	**648162**	0.38
	260×5	695511	691848	0.56	0.78	695308	691882	0.53	0.92	**695183**	**691637**	**0.51**
	260×8	738961	735279	0.57	0.96	739055	734863	0.59	0.90	**738897**	**734750**	**0.56**
	280×3	768973	766249	0.39	0.84	769061	766127	0.41	0.78	**768771**	**765957**	**0.37**
	280×5	809214	804742	0.61	0.64	808825	804507	0.56	0.85	**808554**	**804314**	**0.53**
	280×8	863582	859129	0.53	0.83	863682	859529	0.54	0.77	**863300**	**859016**	**0.5**
	300×3	**462863**	458371	**1.49**	0.30	463111	**456088**	1.54	0.44	464181	458368	1.77
	300×5	948002	942729	0.61	0.75	947670	942500	0.57	0.89	**947445**	**942254**	**0.55**
	300×8	999018	**993456**	**0.56**	0.99	999444	993741	0.6	0.81	**999037**	993470	**0.56**
F=40	80×3	6780	6641	2.74	**0.00**	6727	6604	1.94	**0.00**	**6676**	**6599**	**1.17**
	80×5	17122	17053	**1.84**	**0.00**	16994	16868	1.08	**0.00**	**16920**	**16812**	**0.65**
	80×8	41148	41130	0.18	**0.00**	41189	41152	0.28	**0.00**	**41097**	**41075**	**0.05**
	100×3	23495	23290	3.02	**0.00**	23183	22942	1.65	**0.00**	**23094**	**22806**	**1.26**
	100×5	79837	79771	0.23	**0.00**	79804	79789	0.18	**0.00**	**79712**	**79658**	**0.07**

	100×8	90455	90387	0.4	**0.00**	90419	90353	0.36	**0.00**	**90240**	**90096**	**0.16**
	120×3	60132	59857	0.97	**0.00**	59821	59599	0.44	**0.04**	**59748**	**59556**	**0.32**
	120×5	131439	131323	0.28	**0.00**	131321	131155	0.19	**0.05**	**131224**	**131070**	**0.12**
	120×8	147183	146962	0.21	**0.03**	147207	147084	0.23	**0.00**	**147063**	**146868**	**0.13**
	140×3	90996	90313	2.38	**0.00**	**89295**	88955	**0.46**	0.17	89447	**88883**	0.63
	140×5	197931	197784	0.27	**0.01**	198189	198074	0.4	**0.00**	**197675**	**197392**	**0.14**
	140×8	212916	212648	0.27	**0.05**	**212661**	212447	**0.15**	0.63	212712	**212341**	0.17
	160×3	**247671**	**247301**	0.15	0.32	247967	247570	0.27	0.39	247832	247306	0.21
	160×5	**268837**	**268371**	**0.17**	0.17	269182	268817	0.3	0.74	269118	268481	0.28
	160×8	148889	147542	2.8	**0.00**	**146195**	145353	**0.94**	0.23	146542	**144833**	1.18
	180×3	322449	321765	0.28	0.93	322471	321900	0.29	0.87	**322422**	**321555**	**0.27**
	180×5	**350696**	349777	**0.29**	0.59	350772	349977	0.31	0.73	350894	**349674**	0.35
	180×8	**374419**	373416	**0.33**	0.86	374694	373522	0.4	0.55	374484	**373197**	0.34
	200×3	212192	209163	**2.7**	**0.00**	**209800**	**206605**	**1.55**	0.36	210361	207953	1.82
	200×5	158690	154657	4.18	0.00	155042	153454	1.78	0.00	**153469**	**152330**	**0.75**
	200×8	466986	465910	0.33	0.92	467137	465608	0.36	0.72	**466933**	**465464**	**0.32**
	220×3	134315	133152	**1.54**	**0.00**	133864	132593	1.19	0.76	**133813**	**132284**	**1.16**
	220×5	195046	191619	3.66	**0.00**	191619	189022	1.84	0.27	**191007**	**188164**	**1.51**
	220×8	574455	**572295**	0.38	0.06	**574434**	572368	**0.37**	0.06	576001	573190	0.65
	240×3	316877	314004	**1.59**	**0.01**	316583	312268	1.5	0.14	**315201**	**311917**	**1.05**
	240×5	648484	644985	0.62	0.93	648461	**644486**	0.62	0.95	**648382**	644652	**0.6**
	240×8	685135	681330	0.65	0.56	684873	681331	0.61	0.73	**684505**	**680705**	**0.56**
	260×3	262950	257293	4	**0.00**	258610	255942	2.28	**0.00**	**256400**	**252838**	**1.41**
	260×5	769470	765346	0.59	0.85	**769136**	764995	**0.55**	0.96	769208	**764924**	0.56
	260×8	806615	801577	0.67	0.84	**806249**	801774	**0.62**	0.97	806305	**801279**	0.63
	280×3	440899	437590	1.37	0.65	442122	435644	1.65	0.18	**440524**	**434945**	**1.28**
	280×5	890818	885397	0.65	0.86	**890653**	**885043**	**0.63**	0.79	891133	885490	0.69
	280×8	**938827**	933533	**0.57**	0.97	938900	933741	0.58	1.00	938892	**933489**	0.58
	300×3	**496240**	490705	**1.34**	0.29	496289	**489655**	1.35	0.40	497458	492160	1.59
	300×5	273270	271104	1.28	**0.00**	272678	271077	1.06	**0.00**	**271030**	**269819**	**0.45**
	300×8	**1075634**	1069386	**0.59**	0.94	1076288	1069708	0.65	0.78	1075773	**1069291**	0.61
	80×3	**46069**	46054	**0.05**	0.23	46085	**46048**	0.08	0.04	46073	46054	**0.05**
	80×5	53740	53730	0.25	**0.00**	53700	53678	0.18	**0.00**	**53659**	**53604**	**0.1**
	80×8	62198	62114	0.31	**0.00**	62146	62121	0.23	**0.00**	**62051**	**62006**	**0.07**
	100×3	94493	94457	0.07	**0.00**	94641	94571	0.23	**0.00**	**94460**	**94423**	**0.04**
	100×5	105261	105224	0.25	**0.00**	105343	105281	0.33	**0.00**	**105086**	**104997**	**0.08**
	100×8	116480	116426	0.26	**0.00**	116545	116349	0.31	**0.00**	**116310**	**116181**	**0.11**
	120×3	73908	73453	0.88	**0.00**	**73586**	73280	**0.44**	0.74	73602	**73261**	0.47
	120×5	82049	81495	2.28	**0.00**	80926	80565	0.88	**0.00**	**80566**	**80217**	**0.43**
	120×8	175911	175845	0.38	**0.00**	175730	175675	0.28	**0.00**	**175447**	**175240**	**0.12**
	140×3	211994	211807	0.22	0.03	212169	212061	0.3	**0.00**	**211809**	**211528**	**0.13**
	140×5	232750	232410	0.43	**0.00**	232355	232129	0.26	0.37	**232257**	**231759**	**0.21**
	140×8	249360	248885	0.4	**0.00**	249056	248512	0.28	0.24	**248895**	**248356**	**0.22**
	160×3	284749	284341	0.27	0.12	**284444**	284143	**0.16**	0.68	284507	**283996**	0.18
	160×5	311498	311228	0.4	**0.00**	311044	310818	0.26	0.04	**310727**	**310242**	**0.16**
F=60	160×8	331602	331116	0.34	0.11	331606	331088	0.34	0.11	**331240**	**330469**	**0.23**
	180×3	**368597**	368109	**0.16**	0.24	369041	368620	0.28	0.72	368941	**367999**	0.26
	180×5	204461	202824	3.63	**0.00**	**199304**	**197294**	**1.02**	**0.00**	201062	198653	1.91
	180×8	217711	216204	2.58	**0.00**	**214759**	**212238**	**1.19**	**0.00**	216062	214516	1.8
	200×3	**458440**	457416	**0.23**	0.59	458630	457711	0.27	0.90	458689	**457408**	0.28
	200×5	488345	487032	0.35	0.93	**488229**	487129	**0.33**	0.90	488296	**486647**	0.34
	200×8	519651	517957	0.34	0.49	**519294**	**517899**	**0.27**	0.16	520065	518079	0.42
	220×3	559265	557410	0.33	0.86	**559137**	557532	**0.31**	0.73	559406	**557398**	0.36
	220×5	598030	595835	0.53	0.57	**597715**	**594851**	**0.48**	0.40	598560	595614	0.62
	220×8	632457	**629756**	0.43	0.98	632487	629913	0.43	0.94	**632432**	629924	**0.42**
	240×3	341807	338195	2.09	**0.02**	341381	337283	1.97	0.05	**339599**	**334797**	**1.43**
	240×5	362711	358357	2.35	0.09	**358338**	**354375**	**1.12**	**0.01**	361141	357828	1.91
	240×8	390546	384181	2.23	0.24	**385747**	**382032**	**0.97**	**0.00**	389238	383985	1.89
	260×3	775240	771059	0.81	**0.05**	775490	770569	0.84	**0.04**	**772400**	**769009**	**0.44**
	260×5	839085	834285	0.76	0.19	839078	834110	0.76	0.19	**837262**	**832782**	**0.54**
	260×8	884467	879893	0.68	0.08	886087	880486	0.87	**0.01**	**882335**	**878460**	**0.44**

	280×3	469686	465935	1.58	0.18	470092	**462381**	1.67	0.21	**468501**	464161	**1.32**
	280×5	**967441**	**960791**	**0.69**	0.88	967810	961035	0.73	0.98	967763	961375	0.73
	280×8	**1018316**	**1012227**	**0.6**	0.64	1018515	1012648	0.62	0.72	1019143	1012769	0.68
	300×3	540380	533007	2.2	**0.03**	539994	533509	2.13	0.02	**536813**	**528726**	**1.53**
	300×5	1103278	1095666	0.78	0.83	**1102466**	**1094752**	**0.7**	0.90	1102769	1095260	0.73
	300×8	601601	593393	1.73	**0.02**	**596162**	**591361**	**0.81**	0.23	597482	593013	1.03

表 4-8 当ρ=100 时不带有基于阻塞工件邻域搜索策略的对比结果

		IGA				**IGDLM**				**NIG**		
	N×S	MEAN	BEST	RPD	p-value	MEAN	BEST	RPD	p-value	MEAN	BEST	RPD
F=20	80×3	3592	3585	7.59	**0**	3590	3582	7.5	**0**	**3348**	**3339**	**0.28**
	80×5	7392	7345	5.53	**0**	7355	7335	5	**0**	**7035**	**7005**	**0.43**
	80×8	11538	11459	6.7	**0**	11451	11439	5.9	**0**	**10869**	**10813**	**0.52**
	100×3	15219	15201	6.42	**0**	15197	15157	6.27	**0**	**14499**	**14300**	**1.39**
	100×5	57201	57188	6.12	**0**	57197	57188	6.11	**0**	**53923**	**53901**	**0.04**
	100×8	66791	66783	6.47	**0**	66789	66779	6.47	**0**	**62810**	**62732**	**0.12**
	120×3	31834	31791	4.33	**0**	31793	31749	4.2	**0**	**30897**	**30512**	**1.26**
	120×5	109290	109275	6.89	**0**	109292	109282	6.89	**0**	**102303**	**102247**	**0.05**
	120×8	61195	61082	5.82	**0**	61081	61059	5.62	**0**	**58185**	**57829**	**0.62**
	140×3	77439	77343	6.94	**0**	77283	77259	6.72	**0**	**72889**	**72416**	**0.65**
	140×5	84883	84754	6.24	**0**	84770	84732	6.1	**0**	**80443**	**79896**	**0.68**
	140×8	93869	93686	5.36	**0**	93725	93675	5.2	**0**	**89766**	**89095**	**0.75**
	160×3	222153	222148	7.86	**0**	222138	222132	7.85	**0**	**206302**	**205971**	**0.16**
	160×5	121663	121551	6.13	**0**	121561	121531	6.04	**0**	**115744**	**114636**	**0.97**
	160×8	265960	265932	6.49	**0**	265940	265925	6.48	**0**	**250505**	**249754**	**0.3**
	180×3	300409	300404	8.2	**0**	300407	300397	8.2	**0**	**278264**	**277651**	**0.22**
	180×5	162820	162523	5.98	**0**	162590	162513	5.83	**0**	**155471**	**153635**	**1.2**
	180×8	346935	346914	6.36	**0**	346921	346913	6.35	**0**	**327249**	**326197**	**0.32**
	200×3	388570	388565	9	**0**	388572	388568	9	**0**	**357692**	**356474**	**0.34**
	200×5	418205	418193	8.39	**0**	418207	418196	8.39	**0**	**387846**	**385836**	**0.52**
	200×8	446983	446972	6.79	**0**	446963	446961	6.79	**0**	**420406**	**418561**	**0.44**
	220×3	165399	165117	3.52	**0**	165197	164976	3.39	**0**	**162606**	**159779**	**1.77**
	220×5	256638	256387	5.58	**0**	256402	256369	5.49	**0**	**246357**	**243067**	**1.35**
	220×8	545786	545768	6.64	**0**	545787	545760	6.64	**0**	**514384**	**511802**	**0.5**
	240×3	587250	587244	9.09	**0**	587251	587248	9.09	**0**	**540151**	**538294**	**0.34**
	240×5	312050	311566	5.52	**0**	311637	311523	5.38	**0**	**299173**	**295714**	**1.17**
	240×8	660151	660122	7.41	**0**	660132	660122	7.4	**0**	**617769**	**614620**	**0.51**
	260×3	704790	704785	8.68	**0**	704795	704782	8.68	**0**	**650794**	**648528**	**0.35**
	260×5	751061	751055	8.55	**0**	751070	751063	8.55	**0**	**695308**	**691882**	**0.5**
	260×8	786863	786831	7.08	**0**	786847	786837	7.07	**0**	**739055**	**734863**	**0.57**
	280×3	835004	834996	8.99	**0**	835003	835002	8.99	**0**	**769061**	**766127**	**0.38**
	280×5	872683	872654	8.47	**0**	872680	872658	8.47	**0**	**808825**	**804507**	**0.54**
	280×8	919293	919268	6.95	**0**	919287	919277	6.95	**0**	**863682**	**859529**	**0.48**
	300×3	484586	484023	6.25	**0**	484268	484017	6.18	**0**	**463111**	**456088**	**1.54**
	300×5	1026776	1026758	8.94	**0**	1026775	1026769	8.94	**0**	**947670**	**942500**	**0.55**
	300×8	1066707	1066697	7.34	**0**	1066718	1066696	7.34	**0**	**999444**	**993741**	**0.57**
F=40	80×3	7114	7085	7.72	**0**	7062	7022	6.93	**0**	**6727**	**6604**	**1.86**
	80×5	17931	17887	6.3	**0**	17902	17863	6.13	**0**	**16994**	**16868**	**0.74**
	80×8	43216	43210	5.02	**0**	43219	43207	5.02	**0**	**41189**	**41152**	**0.09**
	100×3	24575	24549	7.12	**0**	24562	24525	7.06	**0**	**23183**	**22942**	**1.05**
	100×5	84840	84804	6.33	**0**	84836	84826	6.33	**0**	**79804**	**79789**	**0.02**
	100×8	94991	94971	5.13	**0**	94998	94979	5.14	**0**	**90419**	**90353**	**0.07**
	120×3	63365	63322	6.32	**0**	63330	63297	6.26	**0**	**59821**	**59599**	**0.37**
	120×5	139556	139546	6.41	**0**	139557	139549	6.41	**0**	**131321**	**131155**	**0.13**
	120×8	154526	154505	5.06	**0**	154545	154528	5.07	**0**	**147207**	**147084**	**0.08**
	140×3	94803	94725	6.57	**0**	94718	94656	6.48	**0**	**89295**	**88955**	**0.38**
	140×5	211144	211114	6.6	**0**	211123	211120	6.59	**0**	**198189**	**198074**	**0.06**
	140×8	224811	224811	5.82	**0**	224806	224806	5.82	**0**	**212661**	**212447**	**0.1**

	160×3	266005	265993	7.45	**0**	265997	265980	7.44	**0**	**247967**	**247570**	**0.16**
	160×5	287159	287147	6.82	**0**	287168	287168	6.83	**0**	**269182**	**268817**	**0.14**
	160×8	153395	153354	5.53	**0**	153310	153262	5.47	**0**	**146195**	**145353**	**0.58**
	180×3	347714	347686	8.02	**0**	347694	347683	8.01	**0**	**322471**	**321900**	**0.18**
	180×5	374238	374237	6.93	**0**	374218	374200	6.93	**0**	**350772**	**349977**	**0.23**
	180×8	394847	394803	5.71	**0**	394818	394811	5.7	**0**	**374694**	**373522**	**0.31**
	200×3	220424	220294	6.69	**0**	220301	220227	6.63	**0**	**209800**	**206605**	**1.55**
	200×5	156858	156421	2.22	**0**	156396	156225	1.92	**0**	**155042**	**153454**	**1.04**
	200×8	495183	495166	6.35	**0**	495157	495142	6.35	**0**	**467137**	**465608**	**0.33**
	220×3	135777	135519	2.4	**0**	135487	135344	2.18	**0**	**133864**	**132593**	**0.96**
	220×5	193111	192665	2.16	**0**	192693	192481	1.94	**0**	**191619**	**189022**	**1.37**
	220×8	608345	608343	6.29	**0**	608301	608293	6.28	**0**	**574434**	**572368**	**0.36**
	240×3	328397	328166	5.17	**0**	328186	328106	5.1	**0**	**316583**	**312268**	**1.38**
	240×5	693720	693717	7.64	**0**	693712	693694	7.64	**0**	**648461**	**644486**	**0.62**
	240×8	722666	722660	6.07	**0**	722680	722663	6.07	**0**	**684873**	**681331**	**0.52**
	260×3	261617	260802	2.22	**0**	261046	260643	1.99	**0**	**258610**	**255942**	**1.04**
	260×5	822035	822033	7.46	**0**	822028	822017	7.46	**0**	**769136**	**764995**	**0.54**
	260×8	856642	856642	6.84	**0**	856636	856631	6.84	**0**	**806249**	**801774**	**0.56**
	280×3	457308	457084	4.97	**0**	457098	457064	4.92	**0**	**442122**	**435644**	**1.49**
	280×5	959763	959720	8.44	**0**	959733	959719	8.44	**0**	**890653**	**885043**	**0.63**
	280×8	993260	993225	6.37	**0**	993252	993237	6.37	**0**	**938900**	**933741**	**0.55**
	300×3	516014	515806	5.38	**0**	515748	515705	5.33	**0**	**496289**	**489655**	**1.35**
	300×5	274247	273864	1.17	**0**	273832	273512	1.02	**0**	**272678**	**271077**	**0.59**
	300×8	1145455	1145443	7.08	**0**	1145410	1145410	7.08	**0**	**1076288**	**1069708**	**0.62**
	80×3	48178	48178	4.63	**0**	48179	48179	4.63	**0**	**46085**	**46048**	**0.08**
	80×5	56421	56408	5.11	**0**	56414	56404	5.1	**0**	**53700**	**53678**	**0.04**
	80×8	64504	64500	3.84	**0**	64519	64519	3.86	**0**	**62146**	**62121**	**0.04**
	100×3	100252	100252	6.01	**0**	100256	100256	6.01	**0**	**94641**	**94571**	**0.07**
	100×5	110945	110921	5.38	**0**	110922	110922	5.36	**0**	**105343**	**105281**	**0.06**
	100×8	121403	121384	4.34	**0**	121388	121382	4.33	**0**	**116545**	**116349**	**0.17**
	120×3	78463	78341	7.07	**0**	78370	78302	6.95	**0**	**73586**	**73280**	**0.42**
	120×5	86175	86090	6.96	**0**	86098	86068	6.87	**0**	**80926**	**80565**	**0.45**
	120×8	185092	185086	5.36	**0**	185085	185085	5.36	**0**	**175730**	**175675**	**0.03**
	140×3	226729	226728	6.92	**0**	226770	226754	6.94	**0**	**212169**	**212061**	**0.05**
	140×5	245941	245916	5.95	**0**	245910	245900	5.94	**0**	**232355**	**232129**	**0.1**
	140×8	261412	261390	5.19	**0**	261386	261381	5.18	**0**	**249056**	**248512**	**0.22**
	160×3	306229	306229	7.77	**0**	306218	306205	7.77	**0**	**284444**	**284143**	**0.11**
	160×5	331202	331198	6.56	**0**	331194	331194	6.56	**0**	**311044**	**310818**	**0.07**
	160×8	349401	349398	5.53	**0**	349436	349426	5.54	**0**	**331606**	**331088**	**0.16**
	180×3	397757	397752	7.9	**0**	397739	397739	7.9	**0**	**369041**	**368620**	**0.11**
	180×5	210176	210062	6.53	**0**	209978	209920	6.43	**0**	**199304**	**197294**	**1.02**
	180×8	223213	223168	5.17	**0**	223127	223090	5.13	**0**	**214759**	**212238**	**1.19**
F=60	200×3	491731	491730	7.43	**0**	491732	491722	7.43	**0**	**458630**	**457711**	**0.2**
	200×5	520271	520267	6.8	**0**	520297	520283	6.81	**0**	**488229**	**487129**	**0.23**
	200×8	549925	549924	6.18	**0**	549933	549923	6.19	**0**	**519294**	**517899**	**0.27**
	220×3	603376	603376	8.22	**0**	603381	603381	8.22	**0**	**559137**	**557532**	**0.29**
	220×5	637692	637692	7.2	**0**	637702	637685	7.2	**0**	**597715**	**594851**	**0.48**
	220×8	665930	665928	5.72	**0**	665945	665934	5.72	**0**	**632487**	**629913**	**0.41**
	240×3	354839	354749	5.21	**0**	354699	354544	5.16	**0**	**341381**	**337283**	**1.22**
	240×5	375595	375432	5.99	**0**	375404	375319	5.93	**0**	**358338**	**354375**	**1.12**
	240×8	397229	397135	3.98	**0**	397124	397036	3.95	**0**	**385747**	**382032**	**0.97**
	260×3	835001	834967	8.36	**0**	835001	834994	8.36	**0**	**775490**	**770569**	**0.64**
	260×5	893764	893742	7.15	**0**	893751	893742	7.15	**0**	**839078**	**834110**	**0.6**
	260×8	934783	934783	6.17	**0**	934774	934774	6.17	**0**	**886087**	**880486**	**0.64**
	280×3	486222	486167	5.16	**0**	486077	485982	5.12	**0**	**470092**	**462381**	**1.67**
	280×5	1036647	1036621	7.87	**0**	1036665	1036643	7.87	**0**	**967810**	**961035**	**0.7**
	280×8	1073346	1073303	5.99	**0**	1073334	1073323	5.99	**0**	**1018515**	**1012648**	**0.58**
	300×3	555735	555623	4.17	**0**	555569	555457	4.13	**0**	**539994**	**533509**	**1.22**
	300×5	1181457	1181431	7.92	**0**	1181487	1181460	7.92	**0**	**1102466**	**1094752**	**0.7**
	300×8	610159	610027	3.18	**0**	610063	609842	3.16	**0**	**596162**	**591361**	**0.81**

表 4-9 当ρ=200 时带有基于阻塞工件邻域搜索策略的对比结果

		IGA				IGDLM				NIG		
	N×S	MEAN	BEST	RPD	p-value	MEAN	BEST	RPD	p-value	MEAN	BEST	RPD
F=20	80×3	3357	3344	0.91	**0.00**	3347	3340	0.61	**0.00**	**3337**	**3327**	**0.29**
	80×5	7081	7021	1.38	**0.00**	7039	7007	0.78	**0.00**	**7008**	**6984**	**0.35**
	80×8	10904	10842	1.67	**0.00**	10870	10802	1.36	**0.00**	**10821**	**10725**	**0.89**
	100×3	14663	14466	3.57	**0.00**	14478	14369	2.27	**0.00**	**14342**	**14157**	**1.3**
	100×5	53864	53840	0.1	**0.00**	**53841**	**53809**	**0.06**	**0.00**	53909	53881	0.19
	100×8	63062	63027	0.55	**0.00**	**62768**	**62715**	**0.08**	**0.00**	62901	62877	0.3
	120×3	31250	30876	3.51	**0.00**	30813	30547	2.06	**0.00**	**30503**	**30190**	**1.04**
	120×5	102229	102171	0.21	**0.00**	102278	102209	0.26	**0.00**	**102086**	**102014**	**0.07**
	120×8	58542	58310	1.66	**0.00**	57942	57661	0.61	0.05	**57859**	**57588**	**0.47**
	140×3	73177	72634	2.04	**0.00**	72679	72329	1.35	0.09	**72546**	**71711**	**1.16**
	140×5	80740	80311	1.35	**0.00**	80432	79918	0.96	**0.01**	**80113**	**79664**	**0.56**
	140×8	89782	**88510**	1.44	**0.02**	89463	88828	1.08	0.36	**89368**	88819	**0.97**
	160×3	206119	205877	0.13	0.34	**206048**	**205846**	**0.1**	0.11	206217	205885	0.18
	160×5	115259	113745	2.05	**0.00**	114604	112983	1.47	0.22	**114197**	**112940**	**1.11**
	160×8	250096	249600	0.28	0.70	**250053**	**249390**	**0.27**	0.58	250171	249656	0.31
	180×3	278135	277645	0.28	0.67	278093	277609	0.26	0.82	**278035**	**277360**	**0.24**
	180×5	155823	153719	2.82	**0.01**	**153993**	**151554**	**1.61**	0.28	154588	152233	2
	180×8	326773	325939	0.33	0.69	326791	325886	0.34	0.66	**326647**	**325700**	**0.29**
	200×3	**357357**	356557	**0.22**	0.83	358045	356900	0.42	0.18	357437	**356555**	0.25
	200×5	387913	386037	0.51	0.90	**387895**	**385956**	**0.5**	0.88	387998	386063	0.53
	200×8	420454	418527	0.51	0.96	420516	418490	0.52	0.88	**420420**	**418337**	**0.5**
	220×3	161978	160338	2.69	**0.00**	161569	159727	2.43	**0.01**	**160589**	**157728**	**1.81**
	220×5	**245253**	242413	1.35	**0.05**	246245	**241981**	**1.76**	0.53	246735	243252	1.96
	220×8	514629	**512059**	0.5	0.94	514751	512165	0.53	0.82	**514572**	512090	**0.49**
	240×3	**539807**	538060	**0.33**	0.95	539870	538029	0.35	0.98	539856	**538005**	0.34
	240×5	299263	296490	1.41	0.99	299996	**295112**	1.66	0.39	**299257**	296828	**1.4**
	240×8	617828	614662	0.53	1.00	**617634**	614677	**0.5**	0.84	617827	**614558**	0.53
	260×3	650604	648376	0.39	0.80	650726	648518	0.41	0.70	**650389**	**648077**	**0.36**
	260×5	695503	691763	0.57	0.92	**695127**	**691529**	**0.52**	0.68	695623	691859	0.59
	260×8	739148	735310	0.58	0.81	739004	735096	0.56	0.91	**738867**	**734864**	**0.54**
	280×3	768906	766162	0.39	0.99	**768698**	766288	**0.36**	0.84	768898	**765910**	0.39
	280×5	808849	804732	**0.53**	0.99	808992	804888	0.55	0.90	**808832**	**804594**	**0.53**
	280×8	863658	859595	0.59	0.73	863715	859679	0.59	0.69	**863212**	**858619**	**0.53**
	300×3	462942	459392	2.18	0.91	**460560**	**453052**	**1.66**	0.07	463060	457415	2.21
	300×5	947732	942147	0.59	0.96	**947388**	**942143**	**0.56**	0.88	947647	942325	0.58
	300×8	999194	993805	**0.58**	0.98	999644	994044	0.63	0.77	**999149**	**993401**	**0.58**
F=40	80×3	6744	6549	4.65	**0.00**	6716	6634	4.23	**0.00**	**6664**	**6444**	**3.42**
	80×5	17115	17060	1.87	**0.00**	16975	16867	1.04	**0.00**	**16895**	**16801**	**0.56**
	80×8	**41076**	**41061**	**0.04**	**0.00**	41175	41171	0.28	**0.00**	41249	41243	0.46
	100×3	23388	23172	3.68	**0.00**	23077	22773	2.3	**0.01**	**22975**	**22559**	**1.84**
	100×5	79855	79772	0.27	**0.00**	**79700**	79670	**0.07**	0.98	79701	**79643**	**0.07**
	100×8	90338	90306	0.2	**0.00**	90497	90435	0.38	**0.00**	**90243**	**90154**	**0.1**
	120×3	60067	59667	1.41	**0.00**	59739	59360	0.86	**0.00**	**59593**	**59232**	**0.61**
	120×5	131346	131255	0.32	**0.00**	131518	131411	0.45	**0.00**	**131170**	**130933**	**0.18**

	120×8	147290	147183	0.4	**0.00**	147240	147135	0.37	**0.00**	**146891**	**146702**	**0.13**
	140×3	90583	89830	2.33	**0.00**	**89051**	**88519**	**0.6**	0.29	89144	88741	0.71
	140×5	197885	197798	0.22	**0.01**	**197554**	**197455**	**0.05**	**0.01**	197724	197515	0.14
	140×8	212905	212791	0.18	**0.01**	**212722**	212586	**0.09**	0.82	212736	**212527**	0.1
	160×3	247725	247501	0.2	0.98	**247565**	**247236**	**0.13**	0.20	247722	247355	0.2
	160×5	268966	268628	0.4	**0.00**	269019	268620	0.42	**0.00**	**268489**	**267897**	**0.22**
	160×8	148746	147457	2.99	**0.00**	**145567**	**144434**	**0.78**	0.75	145649	145060	0.84
	180×3	322183	321697	0.29	0.12	322276	321836	0.32	**0.03**	**321841**	**321251**	**0.18**
	180×5	**350107**	349661	**0.21**	0.21	350123	**349382**	**0.21**	0.30	350422	349565	0.3
	180×8	374379	373795	0.29	0.41	374234	**373315**	0.25	0.84	374171	**373317**	**0.23**
	200×3	212014	210223	3.23	**0.00**	**207481**	**205380**	**1.02**	**0.02**	208952	205717	1.74
	200×5	159128	155510	4.21	**0.00**	154543	153167	1.2	**0.02**	**153702**	**152705**	**0.65**
	200×8	466128	464902	0.3	0.78	466330	465284	0.34	0.47	**466000**	**464735**	**0.27**
	220×3	134254	132683	1.73	**0.00**	133775	132495	1.37	**0.00**	**133154**	**131965**	**0.9**
	220×5	194600	190278	3.44	**0.00**	191451	189134	1.76	**0.00**	**189313**	**188137**	**0.62**
	220×8	574493	572061	0.44	0.85	**574027**	**571956**	**0.36**	0.40	574628	572180	0.47
	240×3	315759	312914	2.81	**0.00**	317366	313117	3.34	**0.00**	**311465**	**307120**	**1.41**
	240×5	647163	644010	0.68	0.13	647520	644326	0.73	0.06	**645581**	**642809**	**0.43**
	240×8	684740	681182	0.69	0.21	685024	681269	0.74	0.13	**683360**	**680025**	**0.49**
	260×3	259972	256395	3.42	**0.00**	255794	252058	1.76	0.93	**255729**	**251379**	**1.73**
	260×5	769690	765286	0.66	0.73	769375	764792	0.62	0.91	**769212**	**764635**	**0.6**
	260×8	806708	**801924**	0.6	0.84	806676	801992	0.59	0.86	**806421**	801955	**0.56**
	280×3	**439274**	**433492**	**1.33**	0.06	442660	437892	2.11	0.56	441923	434927	1.94
	280×5	**890456**	885171	**0.62**	0.82	890735	885058	0.66	0.95	890846	**884935**	0.67
	280×8	**938699**	**933591**	**0.55**	0.76	939208	933942	0.6	0.98	939172	933673	0.6
	300×3	**495556**	**489499**	**1.24**	**0.01**	497364	492065	1.61	0.13	499559	493921	2.06
	300×5	274124	270773	1.76	**0.00**	271793	270035	0.89	0.28	**271437**	**269388**	**0.76**
	300×8	1075543	1069100	**0.62**	1.00	1075763	1069573	0.64	0.91	**1075536**	**1068901**	**0.62**
	80×3	46003	45989	0.12	**0.00**	**45959**	**45946**	**0.03**	**0.00**	46018	46005	0.16
	80×5	53785	53771	0.29	**0.00**	53760	53732	0.25	**0.00**	**53634**	**53627**	**0.01**
	80×8	62150	62113	0.48	**0.00**	62049	61992	0.32	**0.00**	**61903**	**61850**	**0.09**
	100×3	94552	94501	0.15	**0.00**	94600	94560	0.2	**0.00**	**94445**	**94407**	**0.04**
	100×5	105239	105209	0.38	**0.00**	105260	105242	0.4	**0.00**	**104941**	**104842**	**0.09**
	100×8	116526	116443	0.41	**0.00**	116387	116308	0.29	**0.00**	**116152**	**116052**	**0.09**
	120×3	73731	73434	1.15	**0.00**	73398	**72896**	0.69	0.24	**73333**	73001	**0.6**
	120×5	81228	80763	1.04	**0.00**	80822	**80388**	**0.54**	0.68	**80841**	80618	0.56
	120×8	175774	175745	0.26	**0.00**	175646	175551	0.18	**0.00**	**175428**	**175327**	**0.06**
	140×3	212118	212000	0.26	**0.00**	211965	211781	0.19	**0.02**	**211819**	**211566**	**0.12**
F=60	140×5	232494	232322	0.25	**0.00**	**232170**	231981	**0.11**	0.62	232212	**231919**	0.13
	140×8	249152	249053	0.36	**0.00**	248859	248569	0.24	**0.05**	**248676**	**248256**	**0.17**
	160×3	284613	**284234**	0.13	0.69	284735	284577	0.18	0.10	**284566**	284245	**0.12**
	160×5	310764	310497	0.19	0.01	310796	310564	0.2	**0.00**	**310475**	**310173**	**0.1**
	160×8	331570	331230	0.4	**0.00**	330971	330446	0.22	0.69	**330897**	**330242**	**0.2**
	180×3	368783	368384	0.23	0.39	368662	368394	0.2	0.67	**368569**	**367922**	**0.18**
	180×5	200621	198808	1.86	0.44	**198625**	**196950**	**0.85**	**0.00**	200297	198152	1.7
	180×8	216087	214455	1.84	**0.00**	**214012**	**212181**	**0.86**	**0.03**	214925	213034	1.29
	200×3	458374	457722	0.32	0.46	458474	457675	0.34	0.31	**458087**	**456899**	**0.26**
	200×5	487840	486715	0.28	0.70	488099	487177	0.33	0.33	**487665**	**486484**	**0.24**

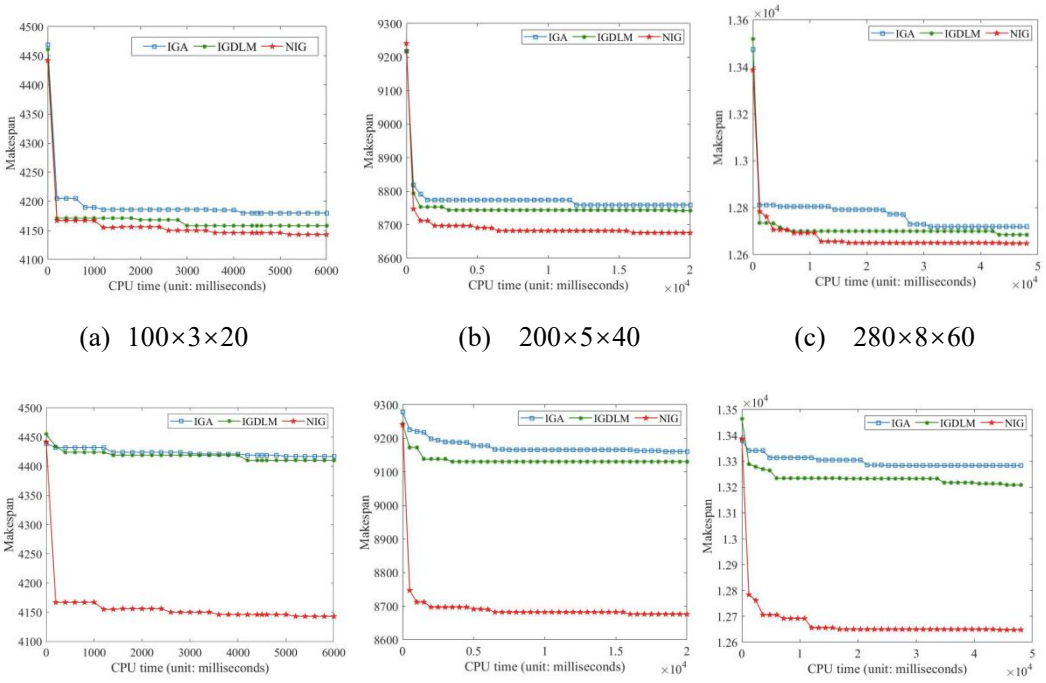
	200×8	519349	518151	0.43	0.57	**518658**	**517128**	**0.3**	0.35	519101	517976	0.38
	220×3	558878	557385	0.38	0.39	559075	557549	0.41	0.24	**558345**	**556784**	**0.28**
	220×5	**597393**	595258	**0.37**	0.80	597514	595351	0.39	0.91	597604	**595174**	0.41
	220×8	**630763**	628797	**0.33**	0.59	630839	**628685**	0.34	0.67	631150	628825	0.39
	240×3	342002	338476	1.54	0.02	340478	**336802**	1.09	0.62	**340057**	337153	**0.97**
	240×5	360269	356867	1.35	0.94	361793	**355485**	1.77	0.12	**360201**	356387	**1.33**
	240×8	385465	383019	0.88	0.35	385846	382158	0.98	0.20	**384879**	**382113**	**0.72**
	260×3	772628	769676	0.46	0.81	772420	769330	0.43	0.95	**772343**	**769113**	**0.42**
	260×5	**835261**	**831430**	**0.46**	0.89	836213	832642	0.58	0.51	835419	831977	0.48
	260×8	883028	879507	0.57	0.49	**882178**	**878036**	**0.47**	0.99	882191	878447	**0.47**
	280×3	466939	461342	2.96	0.37	**464234**	**453503**	**2.37**	0.03	468204	461322	3.24
	280×5	**963544**	958831	**0.51**	0.95	963743	959291	0.53	0.94	963635	**958677**	0.52
	280×8	1014427	**1009742**	0.46	0.83	1014722	**1009986**	0.49	0.67	1014122	1009934	**0.43**
	300×3	535822	532042	**1.01**	0.67	540251	**533739**	1.85	0.00	536377	**530446**	1.12
	300×5	1103136	1095267	0.72	0.79	**1102507**	**1095239**	**0.66**	0.59	1103767	1095645	0.78
	300×8	**595564**	592373	**0.7**	0.25	597433	593146	1.02	0.44	596674	**591426**	0.89

表 4-10 当ρ=200 时不带有基于阻塞工件邻域搜索策略的对比结果

		IGA				**IGDLM**				**NIG**		
	N×S	MEAN	BEST	RPD	p-value	MEAN	BEST	RPD	p-value	MEAN	BEST	RPD
	80×3	3592	3584	7.95	**0**	3589	3581	7.87	**0**	**3337**	**3327**	**0.29**
	80×5	7393	7340	5.85	**0**	7349	7337	5.23	**0**	**7008**	**6984**	**0.35**
	80×8	11508	11451	7.3	**0**	11448	11437	6.74	**0**	**10821**	**10725**	**0.89**
	100×3	15256	15209	7.76	**0**	15186	15156	7.27	**0**	**14342**	**14157**	**1.3**
	100×5	57196	57191	6.15	**0**	57195	57189	6.15	**0**	**53909**	**53881**	**0.05**
	100×8	66789	66785	6.22	**0**	66788	66776	6.22	**0**	**62901**	**62877**	**0.04**
	120×3	31824	31772	5.41	**0**	31781	31746	5.27	**0**	**30503**	**30190**	**1.04**
	120×5	109287	109282	7.13	**0**	109288	109278	7.13	**0**	**102086**	**102014**	**0.07**
	120×8	61156	61145	6.2	**0**	61066	61057	6.04	**0**	**57859**	**57588**	**0.47**
	140×3	77463	77390	8.02	**0**	77268	77259	7.75	**0**	**72546**	**71711**	**1.16**
	140×5	84851	84750	6.51	**0**	84755	84723	6.39	**0**	**80113**	**79664**	**0.56**
	140×8	93899	93837	5.72	**0**	93699	93671	5.49	**0**	**89368**	**88819**	**0.62**
	160×3	222158	222155	7.9	**0**	222137	222131	7.89	**0**	**206217**	**205885**	**0.16**
	160×5	121676	121581	7.74	**0**	121539	121532	7.61	**0**	**114197**	**112940**	**1.11**
	160×8	265944	265939	6.52	**0**	265932	265912	6.52	**0**	**250171**	**249656**	**0.21**
	180×3	300412	300402	8.31	**0**	300403	300397	8.31	**0**	**278035**	**277360**	**0.24**
	180×5	162857	162783	6.98	**0**	162532	162513	6.77	**0**	**154588**	**152233**	**1.55**
F=20	180×8	346961	346925	6.53	**0**	346919	346908	6.51	**0**	**326647**	**325700**	**0.29**
	200×3	388561	388554	8.98	**0**	388572	388569	8.98	**0**	**357437**	**356555**	**0.25**
	200×5	418203	418194	8.33	**0**	418205	418195	8.33	**0**	**387998**	**386063**	**0.5**
	200×8	446972	446964	6.84	**0**	446982	446973	6.85	**0**	**420420**	**418337**	**0.5**
	220×3	165527	165044	4.94	**0**	165181	164994	4.73	**0**	**160589**	**157728**	**1.81**
	220×5	256674	256637	5.52	**0**	256398	256371	5.4	**0**	**246735**	**243252**	**1.43**
	220×8	545782	545769	6.58	**0**	545781	545762	6.58	**0**	**514572**	**512090**	**0.48**
	240×3	587246	587242	9.15	**0**	587250	587245	9.15	**0**	**539856**	**538005**	**0.34**
	240×5	311994	311538	5.11	**0**	311601	311525	4.98	**0**	**299257**	**296828**	**0.82**
	240×8	660122	660115	7.41	**0**	660131	660119	7.42	**0**	**617827**	**614558**	**0.53**
	260×3	704788	704781	8.75	**0**	704795	704788	8.75	**0**	**650389**	**648077**	**0.36**
	260×5	751073	751060	8.56	**0**	751065	751059	8.56	**0**	**695623**	**691859**	**0.54**
	260×8	786885	786882	7.08	**0**	786838	786832	7.07	**0**	**738867**	**734864**	**0.54**
	280×3	835010	835002	9.02	**0**	835001	834997	9.02	**0**	**768898**	**765910**	**0.39**
	280×5	872676	872675	8.46	**0**	872675	872659	8.46	**0**	**808832**	**804594**	**0.53**
	280×8	919283	919276	7.07	**0**	919278	919276	7.06	**0**	**863212**	**858619**	**0.53**
	300×3	484583	484013	5.94	**0**	484150	484006	5.84	**0**	**463060**	**457415**	**1.23**
	300×5	1026774	1026768	8.96	**0**	1026774	1026763	8.96	**0**	**947647**	**942325**	**0.56**

	300×8	1066735	1066729	7.38	**0**	1066726	1066693	7.38	**0**	**999149**	**993401**	**0.58**
F=40	80×3	7083	7071	9.92	**0**	7058	7037	9.53	**0**	**6664**	**6444**	**3.42**
	80×5	17976	17900	6.99	**0**	17896	17872	6.52	**0**	**16895**	**16801**	**0.56**
	80×8	43225	43209	4.8	**0**	43216	43205	4.78	**0**	**41249**	**41243**	**0.02**
	100×3	24631	24580	9.18	**0**	24548	24514	8.82	**0**	**22975**	**22559**	**1.84**
	100×5	84846	84834	6.53	**0**	84824	84810	6.51	**0**	**79701**	**79643**	**0.07**
	100×8	94970	94960	5.34	**0**	94983	94971	5.36	**0**	**90243**	**90154**	**0.1**
	120×3	63392	63334	7.02	**0**	63314	63286	6.89	**0**	**59593**	**59232**	**0.61**
	120×5	139541	139541	6.57	**0**	139539	139534	6.57	**0**	**131170**	**130933**	**0.18**
	120×8	154539	154527	5.34	**0**	154505	154497	5.32	**0**	**146891**	**146702**	**0.13**
	140×3	94761	94736	6.78	**0**	94697	94650	6.71	**0**	**89144**	**88741**	**0.45**
	140×5	211142	211139	6.9	**0**	211140	211121	6.9	**0**	**197724**	**197515**	**0.11**
	140×8	224844	224844	5.8	**0**	224825	224812	5.79	**0**	**212736**	**212527**	**0.1**
	160×3	266007	266001	7.54	**0**	265995	265983	7.54	**0**	**247722**	**247355**	**0.15**
	160×5	287155	287143	7.19	**0**	287141	287135	7.18	**0**	**268489**	**267897**	**0.22**
	160×8	153413	153335	5.76	**0**	153300	153256	5.68	**0**	**145649**	**145060**	**0.41**
	180×3	347696	347694	8.23	**0**	347688	347676	8.23	**0**	**321841**	**321251**	**0.18**
	180×5	374207	374207	7.05	**0**	374197	374190	7.05	**0**	**350422**	**349565**	**0.25**
	180×8	394852	394823	5.77	**0**	394833	394808	5.76	**0**	**374171**	**373317**	**0.23**
	200×3	220420	220222	7.15	**0**	220271	220219	7.07	**0**	**208952**	**205717**	**1.57**
	200×5	156644	156309	2.58	**0**	156372	156225	2.4	**0**	**153702**	**152705**	**0.65**
	200×8	495150	495150	6.54	**0**	495160	495149	6.55	**0**	**466000**	**464735**	**0.27**
	220×3	135716	135569	2.84	**0**	135426	135288	2.62	**0**	**133154**	**131965**	**0.9**
	220×5	193008	192736	2.59	**0**	192680	192442	2.41	**0**	**189313**	**188137**	**0.62**
	220×8	608285	608281	6.31	**0**	608299	608286	6.31	**0**	**574628**	**572180**	**0.43**
	240×3	328374	328267	6.92	**0**	328154	328086	6.85	**0**	**311465**	**307120**	**1.41**
	240×5	693721	693710	7.92	**0**	693701	693673	7.92	**0**	**645581**	**642809**	**0.43**
	240×8	722682	722674	6.27	**0**	722664	722632	6.27	**0**	**683360**	**680025**	**0.49**
	260×3	261605	261234	4.07	**0**	260955	260412	3.81	**0**	**255729**	**251379**	**1.73**
	260×5	822037	822022	7.51	**0**	822022	822004	7.51	**0**	**769212**	**764635**	**0.6**
	260×8	856648	856641	6.82	**0**	856633	856613	6.82	**0**	**806421**	**801955**	**0.56**
	280×3	457279	457124	5.14	**0**	457108	457058	5.1	**0**	**441923**	**434927**	**1.61**
	280×5	959782	959755	8.46	**0**	959729	959717	8.45	**0**	**890846**	**884935**	**0.67**
	280×8	993229	993228	6.38	**0**	993235	993215	6.38	**0**	**939172**	**933673**	**0.59**
	300×3	516011	515732	4.47	**0**	515735	515701	4.42	**0**	**499559**	**493921**	**1.14**
	300×5	274235	274044	1.8	**0**	273689	273494	1.6	**0**	**271437**	**269388**	**0.76**
	300×8	1145480	1145473	7.16	**0**	1145435	1145420	7.16	**0**	**1075536**	**1068901**	**0.62**
F=60	80×3	48191	48180	4.75	**0**	48194	48181	4.76	**0**	**46018**	**46005**	**0.03**
	80×5	56442	56419	5.25	**0**	56411	56405	5.19	**0**	**53634**	**53627**	**0.01**
	80×8	64519	64516	4.31	**0**	64519	64516	4.32	**0**	**61903**	**61850**	**0.09**
	100×3	100261	100256	6.2	**0**	100258	100256	6.2	**0**	**94445**	**94407**	**0.04**
	100×5	110962	110949	5.84	**0**	110934	110931	5.81	**0**	**104941**	**104842**	**0.09**
	100×8	121396	121392	4.6	**0**	121396	121367	4.61	**0**	**116152**	**116052**	**0.09**
	120×3	78428	78344	7.43	**0**	78369	78309	7.35	**0**	**73333**	**73001**	**0.46**
	120×5	86163	86118	6.88	**0**	86086	86031	6.78	**0**	**80841**	**80618**	**0.28**
	120×8	185095	185091	5.57	**0**	185092	185092	5.57	**0**	**175428**	**175327**	**0.06**
	140×3	226741	226712	7.17	**0**	226748	226745	7.18	**0**	**211819**	**211566**	**0.12**
	140×5	245931	245895	6.04	**0**	245934	245928	6.04	**0**	**232212**	**231919**	**0.13**
	140×8	261427	261372	5.31	**0**	261358	261358	5.28	**0**	**248676**	**248256**	**0.17**
	160×3	306226	306212	7.73	**0**	306207	306197	7.73	**0**	**284566**	**284245**	**0.11**
	160×5	331207	331207	6.78	**0**	331205	331205	6.78	**0**	**310475**	**310173**	**0.1**
	160×8	349414	349406	5.81	**0**	349413	349404	5.81	**0**	**330897**	**330242**	**0.2**
	180×3	397747	397747	8.11	**0**	397744	397744	8.11	**0**	**368569**	**367922**	**0.18**
	180×5	210163	210050	6.06	**0**	209971	209928	5.96	**0**	**200297**	**198152**	**1.08**
	180×8	223221	223113	4.78	**0**	223116	223029	4.73	**0**	**214925**	**213034**	**0.89**
	200×3	491726	491713	7.62	**0**	491715	491715	7.62	**0**	**458087**	**456899**	**0.26**
	200×5	520296	520257	6.95	**0**	520281	520274	6.95	**0**	**487665**	**486484**	**0.24**
	200×8	549937	549910	6.17	**0**	549920	549912	6.17	**0**	**519101**	**517976**	**0.22**
	220×3	603383	603375	8.37	**0**	603388	603388	8.37	**0**	**558345**	**556784**	**0.28**
	220×5	637684	637640	7.14	**0**	637700	637667	7.15	**0**	**597604**	**595174**	**0.41**
	220×8	665941	665903	5.9	**0**	665961	665937	5.91	**0**	**631150**	**628825**	**0.37**
	240×3	354785	354612	5.23	**0**	354620	354524	5.18	**0**	**340057**	**337153**	**0.86**

240×5	375538	375449	5.37	**0**	375378	375314	5.33	**0**	**360201**	**356387**	**1.07**
240×8	397243	397089	3.96	**0**	397089	397010	3.92	**0**	**384879**	**382113**	**0.72**
260×3	834985	834967	8.56	**0**	834972	834971	8.56	**0**	**772343**	**769113**	**0.42**
260×5	893754	893728	7.43	**0**	893773	893753	7.43	**0**	**835419**	**831977**	**0.41**
260×8	934777	934767	6.41	**0**	934788	934787	6.41	**0**	**882191**	**878447**	**0.43**
280×3	486177	486054	5.39	**0**	486054	485979	5.36	**0**	**468204**	**461322**	**1.49**
280×5	1036656	1036634	8.13	**0**	1036667	1036645	8.14	**0**	**963635**	**958677**	**0.52**
280×8	1073329	1073328	6.28	**0**	1073323	1073321	6.28	**0**	**1014122**	**1009934**	**0.41**
300×3	555659	555545	4.75	**0**	555535	555419	4.73	**0**	**536377**	**530446**	**1.12**
300×5	1181470	1181436	7.83	**0**	1181478	1181457	7.83	**0**	**1103767**	**1095645**	**0.74**
300×8	610185	610061	3.17	**0**	609994	609790	3.14	**0**	**596674**	**591426**	**0.89**



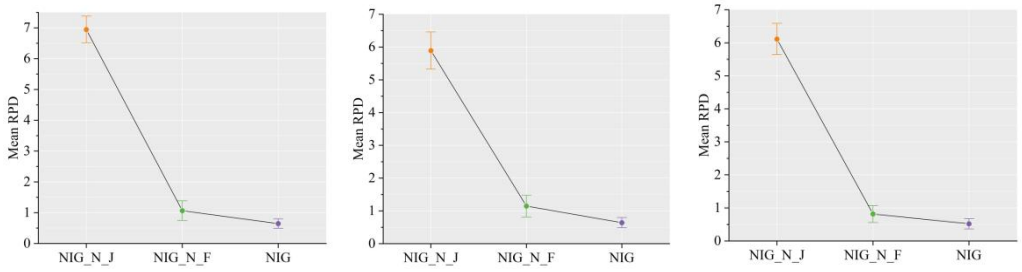
(a) 100×3×20　(b) 200×5×40　(c) 280×8×60

(d) N_100×3×20　(e) N_200×5×40　(f) N_280×8×60

图 4-4 IGA, IGDLM 和 NIG 算法的收敛曲线

4.4.6 统计实验与分析

在本节中，将对上节获得的实验结果进行方差分析。图 4-5 展示了当终止参数 ρ =100 时所有对比算法的均值图，其 HSD 置信区间为 0.95，该分析主要用于检验不同策略或算法，在不同规模和终止条件下的总体水平差异。



(a) F=20　(b) F=40　(c) F=60

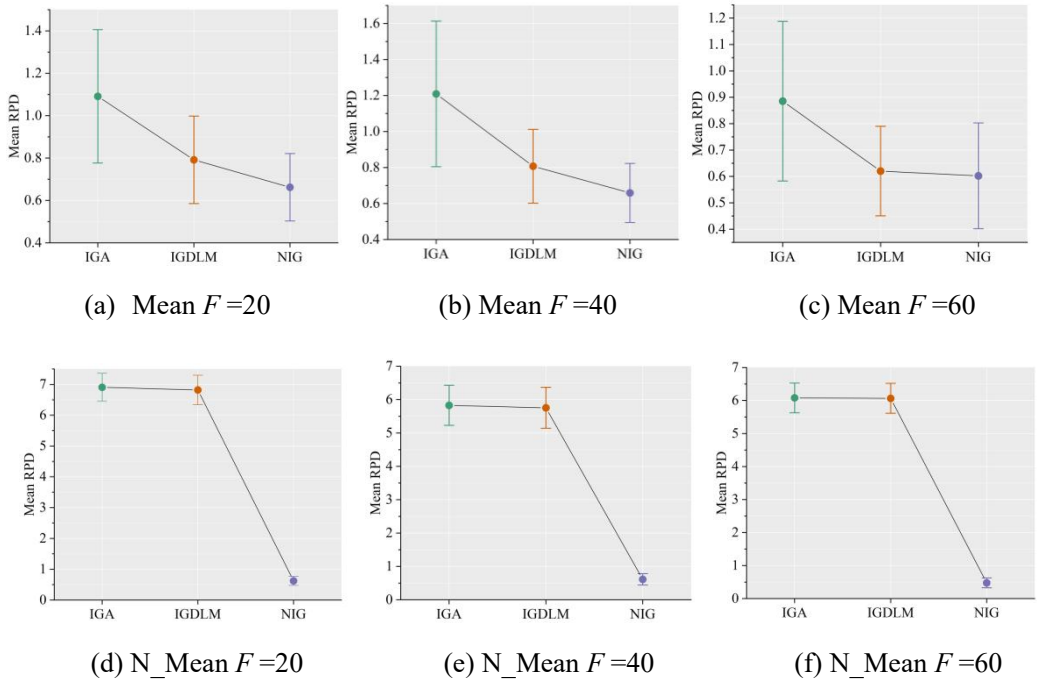
图 4-5 不同组规模下的 NIG_N_J, NIG_N_F 和 NIG 均值图

首先，根据上一节测试数据，绘制不同策略的置信区间来查看影响。本章根据工件组规模将图 4-5 划分成三个子图，由图 4-5 可知，NIG_N_J、NIG_N_F 和 NIG 之间的差异是显著的，并且 NIG 在所有组规模中表现出了最好的性能，从置信区间的分布来看，基于阻塞工件的邻域搜索策略对算法的影响远大于组邻域搜索策略，进一步证实了本章所提针对阻塞约束设计策略思想的重要性。

接下来，本节展示出了 0.95 HSD 置信区间图。图 4-6（a-c）显示了不同组规模下所有对比算法的置信区间：即 F=20、40 和 60。其中，IGA 和 IGDLM 均使用了基于阻塞工件的邻域搜索策略来获取更好的解决方案。 而在图 4-6（d-f）中， IGA 和 IGDLM 均没有使用所提策略来获取解决方案。由图 4-6 可知，无论工件组的规模变为多少，NIG 依旧是性能最佳的优化算法，随后，IGDLM 和 IGA 性能依次排序。当 IGA 和 IGDLM 删除掉基于阻塞工件的邻域搜索策略时，它们的搜索性能将变得更差，并且与 NIG 算法之间的差异显著。

此外，图 4-7 展出了三个实例，即，80×3×20、140×5×40、120×8×60 规模的置信区间为 95%的小提琴图。同样，本节将实验分为两组，一组是对比带有所提阻塞工件邻域搜索策略的实验（a-c），另一组是对比不带有所提策略的实验（d-f）。所有实例被重复执行 30 次，通过实验结果表明，NIG 算法得到的 RPD 值远小于 IGA 和 IGDLM 算法得到的 RPD 值，因此，NIG 算法在获得更好的调度序列方面更有潜力。

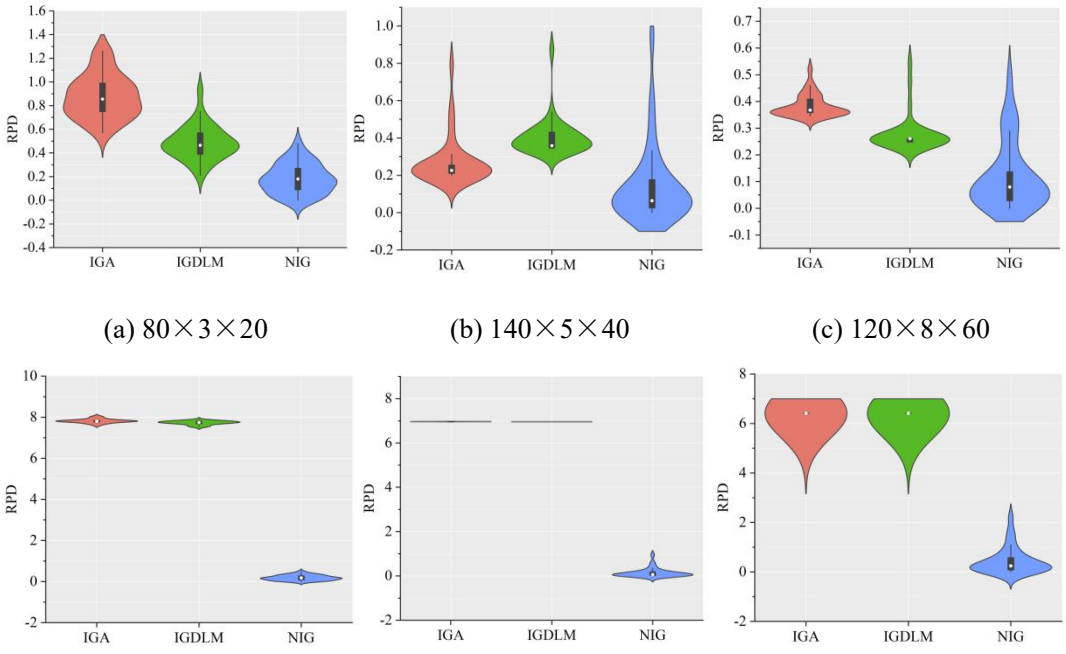
由图 4-6 和图 4-7 可知，在没有使用阻塞工件邻域搜索策略的情况下，IGA 和 IGDLM 与 NIG 算法有很大的性能差异，由此说明了阻塞约束对工件序列完工时间的影响，此外，当 IGA 和 IGDLM 使用了阻塞工件邻域搜索策略时，其性能也不如 NIG 算法，其原因可能是：IGA 和 IGDLM 对机器设定时间的调整能力不如本文设计的组邻域搜索策略，在当前的对比算法中，NIG 仍然是性能最好的算法，这归功于其在全局和局部搜索能力之间的平衡，以及针对问题特征所设计的策略。



(a) Mean F =20　(b) Mean F =40　(c) Mean F =60

(d) N_Mean F =20　(e) N_Mean F =40　(f) N_Mean F =60

图 4-6 在不同组规模下所有算法的置信区间图



(a) 80×3×20　(b) 140×5×40　(c) 120×8×60

(d) 80×3×20N　(e) 140×5×40N　(f) 120×8×60N

图4-7 不同规模下所有对比算法的小提琴图

4.5 本章小结

本章是第一次研究以最大完工时间为优化目标的 BHFGSP，并针对该问题特性设

计了邻域搜索策略来求解。根据大量的实验数据和统计结果证明：所提出的 NIG 算法在求解 BHFGSP 上是非常有效的。综上所述，本章做出了以下四点贡献：1）本章提出了一种具有序列依赖设定时间的 BHFGSP 数学模型。2）提出了新的编码和解码过程来表示和计算一系列加工程序。3）基于组的邻域搜索策略旨在调整组的排列顺序，进一步减少不同组之间的设定时间。4）设计了基于阻塞工件的邻域搜索策略来改变工件序列的阻塞时间。

第五章 求解分布式阻塞混合流水车间调度问题的协同进化迭代贪婪算法

5.1 研究背景

在本文第三章，研究了以最大完工时间为优化目标的 BHFSP，但在现实生活中，为了分散生产压力以及节约成本，企业已经广泛使用分布式工厂的生产模式来代替传统的集中式生产模式。若同时考虑分布式和 BHFSP，该问题也被称作分布式阻塞混合流水车间调度问题（Distributed Blocking Hybrid Flow Shop Scheduling Problem, DBHFSP）。在研究 DBHFSP 时，需要同时考虑一些子问题，如工件排序、工厂分配、机器选择和工件阻塞条件，由于子问题是高度耦合的，因此，可以设计基于邻域的元启发式算法来分步骤完成不同的加工工序。

众所周知，元启发式算法常被用来求解流水车间调度问题，且取得了良好的性能。IG 算法作为元启发式算法的一种，由于参数少、操作简单、过程简单，已被许多学者用来求解相关的流水车间调度问题。该算法在整个过程中只迭代一个解，进而更好地对解邻域更深程度的探索。此外，IG 算法由于其贪婪插入策略，具有较强的局部搜索能力，但也成为了算法的限制因素，导致解的多样性降低。

考虑 IG 算法的优点和局限性，本章提出了一种协同进化 IG（Coevolution IG, CIG）算法来求解分布式阻塞混合流水车间调度问题（Distributed Blocking Hybrid Flow Shop Scheduling Problem，DBHFSP）。在 CIG 中，本章使用了一种新的初始化方案为异构工厂分配不同的工件。随后，本章提出了两种跨工厂的邻域搜索策略来重新调度工件序列，紧接着，本章对每个工厂的工件序列执行破坏和重构操作，并使用局部强化策略进一步缩短工件序列的最大完工时间。

本章的贡献如下：

1） 建立了带阻塞约束的 DBHFSP 的数学模型。

2）为了减少阻塞约束对加工过程的影响，本章提出了 CIG 算法来求解 DBHFSP。其中，设计了 Nawaz–Enscore–Ham Increase（NEH_IN）初始化策略来将工件分配给异构工厂。

3） 为了进一步提高算法的全局搜索能力，减少序列的阻塞状况，本章分别提出了跨工厂和内部工厂邻域搜索策略，使算法可以协同优化。

4） 为了进一步提高 CIG 算法的局部搜索能力，减少工件序列的完工时间，本章提出了局部强化策略来调整各工厂中的序列排序。

本章的其余部分组织如下：5.2 节建立了 DBHFSP 的 MILP 模型。5.3 节详细阐述了所提的 CIG 算法，包括框架和策略的细节。5.4 节测试了 CIG 算法的不同参数和所使用的策略性能，并将其与最先进的算法进行比较。第 5.5 节给出了本章的小结。

5.2 问题描述与数学模型

在 DBHFSP 中，存在 F 个异构工厂，每个工厂具有相同数量的处理阶段。在任意一个加工工厂中，至少有一个处理阶段包含两台或两台以上的并行机器。在加工阶段 s，有 m（m≥1）个加工机器，在各台机器中，没有用于存储已加工完毕的缓冲区。此外，还有以下条件限制：

1） 每台机器都是可使用且无故障的。

2） 有 n 个工件必须在这 F 个工厂中处理。

3） 每个工件都是按阶段顺序依次处理的。

4） 工件一旦确定被某台机器加工，则在加工过程中不能被中断。

总之， DBHFSP 由三个子问题组成，即安排工件的排列顺序、为工件选择工厂和为工件分配机器。在本章，该问题的优化目标是最大完工时间，基于上述定义，本章设计了 DBHFSP 的 MILP 模型。

参数设定：

J：工件数量.

F：工厂数量.

S：每个工厂中的阶段数.

j：工件编号， $j \in \{1,2,...,J\}$.

f：工厂编号， $f \in \{1,2,...,F\}$.

s：阶段编号， $s \in \{1,2,...,S\}$.

m：每个阶段的机器编号.

$m_{f,s}$：在工厂 f 的阶段 s 的平行机数量.

$p_{j,s}$：工件 j 在 s 的加工时间.

决策变量：

$C_{\max}$：工件序列的最大完工时间.

$B_{j,s}$：工件 j 在阶段 s 的开始时间.

$C_{j,s}$：工件 j 在阶段 s 的完工时间.

$D_{j,s}$：工件 j 在阶段 s 的离开时间.

$x_{f,j}$：决策变量，若工件 j 在工厂 f 内加工，等于 1；否则，等于 0.

$y_{f,s,j,m}$：决策变量，若工件 j 在工厂 f 中阶段 s 的机器 m 加工，等于 1；否则，等于 0.

$z_{f,s,j,j'}$：决策变量，若在工厂 f 的阶段 s 中，工件 j 的位置在工件 j' 的前面，等于 1；否则，等于 0.

优化目标：

最小化 $C_{\max}$ （5-1）

约束条件：

$$\sum_{f=1}^{F} x_{f,j} = 1, \forall j \quad （5\text{-}2）$$

$$\sum_{m=1}^{m_{f,s}} y_{f,s,j,m} = x_{f,j}, \forall f, j, s \quad （5\text{-}3）$$

$$B_{j,s} \geq 0, \forall j, s \quad （5\text{-}4）$$

$$C_{j,\mathrm{s}} = B_{j,\mathrm{s}} + p_{j,s}, \forall j, s \quad （5\text{-}5）$$

$$z_{f,s,j,j'} + z_{f,s,j',j} \leq 1, \forall f, s, j \neq j' \quad （5\text{-}6）$$

$$z_{f,s,j,j'} + z_{f,s,j',j} \geq y_{f,s,j,m} + y_{f,s,j',m} - 1, \quad \forall f, s, j' \neq j,\ m \in \{1,2...,m_{f,s}\} \quad （5\text{-}7）$$

$$B_{j',s} - D_{j,s} + U \cdot \left(3 - y_{f,s,j,m} - y_{f,s,j',m} - z_{f,s,j,j'}\right) \geq 0, \quad \forall j \neq j', f, s, m \in \{1,2...,m_{f,s}\} \quad （5\text{-}8）$$

$$D_{j,s} = B_{j,s+1}, \forall j, s \in \{1,...,S-1\} \quad （5\text{-}9）$$

$$C_{j,s} <= D_{j,s}, \forall j, s \quad （5\text{-}10）$$

$$C_{\max} >= D_{j,S}, \forall j \quad （5\text{-}11）$$

（5-1）是 DBHFSP 的优化目标。约束（5-2）确保工件只能被分配到一个工厂里加工。约束（5-3）确保一个工件只能分配给一个工厂，且在每个阶段中必须只由一台机器加工。约束（5-4）确保工件在各个阶段的开始时间不小于 0。约束（5-5）说

明在每个阶段，一个工件的完成时间等于其开始时间与加工时间之和。约束条件（5-6）-（5-7）确保每台加工机器一次只能处理一个工件，并且工件只能由一台机器处理。约束（5-8）表示在同一台机器上，只有完成上一个工件的加工任务后，才能处理后续的工件。约束（5-9）定义，除最后一个阶段外，工件的开始时间等于其前一阶段的离开时间。约束（5-10）表示工件的离开时间不小于其在同一阶段的完成时间。约束（5-11）确保最大完工时间不小于最后阶段所有工件的离开时间。

为了更直观展示出不同的工件顺序和工厂分配所带来的影响，本章给出了带有两个异构工厂、两个阶段和六个工件的甘特图来举例说明这一过程。表 5-1 列出了每个工件的处理时间。

在工厂 1 中，第一阶段有一台机器，第二阶段有两台平行机器。在工厂 2 中，第一阶段设有两台平行机，第二阶段设有一台平行机，所有工件在同一阶段都有相同的加工时间。

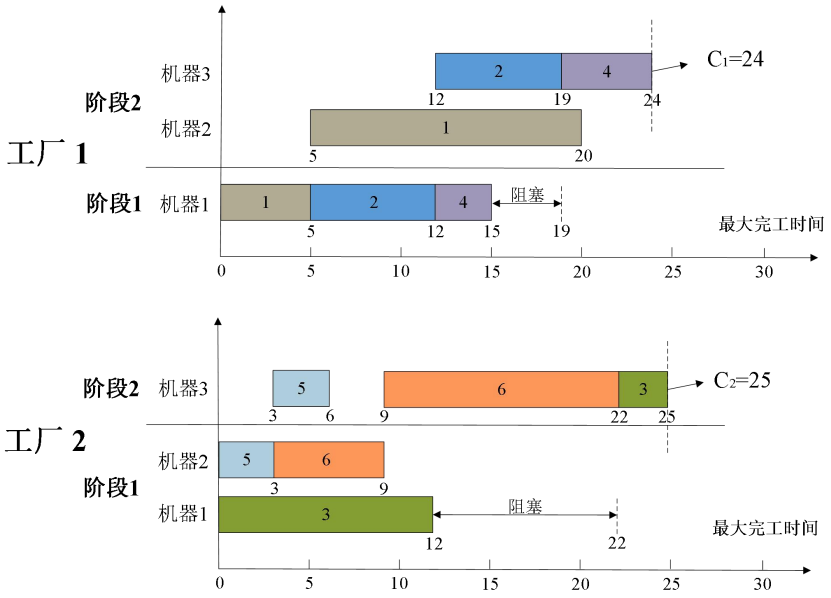
在图 5-1（a）中，工件序列为π={1，2，4；3，5，6}，说明工件 1，2，4 被分配到工厂 1 的第一阶段按顺序处理，而工件 3，5，6 被分配到工厂 2 的第一阶段上处理。在图 5-1（b）中，工件序列为π={1，3，6；2，4，5}，说明工件 1，3，6 被分配到工厂 1 的第一阶段中按顺序处理，而工件 2，4，5 在工厂 2 的第一阶段被按顺处理。如图 5-1（a）和（b）所示，在相同的环境下，不同的调度序列会产生不同的最大完工时间。序列{1，2，4；3，5，6}和{1，3，6；2，4，5}的生成时间分别为 25 和 36，这两种调度方案之间的差距很大。此外，随着工件规模的不断扩大，间隙（机器处于空闲和阻塞状态的时间间隔之和）可能会由于较差的排列顺序而随之扩大，因此，设计合理的调度策略来帮助企业缩短完工时间就变得至关重要。

表 5-1 各阶段不同工件的处理时间

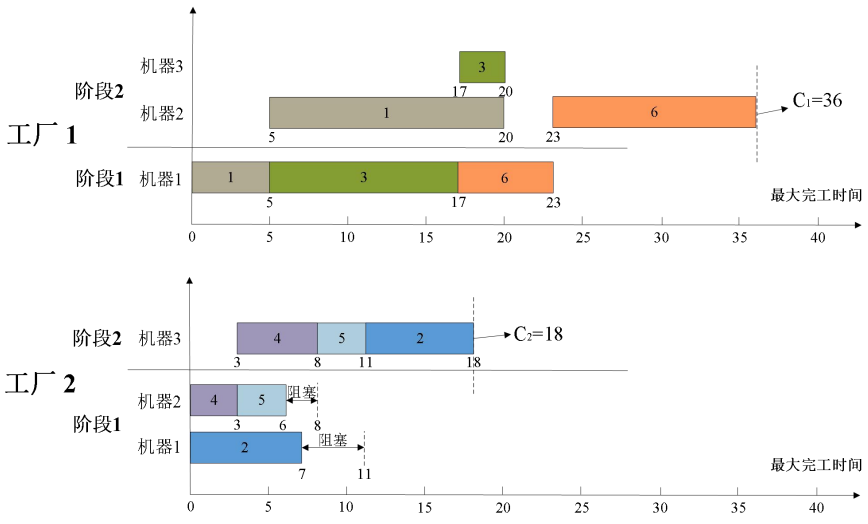
Job	Stage1	Stage2
1	5	15
2	7	7
3	12	3
4	3	5
5	3	3
6	6	13

图 5-1（c）的调度顺序与图 5-1（b）相同。但是机器数量不同。即，工厂是异构的。通过比较图 5-1（b-c）可知，虽然工件的加工顺序相同，但由于在异构工厂中机

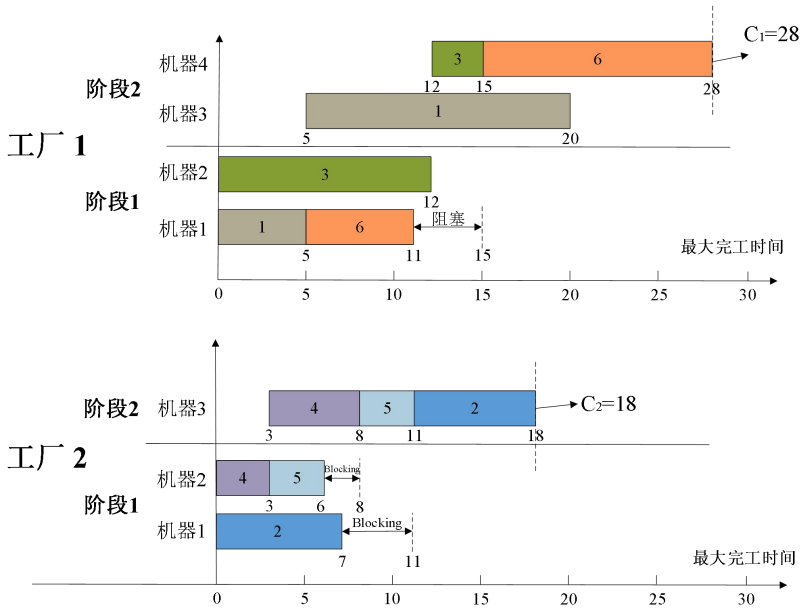
器数量配置的不同，其完成时间也不同，两个加工厂的完工时间差为 8。由此可知，异构工厂的配置对完工时间有着显著的影响。



(a)



(b)



(c)

图 5-1 不同例子甘特图. (a) 和 (b) 是相同工厂配置下不同调度序列的比较. (b) 和 (c) 是工厂中机器配置不同但调度序列相同的比较

5.3 基于协同进化的迭代贪婪算法

本节首先描述了所提 CIG 算法的伪代码框架（见算法 5-1）。CIG 包括初始化策略、邻域搜索策略、破坏重构和局部强化策略。首先，根据分布式工厂的问题特征，分别使用升序 Nawaz Enscore Ham（*NEH_F_asc*）和降序（*NEH_F_des*）初始化方法将工件分配给工厂。随后，本节提出了跨工厂和内部工厂的邻域搜索策略，以探索全局更好的解决方案。其次，采用破坏重建方法改进工件的排列顺序。最后，使用局部强化策略进一步提高解的质量。

算法 5-1 CIG 算法的框架

Input: $\pi=(1,2,3...,n)$, 参数 d

Output: π^{best} 和 makespan

Begin:

$\pi = NEH\_F\_asc(\pi^{origin})$， $\pi' = NEH\_F\_des(\pi^{origin})$

While 终止条件没有满足 **do**

$\pi^1 \leftarrow$ 随机关键工厂扰动策略(π) % 跨工厂邻域搜索策略

$\pi^2 \leftarrow$ 随机任意工厂扰动策略(π')

For *f=1* **to** *F* % 工厂内部邻域搜索策略

$\pi^{1'} \leftarrow$ 破坏重构策略(π^1, d)

$\pi^{2'} \leftarrow$ 破坏重构策略(π^2, d)

$\pi^{1"} \leftarrow$ 局部强化策略 $(\pi^{1'})$
$\pi^{2"} \leftarrow$ 局部强化策略$(\pi^{2'})$
If $f(\pi^{2"}) < f(\pi^{1"})$ **then**
$\pi=\pi^{1'}=\pi^{2"}$
Else
$\pi=\pi^{2'}=\pi^{1"}$
End
End
If $f(\pi) < f(\pi^{best})$ **then**
$\pi^{best}=\pi$
makespan $= f(\pi^{best})$
End
End
End

值得注意的是，求解 DBHFSP 的一个关键问题是如何将工件分配给合适的工厂，以及如何在最大完工时间最小的机器上生成其调度序列。由于上述问题是高度耦合的，因此有必要在跨工厂和工厂内部之间建立协作。首先，本节对不同的工厂执行不同的邻域搜索策略，改变分配给工厂的工件排序。随后，执行破坏重构策略和局部强化策略，以提高每个工厂的解决方案质量。紧接着，通过接受准则获得的当前最优解将用于更新邻域解，并继续参与下一次循环。图 5-2 给出了 CIG 算法的基本流程图。

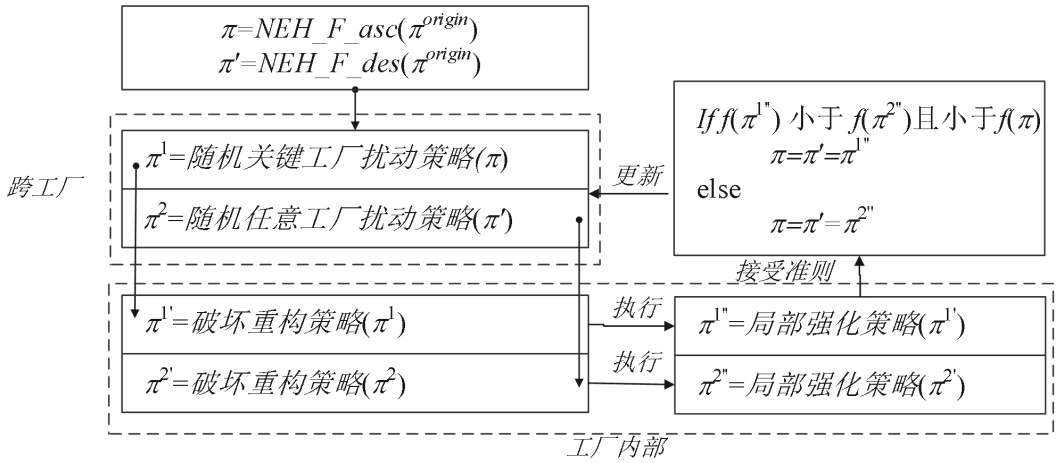


图 5-2 跨越工厂和内部工厂之间的基本流程图

5.3.1 编码与解码

一个好的码解码方法可以帮助算法获得优秀的调度序列，特别是在求解复杂的组合优化问题时，更是展现了其优异性能。在 DBHFSP 中，工件的排列顺序和机器选择应在每个工厂中提前确定下来。其解决方案可以表示为：$\pi = \{\pi^1;\pi^2;...;\pi^F\} = \left\{\pi_1^1, \pi_2^1,...,\pi_{sum1}^1;\pi_{sum1+1}^2,\ldots,\pi_{sum2}^2;...;\ \pi_{sumF-1}^F,...,\pi_{sumF}^F\right\}$，$\pi^f$ 表示为分配

给工厂 f 的工件序列，π_j^f 表示工厂 f 中的第 j 个工件，$f = 1, 2, \ldots, F$。$sumf$ 表示前 f 个工厂中所有工件的总和，每个工件必须分配到一个工厂中进行处理。如图 5-1（a）所示，序列π为 $\{\pi^1;\pi^2\}$，$\pi^1=\{\pi_1^1,\pi_2^1,\pi_3^1\}=\{1,2,4\}$，$\pi^2=\{\pi_4^2,\pi_5^2,\pi_6^2\}=\{3,5,6\}$，即，工件 1、2 和 4 在工厂 1 中按顺序处理，工件 3、5 和 6 在工厂 2 中处理。

针对所有工厂，解码过程采用先来先服务的规则来确定调度序列，并采用第一个可用机器规则来将工件分配给机器（见图 5-1）。

5.3.2 初始化方案

阻塞约束的存在会影响工件序列的完工时间，对于 CIG 算法而言，初始解的质量直接影响了算法的后续迭代。*NEH_F* 方法已被证明是一种有效的初始化策略[62]，在此基础上，本章设计了一种按处理时间升序的初始化方法 *NEH_F_asc* 来完成解的初始化操作。为了提高解的多样性和收敛性，本章同时选择 *NEH_F_des* 和 *NEH_F_asc* 策略生成两个初始解。*NEH_F_des* 和 *NEH_F_asc* 之间的唯一区别在于，初始解决方案是根据工件降序排列的总加工时间对工件进行排序的。

算法 5-2 NEH_F_asc 初始化策略

Input: $\pi^{origin}=(1,2,3...n)$
Output: π
Begin:
π = 升序排列$(\sum_{s=1}^{S} p_{j,s})$, *j =1,2...n*
For *f*=1 to *F*
$\pi^f=\pi_f$ % 将第 f 个工件分配到工厂 f 中
End
For *j=F+1* **to** *n*
For *f=1* **to** *F*
$\pi^{f\_temp} \xleftarrow[i=1 \ \ to \ \ |\pi^f|]{\text{插入到第}i\text{个位置}}$ 提取(π_j)
$\pi^f = \arg\min_{i=1}^{|\pi^f|} f(\pi^{f\_temp})$
End
End
End

在 *NEH_F_asc* 中，本章首先按升序排列工件序列，其次，提取与工厂数量相同的工件数量，随后，将这些工件逐个安置到每个工厂中，以确保每个工厂都拥有一个工件。针对序列中的剩余工件，（1）取出第一个工件，尝试将其插入到所有工厂的所有位置，随后选择最大完工时间最小的位置作为该工件的最终插入位置。（2）从π中删除第一个工件。（3） 重复上述步骤（1）和（2），直到所有工件都被分配到工厂中

合适的位置上。此时，可以保证每个工厂的最大完工时间尽可能小，但不能保证所有工厂的工作数量相同或平均。*NEH_F_asc* 的伪码如算法 5-2 所示。

5.3.3 邻域搜索策略

IG 算法局部搜索能力较强，但其全局搜索能力并不突出。据作者所知，基于交换的策略的时间复杂度小于基于插入策略的时间复杂度（$O(n^2)$ vs. $O(n^3)$），例如，本章对 n 个工件执行交换操作。每个工件均与所有其他工件交换位置，因此总共有 $n-1$ 次位置交换，总时间复杂度为 $O(n^2)$。当对 n 个工件执行插入操作时，每个工件首先选择位置，有 $n-1$ 个位置可以选择。一旦选择了位置 p，则应移动 $n-p$ 个工件。如果 n 个工件都要执行插入操作，则总时间复杂度为 $O(n^3)$，大于交换操作的时间复杂度。

因此，在该算法中，针对以上两种解决方案，提出了基于交换算子的邻域搜索策略，以增强 IG 算法的全局搜索能力，提高工件排序的效率，减少阻塞约束对最大完工时间造成的影响。此外，针对 DBHFSP 的异构特性，将其划分为两种工厂，即关键工厂和非关键工厂。由于目前缺乏为这两类工厂分配工作的邻域扰动策略，因此，在本章邻域搜索策略中，考虑了随机关键工厂扰动和随机任意工厂扰动策略。

1） 随机关键工厂扰动策略：（a）选择工厂 f 中具有最大完成时间（f_{max}）的序列，随后，随机选择工厂，记为 f_{random}。（b）从被选的两个工厂中各自随机选择一个工件进行交换。（c）若获得的新序列的最大完工时间小于原始序列的最大完工时间，新序列将替换旧序列。（d）重复步骤（a-c），直到满足终止标准。

2） 随机任意工厂扰动策略：（a）随机选择两个不同的工厂。（b）从每个选择的工厂中随机选择一个工件并交换它们。（c）如果新序列的最大完工时间小于原始序列的完工时间，则替换原始序列。（d）转到步骤（a），直到满足终止条件。

算法 5-3 给出了邻域搜索策略的框架。

算法 5-3 邻域搜索策略

Input: π, π'
Output: π^1, π^2
Begin:
$\pi^{temp1}=\pi$， $\pi^{temp2}=\pi'$
π^{temp1}**：执行随机关键工厂扰动策略**
$f_{\max}=\arg\min_{f=1}^{F} f(\pi^{temp1})$ % 找到带有最大 makespan 的工厂
do {
$f_{randonm}=rand()\%F$
} **While** ($f_{randonm} == f_{\max}$)
For j=1 **to** n^2

$pos1 = rand()\%|\pi^{temp1\_f_{\max}}|$， $pos2 = rand()\%|\pi^{temp1\_f_{random}}|$

$\pi^{temp1\_new} = 交换(\pi^{temp1}_{pos1}, \pi^{temp1}_{pos2})$

If $f(\pi^{temp1\_new}) < f(\pi^{temp1})$ **then**

$\pi^{temp1} = \pi^{temp1\_new}$

Else

$\pi^{temp1\_new} = \pi^{temp1}$

End

End

If $f(\pi^{temp1}) < f(\pi)$ **then**

$\pi = \pi^{temp1}$

End

π^{temp2}**：执行随机任意工厂扰动策略**

$f_{randonm1} = rand()\%F$

do {

$f_{randonm2} = rand()\%F$

} **While** ($f_{randonm1} == f_{randonm2}$)

For j=1 **to** n^2

$pos3 = rand()\%|\pi^{temp2\_f_{random1}}|$， $pos4 = rand()\%|\pi^{temp2\_f_{random2}}|$

$\pi^{temp2\_new} = 交换(\pi^{temp2}_{pos3}, \pi^{temp2}_{pos4})$

If $f(\pi^{temp2\_new}) < f(\pi^{temp2})$ **then**

$\pi^{temp2} = \pi^{temp2\_new}$

Else

$\pi^{temp2\_new} = \pi^{temp2}$

End

End

If $f(\pi^{temp2}) < f(\pi^2)$ **then**

$\pi^2 = \pi^{temp2}$

End

End

5.3.4 破坏重构策略

在传统的 IG 算法中，破坏和重构策略可以更深入地探索局部邻域，此操作对找到更好的调度序列有直接影响。为了增强算法的局部搜索能力，本算法引入了破坏和重构策略，以减少阻塞约束对最大完工时间造成的影响。其步骤如下（见算法 5-4）：

（1）在工厂中，从当前工件序列中随机提取 d 个工件。

（2）将第一个提取的工件插入到剩余序列的所有位置中。

（3）选择最大完工时间值最小的序列作为当前剩余序列。

（4）重复步骤（2-3），直到 d 个工件都插入到剩余序列中。

算法 5-4 列出了破坏和重构策略的具体细节。

算法 5-4 破坏重构策略

Input: π^1, π^2, 参数 d

Output: $\pi^{1'}, \pi^{2'}$

Begin:

For $f = 1$ **to** F

$\pi^{temp1\_f} = \pi^{1\_f}$ % π^{1_f}：工厂 f 的工件序列

$U_{i=1}^{d} =$ 提取(π^{temp1_f})

For $j = 1$ **to** d

$\pi^{temp1\_f'} = \pi^{temp1\_f} \setminus U_j$, $\pi^{temp1\_f'} \xleftarrow[i=1 \quad to \quad |\pi^{temp1\_f}|]{\text{插入到第}i\text{个位置}} U_j$

$\pi^{temp1\_f} = \arg\min_{i=1}^{|\pi^{temp1\_f}|} f(\pi^{temp1\_f'})$

End

If $f(\pi^{temp1\_f}) < f(\pi^{1\_f})$ **then**

$\pi^{1'\_f} = \pi^{temp1\_f}$

End

End

π^2 执行与 π^1 相同的过程

End

5.3.5 局部强化策略

工件的阻塞延长了调度序列的完工时间，降低了企业的生产效率。为了进一步提高算法的性能，减少阻塞时间，本节提出了一种基于交换算子的局部强化策略来求解DBHFSP。

局部强化策略包含两种交换算子：随机交换和顺序交换算子，根据从破坏和重构策略中得到序列的$\pi^{1'}$和$\pi^{2'}$，分别使用以下策略对其进行改进：

1） 随机交换：（a）在同一序列中随机选择两个不同的工件。（b）交换所选工件的位置。（c）若序列的完工时间缩短，则使用新序列替换旧序列。（d）重复步骤（a-c），直到满足终止标准。

2） 顺序交换：设置 i=1，j=1。（a）按顺序选择第 i 个工件。（b）在相同序列中选择第 j 个工件。（c）交换第 i 个工件和第 j 个工件的位置。若工件序列得到改进，则替换原始序列。（d）j++，重复步骤（c），直到 $j == n$.（e）i++，跳到步骤（b），直到满足 $i == n$。

下面给出该策略的伪代码：

算法 5-5 局部强化策略

Input: $\pi^{1'}, \pi^{2'}$

Output: $\pi^{1''}, \pi^{2''}$

Begin:

$r = rand()\%2$

Case $r == 0$:

***随机交换*:**

For $f = 1$ **to** F

For $j = 1$ **to** n^2

$\pi^{temp1\_f} = \pi^{1'\_f}$ $\%\ \pi^{1'\_f}$：工厂 f 的工件序列

For $j = 1$ **to** n^2

$p_1 = rand()\%|\pi^{temp1\_f}|$

do {

$p_2 = rand()\%|\pi^{temp1\_f}|$

} **While** ($p_1 == p_2$)

交换($\pi_{p_1}^{temp1\_f}, \pi_{p_2}^{temp1\_f}$)

If $f(\pi^{temp1\_f}) < f(\pi^{1'\_f})$ **then**

$\pi^{1''\_f} = \pi^{temp1\_f}$

End

End

End

End

Case $r == 1$:

***按序交换*:**

For $f = 1$ **to** F

For $i = 1$ **to** $|\pi^{temp1\_f}|$

$\pi^{temp1\_f} = \pi^{1'\_f}$

For $j = 1$ **to** $|\pi^{temp1\_f}|$

交换($\pi_i^{temp1\_f}, \pi_j^{temp1\_f}$)

If $f(\pi^{temp1\_f}) < f(\pi^{1'\_f})$ **then**

$\pi^{1''\_f} = \pi^{temp1\_f}$

End

End

End

End

$\pi^{2'}$ 执行与 $\pi^{1'}$ 相同的过程

End

5.3.6 CIG 算法时间复杂度

假设有 n 个工件，f 个工厂，每个工厂的工作数量和阶段分别为 n/f 和 s。整个 CIG 算法的计算复杂度主要由初始化、邻域搜索、破坏与重建以及局部强化策略决定。

初始化策略时间复杂度为 $O(n*f*s*n/f)$，接近 $O(n^2s)$。在 while 循环中，假设邻域搜索策略的迭代次数为 w_1，该策略的时间复杂度为 $O(w_1n^2)$。破坏重构策略和局部强化策略的时间复杂度分别为 $O(w_2*d*f*s*n/f)$和 $O(w_3(n^4+s\ n^3/f^2))$，其中 w_2 和 w_3 分别是破坏重建和局部强化策略的迭代次数。因此，对于整个 CGI 算法，CIG 的时间复杂度为 $O(n(s\ n + w_1\ n + w_2ds + w_3(n^4+s\ n^2/f^2))$，接近 $O(n^5)$。此外，在所提算法中，还设计了基于交换算子的邻域搜索策略来取代贪婪插入操作。与其他 IG 算法相比，该策略降低了时间复杂度，因此，CIG 可以在相同的终止条件下执行更多次。与其他群智能比较算法相比，该算法的解数远远少于这些比较算法。于是，在整个迭代过程中，只需要在更深的层次上迭代这两个解。

5.4 实验结果与分析

本章所有的策略和方法同样使用第三章的实验环境与编程语言。与前两章相同，本章依旧采用相同的 CPU 运行时间作为所有对比算法的终止条件。

5.4.1 测试数据

为了更系统地验证所有算法的性能，本节提供了相关设定值。其工件集的数目是{100、200、300、400、500}，工厂集是{2、3、4、5}，阶段集是{5、8、10}。工件、工厂和阶段的数量构成了不同的算例规模，因此，基准集由 60 个不同规模的实例组成。本章根据参考文献[107]来设置终止条件，即ω为控制运行时间长度的参数，在本章中，将ω的值设置为 5 和 10 来控制算法的运行时间长度。工件的加工时间在范围[1，99]内随机生成，以上实验设置在流水车间调度研究中是通用的，此外，加工机器的数量是从范围[1，5]中随机生成的。本章利用 RPI 来估计当前值与最优值之间的差异，具体公式如下所示：

$$RPI = (c_i - c_{best}) / c_{best} \times 100 \qquad (5\text{-}12)$$

其中，RPI 是相对百分增强，c_i 是由独立执行多次的算法获得的最大完工时间平均值，c_{best} 是由独立执行多次的所有比较算法获得的最大完工时间最小值。本章首先计算每个实例的 RPI，随后计算所有实例的 RPI 平均值。值得注意的是，根据模拟实验结果，不同规模得到的 RPI 值范围略有差异，因此，在下表中，使用“mean”表示所有实例的 RPI 平均值，以测试不同工厂配置的总体性能。

5.4.2 验证 MILP 模型

在本节，将验证所提的 MILP 模型的正确性及有效性，其具体设定为：在 12 个小规模实例上分别测试 MILP 和 CIG 算法的性能。在 CPLEX 12.6 软件中对 DBHFSP 的 MILP 进行编码，最大终止准则设定为 1000 秒。如果能在 1000 秒内找到最佳解决方案，程序将提前终止。针对 CIG 算法，终止时间设置为 $TimeLimit = 10 \times n \times f \times S$。此外，CIG 算法对每个实例独立执行 20 次。表 5-2 给出了通过 MILP 和 CIG 算法获得的 RPI 值、最大完工时间和花费时间。其中，CIG 算法的最大完工时间是 20 个实验结果的平均值。

由表 5-2 可知，MILP 分别为前七个实例和前九个实例获得了最佳 RPI 和最大完工时间值。然而，随着问题规模的不断扩大，即 2_20_3 and 2_20_4，CIG 算法在 RPI、完工时间和运行时间方面优于 CPLEX 算法。根据表 5-2 中的数据可知，针对大规模实例，很难在短时间内获得好的解决方案。因此，所提出的 CIG 算法可以有效地求解 DBHFSP。

表 5-2 MILP 模型与 CIG 算法的 RPI 值和最大完工时间

	MILP			CIG		
F_n_S	RPI	makespan	Time(s)	RPI	makespan	Time(s)
2_8_2	**0.00**	**201**	0.66	4.98	211	**0.32**
2_8_3	**0.00**	**263**	0.73	1.90	268	**0.48**
2_8_4	**0.00**	**304**	0.72	2.63	312	**0.64**
2_12_2	**0.00**	**316**	1000	**0.00**	**316**	**0.48**
2_12_3	**0.00**	**385**	29.04	0.31	386.2	**0.72**
2_12_4	**0.00**	**419**	126.16	0.72	422	**0.96**
2_16_2	**0.00**	**439**	1000	0.06	439.25	**0.64**
2_16_3	**0.63**	**478**	1000	1.23	480.85	**0.96**
2_16_4	**0.57**	**530**	1000	0.99	532.2	**1.28**
2_20_2	**0.00**	**528**	1000	**0.00**	**528**	**0.8**
2_20_3	1.20	592	1000	**0.24**	**586.4**	**1.2**
2_20_4	4.38	644	1000	**0.52**	**620.2**	**1.6**

5.4.3 CIG 算法参数测试

虽然 CIG 算法的参数很少，但参数 *d* 的设定却很关键，它直接影响算法的性能。因此，本小节选择不同数量的 *d* 值，*d*={2、3、4、5、6、7}，以测试不同参数值对实验结果的影响。此外，本节将从多个角度对问题进行分类测试，即，按工厂规模 *f*={2，3，4，5}，工作规模 *n*={100，200，300，400，500}以及 *S*={5，8，10}进行组合，因此，共有 5×4×3=60 个不同的组合。针对 60 个实例，本节给出了按工厂数量和工件

数量分类的平均 RPI（Mean）值，以从不同角度检测参数 *d* 的影响。

由表 5-3 可知，当参数 *d* 的值等于 6 时，不同工厂与工件数量分类实验所得到的 Mean 值最小。表 5-3 列出的结果表明，参数 *d* 的不同取值对算法性能的影响较大，并且对于不同规模的工厂和工作，其影响的趋势是一致的。事实上，随着 *d* 值的增加，越来越多的工件被提取并插入到新序列中进行测试。然而，当 *d* 的值大于 6 时，其 Mean 值在逐渐减小，其原因可能是：将 *d* 设置为较大的值将花费更多时间来探索，同时失去了多次迭代生成有希望解的机会，这将导致算法性能的降低。为了更清楚地观测所有实例中不同 *d* 值的差异性，本节将为所有结果绘制了一个盒图，其结果如图 5-3 所示。当 *d*=6 时，通过分析表 5-3 中的值，均值线在矩形中的位置最低。最终，通过对参数 *d* 的实验测试，本章选择 *d*=6 作为该算法中使用的参数值。

表 5-3 不同规模的工厂和工件的 Mean 值（最小 Mean 值用粗体表示）

	d=2	***d=3***	***d=4***	***d=5***	***d=6***	***d=7***
f=2	**0.25**	0.68	0.27	0.41	0.31	0.33
f=3	**0.62**	3.46	0.67	1.14	0.69	0.8
f=4	2.63	2.8	2.03	2.61	**0.91**	1.23
f=5	3.28	6.22	1.34	4.25	**1.31**	1.64
Mean	1.7	3.29	1.08	2.1	**0.81**	1
n=100	0.86	1.34	0.92	0.85	0.73	**0.66**
n=200	1.24	3.27	1.07	1.33	**1.04**	1.9
n=300	0.71	2.87	**0.25**	2.56	0.34	0.5
n=400	4.88	2.99	1.48	3.65	1	**0.97**
n=500	**0.59**	4.83	1.5	1.45	0.93	0.91
Mean	1.66	3.06	1.04	1.97	**0.81**	0.99

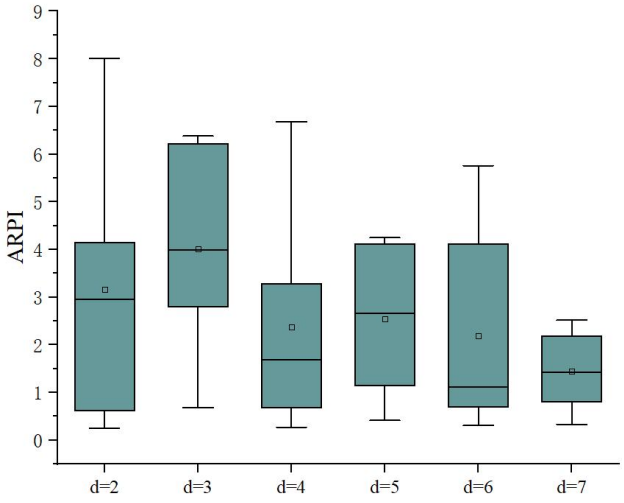


图 5-3 不同 d 值的盒图

5.4.4 不同初始化策略对比实验

为了进一步验证所提初始化策略的性能，针对 60 个不同的测试实例，本节使用 ARPI 指标列出了不同初始化策略所获得的结果。其中，CIG_des_asc（CIG）使用 *NEH_F_asc* 和 *NEH_F_des* 策略分别生成了两个初始解决方案。CIG_des_des 使用 *NEH_F_des* 和 *NEH_F_des* 策略生成两个初始解决方案。 CIG_asc_asc 使用 *NEH_F_asc* 和 *NEH_F_asc* 策略生成初始解决方案。 由表 5-4 中结果可知，CIG_des_asc 优于其他算法，表明所提出的初始化策略与 *NEH_F_des* 策略相结合的性能优于其他策略组合，选择加工时间短和加工时间长的工件分别形成调度序列，可以有效地减少阻塞约束对完工时间的影响。所提出的 *NEH_F_asc* 策略不仅提高了初始解的质量，而且还提高了后续迭代的效率。此外，本节还对 60 个实例的 RPI 值进行了统计检验，如图 5-4 所示，盒图展示了不同水平下初始化策略的性能。其中，采用 *NEH_F_asc* 策略的初始化策略是有效的。考虑到表 5-4 和图 5-4 中获得的平均值，最终选择 CIG_des_asc 作为初始化策略。

表 5-4 比较不同初始化策略的 ARPI 值

ARPI	CIG_des_asc	CIG_des_des	CIG_asc_asc
f=2	0.244	0.410	**0.134**
f=3	**0.299**	0.882	1.058
f=4	**0.532**	1.270	0.854
f=5	2.331	3.103	**1.492**
Mean	**0.851**	1.416	0.884

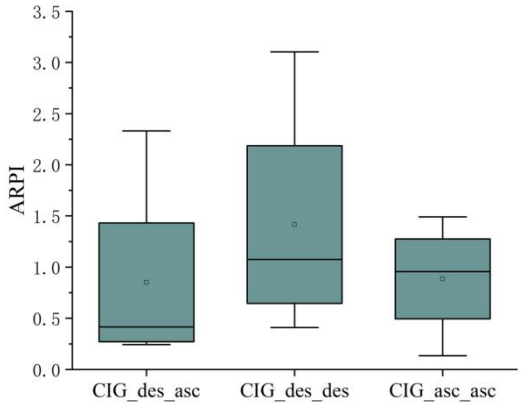


图 5-4 不同初始化策略的盒图

5.4.5 邻域搜索和局部强化策略的性能

针对 CIG_N_NSS 和 CIG_N_LIS 的测试，使用 *NEH_F_asc* 和 *NEH_F_des* 的初

始化策略为其提供初始解。 CIG_N_NSS 是没有使用邻域搜索策略的算法。CIG_N_LIS 是没有使用局部强化策略的算法。从实验结果来看，针对每个工厂（如表 5-5 和图 5-5 所示），由 ARPI 值和盒图中的间隔范围可知，采用上述两种策略的 CIG 算法均获得了最小的 Mean 值。此外，由图 5-5 可知，CIG 算法显著性水平是最好的，原因可能是邻域搜索策略提高了算法的全局搜索能力，局部强化策略提高了算法的局部探索能力。如果没有该策略，算法的质量将急剧下降。所提策略有可能对序列排序进行更深入的探索，发现许多未知邻域，从而提高解的多样性。总之，所提出的策略有效地减少了工件序列的阻塞状况。

表 5-5 比较有无邻域和强化策略的 ARPI 值

ARPI	CIG	CIG_N_NSS	CIG_N_LIS
f=2	**0.244**	1.581	3.221
f=3	**0.299**	0.675	9.823
f=4	**0.532**	2.534	12.574
f=5	2.331	**0.533**	7.531
Mean	**0.851**	1.331	8.287

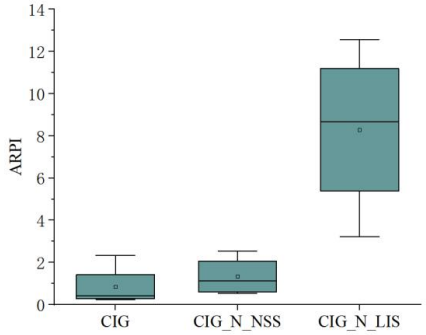


图 5-5 有无邻域和强化策略的盒图

5.4.6 算法性能对比

为了更全面、直观地评估 CIG 算法的性能，本节将 CIG 与求解相关问题最新颖的元启发式算法：DDE 算法[108]、EA 算法[63]、模态和多邻域迭代贪婪算法（modeling and multi-neighborhood iterated greedy algorithm , MN_IG）[75]、IGA 算法[68]，以及 DPSO 算法[26]进行了对比，针对 DBHFSP，有两个关键问题需要解决，一是如何将工件分配给合适的工厂，二是如何找到目标值小的调度序列。在没有分布式环境的情况下，已经针对 HFSP 设计了有效的 DPSO 算法，然而，DPSO 只处理了 DBHFSP 的第二个问题而无法处理第一个工厂分配的问题。因此，为公平起见，DPSO 采用与 CIG 算法相

同的分配策略来解决 DBHFSP 的第一个问题。具体来说，DPSO 首先采用本文提出的 *NEH_F_asc* 和 *NEH_F_des* 的分配策略来初始化种群。随后，针对每个工厂，根据原始文献的步骤实施具体的策略。因此，DPSO 能够实现良好的性能，其原因是进化策略在优化每个工厂的调度序列时是足够有效的。

除 DPSO 算法外，还针对分布式调度问题开发了具有较高性能的算法，如 DDE、EA、IG 和 MN_IG 算法。MN_IG 使用了 *NEH_F_asc* 初始化策略将工件分配给不同的工厂。在分配工厂之后，根据原始文献中提出的策略调整每个工厂中工件的顺序，EA、DDE 和 IG 算法均根据原文献中的策略将工件分配给工厂。在代码中，作者将问题从算法中分离出来，当工厂分配问题完成时，算法的后续操作不会与研究问题相互干扰。因此，选择上述比较算法是合理的。

为了更好地展示所有算法的性能，本节的对比算法使用了与原文献相同的参数值。为了保证对比的公平性，作者在相同的环境下运行所有对比算法，并采用相同的最大 CPU 时间 *Timelimit* 作为终止标准。此外，所有测试算法的平均值和 95%最小显著性差异（LSD）置信区间（验证两个值之间是否存在显著性差异）已在图 5-6 中展示。

表 5-6 当 ω=5 时所有测试算法的最好目标值和 RPI 值的实验结果

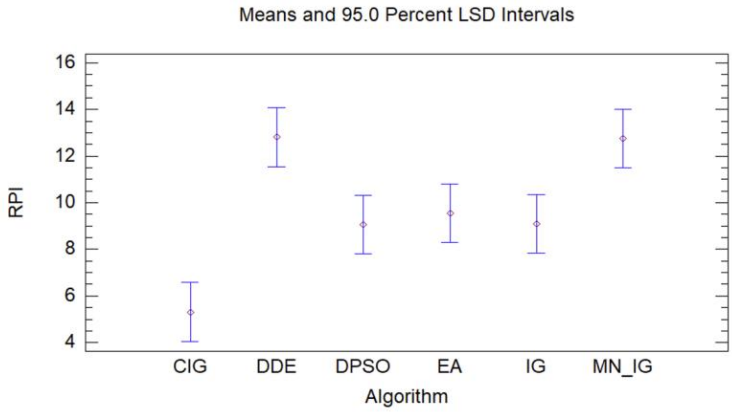
		CIG		DDE[2018]		EA[2018]		IG[2019]		DPSO[2019]		MN_IG[2020]	
f	*n*×*s*	best	RPI	best	RPI	best	RPI	best	RPI	best	RPI	best	RPI
	100×5	**2759**	**0.32**	2928	6.13	2791	2.02	2814	3.42	2772	1.82	2932	6.27
	100×8	**2682**	**1.15**	3348	24.83	3050	15.72	3035	15.53	3030	15.27	3383	26.14
	100×10	**2858**	**1.6**	3329	16.48	3101	9.41	3104	10.52	3081	9.83	2996	4.83
	200×5	**5113**	**0.22**	5176	1.23	5138	0.9	5119	0.33	5128	0.73	5235	2.39
	200×8	**5778**	**0.26**	6166	6.8	5870	2.51	6079	5.66	5806	1.16	5855	1.33
	200×10	**4970**	**0.34**	5280	8.59	5107	3.88	5261	6.81	5024	2.12	5037	1.35
	300×5	**4402**	**2.52**	5199	18.11	5078	15.88	4999	14.56	5062	16.76	5341	21.33
f=2	300×8	**7538**	**0.35**	7764	4.7	7603	1.7	7840	4.03	7562	0.62	7694	2.07
	300×10	**7732**	**1.02**	8571	10.85	8158	6.52	8119	5.92	8066	5.26	8513	10.1
	400×5	**10152**	0.32	10329	2.21	10203	0.94	10318	1.91	10168	0.34	**10152**	**0**
	400×8	**9871**	**0.83**	10351	4.86	10296	4.99	10153	3.37	10121	3.21	10739	8.79
	400×10	**10788**	**1.85**	13564	25.73	13164	23.78	12842	20.14	12561	17.98	13211	22.46
	500×5	**6335**	**1.17**	7035	11.05	6963	10.29	6866	8.96	7018	11.02	7234	14.19
	500×8	**6832**	**3.26**	8112	18.74	8040	18.07	7930	17.41	8087	18.7	8441	23.55
	500×10	**7539**	**4**	9098	20.68	8988	19.73	8731	16.61	8869	19.19	9308	23.46
	mean	**6357**	**1.28**	7083.3	12.07	6903.3	9.09	6880.7	9.01	6823.7	8.27	7071.4	11.22
	100×5	**1801**	**2.79**	2039	13.21	1900	7.77	2065	17.04	1874	6.22	1974	9.61
	100×8	**1828**	**1.79**	2210	20.9	1938	7.54	1976	10.67	1882	4.57	2080	13.79

	100×10	**2394**	**3.3**	2924	22.14	2745	16.45	2739	17.21	2728	16.57	2945	23.02
	200×5	**1917**	**4.1**	2273	18.57	2158	14.31	2109	11.86	2151	13.57	2243	17.01
	200×8	**3701**	**5.53**	4521	22.16	4307	18.15	4207	15.44	4216	16.05	4712	27.32
	200×10	**4276**	**4.28**	5256	22.92	5091	20.26	5016	18.59	4949	17.27	5231	22.33
	300×5	**5766**	**17.6**	7118	23.45	6984	21.69	6731	18.29	6865	20.34	7224	25.29
f=3	300×8	**5460**	2.96	5782	5.9	5555	2.57	5608	3.17	5519	**2**	5693	4.27
	300×10	**5754**	15.63	6742	17.17	6724	17.63	6536	15.35	6533	**15.17**	7140	24.09
	400×5	**7353**	17.4	8721	18.6	8745	19.33	8391	**15.74**	8658	18.43	9174	24.77
	400×8	**6899**	3.28	7046	2.13	6999	1.81	6952	**1.27**	6981	1.79	7205	4.44
	400×10	7182	2.93	7532	6.38	**7119**	1.19	7357	4.48	7125	**0.67**	7382	3.69
	500×5	**8689**	2.43	9010	3.69	8848	2.48	8777	1.92	8749	**1.46**	9040	4.04
	500×8	**8500**	3.23	9096	7.01	8836	4.63	8719	3.71	8763	3.82	9085	6.88
	500×10	**8988**	5.77	9438	5.01	9345	4.28	9308	4.4	9223	**3.4**	9580	6.59
	mean	**5367**	**6.2**	5980.5	13.95	5819.6	10.67	5766.1	10.61	5747.7	9.42	6047.2	14.47
	100×5	**1290**	**2.48**	1444	11.94	1336	6.99	1380	8.06	1322	4.37	1380	6.98
	100×8	**1550**	**2**	1849	19.29	1649	7.88	1618	7.04	1603	5.26	1709	10.26
	100×10	**1714**	**4.8**	2021	17.91	1891	11.86	1824	8.81	1900	13.45	2077	21.18
	200×5	**1419**	**9.51**	1668	17.55	1620	15.27	1591	13.5	1650	18.33	1669	17.62
	200×8	**1484**	**8.28**	1835	23.65	1660	13.73	1631	11.73	1659	13.35	1729	16.51
	200×10	**2800**	4.08	2968	6	2884	4.22	2847	**2.62**	2858	3.03	3013	7.61
	300×5	4104	3.58	4075	0.3	4074	**0.29**	**4063**	0.35	4107	1.08	4230	4.11
f=4	300×8	**3980**	5.36	4204	5.63	4136	4.82	4084	**3.26**	4097	4.92	4379	10.03
	300×10	**4324**	3.61	4521	4.8	4332	1.85	4314	**0.61**	4330	1.33	4465	3.5
	400×5	**5095**	2.17	5189	1.84	5165	1.69	5133	**1.09**	5161	1.59	5191	1.88
	400×8	**5774**	**12.23**	6990	21.06	6782	18.7	6656	17	6765	18.95	7255	25.65
	400×10	**5220**	**5.07**	5657	8.37	5541	7.04	5468	5.68	5498	6.12	5778	10.69
	500×5	**6342**	1.92	6386	0.69	6372	**0.56**	6382	0.83	6386	0.99	6479	2.16
	500×8	**2816**	**9.38**	3365	19.5	3307	18.01	3243	15.83	3296	17.87	3283	16.58
	500×10	**6508**	**0.67**	6915	6.25	6634	2.38	6842	5.21	6557	1.29	6561	0.81
	mean	**3628**	**5.01**	3939.1	10.99	3825.5	7.69	3805.1	6.77	3812.6	7.46	3946.5	10.37
	100×5	**1173**	**2.63**	1359	15.86	1228	7.74	1232	9.68	1208	4.61	1282	9.29
	100×8	**1244**	**1.7**	1521	22.27	1308	8.43	1399	14.85	1286	4.97	1338	7.56
	100×10	**1400**	**3.93**	1646	17.57	1565	13.11	1458	7.69	1562	13.6	1714	22.43
	200×5	**2418**	15.16	2784	15.14	2715	**13.25**	2705	14.76	2794	15.55	2984	23.41
	200×8	**1406**	12.94	1525	8.46	1516	8.39	1448	**5.34**	1525	8.46	1655	17.71
	200×10	**1343**	16.76	1619	20.55	1566	18.59	1472	**10.77**	1539	16.76	1617	20.4
	300×5	**1668**	**8.35**	1911	14.57	1857	12.84	1823	10.39	1847	12.07	1908	14.39
f=5	300×8	**3355**	**5.39**	3671	9.42	3552	6.61	3547	7.4	3569	7.28	3746	11.65
	300×10	**3185**	**3.1**	3538	11.08	3337	5.91	3378	6.88	3288	4.53	3384	6.25
	400×5	**2049**	**7.73**	2403	17.28	2338	15	2305	13.28	2360	16.15	2345	14.45
	400×8	**4506**	4.77	4555	1.09	4541	1.02	4512	**0.91**	4622	2.57	4937	9.57
	400×10	**5078**	15.76	6047	19.08	5808	15.09	5676	**12.79**	5945	18.26	6129	20.7
	500×5	**5023**	**0.58**	5258	4.68	5072	1.52	5212	4.06	5042	0.86	5078	1.09
	500×8	**5767**	**20.53**	7068	22.56	6969	21.72	6932	21	7193	26.68	7345	27.36
	500×10	**5927**	12.15	6737	13.67	6530	11.66	6443	**10.41**	6738	14.66	6939	17.07
	mean	**3036**	**8.77**	3442.8	14.22	3326.8	10.72	3302.8	10.01	3367.9	11.13	3493.4	14.89

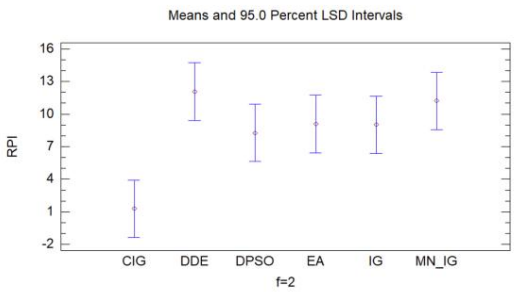
表 5-7 当 ω=10 时所有测试算法的最好目标值和 RPI 值的实验结果

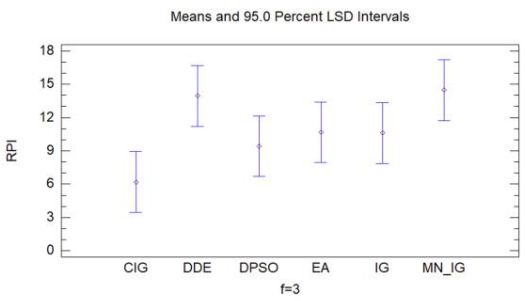
		CIG		DDE(2018)		EA(2018)		IG(2019)		DPSO(2019)		MN_IG(2020)	
f	*n*×*s*	best	RPI	best	RPI	best	RPI	best	RPI	best	RPI	best	RPI
	100×5	**2762**	**0.11**	2928	6.01	2789	1.44	2806	3.15	2772	1.37	2932	6.15
	100×8	**2644**	**0.42**	2838	7.34	2678	2.66	2690	3.82	2654	1.44	2752	4.08
	100×10	**2684**	**0.59**	3062	16.83	2768	4.73	2841	8.98	2729	3.22	2819	5.03
	200×5	**1765**	**2.47**	2033	15.18	1952	11.88	1968	12.86	1995	14.71	2163	22.55
	200×8	**2984**	**2.02**	3553	19.07	3346	14	3352	13.84	3402	15.63	3665	22.82
	200×10	**5301**	**1.99**	6821	28.67	6207	18.67	6250	20.62	6149	17.56	6476	22.17
	300×5	**7617**	**0.29**	8148	6.97	7853	3.66	7851	4	7786	2.58	8076	6.03
f=2	300×8	**7607**	**0.7**	8254	8.51	7911	4.76	7921	4.78	7767	2.94	8178	7.51
	300×10	**8124**	**1.06**	8577	5.88	8289	2.79	8352	3.27	8184	1.43	8342	2.68
	400×5	**10277**	**0.91**	10825	5.33	10698	4.54	10634	4.18	10499	3.11	####	6.9
	400×8	**9959**	0.72	10125	1.67	10011	1.01	10004	1.06	9977	**0.69**	####	1.81
	400×10	**9975**	1.16	10178	2.04	10106	1.61	10060	1.27	10027	**0.86**	####	2.02
	500×5	**12498**	**0.48**	13156	5.26	12872	3.51	12813	3.41	12663	1.8	####	4.38
	500×8	**12995**	**2.48**	16323	25.61	15906	22.91	15465	20.28	15296	18.8	####	29.63
	500×10	**13391**	**2.26**	16515	23.33	16350	22.91	15870	19.5	15588	18.06	####	27.03
	mean	**7372**	**1.18**	8222.4	11.85	7982	8.07	7925.1	8.33	7833	6.95	8240	11.39
	100×5	**1746**	**1.38**	1955	11.97	1792	3.66	1855	6.84	1760	1.96	1816	4.01
	100×8	**1834**	**2.07**	2125	15.87	1933	7.6	1938	7.57	1956	8.26	2106	14.83
	100×10	**1977**	**1.75**	2491	26	2201	13.36	2221	16.08	2212	14.33	2293	15.98
	200×5	**3043**	**1.23**	3247	6.7	3118	3.22	3161	4.34	3066	1.99	3112	2.27
	200×8	**3888**	**3.41**	4902	26.08	4607	20.36	4543	18.18	4601	20.32	4931	26.83
	200×10	**3880**	**3.71**	4976	28.25	4434	15.9	4368	13.85	4399	15.09	4925	26.93
	300×5	**5419**	1.72	5505	1.59	5460	1.27	5433	**1.17**	5484	1.56	5724	5.63
f=3	300×8	**5660**	3.35	5967	5.42	5781	2.53	5741	**2.18**	5782	3.64	6164	8.9
	300×10	**5348**	**8.02**	6219	16.29	6014	13.35	5878	11.47	6013	13.69	6303	17.86
	400×5	**6737**	2.29	6862	1.86	6786	1.14	6796	1.25	6761	**0.81**	6844	1.59
	400×8	**7651**	**14.6**	9523	24.47	9319	22.61	9027	18.8	9126	21.6	9603	25.51
	400×10	**4161**	**14.4**	4915	18.12	4901	18.42	4841	17	4915	18.12	5287	27.06
	500×5	8585	1.31	8590	0.25	8588	**0.24**	**8569**	0.25	8610	0.82	8652	0.97
	500×8	**8239**	1.78	8436	2.39	8338	1.85	8347	1.57	8310	**1.28**	8459	2.67
	500×10	**8421**	2.85	8692	3.22	8566	2.32	8555	2.13	8510	**1.81**	8745	3.85
	mean	**5106**	**4.26**	5627	12.56	5456	8.52	5418.2	8.18	5434	8.35	5664	12.33
	100×5	**1352**	**3.49**	1621	19.9	1431	8.26	1428	8.14	1441	8.89	1563	15.61
	100×8	**1585**	**2.51**	1848	16.59	1656	6.96	1610	4.34	1675	7.88	1738	9.65
	100×10	**1834**	**2.29**	2214	20.72	2044	13.3	1978	11.27	2032	13.28	2221	21.1
	200×5	**2521**	1.82	2604	3.29	2562	2.39	2559	2.31	2531	**1.45**	2590	2.74
	200×8	**2884**	**9.88**	3485	20.84	3305	16.61	3219	13.98	3331	16.83	3655	26.73
	200×10	**1684**	**6.36**	2099	24.64	1930	15.91	1857	12.71	1939	16.46	2049	21.67
	300×5	**1985**	**6.12**	2202	10.93	2134	8.47	2058	4.88	2143	9.31	2210	11.34
f=4	300×8	**2109**	**7.76**	2475	17.35	2334	12.77	2342	12.08	2324	12.28	2420	14.75
	300×10	**4189**	**12.4**	5190	23.9	4828	16.65	4721	14.39	4794	16.15	5277	25.97
	400×5	**2843**	**14.2**	3425	20.47	3344	18.56	3199	13.55	3363	19.57	3352	17.9
	400×8	**5280**	5.45	5612	6.29	5475	4.63	5451	**3.93**	5502	5.56	5610	6.25

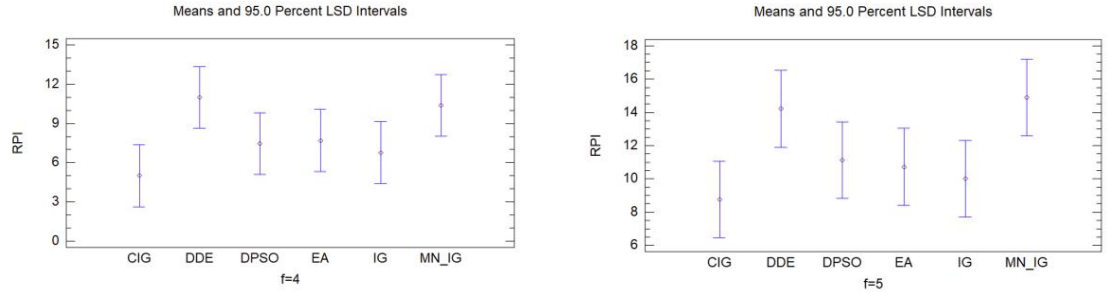
	400×10	**5106**	**1.11**	5507	7.85	5220	2.98	5353	5.6	5144	1.55	5251	2.84
	500×5	**6407**	1.43	6660	3.95	6469	1.39	6653	4.01	6428	**0.8**	6536	2.01
	500×8	**6758**	5.22	7002	3.61	6790	1.56	**6758**	**0.88**	6816	1.91	7061	4.48
	500×10	**7160**	17.1	8385	17.11	8098	13.92	8165	**15.57**	8171	15.83	8651	20.82
	mean	**3580**	**6.48**	4021.9	14.5	3841	9.63	3823.4	8.51	3842	9.85	4012	13.59
	100×5	**1013**	**3.95**	1135	12.04	1059	7.49	1037	4.78	1045	4.92	1156	14.12
	100×8	**1395**	**4.7**	1734	24.3	1524	11.6	1553	17.6	1524	11.57	1676	20.14
	100×10	**1230**	**4.59**	1486	20.81	1348	12.15	1315	9.46	1324	10.53	1439	16.99
	200×5	**1177**	**7.37**	1339	13.76	1294	11.02	1250	7.47	1299	11.95	1417	20.39
	200×8	**2409**	**6.47**	2809	16.6	2581	8.66	2548	9.4	2583	10.03	2679	11.21
	200×10	**2339**	**3.85**	2666	13.98	2437	6.02	2472	7.21	2401	4	2555	9.23
	300×5	3492	5	3643	6.93	3423	1.58	**3407**	**1.35**	3456	2.77	3614	6.08
f=5	300×8	3172	3.15	3518	10.98	**3170**	**2.09**	3386	9.85	3240	3.34	3264	2.97
	300×10	**3508**	6.15	3838	9.41	3642	5.08	3565	**2.61**	3664	6.28	3751	6.93
	400×5	**4976**	17.5	5655	13.65	5611	**13.02**	5579	13.3	5780	16.16	6122	23.03
	400×8	4275	1.85	4571	7.02	**4271**	**1.15**	4494	7.15	4331	2.4	4387	2.72
	400×10	**4646**	19.6	5528	18.98	5382	16.76	5208	**14.46**	5376	17.71	5737	23.48
	500×5	**5332**	5.82	5484	2.85	5442	**2.28**	5444	3.22	5655	6.65	5786	8.51
	500×8	**2767**	**7.4**	3115	12.58	2986	9.4	2951	7.42	3004	9.39	3103	12.14
	500×10	**5275**	6.54	5541	5.04	5473	4.32	5399	**3.21**	5489	5.07	5762	9.23
	mean	**3134**	**6.93**	3470.8	12.6	3310	7.51	3307.2	7.9	3345	8.18	3497	12.48



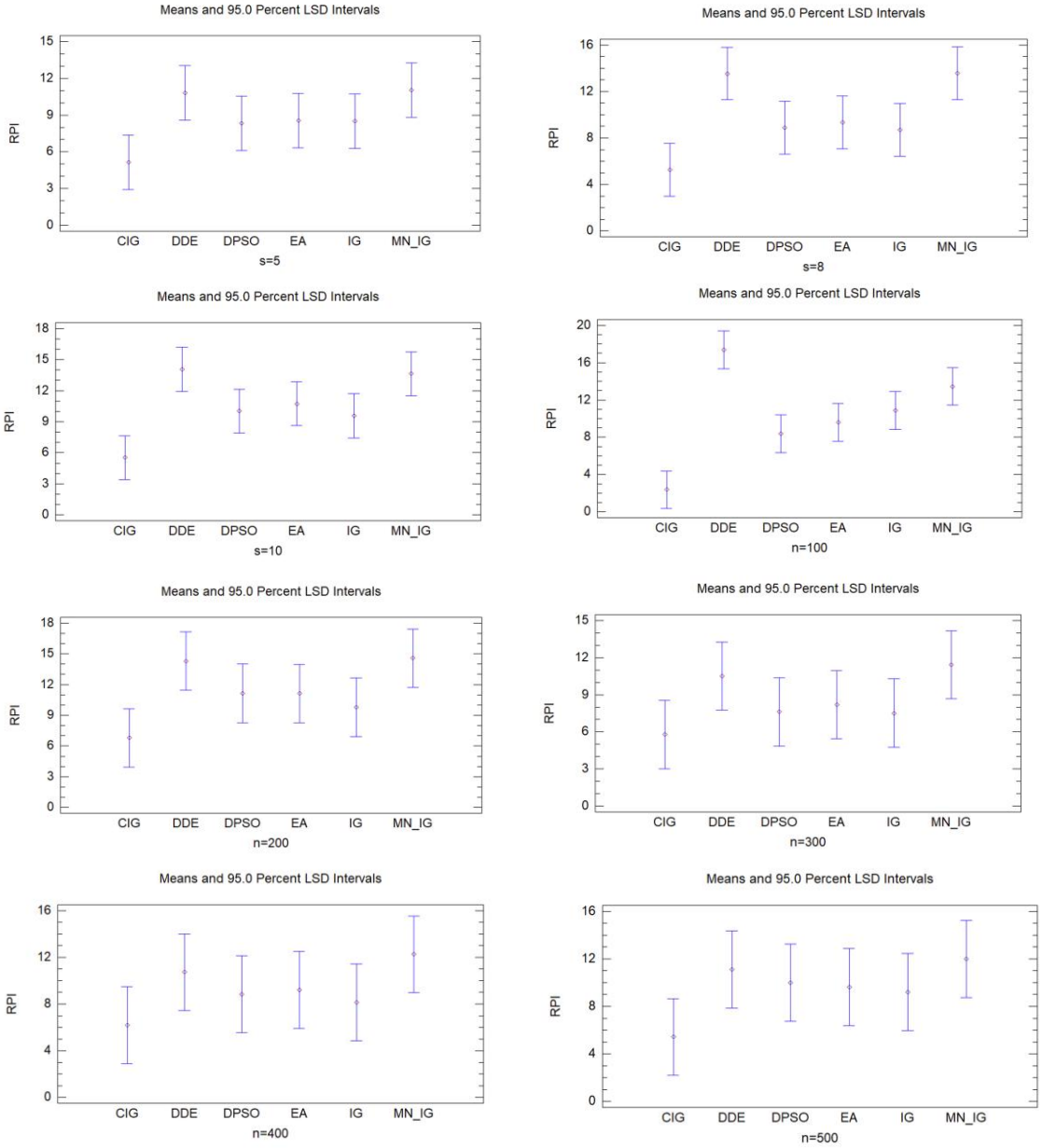
(a) 算法均值图







(b) 针对不同数量工厂的 LSD 置信区间图



(c) 针对不同工件和阶段的 LSD 置信区间图

图 5-6 所有对比算法的均值图和 95%LSD 置信区间图

表 5-6 和表 5-7 列出了所有算法的 RPI 值和最好的目标值，其中，最小的 RPI 值与目标值以粗体形式标记。由表中数据可知，CIG 算法产生的最小目标值的数量在表 5-6 的 60 个测试实例中为 58 个，在表 5-7 的测试实例中为 56 个。 DDE、EA、MN_IG、IG 和 DPSO 算法分别获得了 0、1、1、1、0 个最小目标值，以上算法获得的最小值数目均明显少于 CIG 算法。针对 RPI 值，CIG 算法获得的最优值最多，其次是 IG、EA、DPSO、MN_IG 和 DDE 算法。针对每个工厂实例的平均值，由表 5-6 和表 5-7 可知，CIG 算法可以得到 6/6 最小目标值和 6/6 最小 RPI 值。EA、DDE、MN_IG 和 IG 算法分别获得 0/6 最小目标值和 0/6 最小 RPI 值。由表 5-6 和表 5-7 展示的总体性能来看，所提 CIG 算法优于其他对比算法，表明了所提策略对于求解 DBHFSP 是十分有效的。

为了进一步评估该算法的性能，本节使用多因素 ANOVA 分析来说明 CIG 得到的结果是否与对比算法不同，工件、工厂和阶段数量被视为影响因素。图 6-6 展示了当 ω=5 时，所有对比算法的均值图和 95%最小显著性差异（LSD）置信区间，图 5-6（a）展示出了算法的均值图，图 5-6（b）展示了不同工厂数量（f）的 LSD 置信区间图，图 5-6（c）给出了不同阶段和工件数量（s 和 n）的 LSD 置信区间图。此外，为了进一步说明这些算法之间的差异，本节随机选择了六个不同规模的算例来绘制盒图，详情如图 5-7（a-f）所示。

为了丰富统计结果，本节对实验结果进行了 Friedman 检验。Friedman 检验是一种非参数检验，第一步是假设所有算法都是零假设。判断结果是否拒绝假设，若拒绝，则表明这两种算法在统计上存在显著性差异；否则，两种算法之间没有显著性差异。Friedman 检验如表 5-8 所示，它包含以下元素：性能排名（Ranks）、测试编号（N）、均值（Mean）、标准偏差（Std. Deviation）、最小值和最大值（Min and Max）。

由图 5-6（a）可知，所提 CIG 算法与所有对比算法都有显著的不同。CIG 性能是最好的；IG、DPSO、EA、DDE 和 MN_IG 性能依次递减。它们在求解带有阻塞约束的 DHHFSP 时都表现出了良好的性能，在图 5-6（b）中，随着工厂数量的增加，所提算法与其他算法之间的性能差距正在逐渐缩小，然而，仍有一些区域与对比算法存在显著差异，如图 5-6（c）所示，在不同的加工阶段和工件数量规模下，所提算法与其他对比算法存在显著不同。通过 Friedman 可以观察到，所提算法的性能最好，其 ranks 值为 1.05。此外，与其他算法相比，CIG 的标准偏差、均值、最小值和最大

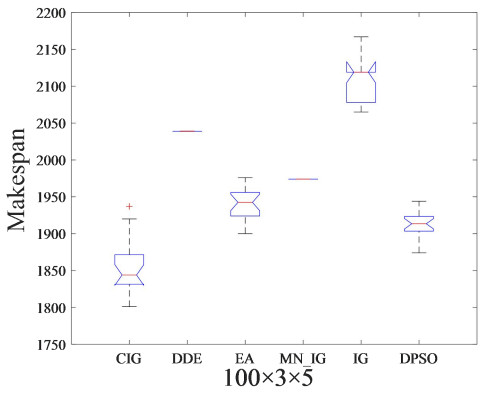
值都是最小的。由图 5-7（a-f）可知，CIG 算法盒图中的目标值小于 DDE、EA、MN_IG、IG 和 DPSO 算法的目标值。在随机选取的 6 个算例中，该算法与其他算法有显著差异，总体来说，CIG 算法的性能最好。

CIG 算法性能优越的原因可能是：首先，所提出的 *NEH_F_asc* 策略可以优先处理加工时间短的工件来提高产品的生产率，有效减少了许多小型工件的阻塞时长。其次，邻域搜索策略可以快速调整工件序列以搜索更好的解，提高全局搜索能力。随后，局部增强策略提高了算法的局部搜索能力，该策略可以更深入地探索单个工厂的解决方案，此外，跨越工厂和内部工厂搜索策略可以在全局搜索和局部搜索之间保持平衡，在完成一次迭代后，解决方案的质量将进一步提高。

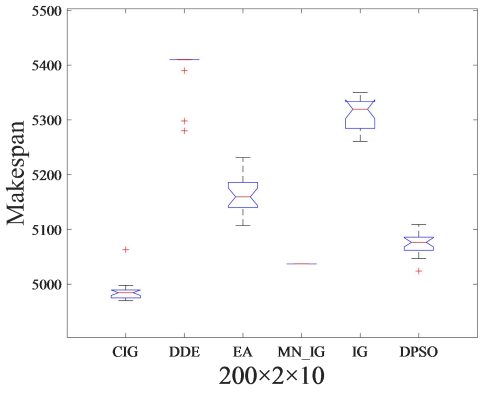
综上所述，根据对比算法的实验结果，所提 CIG 算法最适合求解以最大完工时间为优化目标的 DBHFSP。

表 5-8 Friedman 测试的实验结果（α = 0.05）

算法	Ranks	N	Mean	Std. Deviation	Min	Max
CIG	**1.05**	150	**1.91**	**1.84**	**0.00**	**14.66**
DDE	5.47	150	16.70	5.95	6.13	24.83
EA	3.13	150	9.31	4.70	1.16	18.38
MN_IG	4.64	150	12.77	7.66	4.83	26.14
IG	4.13	150	11.47	4.94	1.99	20.32
DPSO	2.58	150	8.17	5.27	0.47	18.42



(a)



(b)

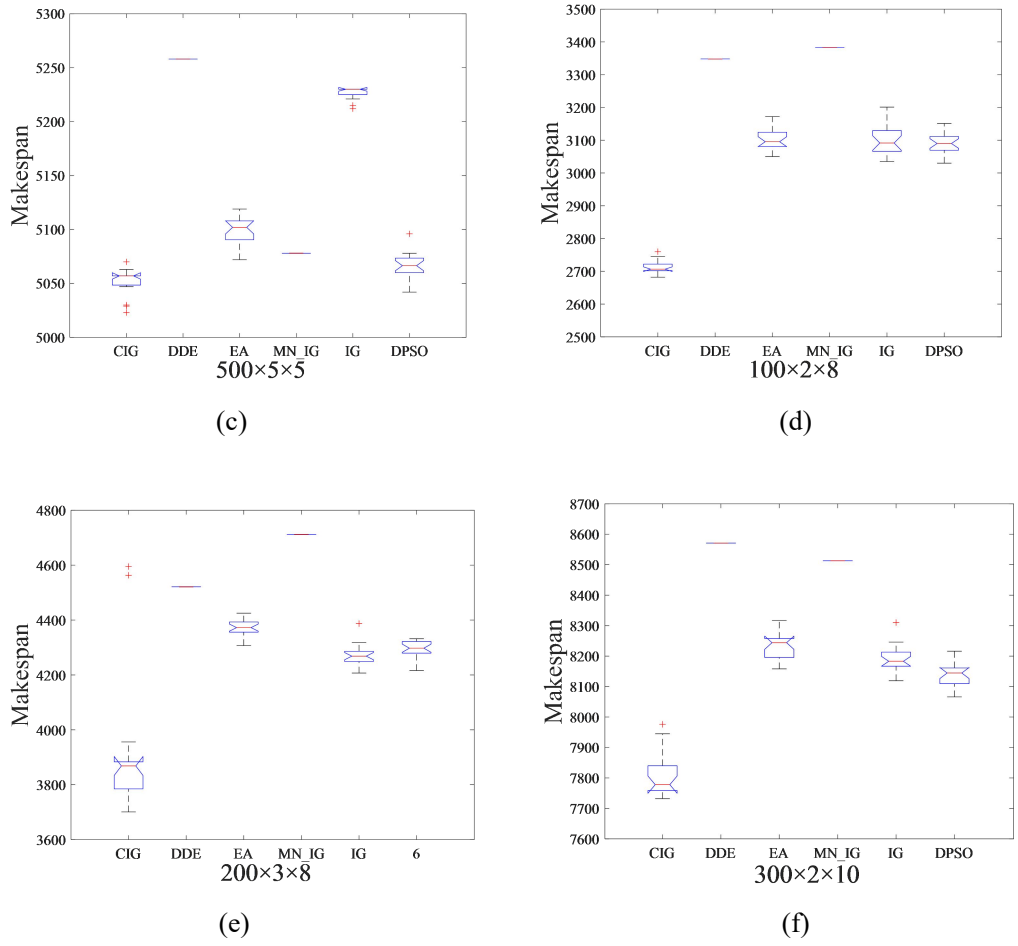


图 5-7 所有对比算法的盒图

5.5 本章小结

本章提出了一种有效的 CIG 算法来求解 DBHFSP 以最小化工件序列的最大完工时间。为求解该问题，本章首先建立了 DBHFSP 的数学模型，随后，本章结合 *NEH_F_des* 设计了 *NEH_F_asc* 初始化策略来尽可能减少早期阻塞因素对算法性能的影响，紧接着，本章提出了带有两种搜索算子的邻域搜索策略，以提高 CIG 算法的全局搜索能力，并快速调整不同工厂的工件序列。此外，CIG 算法利用基于交换算子的局部强化策略进一步探索解的邻域，进一步提高算法的局部搜索性能。

第六章 总结与展望

调度问题普遍存在于现实生活中，譬如，制药业、化学化工业、半导体工业、炼钢连铸加工制造业等，针对问题特性研究有效的求解算法对提高企业生产水平，提高国家的国际竞争力和经济水平都起着非常重要的作用。

不同的调度问题存在着不同的约束条件及优化目标，通常为复杂的组合优化问题，若使用传统的数学计算方法在短时间内难以求得令人满意的解决方案，此时，可以使用智能优化算法来近似求解此类离散问题，尤其是在大规模实例下，智能优化算法在短时间内获得的结果通常比传统数学精确算法要好，根据问题的约束及优化目标设计相应的求解算法并不是固定不变的，另外可以结合强化学习，神经网络，大数据等主流技术进一步对调度问题进行求解。

鉴于此，本文根据现存的并行机调度，流水车间调度，阻塞约束，分布式调度环境以及成组约束等，设计了一系列针对这些特性的 IG 算法来求解。

本课题是计算机科学、自动化、控制学、管理科学、统计学、能源、数学等各个领域学科的交叉方向，其具有广阔的研究空间和应用价值，其完成的研究成果不仅可以丰富现有的调度学术领域的研究，更对现实生活中的相关调度问题具有启发作用。

6.1 本文所做的工作

针对一系列阻塞混合流水车间调度问题，本文建立了相应的数学模型并验证了其正确性和有效性，并针对问题特性设计了相应的 IG 算法，优化了调度序列的最大完工时间。其研究成果总结如下：

（1） 建立以最大完工时间为优化目标的阻塞混合流水车间调度问题的数学模型并设计了其求解算法：本文第三章首先建立了 BHFSP 的数学模型，随后提出了一种融入变异策略的双层变异 IG 算法，所提策略弥补了传统 IG 算法中全局搜索能力的不足，提高了算法的搜索性能，进而提高了解的多样性。

（2） 建立 BHFGSP 的数学模型并设计了其求解算法：第四章是对第三章内容的进一步扩展，此时将工件成组的约束考虑进 BHFSP 中，在现实生活中，为了提高企业的生产效率，在加工过程中开始使用可以处理具有相同属性工件的机

器，该操作可以减少不必要的机器换件换洗时间，进而提高产品的产出率。于是，基于这一目标，第四章建立了 BHFGSP 的数学模型并在实验部分验证了其正确性和有效性，同时设计了新的编码和解码方法，并针对工件成组和阻塞约束设计了一种新的 IG 算法来求解，所提算法基于不同组和基于组内工件序列的排序问题，共提出了 8 种新的局部搜索算子，并将其嵌入到邻域搜索策略中来寻找性能较好的解决方案。最后的仿真实验验证了所提策略的高效性。

（3） 建立 DBHFSP 的数学模型并设计了其求解算法：第五章是对第三章内容的进一步扩展，将分布式的调度环境考虑进 BHFSP 中。在该问题中，要解决的问题是为工厂分配工件以及安排工件序列的加工顺序。在第五章中，首先建立了该问题的数学模型，并使用 CPLEX 验证了其有效性，其次，针对其问题特性提出了一种新的协同进化 IG 算法，在所提算法中，使用双个体来提高算法的全局搜索能力，并针对阻塞约束和分布式工厂设计了相应的初始化策略以及邻域搜索策略，最后的仿真实验验证了所提算法的性能。

6.2 未来展望

本文针对 BHFSP 提出了多种不同的数学模型并根据其问题特性设计了高效的求解算法，但本文所做的工作只是本领域内很小的一部分，目前仍尚存很多问题需要进一步深入研究。

如，本文研究了在分布式调度环境下以及工件成组中的 BHFSP，可以针对以上问题同时考虑加工的能耗指标，并在此基础上加入机器的关闭重启策略，但关闭重启次数不应该过多，若启动关闭多次，则有可能对机器造成损害，从而减少了机器的使用寿命。因此，有必要在此基础上做进一步的改进措施，如降低机器的运行速率但不完全关闭，这样既能在一定程度避免机器寿命上的损耗又能减少加工过程中的能源消耗。此外，一些不确定性的情况也有可能发生，如机器的故障，若机器在加工过程中突然发生故障，则应采取什么措施来防止生产的停滞也是值得探讨的研究课题。

除以上考虑外，三个及以上的优化目标也是可以研究的方向，如机器的能源消耗，工件序列的最大完工时间，总流经时间，总延迟及总延误等优化目标同时作为优化目标的情况，此时，数学模型应该重新设计，算法策略可能会进行更大程度的修改。

参考文献

[1] 黄敏, 付亚平, 王洪峰, 等. 设备带有恶化特性的作业车间调度模型与算法[J]. 自动化学报, 2015, 41(3):551-558.

[2] 王凌. 车间调度及其遗传算法[M]. 北京:清华大学出版社, 2003.5.

[3] 俞胜平, 柴天佑, 郑秉霖. 炼钢连铸混合智能优化调度方法及应用[J]. 系统工程学报, 2010, 25(3): 379-386.

[4] 韩彪, 江永亨, 王凌, 等. 基于即时交货的离散时间模型及其在炼油过程调度优化中的应用[J]. 控制与决策, 2020, 35(6): 1361-1368.

[5] 俞胜平, 柴天佑, 郑秉霖. 炼钢连铸混合智能优化调度方法及应用[J]. 系统工程学报, 2010, 25(3): 379-386.

[6] 俞胜平, 柴天佑. 开工时间延迟下的炼钢-连铸生产重调度方法[J]. 自动化学报, 2016, 42(3): 358-374.

[7] 张龙, 许川佩, 刘民, 等. 微电子生产过程调度问题基于指标快速预报的分解算法[J]. 控制与决策, 2020,35(1): 139-146.

[8] Pan Q. K., Tasgetiren M. F., Suganthan P. N., et al. A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem[J]. Information sciences, 2011, 181(12): 2455-2468.

[9] Lei D. M., Gao L., and Zheng Y. L. A novel teaching-learning-based optimization algorithm for energy-efficient scheduling in hybrid flow shop[J]. IEEE Transactions on Engineering Management, 2017, 65(2): 330–340.

[10] Li J. Q., Pan Q. K., Mao K. A Hybrid Fruit Fly Optimization Algorithm for the Realistic Hybrid Flowshop Rescheduling Problem in Steelmaking Systems[J]. IEEE Transactions on Automation Science & Engineering, 2016, 13(2): 932-949.

[11] Pan Q. K., Wang L., Mao K., Zhao J. H., Zhang M. An effective artificial bee colony algorithm for a real-world hybrid flowshop problem in steel making process[J]. IEEE Transactions on Automation Science and Engineering. 2013, 10(2): 307–322.

[12] Naderi B., Ruiz R., and Zandieh M. Algorithms for a realistic variant of flowshop scheduling[J]. Computers & Operations Research, 2010, 37(2): 236–246.

[13] Salvador M. S. A Solution to a Special Class of Flow Shop Scheduling Problems, Symposium on the Theory of Scheduling and Its Applications[J]. Springer Berlin Heidelberg, 1973: 83-91.

[14] Ruben R., JA Vázquez-Rodríguez. The hybrid flow shop scheduling problem[J]. European Journal of Operational Research, 2010, 205(1): 1-18.

[15] Kis T., Pesch E. A review of exact solution methods for the non-preemptive multiprocessor flowshop problem[J]. European Journal of Operational Research, 2005, 164(3): 592-608.

[16] Hua X., Tang L. Scheduling a hybrid flowshop with batch production at the last stage[J]. Computers & Operations Research, 2007, 34(9): 2718-2733.

[17] Nawaz M. M., Enscore E., Ham J. I. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem[J]. Omega, 1983, 11(1): 91-95.

[18] Ronconi D. P. A note on constructive heuristics for the flowshop problem with blocking[J]. International Journal of Production Economics, 2004, 87(1): 39-48.

[19] 唐立新, 吴亚萍. 混合流水车间调度的遗传下降算法[J]. 自动化学报, 2002, 28(4): 637-641.

[20] Nejati M., Mahdavi I., Hassanzadeh R., et al. Multi-job lot streaming to minimize the weighted completion time in a hybrid flow shop scheduling problem with work shift constraint[J]. The International Journal of Advanced Manufacturing Technology, 2014, 70(1): 501-514.

[21] 雷德明, 杨冬婧. 基于新型蛙跳算法的低碳混合流水车间调度[J]. 控制与决策, 2020, 35(6): 1329-1337.

[22] 王圣尧, 王凌, 许烨, 等. 求解混合流水车间调度问题的分布估计算法[J]. 自动化学报, 2012, 38(3): 437-443.

[23] 崔琪, 吴秀丽, 余建军. 变邻域改进遗传算法求解混合流水车间调度问题[J]. 计算机集成制造系统, 2017, 23(9): 1917-1927.

[24] Zhang B., Pan Q. K., Gao L., et al. An effective modified migrating birds optimization for hybrid flowshop scheduling problem with lot streaming[J]. Applied Soft Computing, 2017, 52: 14-27.

[25] Pan Q. K., Wang L., Li J. Q., et al. A novel discrete artificial bee colony algorithm for the hybrid flowshop scheduling problem with makespan minimisation[J]. Omega, 2014, 45: 42-56.

[26] Marichelvam M. K., Geetha M., Tosun Ö. An improved particle swarm optimization algorithm to solve hybrid flowshop scheduling problems with the effect of human factors — A case study[J]. Computers & Operations Research, 2020, 114: 1-9.

[27] Qin W., Zhuang Z. L., Yang L., and Tang Ou. A two-stage ant colony algorithm for hybrid flow shop scheduling with lot sizing and calendar constraints in printed circuit board assembly[J]. Computers & Industrial Engineering, 2019, 138: 1-12.

[28] Hz A., Bna B., Mm A., and V. C, Reformulation, linearization, and a hybrid iterated local search algorithm for economic lot-sizing and sequencing in hybrid flow shop problems[J]. Computers & Operations Research, 2019, 104:127-138.

[29] Kurdi M. Ant colony system with a novel non-daemonactions procedure for multiprocessor task scheduling in multistage hybrid flow shop[J]. Swarm and Evolutionary Computation, 2018, 44: 1-16.

[30] Liu S. W., Pei J., Cheng H., Liu X. B., et. al. Two-stage hybrid flow shop scheduling on parallel batching machines considering a job dependent deteriorating effect and non-identical job sizes[J]. Applied Soft Computing, 2019, 84: 1-15.

[31] Ztop H., Tasgetiren M. F., Eliiyi D. T., and Pan Q. K. Metaheuristic algorithms for the hybrid flowshop scheduling problem[J]. Computers &Operations Research, 2019, 111:177-196.

[32] Yu C., Semeraro Q., and Matta A. A genetic algorithm for the hybrid flow shop scheduling with unrelated machines and machine eligibility[J]. Computers & Operations Research, 2018, 100(12): 211-229.

[33] Tao X. R., Li J. Q., Huang T. H., et al. Discrete imperialist competitive algorithm for the resource-constrained hybrid flowshop problem with energy consumption[J]. Complex & Intelligent Systems, 2021, 7: 311-326.

[34] Zhang Z., Wu L., Peng T., et al. An improved scheduling approach for minimizing total energy consumption and makespan in a flexible job shop environment[J]. Sustainability, 2018, 11(1): 1-21.

[35] Lei D. M., Li M., Wang L. A two-phase meta-heuristic for multi objective flexible job shop scheduling problem with total energy consumption threshold[J]. IEEE Transactions on Cybernetics, 2019, 49(3): 1097-1109.

[36] Zhang B., Pan Q. K., Gao. L., Meng L. L., Li X. Y., Peng K. K. A three-stage multi objective approach based on decomposition for an energy-efficient hybrid flow shop scheduling problem[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2020, 50(12): 4984-4999.

[37] Peng K., Pan Q. K., Gao L., et al. An improved artificial bee colony algorithm for real-world hybrid flowshop rescheduling in steelmaking-refining continuous casting process[J]. Computers & Industrial Engineering, 2018, 122(8): 235-250.

[38] Choi H. S., Kim J. S., Lee D. H. Real-time scheduling for reentrant hybrid flow shops: A decision tree based mechanism and its application to a TFT-LCD line[J]. Expert Systems with Applications, 2011, 38(4): 3514-3521.

[39] Kim Y. D., Shim S. O., Choi B., et al. Simplification methods for accelerating simulation-based real-time scheduling in a semiconductor wafer fabrication facility[J]. IEEE Transactions on Semiconductor Manufacturing, 2003, 16(2): 290-298.

[40] Luo H., Huang G. Q., Zhang Y., et al. Two-stage hybrid batching flowshop scheduling with blocking and machine availability constraints using genetic algorithm[J]. Robotics and computer-integrated manufacturing, 2009, 25(6): 962-971.

[41] Nakkaew P., Kantanantha N., Wongthatsanekorn W. A comparison of genetic algorithm and artificial bee colony approaches in solving blocking hybrid flowshop scheduling problem with sequence dependent setup/changeover times[J]. KKU engineering journal, 2016, 43(2).

[42] Elmi A., Topaloglu S. A scheduling problem in blocking hybrid flow shop robotic cells with multiple robots[J]. Computers & Operations Research, 2013, 40(10): 2543-2555.

[43] Missaoui A., Boujelbene. Y. An effective iterated greedy algorithm for blocking hybrid flow shop problem with due date window[J]. RAIRO - Operations Research, 2021, 55(3): 1603-1616.

[44] Moccellin J. V., Nagano M. S., Neto A., et al. Heuristic algorithms for scheduling hybrid flow shops with machine blocking and setup times[J]. Journal of the Brazilian Society of Mechanical Sciences & Engineering, 2018, 40(2): 40.

[45] Meng. L. L., Zhang C., Ren C., et al. MILP Models and an Improved BSA for Hybrid Flow Shop Scheduling Problem with Blocking[J]. Zhongguo Jixie Gongcheng/China Mechanical Engineering, 2018.

[46] Meng L. L., Zhang C., Zhang B., and Li J. Q. Modeling of energy saving blocking hybrid flow shop scheduling problem[J]. Journal of Huazhong University of science and technology, 2021，49(7): 127-132.

[47] Trabelsi W., Sauvey C., Sauer N. Mathematical model and lower bound for hybrid flowshop problem with mixed blocking constraints[J]. IFAC Proceedings Volumes, 2012, 45(6): 1475-1480.

[48] Aqil S., Allali K. Two efficient nature inspired meta-heuristics solving blocking hybrid flow shop manufacturing problem[J]. Engineering Applications of Artificial Intelligence, 2021, 100:104196.

[49] Deng J., Wang L. A competitive memetic algorithm for multi-objective distributed permutation flow shop scheduling problem[J]. Swarm and Evolutionary Computation, 2016: 107-112.

[50] Wang G. C., Gao L., Li X. Y., et al. Energy-efficient distributed permutation flow shop scheduling problem using a multi-objective whale swarm algorithm[J]. Swarm and Evolutionary Computation, 2020, 57.

[51] Shao Z. S., Shao W. S., Pi D. C. Effective heuristics and metaheuristics for the distributed fuzzy blocking flow-shop scheduling problem[J]. Swarm and Evolutionary Computation, 2020, 59:100747.

[52] Wang J. J., Wang L. A bi-population cooperative memetic algorithm for distributed hybrid flow-shop scheduling. IEEE Transactions on Emerging Topics in Computational Intelligence, 2021, 99:1-15, DOI: 10.1109/TETCI.2020.3022372.

[53] Naderi B., Ruiz R. The distributed permutation flowshop scheduling problem[J]. Computers & Operations Research, 2010, 37(4): 754-768.

[54] Hatami R., Ruiz S., and Andres-Romano C., The distributed assembly permutation flowshop scheduling problem[J]. International Journal of Production Research, 2013, 51(17-18): 5292-5308.

[55] Li J. Q., Song M. X., Wang L., et al. Hybrid artificial bee colony algorithm for a parallel batching distributed flow shop problem with deteriorating jobs[J]. IEEE Transactions on Cybernetics, 2020, 50: 2425-2439.

[56] Lin S. W., and Ying K. C. Minimizing makespan for solving the distributed no-wait flow shop scheduling problem[J]. Computers & Industrial Engineering, 2016, 99(9): 202-209.

[57] Shao W. S., Pi D. C., and Shao Z. S. A pareto-based estimation of distribution algorithm for solving multi objective distributed no-wait flow shop scheduling problem with sequence-dependent setup time[J]. IEEE Transactions on Automation Science and Engineering, 2019, 16(3): 1344-1360.

[58] Pan Q. K., Gao L., Wang L., Liang J., and Li X. Y., Effective heuristics and metaheuristics to minimize total flowtime for the distributed permutation flowshop problem[J]. Expert Systems with Applications, 2019, 124(6): 309-324.

[59] Fu Y. P., Tian G. D., et al. Stochastic multi objective modelling and optimization of an energy-conscious distributed permutation flow shop scheduling problem with the total tardiness constraint[J]. Journal of Cleaner Production, 2019, 226: 515-525.

[60] Lin J., Wang Z. J., and Li X. A backtracking search hyper-heuristic for the distributed assembly flow-shop scheduling problem[J]. Swarm and Evolutionary Computation, 2017, 36: 124-135.

[61] Wang G. C., Gao L., Li X. Y., et al. Energy-efficient distributed permutation flow shop scheduling problem using a multi objective whale swarm algorithm[J]. Swarm and Evolutionary Computation, 2020, 57: 1-17.

[62] Huang J. P., Pan Q. K., and Gao L. An effective iterated greedy method for the distributed permutation flowshop scheduling problem with sequence-dependent setup times[J]. Swarm and Evolutionary Computation, 2020, 59: 1-18.

[63] Fernandez V., Perez-Gonzalez P., and Framinan J. M. The distributed permutation flow shop to minimise the total flowtime[J]. Computers & Industrial Engineering, 2018, 118: 464-477.

[64] Bargaoui H., Belkahla D. O., and Ghedira K. A novel chemical reaction optimization for the distributed permutation flowshop scheduling problem with makespan criterion[J]. Computers & Industrial Engineering, 2017, 111: 239-250.

[65] Gao J., Chen R., and Deng W. An efficient tabu search algorithm for the distributed permutation flowshop scheduling problem[J]. International Journal of Production Research, 2013, 51(3-4): 641-651.

[66] Rifai A. P., Nguyen H. T., and Dawal S. Z. Multi-objective adaptive large neighborhood search for distributed reentrant permutation flow shop scheduling[J]. Applied Soft Computing, 2016, 40: 42-57.

[67] Pan Q. K., Gao L., et al. Effective constructive heuristics and meta-heuristics for the distributed assembly permutation flowshop scheduling problem[J]. Applied Soft Computing. 2019, 81: 105492.

[68] Ruiz R., Pan Q. K., and Naderi B., Iterated greedy methods for the distributed permutation flowshop scheduling problem[J]. Omega, 2019, 83(3): 213-222.

[69] Ochi H., and Driss O. B. Scheduling the distributed assembly flowshop problem to minimize the makespan[J]. Procedia Computer Science, 2019, 164: 471-477.

[70] Huang Y. Y., Pan Q. K., Huang J. P., et. al. An improved iterated greedy algorithm for the distributed assembly permutation flowshop scheduling problem[J]. Computers & Industrial Engineering, 2020, 152(3): 1-11.

[71] Shao Z., Shao W., and Pi D. Effective constructive heuristic and iterated greedy algorithm for distributed mixed blocking permutation flow-shop scheduling problem[J]. Knowledge-Based Systems, 2021, 221(5): 1-19.

[72] Chen S., Pan Q. K., and Gao L. Production scheduling for blocking flowshop in distributed environment using effective heuristics and iterated greedy algorithm[J]. Robotics and Computer-Integrated Manufacturing, 2021, 71(3): 1-16.

[73] Ying K. C., and Lin S. W. Minimizing makespan for the distributed hybrid flowshop scheduling problem with multiprocessor tasks[J]. Expert Systems with Applications, 2018, 92(2): 132-141.

[74] Lei D. M., and Wang T. Solving distributed two-stage hybrid flowshop scheduling using a shuffled frog-leaping algorithm with memeplex grouping[J]. Engineering Optimization, 2019, 12: 1-14.

[75] Shao W. S., Shao Z. S., and Pi D. C. Modeling and multi-neighborhood iterated greedy algorithm for distributed hybrid flow shop scheduling problem[J]. Knowledge-Based Systems, 2020, 194: 1-17.

[76] Zheng J., Wang L., and Wang J. J. A cooperative coevolution algorithm for multi-objective fuzzy distributed hybrid flow shop[J]. Knowledge Based Systems, 2020, 194: 1-11.

[77] Cai J. C., Zhou R., and Lei D. M. Dynamic shuffled frog-leaping algorithm for distributed hybrid flow shop scheduling with multiprocessor tasks[J]. Engineering Applications of Artificial Intelligence, 2020, 90: 1-13.

[78] Li Y. L., Xin Y. L., Gao L., and Meng L. L. An improved artificial bee colony algorithm for distributed heterogeneous hybrid flowshop scheduling problem with sequence-qependent setup times[J]. Computers & Industrial Engineering, 2020, 147: 1-12.

[79] Wilson A. D., King R. E., and Hodgson T. J., Scheduling non-similar groups on a flow line: Multiple group setups[J]. Robotics and Computer-Integrated Manufacturing, 2004, 20(6): 505-515.

[80] Schaller J. E., Gupta J. N. D., and Vakharia A. J. Scheduling a flowline manufacturing cell with sequence dependent family setup times[J]. European Journal of Operational Research, 2000, 125(2): 324-339.

[81] Salmasi N., Logendran R., and Skandari M. R., Total flow time minimization in a flowshop sequence-dependent group scheduling problem[J]. Computers & Operations Research, 2010, 37(1): 199-212.

[82] He X., Pan Q. K., Gao L., Wang L., Ponnuthurai N.S. A Greedy Cooperative Co-evolutionary Algorithm with Problem-specific Knowledge for Multi-objective Flowshop Group Scheduling Problems[J]. IEEE Transactions on Evolutionary Computation, 2021. https://doi.org/10.1109/TEVC.2021.3115795.

[83] Pan Q. K., Gao L., Wang L. An Effective Cooperative Co-Evolutionary Algorithm for Distributed Flowshop Group Scheduling Problems[J]. IEEE Transactions on Cybernetics, 2020, 99: 1-14.

[84] Lin S. W., and Ying K. C. Makespan optimization in a no-wait flowline manufacturing cell with sequence-dependent family setup times[J]. Computers and Industrial Engineering, 2019, 128: 1-7.

[85] Neufeld J. S., Gupta J. N. D., and Buscher U. Minimising makespan in flowshop group scheduling with sequence-dependent family set-up times using inserted idle times[J]. International Journal of Production Research, 2015, 53(6): 1791-1806.

[86] Feng H., Xi L., Xiao L., Xia T., and Pan E. Imperfect preventive maintenance optimization for flexible flowshop manufacturing cells considering sequence-dependent group scheduling[J]. Reliability Engineering & System Safety, 2018, 176: 218-229.

[87] Neufeld J. S., Gupta J. N. D., and Buscher U. A comprehensive review of flowshop group scheduling literature[J]. Computers & Operations Research, 2016, 70: 56-74.

[88] Costa A., Cappadonna F. A., and Fichera S., A hybrid genetic algorithm for minimizing makespan in a flow-shop sequence-dependent group scheduling problem[J]. Journal of Intelligent Manufacturing, 2017, 28(6): 1269-1283.

[89] Costa A., Cappadonna F. V., and Fichera S. Minimizing makespan in a Flow Shop Sequence Dependent Group Scheduling problem with blocking constraint[J]. Engineering Applications of Artificial Intelligence, 2020, 89: 103413.

[90] Naderi B., and Salmasi N. Permutation flowshops in group scheduling with sequence-dependent setup times[J]. European Journal of Industrial Engineering, 2012, 6(2): 177-198.

[91] Liou C. D., and Hsieh Y. C. A hybrid algorithm for the multi-stage flow shop group scheduling with sequence-dependent setup and transportation times[J]. International Journal of Production Economics, 2015, 170: 258-267.

[92] Keshavarz T., Salmasi N., and Varmazyar M. Flowshop sequence dependent group scheduling with minimisation of weighted earliness and tardiness[J]. European Journal of Industrial Engineering, 2019, 13(1): 54–80.

[93] Ruben R., Thomas S. A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem[J]. European Journal of Operational Research, 2007, 177(3): 2033-2049.

[94] Ding J. Y., et al. An improved iterated greedy algorithm with a Tabu-based reconstruction strategy for the no-wait flowshop scheduling problem[J]. Applied Soft Computing, 2015, 30: 604-613.

[95] Fernandez V., Valente J. M. S., Framinan J. M. Iterated-greedy-based algorithms with beam search initialization for the permutation flowshop to minimise total tardiness[J]. Expert Systems with Applications, 2018, 94(15): 58-69.

[96] Ying K. C. An iterated greedy heuristic for multistage hybrid flowshop scheduling problems with multiprocessor tasks[J]. Journal of the Operational Research Society, 2009, 60(6): 810-817.

[97] Rodriguez F. J., Lozano M., Blum C., et al. An iterated greedy algorithm for the large-scale unrelated parallel machines scheduling problem[J]. Computers & Operations Research, 2013, 40(7): 1829-1841.

[98] Fbo A., Ms B. Iterated greedy algorithms enhanced by hyper-heuristic based learning for hybrid flexible flowshop scheduling problem with sequence dependent setup times: A case study at a manufacturing plant[J]. Computers & Operations Research, 2020, 125: 105044.

[99] 宋存利. 求解混合流水车间调度的改进贪婪遗传算法[J]. 系统工程与电子技术, 2019, 41(5): 148-155.

[100] Osman I., Potts C. Simulated annealing for permutation flow-shop scheduling[J]. Omega, 1989, 17(6): 551-557.

[101] Nejati M., Mahdavi I., Hassanzadeh R, et al. Multi-job lot streaming to minimize the weighted completion time in a hybrid flow shop scheduling problem with work shift constraint[J]. International Journal of Advanced Manufacturing Technology, 2014, 70(1): 501-514.

[102] Bargaoui H., Belkahla D. O., Khaléd G. A novel chemical reaction optimization for the distributed permutation flowshop scheduling problem with makespan criterion[J]. Computers & Industrial Engineering. 2017, 111: 239-250.

[103] Wang J. J., and Wang L. A cooperative memetic algorithm with learning-based agent for energy-aware distributed hybrid flow-shop scheduling[J]. IEEE Transactions on Evolutionary Computation, 2021, doi: 10.1109/TEVC.2021.3106168.

[104] Wang S. Y., Wang L. An estimation of distribution algorithm-based memetic algorithm for the distributed assembly permutation flow-shop scheduling problem[J]. IEEE Transactions on Systems Man & Cybernetics Systems, 2015, 46(1): 139-149.

[105] Aqil S., Allali K. Local search metaheuristic for solving hybrid flow shop problem in slabs and beams manufacturing[J]. Expert Systems with Applications, 2020, 162: 113716.

[106] Qin H. X., Han Y. Y., Chen Q. D., et al. A Double Level Mutation Iterated Greedy Algorithm for Blocking Hybrid Flow Shop Scheduling[J]. Control and Decision, 2021: 1-10. https://doi.org/10.13195/j.kzyjc.2021.0607.

[107] Taillard E. Benchmarks for basic scheduling problems[J]. European Journal of Operational Research, 1993, 64: 278-285.

[108] Zhang G., Xing K., Cao F. Discrete differential evolution algorithm for distributed blocking flowshop scheduling with makespan criterion[J]. Engineering Applications of Artificial Intelligence, 2018, 76: 96-107.

致 谢

转眼间，三年的研究生生活即将步入尾声，在这三年的时光里，我学到了很多，收获了很多，回首过往，我心中充满了感激，感谢在我的研究生生涯里一直帮助我的人！

从步入校门的那一刻我就决定，一定要好好读研，不辜负自己的每一天，幸运的是，我遇到了自己人生中的贵人，我的导师：韩玉艳老师。

自己从开始的科研小白到今天这一步，每一次挫折每一处苦难，我走过的，韩老师也一同陪着我走了一遍。不管何时，老师都会很有耐心的对我论文的写作进行详细指导。从帮我梳理研究思路和论文逻辑，到代码讨论，论文撰写，软件使用，再到论文投稿，送审，返修，录用为止，可以说，论文的每一处都有着韩老师的心血。至今，我仍忘不了有一次韩老师无意间说过的一句话：“大年三十那天还帮你改论文呢。”

凭着这一句话，我更加下定决心要多做些研究成果出来，在这里，向韩老师致以最崇高的敬意和最真挚的感谢!

不光如此，韩老师还给我介绍了许多其他杰出的老师，多亏了这些杰出、无私、有耐心的好老师们，我的科研能力再次得到了极大地提升，在这里，要特别感谢清华大学的王凌老师，上海大学的潘全科老师，中国矿业大学的巩敦卫老师，华南理工大学的黄翰老师，向毅老师，湖南大学的刘益萍老师，东北大学的陈庆达老师，南方科技大学的 Hisao Ishibuchi 老师，柳青老师，彭丽敏老师，土耳其巴斯干大学的 Tasgetiren, M. Fatih 老师，聊城大学的李俊青老师，桑红燕老师，王玉亭老师，张彪老师，孟磊磊老师，高开周老师，段朋老师，感谢以上老师对我研究问题以及论文撰写过程中的耐心解答与指导。此外，还要衷心的感谢贾仰理老师，贾保先老师，姜华老师，赵阳老师，荆雪蕾老师，赵传申老师，郑丽萍老师，赵海勇老师，对我生活和学习的帮助，没有这些老师的栽培，就不会有今天的我。

此外，还要感谢聊城大学的韩雪，李晗潇以及杨历阁学长，华中科技大学的黄江平学姐，以及上海大学的陶昕瑞学长与何轩学长，感谢你们对我论文写作的帮助。感谢我的舍友们，宁上峰，于振翱，林臻，以及 2020 级各位同学平日的陪伴与帮助。 此外，还要感谢我的师弟师妹们，王宇航，王勇，李成帅，张晨瑶，刘金黎，纪千惠，

李文龙，刘超和王一正，很高兴与你们一同度过了读研期间的愉快时光。

将特别的感谢送给我的父母，爷爷奶奶，姥爷姥姥，感谢你们的养育之恩和对我学习生活上的支持与理解。将真挚的感谢送给我的女朋友薛暖颜，感谢这三年期间里你的陪伴，帮助，支持，你的鼓励一直是我前进的最大动力，真的非常感谢你!!

最后，衷心的感谢所有评审组和负责答辩的老师们！感谢你们默默无闻的奉献，这将不断激励着我在攀登学术高峰的道路上继续勇往直前。

攻读硕士学位期间取得的学术成果和奖励

[1] **Qin H. X.,** Han Y. Y., Chen Q. D., Wang L., Wang Y.T., Li J.Q., and Liu Y. P. Energy Efficient Iterative Greedy Algorithm for the Distributed Hybrid Flow Shop Scheduling with Blocking Constraints. IEEE Transactions on Emerging Topics in Computational Intelligence. (Accepted, 2023, DOI: 10.1109/TECI.2023.3271331)

[2] **Qin H. X.,** Han Y. Y., Wang Y. T., Liu Y. P., Li J. Q., and Pan Q. K. Intelligent Optimization Under Blocking Constraints: A Novel Iterated Greedy Algorithm for the Hybrid Flow-Shop Group Scheduling Problems. knowledge-based systems, 2022, 258(22): 109962. (SCI 一区，对应第四章内容).

[3] **Qin H. X.,** Han Y. Y., Liu Y. P., Li J. Q., Pan Q. K., and Xue Han. A collaborative iterative greedy algorithm for the scheduling of distributed heterogeneous hybrid flow shop with blocking constraints. Expert Systems with Applications, 2022, 201: 117256. (SCI 一区，对应第五章内容).

[4] **Qin H. X.,** Han Y. Y., Zhang B., Meng L. L., Liu Y. P., Pan Q. K., and Gong D. W. An improved iterated greedy algorithm for the energy-efficient blocking hybrid flow shop scheduling problem, Swarm and Evolutionary Computation, 2021, 69: 100992. (SCI 一区).

[5] **秦浩翔，**韩玉艳，陈庆达，李俊青，桑红燕. 求解阻塞混合流水车间调度的双层变异迭代贪婪算法. 控制与决策，2021, 37(9): 2323-2332. （北大核心，EI 检索，对应第三章内容）

[6] **Qin H. X.,** Han Y. Y., Li J. Q., Wang L., Gao K. Z., Liu Y. P., and Tasgetiren M. F. A problem-specific iterative greedy algorithm for solving the blocking hybrid flow shop with machine turning on/off strategy. (SCI 在审).

[7] **Qin H. X.,** Han Y. Y., Li J. Q., Sang H. Y., Chen Q. D., Meng L. L., and Zhang B. A quick and effective iterated greedy algorithm for energy-efficient hybrid flow shop scheduling problem with blocking constraint. 2021 11th International Conference on Information Science and Technology (ICIST), 2021, May. DOI:10.1109/ICIST52614.2021.9440648. (EI 国际会议).

[8] **Qin H. X.,** Wang Y. T., Han Y. Y., Chen Q. D., and Li J. Q. Adapting a reinforcement learning method for the distributed blocking hybrid flow shop scheduling problem, The 5th Asian Conference on Artificial Intelligence Technology (ACAIT), 2022, March. DOI:10.1109/ACAIT53529.2021.9731228. (EI 国际会议).

[9] Han X., Han Y. Y., Zhang B., **Qin H. X.,** Wang Y. T., Li J. Q., Liu Y. P., and Gong D. W. An effective iterative greedy algorithm for distributed blocking flowshop scheduling problem with balanced energy costs criterion. Applied soft computing, 2022, 129: 109502. (SCI 二区).

[10] Han X., Han Y. Y., Liu Y. P., Pan Q. K., **Qin H. X.,** and Li J. Q. An improved iterated greedy algorithm for the distributed flow shop scheduling problem with sequence-dependent setup times. 2021 11th International Conference on Information Science and Technology (ICIST), 2021, May.

DOI:10.1109/ICIST52614.2021.9440648. (EI 国际会议).

[11] 2022 年度山东省研究生优秀创新成果三等奖，位次 1.