武汉理工大学

硕士学位论文

求解作业车间调度问题的禁忌演化算法

姓名：成浩

申请学位级别：硕士

专业：机械制造及其自动化

指导教师：胡业发

20060501

摘 要

现代制造型企业的主要活动之一是生产管理，即利用企业资源，根据生产任务和任务顺序约束制定和执行生产计划。有效的生产调度方法和优化技术的研究和应用是实现先进制造和提高生产效益的基础和关键。本文主要工作分为两个部分：

第一部分为作业车间调度理论算法的研究。具体包括：通过分析对国内外的研究进展，阐述精确算法和近似算法各自的特点。禁忌搜索一种求解作业车间调度问题的有效局部搜索算法，但其搜索最优解的能力取决于初始解的选择。演化算法是通过选择、交换和变异等操作使群体进化来进行全局优化搜索的，它本身不能减小搜索空间的大小，但由于群体搜索的机制使得它能有效地覆盖较大的解空间。将禁忌搜索纳入到演化计算的框架中，增加局部搜索能力，提出禁忌演化算法。新算法通过禁忌策略产生初始种群，并引入了分级策略，对种群按照适应值进行划分，对处于不同级别的个体采用不同的遗传操作。通过对个体的分级，可以区分个体在搜索过程中的职能：优秀的个体进行局部极小值的开采；其它的个体进行搜索空间的探索，以发现新的局部极小值。通过测试 Benchmark 问题，数值实验表明算法收敛速度快，结果较好。

第二部分为禁忌演化算法的应用研究。具体包括：测试与 JSSP 类似的组合优化问题 TSP 问题，并阐述将禁忌演化算法植入实际调度系统，生产排产系统中的开发过程。通过分析智能算法应用于实际调度系统中失败的原因，以此为基础提出一种智能调度框架，该框架的特点在于生产反馈信息的收集和调度器使用多种算法共同决策的模式。

关键词：作业车间调度；演化计算；禁忌搜索算法；生产调度

Abstract

Production Management is one of main activities in modern manufacture enterprise, which executes production scheduling by allocating resources in order to perform a number of tasks, which are assigned to resources in a temporal order. The research and application of effective production scheduling methods and optimization techniques are the key elements to implement modern manufacture and promote production efficiency. This paper consists of two major parts.

The 1st part is the research on theory algorithms of Job Shop Scheduling Problem (JSSP). By analyzing the domestic and foreign research developments, this paper notes exact and approximate methods' characteristics respectively. Tabu Search is an effective local search algorithm for the job shop scheduling problem, but the quality of the best solution found depends on the initial solution. Evolutionary algorithm is a global search algorithm by choosing, crossover and mutation operations to operate the population. The method itself cannot decrease the search space, but the population search mechanism can cover a large scale solution spaces. To overcome this problem we present a new Evolutionary-Tabu alogrithm (ETA) that uses a population of Tabu Search runs in an Evolutionary Computation framework. ETA adopts a search strategy to classify the individuals by their fitness. Individuals' classification differentiate respective function in search process, that's the excellent individuals mine the local optimal solution and others explore the search domain to find new local optimal solution. By testing benchmark instances the results show the new algorithm is satisfactory.

The 2nd part is the application of ETA. Testing Traveling Salesman Problem (TSP), which is a similar combination optimization problem to JSSP. In addition, we explain how to design production scheduling system by planting ETA into a real-world scheduling system. Based on analyzing the failures using intelligent algorithms in real-world scheduling system, we put forward an intelligent scheduling framework. The characteristics of the framework are collecting the production response information and the judge modal of scheduler using multi algorithms.

Keywords: job shop scheduling; evolutionary computation; tabu search; production scheduling

第一章 绪论

本章从现代生产系统的运行开始，介绍生产调度理论和技术国内外发展现状，分析相关领域的最新研究成果，明确混合优化算法求解作业车间调度问题的理论及实际意义，以及课题提出的背景、目的、和研究的内容。

1.1 前言

生产管理是制造型企业的主要活动之一，其完成的好坏取决于生产计划的制订。随着市场竞争的激烈，工厂品种不断增多，批量不断减少，生产计划不但要以市场需求和客户个性化的要求来确定，还要根据企业制造资源的实际能力和库存、生产进度的动态变化来调整，制造过程的优化和监控成为提高企业核心竞争力不可回避的环节[1],[2]。

企业信息化的不断深入，如企业资源计划系统(Enterprise Resource Planning, ERP) 集中信息技术与管理思想，反映企业需要对自身资源合理调配。但当数据集成后，却无法合理的运用，也使设计者不断反思"We've been warehousing our data for the past 10 years and I'm still not sure what to do with it?"[3]。如何做好生产计划？以应对不准确的预测和不断变化的客户需求？当内部能力发生变化时如何进行主生产计划排程与调整？如何妥善安排进度，既保证生产指标的实现，又保证企业生产秩序与生产线的相对稳定？

目前企业仍在采用传统的手工方式编制生产作业计划，这种静态的、工作量极大的计划会因生产过程中每一要素的变化而受到限制，而调整一般只能通过生产调度员的经验来进行，因此，整个生产作业计划的编制具有很大的盲目性。制定满足企业资源约束（设备、人员等）的高效的生产计划是企业当前面临的主要问题，该问题的解决，将解决企业发展的瓶颈，为企业快速发展提供良好的保障。企业如果使用实施基于生产调度自动化的 ERP，可以从各方面提升企业管理能力。现代生产系统运行的核心是生产调度方法和技术，由于现代生产系统的运行管理十分复杂，故其生产计划调度与控制十分重要。它对降低生产成本、缩短制造周期、提高生产效益均具有重要的意义。

1.2 生产调度问题的分类

对于生产调度问题，Graves[4]等人进行了分类整理。根据应用环境的不同而有不同的类型。按照职能区分调度问题的标准：(1)开环车间和闭环车间；(2)过程复杂度；(3)约束标准；(4)参数可变性；(5)调度环境。

由标准(1)，根据需求的不同，生产调度问题分为开环车间和闭环车间。在开环车间中，调度问题基于订单，而不考虑库存。在闭环车间里，调度问题基于现有库存。

由标准(2)，考虑过程复杂度，根据工作阶段和工作位置的不同，生产调度问题为单机(single machine)，多台并行机(parallel machine)，流水作业调度(flow shop)和作业车间调度(job shop)。单机调度问题是所有的操作任务都在单台机器上完成，为此存在任务的优化排队问题：多台并行机的调度问题更复杂，因而优化问题更突出；流水作业调度假设所有作业都在同样的设备上加工，并有一致的加工操作和加工顺序；作业车间调度是最典型的调度类型，不同的作业具有不同的加工操作和加工顺序，并不限制作业的加工设备。

由标准(3)，约束标准说明了车间调度要达到的目标，车间调度问题的约束标准有很多，并且非常复杂，有时还互相冲突。经常使用的调度约束标准主要有：总延迟最小、误工最少、系统/资源效率最优、资源使用平衡和生产效率最高等。

由标准(4)，参数可变性即调度问题中可变参数不确定的程度。如果参数波动的数量级比参数本身要小很多，那么这个调度问题可以称为是确定性的;反之则称为不确定性的，或称随机的。

由标准(5)，调度环境定义了一个调度问题是静态的还是动态的。如果任务的数量和相关特性并不随着时间改变，则这个问题是静态的；反之，如果任务的数量和相关特性随着时间不断的变化，则称这个问题是动态的。

1.3 生产调度理论国内外发展现状

作业车间调度问题(Job Shop Scheduling Problem, JSSP)是一类满足任务配置和顺序约束要求的资源分配问题，是最困难的组合优化问题之一，同时也可以视为排序问题。一般说来，有不同的任务在只能使用有限的资源的条件下要完成，都可以归为排序问题。当然，调度不只是排序，它还根据这个得到的排序，

确定各个任务的开始时间和结束时间[5], [6], [7], [8]。

生产调度问题作为现代制造系统的一个研究热点，由于系统建模方法的多样性，以及问题的侧重点不同，调度方法和研究对象也明显不同。就对象而言，有确定性和随机性调度、离散事件和连续事件调度、静态和动态调度等；就调度方法而言，有动态规划、排对论方法、规则调度方法等等；就调度优化目标而言，有正规性能指标和非正规性能指标，如生产成本、E/T 指标等[5]。作业车间调度问题的研究早在 1950 就展开，国内外对于生产调度问题的诸多研究可以归结为两个方面：生产调度问题的建模和生产调度问题算法设计与分析。生产调度问题建模的主要目的是根据理论研究的需要或根据实际生产系统的特点提出一个或一类生产调度问题的子问题，并以一定的形式对其进行描述主要包括：问题的语义描述、问题中各要素的描述、问题的符号表示、问题的公式化描述、问题的图论描述等它是生产调度问题算法设计与分析的基础。生产调度问题算法设计的目的是对建立的模型进行求解，获得优化的调度解。而算法分析的目的则是对算法正确性进行验证、对调度结果进行评估与预测、对算法的时间复杂度及空间复杂度进行分析等[9]。典型研究和应用具体有：

国外 LiSA (a library of scheduling algorithms)由德国 Heidemarie Bräsel, Thomas Tautenhahn, Per Willenius, Martin Harborth 等人发起，并受德国 Saxony-Anhalt 资助，从 1997 年至今一直更新，属开源软件[10]；HeuristicLab 由澳大利亚 Stefan Wagner 和 Michael Affenzeller 发起，它设计了启发示算法框架，对 JSSP 问题的研究是其中一分支，软件从 1992 年至今一直更新[11]；葡萄牙 José Fernando Gonçalves[12]，AT&T 实验室 Maurício G.C. Resende[12]等采用混合遗传算法；意大利 A.MORAGLIO 研究了局部搜索在 JSP 中的作用[13]；Starkweather 等人在一个啤酒厂中应用遗传算法解决了多目标 Job Shop 调度问题，这些目标包括工厂中平均存货时间的最小化和客户订单平均等待时间的最小化等[1]；

国内清华大学王凌出版专著《车间调度及其遗传算法》[5]；中国科学院软件研究所徐俊刚等分析了经典调度理论和实际调度问题之间存在的鸿沟，例举主要的生产调度方法和典型应用[1]；台湾中国文化大学Feng-Tse Lin 用不精确数据进行模糊调度[14]；大连铁道学院黄明等将遗传算法与禁忌搜索结合来研究机器调度问题[15]；鄢烈祥提出列队竞争算法处理函数优化组合优化，其思想也可用于JSSP加速收敛[16], [17]；东北大学彭志刚利用基于遗传算法和禁忌搜索算法结合的混合搜索算法解决一机两流的连铸生产计划编制问题[18]；浙江大学余琦玮[19]，

大连理工大学陈东升[20],南昌大学万芳[21]等[22-27]开展了遗传算法在JSSP中的运用。大连海事大学徐本强将基于MAS的调度技术植入大连奥拓有限公司的ERP系统中[28]；浙江工业大学赵巍设计基于多智能体的生产调度方法[29]；南京航空航天大学李进引入多代理系统(Multi-Agent System, MAS)，通过MAS实现动态过程和动态事件管理，通过MAS和GA共同协作构建动态车间调度系统[30]；武汉理工大学梁书锋开展了柔性车间管理系统的设计与实现的研究[31]；武汉科技大学吴培栋开展实际生产系统中考虑任务相关性的作业计划规则调度算法研究[32]；哈尔滨理工大学朱红用DNA算法求解车间调度问题[33]；华中科技大学熊禾根模具企业动态车间作业计划研究[9]。

通过对以上文献的研究，可以发现求解的方法以启发式算法为主，基于优先权规则。即从未排序的工序特定子集中选用工序的规则。如枚举法，这种策略计算量大不适于大规模计算；移动瓶颈法，这种策略能取得很好的优化结果，但算法的实现过程相当复杂，不适于工程移植；模拟退火，禁忌搜索等算法的求解依赖于参数的设定，运算时间长；鉴于精确方法仅适合于小规模问题，构造性方法优化质量较差且缺少柔性，先进的邻域搜索方法等智能优化算法在生产调度领域受到广泛重视和研究，其中以遗传算法的研究居多。其操作的基本思想是预先排出由工序组成的序列（初始种群），对这些序列进行遗传操作，以达到优化调度的性能指标。确定性算法收敛速度快，但其依赖初始点的选取，易陷入局部最优，对大规模问题无法求解。遗传算法是一种比较通用的优化算法，其有编码技术与遗传操作比较简单，优化不受限制性的约束的特点，其有全局寻优的能力。通过对MAS技术可以弥补调度理论的不足，可以和ERP结合使用，其不足之处是理论还有待进一步完善，标准化不够。模糊理论的显著特点是表达逻辑，但它本质没有获取知识的能力，模糊的规则也难以确定[14]。

本文主要研究静态作业车间调度问题，试图解决求解此类问题时算法收敛速度慢，难以用于实际调度问题的矛盾。

1.4 演化计算的研究及发展动态

演化计算(Evolutionary Computation, EC)是人工智能(Artificial Intelligence, AI)研究领域的一项学科[37], [38]。现今已经发展出很多的方法，其原理大都来自于自然界生物演化的机制－就是生物学家达尔文(Darwin)提出的“物竞天择，适者生

存"的演化论[39]。演化论的大意是说，在有限资源的环境下，生态群体(population)中的各类生物个体(individual)，必须为了生存(survival)而互相竞争(competition)。在竞争中，失败者被选择(selection)淘汰，而胜利者除了生存外，也增加它繁殖子代的机会。个体所繁殖的子代会经由各种遗传(genetic)机制，而在演化式计算的群体中，大量个体追随环境(niche)，聚集而形成各式族群(species)。不同族群也就代表了用以解决问题的不同途径与其资讯—这就是演化式计算含有多样性(diversity)以解决问题的能力[40]。尤其在大部份复杂非线性的数学问题上，因为缺乏解析决定的求解方法，演化式计算便成为最有效的数值分析工具[41],[43]。

演化计算求解问题的基本思想相当简单：由问题的候选解组成一个种群，然后通过随机变化和选择等算子进行演化。其中随机变化提供了发现新解的机制，选择则确定保持哪些解作为下一步搜索的基础[42]。

应用这类算法的历史可以追溯到二十世纪 50 年代早期，这些算法彼此之间稍有差别。如最具代表性、最基本的遗传演算法(Genetic Algorithm)，较偏数值分析的演化策略(Evolution Strategy)，介于数值分析和人工智能间的演化式规划(Evolutionary Programming)，偏向以程式表现人工智能行为的遗传规划(Genetic Programming)，适应动态环境学习的分类元系统(Classifier System)，用以观察复杂系统互动的各种生态模拟系统(Echo System and etc.)，研究人工生命(Artificial Life)的格构自动机(Cellular Automata)，模拟蚂蚁群体行为的蚁元系统(Ant System)。到了 90 年代，重要的理论和实验均证明：这些方法在功能上彼此等价，没有哪种方法比其它方法更优越。并且，随着人们对于演化计算研究兴趣的日益增加，这些方法中的许多思想也被相互借鉴、交换和修改，使得各种演化方法之间不再有明显的区分界限。

尽管解决这些问题的具体方法有所不同，但演化计算的实现却相当容易。基本过程都是一样的：

(1) 创建一个个体的种群，表示所解决问题的可行解集；

(2) 评估个体；

(3) 引入某种选择压力，保留较好个体，淘汰较差个体；

(4) 应用变化算子产生待测试的新个体。

重复执行评估—选择—变化（步骤 2-4）的过程，演化循环若干次。

EC 有以下几方面的优异特性：

(1) 以优化变量的遗传编码为运算、搜索对象。传统的优化算法往往直接对

优化变量进行优化计算，但 EC 是以优化变量的某种形式的遗传编码为运算对象，它们可以是字符串、图或抽象公式。

(2) 应用“适应值”信息，而不需应用目标函数的具体值及其它辅助信息。传统优化算法不仅依赖于目标函数的具体值，而且也需要应用目标函数的导数值等其它一些辅助信息来确定搜索方向。而 EC 通常仅使用经过目标函数变换得到的适应度值。

(3) 使用群体搜索策略。传统优化算法往往是从解空间中的一个初始点开始进行单点迭代，形成解空间中的一条轨迹。EC 的搜索是由多个个体所组成的初始种群起始的种群空间中的迭代过程，形成个体空间中的多条轨迹，其搜索过程的每一步利用了种群中各个体所提供的群体信息。

(4) 使用随机搜索机制。很多传统优化算法使用确定性搜索方法，从一个搜索点到另一个搜索点的转移有确定的转移关系，使得算法的搜索具有定向性，其数值稳定性不好。EC 使用一类导向的随机搜索技术，其杂交、变异等操作以概率方式进行，它能以一定概率接受适应值较差的个体，从而提高算法跳出局部极小值陷阱的能力。

可以看出，EC 具有通用、并行、稳健、简单与全局优化能力强等突出优点。JSSP 调度是生产调度领域的典型代表，EC 则是智能优化技术的代表，因此基于 EC 的 JSSP 研究具有重要的理论意义和工程价值。

1.5 本文的主要工作

本文的研究得到了国家自然科学重点基金“在网络环境下的数字制造基础理论与关键技术”(编号：50335020)和湖北省数字制造重点实验室基金(编号：SZ0404)的资助。

本文将从以下五个方面对其进行具体分析，具体章节安排如下：

第一章分析了现代企业对生产调度的要求，明确进行作业车间调度算法研究选题的意义和国内外的研究现状，描述以演化计算结合禁忌搜索方法院的研究的主要思路和主要内容。

第二章描述作业车间调度问题的定义，分类，表示并建立数学模型。

第三章确定以混合优化算法为主要的研究手段，介绍已有的求解调度问题的启发示算法，描述其于相似性的演化算法及其与禁忌搜索算法结合的新算法。

第四章以典型的作业车间调度问题的 Benchmark 测试算例对算法进行了试验，并进行了性能分析。

第五章为禁忌演化算法的应用，测试与 JSSP 类似的组合优化问题 TSP 问题，并阐述将禁忌演化算法植入实际调度系统，生产排产系统中的开发过程。

文章最后对本文进行了总结和展望。

第二章 作业车间调度问题

2.1 JSSP 问题的定义

上世纪初，企业工程师和管理顾问讨论“如何分派(dispatching)工作以及能够胜任这种作的调度员(scheduler)”。随着工业化程度的深入，人们在理论研究中将车间作业调度问题常定义为“分配一组资源来执行一组任务”。如今，车间调度主要是针对一项可分解的工作(如产品制造)，探讨在尽可能满足约束条件(如交货期、工艺路线、资源情况)的前提下，通过下达生产指令，安排其组成部分(操作)使用哪些资源、其加工时间及加工的先后顺序，以获得产品制造时间或成本的最优化[1]。Kishan Mehrotra 博士[51]有如下的定义 In the job shop scheduling problem (JSSP), a finite set of jobs is processed on a finite set of machines. Each job is characterized by a fixed order of operations, each of which is to be processed on a specific machine for a specified duration. Each machine can process at most one job at a time and once a job initiates processing on a given machine it must complete processing uninterrupted. The objective of the JSSP is to find a “schedule” that minimizes the “makespan” of the jobs. 今后，随着集成化程度的加深，人工智能，实时反馈等手段的运用，作业车间调度将不在仅仅局限于车间-机器-工件-工序这样的关系，会与制造企业的信息流、组织结构、决策方式关联起来。

2.2 JSSP 问题的计算复杂性

算法的时间和空间复杂性对计算机的求解能力有重大影响。算法对时间和空间的需要量称为算法的时间复杂性和空间复杂性。按照计算复杂性理论研究问题求解的难易程度，可把问题分为P类、NP类和NP完全类。作业车间调度问题不仅是NP难解，还被认为是最难的组合最优化问题之一，至今没有找到可以精确求得最优解的多项式时间算法[5]。

JSSP问题解空间容量巨大，$n\times m$ JSSP 包含 $(n!)^m$种排列。一个20x10问题有 7.2651×10^{183} 个可行解。

JSSP 问题的研究大量局限于理想的数学模型，则大量的具有建模困难、计算复杂、存在动态性、随机性、约束性以及多目标等难点的实际问题仍无法得到解决。该如何量化影响生产的因素，这些因素通常看起来无法估量。

2.3 JSSP 的描述方法

2.3.1 甘特图表示

对于 m 台机器(machine)$\{M1,M2,...Mm\}$,n 个工件(job)$\{J1,J2,..Jn\}$的加工过程，分配各工件在各机器上的加工时间。调度通常用甘特图表示。甘特图（Gantt chart）是在 1917 年由亨利甘特（Henry Laurence Gantt）开发的[7]。它基本上是一种线条图，横轴表示时间，纵轴表示要安排的活动，线条表示在整个期间上计划的和实际的活动完成情况。甘特图直观地表明任务计划在什么时候进行，以及实际进展与计划要求的对比。甘特图使车间的计划安排情况一目了然.成为管理人员了解全局，安排车间进度的有效工具。

以一个 3 个工件 3 个机器的 JSSP 为例说明，表 2-1 中每个括弧中第一个数据表工件在哪台机器上加工，第二个数据表示加工时间。图 2-1 表示一个可行调度，以甘特图表示。

表 2-1 一个 3×3 JSSP（3 工件，3 机器）问题

Job	Machine Sequence		
1	(3, 1)	(1, 2)	(2, 2)
2	(3, 1)	(2, 4)	(1, 1)
3	(2, 2)	(3, 3)	(1, 1)

2.3.2 析取图表示

析取图是描述 JSP 的常用工具，对于 n 个工件、m 台机器(共 N 个操作)的 JSP，所对应的析取图 $G=(V,A,E)$如图 2-2 示。其中，V 为所有操作构成的顶点集，包括 0 和 $N+1$ 两个虚拟操作(分别表示加工开始和结束)；A 为 n 条子边构成的边集，子边（实线）表示某工件按约束条件在所有机器上从开始到结束的加工路径；E 为 m 条子边构成的弧集，子弧（虚线）表示同一机器上加工各操作的连接。

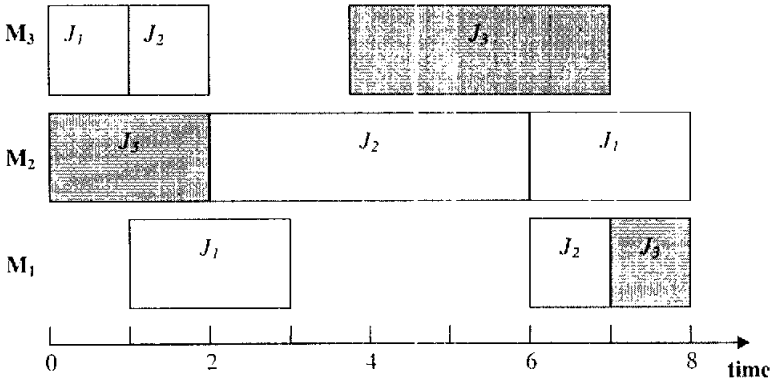

图 2-1　3 工件 3 机器, 3×3 JSSP 甘特图

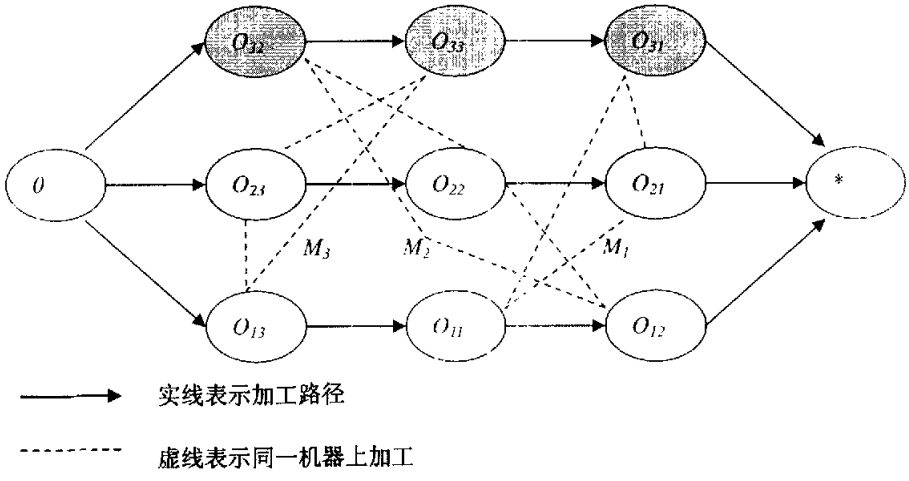

图 2-2　3 工件 3 机器,3×3 JSSP 析取图

若以最大完成时间为指标，则对 JSP 的求解就归结为到各弧（即机器）上作为优先决策的各操作的一组顺序（即走向），当同一机器上有多个操作出现冲突时，上述顺序用于决定各操作的先后，最终得到各操作间没有冲突的一个有向非循环图，其关键路径长度即为最大完成时间。

2.3.2 优化目标

生产调度系统里的指标费用、质量、时间、数量都可以作为目标。选择什么样的指标作为优化目标取决于实际需要，因为生产是基于时间，其它相关目标，费用等目标都可在时间目标上衍生。给出工件完成时间，C_j 表示工件 j 在系统中存在的时间（即工件在最后一台机床上完成加工后的时间）

最大完成时间 Makespan (C_{max})。定义最大完成时间为 $C_{\max} = \max\{C_j\}$，$j = 1,2,..n$，等于最后一个工件完成加工后离开系统的时间，极小化最大完成时间通常暗示机器的最大利用率。

最大滞后时间 Maximum lateness (L_{max})。定义最大滞后时间为 $L_{\max} = \max\{L_j\}$，$j = 1,2,...,n$。它测量最大完成期（最迟完成时间）。

2.4 JSSP 问题优化的数学模型

JSSP 调度问题是许多实际生产调度问题的简化模型。JSSP 研究 n 个工件在 m 台机器上的加工。已知每个工件在各个机器上的加工次序和每个工件的各个工序的加工时间。要求确定与工艺约束条件相容的各机器上所有工件的加工开始时间或完成时间或加工次序，使加工性能指标达到最优。各个工件和机器应满足以下约束[5],[14]：

(1) 在整个加工过程中，每个工件只能被所有的机器都加工并且只加工一次；

(2) 各个工件必须按工艺路线以指定的次序在机器上加工；

(3) 加工过程不能间断；

(4) 每一时刻每一台机器只能加工一个工件。

JSSP 问题的求解就是要找到一个合理的安排使每个工件都能在满足工艺约束的条件下在各台机器上加工，使得总的加工时间最短。对于工件 i，c_{ik} 为第 i 工件在机器 k 上的完成时间，p_{ik} 为第 i 工件在机器 k 上的加工时间。L 是一个足够大的正数。

一般用 $n/m/A/B$ 表示标准调度问题，其中 n 表示工件数，m 表示机器数，A 表示流经机器的类型（Job Shop 用 G 表示，Flow Shop 用 F 表示，置换 Flow Shop 用 P 表示），B 表示性能指标（如 C_{max}，L_{max} 等）。因此一个 $n/m/G/C_{max}$ 的 JSSP 问题的数学模型可描述如下：

$$\min \max_{1\le k\le m}\left\{\max_{1\le i\le n}\{c_{ik}\}\right\} \tag{2-1}$$

$$\text{s.t.}\quad c_{ik}-p_{ik}+L(1-x_{ihk})\ge c_{ih},\quad i,j=1,2,\cdots,n,\quad k=1,2,\cdots,m \tag{2-2}$$

$$c_{jk}-c_{ik}+L(1-y_{ijk})\ge p_{jk},\quad i,j=1,2,\cdots,n,\quad k=1,2,\cdots m \tag{2-3}$$

$$c_{ik}\ge 0,\quad i,j=1,2,\cdots,n,\quad k=1,2,\cdots m \tag{2-4}$$

$$x_{ihk}=0\ or\ 1,\quad i,j=1,2,\cdots,n,\quad k=1,2,\cdots m \tag{2-5}$$

$$y_{ijk}=0\ or\ 1,\quad i,j=1,2,\cdots,n,\quad k=1,2,\cdots m \tag{2-6}$$

式(2-1)表示目标函数，式(2-2)表示各工件的各操作的先后加工顺序，式(2-3)表示加工各个工件的各机器的先后顺序。x_{ihk}，y_{ijk}分别为指示系数和指示变量，其意义为

$$x_{ihk}=\begin{cases}1,\text{若机器}h\text{先于机器}k\text{加工工件}i\\0,\text{非上述情况}\end{cases} \tag{2-7}$$

$$y_{ij}=\begin{cases}1,\text{若工件}i\text{先于工件}j\text{在机器}k\text{上加工}\\0,\text{非上述情况}\end{cases} \tag{2-8}$$

2.5 遗传算法求解 JSSP 问题的主要操作

下面将介绍用演化算法求解 JSSP 问题的主要操作，分别为编码设计，交叉操作设计、变异操作设计[5], [44]。

2.5.1 编码设计

编码问题时设计演化算法的首要和关键问题。演化算法的编码技术必须考虑“染色体”的合法性、可行性、有效性以及对问题解空间表征的完全性，求解 JSSP 同样如此。下面，我们介绍几种适用于 Job Shop 调度问题的演化算法编码方式。

基于操作的编码（Operation-based Representation）方式将每个染色体用$n\times m$个代表操作的基因组成，是所有操作的一个排列，其中各工件号均出现m次。

解码过程是：先将染色体转化为一个有序的操作表，然后基于该表和工艺约束对各操作以最早允许加工时间逐一进行加工，从而产生调度方案。

基于工件的编码（Job-based Representation）方式将每个染色体用n个代表工件的基因组成，是所有工件的一个排列。解码过程是：先加工第 1 号工件的所有操作，然后依次以最早允许加工时间加工后面各工件的所有操作。

基于先后表的编码（Preference List-based Representation）方式将每个染色体用分别对应于m台不同机器的m个子串构成，各子串是一个长度为n的符号串，用于表示一种优先表，各符号表示相应机器上的加工操作。解码过程通过对染色体的仿真得到，即分析机器上当前等待队列的状态并判断是否用先后表来确定调度，也就是说，最先出现在先后表的操作将被选中。

基于优先规则的编码（Priority Rule-based Representation）方式将每个染色体用一个长度为$n \times m$的优先分配规则序列构成，每个基因即为一种优先调度规则。算法的优化结果是一个满意的规则序列，然后依次用这些优先规则产生或修改调度方案。常用的优先调度规则有：SPT（Shortest Processing Time），即选择加工时间最短的操作。LPT（Longest Processing Time），即选择加工时间最长的操作。MWR（Most Work Remaining），即选择总剩余加工时间最长的操作。LWR（Least Work Remaining），即选择总剩余加工时间最短的操作。MOR（Most Operations Remaining），即选择总剩余操作数最多的操作。LOR（Least Operations Remaining），即选择总剩余操作数最少的操作。EDD（Earliest Due Date），即选择交货期最早的工件。FCFS（First Come First Served），即选择同台机器上工件队列种最先的操作。RANDOM（Random），即随机选择。

基于析取图的编码（Disjunctive Graph-based Representation）方式将染色体用一个长度$n \times m$的 0-1 字符串来表示，该染色体（由各弧的操作顺序组成）作为优先决策，以决定同台机器上发生操作冲突时各操作的顺序。它也可以认为是一种基于工件对关系的编码关系。

基于完成时间的编码（Completing Time-based Representation）方式利用各操作完成时间的有序表来表示染色体。

基于机器的编码（Machine-based Representation）方式将每个染色体用所用机器的排列，并以此通过移动瓶颈方法来构造调度。

随机键编码（Random Key Representation）将调度解码成随机数，这些值用作排列键来解码。具体而言，每个基因包括两部分，即$\{1,2,\cdots,m\}$集合中的一个

整数以及(0,1)中的随机分数。任意随机键的整数部分解释为工作的机器分配，对分数部分的排列则提供每一机器上工件的顺序。

编码方式决定了 JSSP 的问题表达及遗传操作方式。为保证编码策略不遗漏问题的全局最优解，使优化操作容易处理状态的可行性和合法性。

2.5.2 交叉操作设计

交叉操作的目的是利用父代个体组合出后代新个体。鉴于 JSSP 的特殊性，交叉操作需结合所采用的编码技术设计。在以下针对置换编码的交叉操作举例中分别用 P_1 和 P_2 代表父个体，C_1 和 C_2 为他们产生的后代。

部分映射交叉（Partially Mapping Crossover，PMX）。首先随机选取两个交叉点，交换父代个体交叉点之间的片段，对于交叉点外的基因，若它不与换过来的片段冲突则保留，若冲突则通过部分映射来确定直到没有冲突的基因，从而获得后代个体。例如两个父代个体为 P_1 = [2 6 4 7 3 5 8 9 1]，P_2=[4 5 2 1 8 7 6 9 3]。交叉位置为 3，7，则片段 7 3 5 8 和 1 8 7 6 将交换，得 C_1=[* * * 1 8 7 6 * *]，C_2 = [* * * 7 3 5 8 * *]。*表示基因待定。从各自得父体中填入无冲突得城市得 C_1 = [2 3 4 1 8 7 6 9 5]，C_2=[4 1 2 7 3 5 8 9 6]。

次序交叉（Order Crossover，OX）。首先随机确定两个交叉位置，并交换交叉点之间的片段，并从第 2 交叉位置起在原先父代个体中删除将从另一父代个体交换过来的基因，然后从第 2 交叉位置后开始填入剩余基因。例如，两个父代个体及交叉点同上，则 C_1 = [4 3 5 1 8 7 6 9 2]，C_2=[2 1 6 7 3 5 8 9 4]。

线性次序交叉（Linear Order Crossover，LOX）。首先随机确定两个交叉位置，并交换交叉点之间的片段，并在原先父代个体中删除将从另一父代个体交换过来的基因，然后从第 1 个基因位置起依次在两交叉位置外填入剩余基因。例如，两个父代个体及交叉点同上，则 C_1 = [2 4 3 1 8 7 6 5 9]，C_2=[4 2 1 7 3 5 8 6 9]。

基于位置的交叉（Position-based Crossover，PX）。PX 随机选取一些位置，然后交换被选中位置上的基因，并在原先父代个体中删除从另一父代个体交换过来的基因，接着从第 1 个基因位置起依次在未选中位置填入剩余基因。例如，若父代个体同前，假设随机选取的位置点为 2，3，6，8，则后代为 C_1 =[6 5 2 4 3 7 8 9 1]，C_2 =[2 6 4 1 8 5 7 9 3]。

循环交叉（Cycle Crossover，CX）。CX 将另一个父代个体作为参照，以对当前父代个体中的位置进行重组。先与另一父代个体实现一个循环链，并将对应

位置的基因填入相应的位置，循环组成后再将另一个父代个体各位置的基因填入相同的位置。例如，父代个体同上，得到$C_1=[2\,5\,4\,1\,8\,7\,6\,9\,3]$类似地可得到$C_2=[4\,6\,2\,7\,3\,5\,8\,9\,1]$。

2.5.3 变异操作设计

变异操作的目的是通过随机改变染色体的某些基因来引入新个体，增加种群多样性，并在一定程度上进行局部搜索。相对于交叉操作的设计，变异操作的设计更为简单和灵活。常用的针对置换编码的变异操作包括：

互换（Swap）：即随机交换若干不同位置上的不同基因。例如，父代个体为$P=[1\,2\,3\,4\,5]$将位置2，5的基因进行交换，得$C=[1\,5\,3\,4\,2]$。本文算法采用了互换变异操作。

逆序（Inverse）：即将某两个随机位置间的基因串逆序。例如，父代个体为$P=[1\,2\,3\,4\,5]$将基因位置2，4之间的片段逆序排列，得$C=[1\,4\,3\,2\,5]$。

插入（Insert）：即将某一位置上的基因（或某一段位置上的基因串）插入到另一位置之前或之后。例如，父代个体为$P=[1\,2\,3\,4\,5]$将位置5的基因插入到位置2后，得$P=[1\,2\,5\,3\,4]$。

2.6 本章小结

本章介绍了标准JSSP问题。分别阐述了JSSP的定义、分类、甘特图及析取图表示方法及数学模型。GA是求解JSSP的一种有效算法，其对于JSSP问题的操作是设计演化算法的一个主要部分。介绍了编码设计、交叉操作设计、变异操作设计方法。

第三章 禁忌演化算法

3.1 禁忌搜索算法

禁忌搜索是对人类思维过程本身的一种模拟，它通过对一些局部最优解的禁忌（也可以说是记忆）达到接纳一部分较差解，从而跳出局部搜索的目的[38]。

禁忌搜索算法是由 Glover 在 1986 年首先提出，1997 年 Glover 和 Laguna 详细地描述了算法的基本概念[5]。TS 明确使用历史记录，它不仅考虑如何跳出局部极小值陷阱，而且考虑搜索策略的实现。TS 算法使用改进算子为局部搜索的基本因素，使用短期记忆来跳出局部极小值陷阱和避免循环搜索。它使用一个禁忌表，这个表记录最近访问的解的轨迹并且禁止搜索向它们移动，以次来实现短期记忆。它规定在候选解的邻域内不容许包含属于禁忌表中的解，不包含属于禁忌表中的解的集合称为容许集[35],[36]。

为了找到“全局最优解”，就不应该执着于某一个特定的区域。局部搜索的缺点就是太贪婪地对某一个局部区域以及其邻域搜索，导致一叶障目，不见泰山。禁忌搜索就是对于找到的一部分局部最优解，有意识地避开它（但不是完全隔绝），从而获得更多的搜索区间。兔子们找到了泰山，它们之中的一只就会留守在这里，其他的再去别的地方寻找。就这样，一大圈后，把找到的几个山峰一比较，珠穆朗玛峰脱颖而出。

当兔子们再寻找的时候，一般地会有意识地避开泰山，因为他们知道，这里已经找过，并且有一只兔子在那里看着了。这就是禁忌搜索中“禁忌表(tabu list)”的含义。那只留在泰山的兔子一般不会就安家在那里了，它会在一定时间后重新回到找最高峰的大军，因为这个时候已经有了许多新的消息，泰山毕竟也有一个不错的高度，需要重新考虑，这个归队时间，在禁忌搜索里面叫做“禁忌长度（tabu length)”；如果在搜索的过程中，留守泰山的兔子还没有归队，但是找到的地方全是华北平原等比较低的地方，兔子们就不得不再次考虑选中泰山，也就是说，当一个有兔子留守的地方优越性太突出，超过了“best so far”的状态，就可以不顾及有没有兔子留守，都把这个地方考虑进来，这就叫“特赦准则(aspiration criterion)”。这三个概念是禁忌搜索和一般搜索准则最不同的地方，

算法的优化也关键在这里。

禁忌搜索法中重要的构造之一就是禁忌结构的确定。禁忌结构包含两种操作：

1) 通过禁忌表防止循环现象：将L个访问过的解的移动或移动标志存在禁忌表中，并随着表的溢出，不断更新表中的内容，禁忌表的主要作用就是考察当前解的邻域中的解是否曾经被访问过；

2) 通过期望函数及其标准的规定来解除禁忌表中的某些移动。具体地讲，当出现某一个移动被禁忌时，若它能通过期望函数的标准,则该移动的禁忌状态被解除，并执行该移动产生一新解，作为下一个初始解。这里移动是指一个函数，能将一个解转换成另外一个解,对任意一个解，需定义一个可适用它的移动子集，将这个子集中的移动作用于解，便产生解的邻域。

基本禁忌搜索算法[5]：

步骤 1：给定算法参数，随机产生初始解 x，置禁忌表为空；

步骤 2：判断算法终止条件是否满足？若是，则结束算法并输出优化结果；否则，继续以下步骤；

步骤 3：利用当前解 x 的邻域函数产生其所有（或若干）领域解，并从中确定若干侯选解；

步骤 4：对侯选解判断藐视准则是否满足？若成立，则用满足藐视准则的最佳状态 y 替代 x 成为新的当前解，即 x=y，并用与 y 对应的禁忌对象替换最早进入禁忌表的禁忌对象，同时用 y 替换“best so far”状态，然后转步骤 6；否则，继续以下步骤；

步骤 5：判断侯选解对应的各对象的禁忌属性，选择侯选解集中非禁忌对象对应的最佳状态为新的当前解，同时用与之对应的禁忌对象替换最早进入禁忌表的禁忌对象元素；

步骤 6：转步骤 2。

以上程序中有关键的几点：

(1) 禁忌对象：可以选取当前的值作为禁忌对象放进 tabu list，也可以把和当前值在同一“等高线”上的都放进 tabu list。

(2) 为了降低计算量，禁忌长度和禁忌表的集合不宜太大，但是禁忌长度太小容易循环搜索，禁忌表太小容易陷入“局部极优解”。

(3) 上述程序段中对 best so far 的操作是直接赋值为最优的“解禁候选解”，但是有时候会出现没有大于 best so far 的，候选解也全部被禁的“死锁”状态，这个时候，就应该对候选解中最佳的进行解禁，以能够继续下去。

(4) 终止准则：给定一个迭代步数；设定与估计的最优解的距离小于某个范围时，就终止搜索；当与最优解的距离连续若干步保持不变时，终止搜索；

3.2 演化算法

3.2.1 简单的演化算法

经典演化算法首先对参数进行编码，生成一定数量的个体，形成初始种群。其中每个解可以是一维或多维矢量，以二进制数串表示，称为染色体。染色体的每一位进制数称为基因。自然界生物被客观环境选择，优胜劣汰，算法中则需要设计适应度函数作为评判每个个体性能优劣的标准，性能好的个体以一定概率被选择出来作为父代参加以后的遗传操作以生成新一代种群。算法中基本的遗传算子为染色体选择，染色体上基因杂交和基因变异。交叉的结果使父代的特性更集中，变异则有益于产生新个体，利于进化。生成新一代种群后，算法循环进行适应度评价、遗传操作等步骤，逐代优化，直至满足结束条件。

简单演化算法的流程如下：

步骤 1：初始化群体；

步骤 2：计算群体上每个个体的适应值；

步骤 3：按由个体适应值所决定的某个规则选择将进入下一代的个体；

步骤 4：按概率 P_c 进行杂交操作；

步骤 5：按概率 P_m 进行变异操作；

步骤 6：若满足某种停止条件，则执行步骤 7，否则执行步骤 2；

步骤 7：输出种群中适应值最优的染色体作为问题的满意解或最优解。

一般情况下，算法的终止条件包括：1、完成了预先给定的进化代数；2、种群中的最优个体在连续若干代没有改进或平均适应度在连续若干代基本没有改进；3、所求问题最优解的达到了某一精度。

3.2.2 简单演化算法的分析

在演化算法中，变异算子经群体中带来新的遗传基因以恢复出于选择算子的作用而失去的个体多样性；杂交算子对群体内现有的信息进行重组以发现与环境更为适应的个体。而选择则起着向导的作用以使搜索朝着搜索空间中的可能的最优区域进行。杂交与变异的作用是“勘探”搜索空间以寻找那些可能的最优区域，以保持群体内的多样性为其主要目的;而选择的作用则是“开采”搜索空间以充分利用群体内当前所具有的有效信息，它使算法将搜索的侧重点放在那些具有较高适应值的个体上。简单演化算法中，“勘探”与“开采”始终是影响演化算法收敛速度与全局寻优的一对矛盾。演化初期选择压力较大，适应度高的个体纷纷被选中，这些个体很快控制了演化过程。造成群体的过早收敛，产生局部最优;演化后期群体适应度差异较小，选择压力自然减小，使收敛速度大大降低。演化算法可以用很快的速度达到次优解，但继续操作要达到真正的最优解，则要花费很长的时间。

综上分析，对于简单的演化算法，搜索前期容易出现“超级个体”，而搜索后期又容易出现停滞不前的局面：杂交和变异操作往往会产生良种丢失，从而导致种群退化；有时无法从局部最优解中脱出。为此，很多专家为改进演化算法的性能进行了卓有成效的研究，提出了各种改进的演化算法。下面介绍几种改进的演化算法[42],[53]。

3.2.3 演化算法的改进

(1) 自适应演化算法

自适应演化算法根据种群的进化情况来自动调整杂交概率 P_c 和变异概率 P_m，以达到克服过早收敛及加快搜索速度的目的。设 $f_{\max}$ 为某一代群体中最佳个体的适应度值，f_{avg} 为此代群体的平均适应度值，又设 $\Delta = f_{\max} - f_{avg}$ Δ 则可认为 Δ 越小，此代种群中个体之间的差异越小，群体趋向于收敛。Δ 越大，此代种群中个体之间差异越大，越分散，收敛速度慢，因此为防止群体过早收敛，当 Δ 较小时，应增大 P_c 和 P_m；当 Δ 较大时，应减小 P_c 和 P_m，$P_c = K_1/(f_{\max} - f_{avg})$，$P_m = K_2/(f_{\max} - f_{avg})$。上式表明在进化过程中，根据实际情况随时调整 P_c 和 P_m，使演化算法具有更强的搜索能力。但上式也同样表明，当进化已收敛到全局最优时，最优个体遭破坏的概率也很大，为了克服这一缺点，自适应演化算法通常写成：

$$\begin{cases} P_c = K_1/(f_{\max} - F)(f_{\max} - f_{avg}) & F \geq f_{avg} \\ P_c = K_3 & F < f_{avg} \end{cases}$$

$$\begin{cases} P_m = K_2/(f_{\max} - f)(f_{\max} - f_{avg}) & f \geq f_{avg} \\ P_m = K_4 & f < f_{avg} \end{cases}$$

式中，F 为两个杂交个体中适应度最大的一个，f 为要变异个体的适应度值，只要设定 K_1，K_2，K_3，K_4 取 (0,1) 区间的值，就可以自适应调整了。自适应演化算法在保持群体多样性的同时，保证演化算法的收敛性。

(2) 精英保留策略

这种方法是对于每代中一定数量的最优个体，使之直接进入下一代。这样可以防止优秀个体由于复制、杂交或变异中的偶然因素而被破坏掉。这是增强算法稳定性和收敛性的有效方法。但同时也可能使演化算法陷入局部的极值范围。

(3) 移民法

移民算法是为了加速淘汰差的个体以及引入个体多样性的目的而提出的。所需的其它步骤是用杂交产生出的个体替换上一代中适应值低的个体，继而按移民的比例，引入新的外来个体来替换新一代中适应值低的个体。这种方法的主要特点是不断地促进每一代的平均适应值的提高。但由于低适应值的个体很难被保存至下一代，而这些低适应值的个体中也可能包含着一些重要的基因模式块，所以这种方法在引入移民增加个体多样性的同时，由于抛弃低适应值的个体又减少了个体的多样性。

(4) 部分替换法

设 P_G 为上一代进化到下一代时被替换的个体的比例，则按此比例，部分个体被新的个体所取代，而其余部分的个体则直接进入下一代。P_G 越大，进化得越快，但算法的稳定性和收敛性将受到影响;而 P_G 越小，算法的稳定性较好，但进化速度将变慢。可见，应该寻求运行速度与稳定性、收敛性之间的协调平衡。

(5) 分布式演化算法

对于一个问题，首先随机地生成个 $N \times n$ 样本，然后分成 N 个子种群，各子种群将具有略微不同的基因模式，每个子种群独立地运行各自的演化算法，因而进化的方向也略有差异，从而保证了搜索的充分性及收敛结果的全局最优性。另一方面，在各子种群之间又以一定的比率定期地进行优良个体的迁移，即每个子种群将其中最优的几个个体轮流送到其它子种群中，这样做的目的是期望

使各子种群能共享优良的基因模式以防止某些子种群向局部最优方向收敛。

分布式演化算法模拟了生物进化过程中的基因隔离和基因迁移，即各子种群之间既有相关的封闭性，又有必要的交流和沟通。研究表明，在总的种群个数相同的情况下，分布式演化算法可以得到比单一种群演化算法更好的效果。

3.2.3 混合演化算法

梯度法、爬山法、模拟退火法等一些优化算法具有很强的局部搜索能力，而另一些含有问题相关的启发知识的启发式算法的运行效率也比较高。如果融合这些优化方法的思想，构成一种新的混合演化算法，是提高演化算法运行效率和求解质量的一个有效手段。目前，混合演化算法实现方法体现在两个方面，一是引入局部搜索过程，二是增加编码变换操作过程。在构成混合演化算法时，De Jong[53]提出下面三个基本原则：

(1) 尽量采用原有算法的编码；

(2) 利用原有算法全局搜索的优点；

(3) 改进遗传算子。

本文设计算法的主要操作之一就是引入局部搜索过程。

3.3 一种基于相似性的演化算法

演化算法理论解释了为什么对一个给定问题表达，能收敛到欲求的最优点而实际的应用并不总是遵循这一理论，主要的原因是：

- 对问题的编码经常使演化算法在不同于问题本身的空间运行。
- 理论假定迭代次数是无限的而实际是有限的。
- 理论假定群体规模是无限的，而实际是有限的。

这说明演化算法在某种条件下不能找到最优解；这种失败是由于过早收敛到局部最优造成的。过早收敛是演化算法和其它优化算法共同的问题。如果收敛发生得太快，包含在群体中的有价值的信息常常会失去。而演化算法的执行趋向于在找到最优解前过早收敛。

演化算法有较强的全局搜索能力，而传统算法有较强的局部搜索能力，如果能将两者的优点结合在一起，则能提高其搜索效率和效果。一个算法的目标是如何达到效果和效率的平衡，效果是要使算法得到的结果更接近最优解，效

率是如何使算法以更快的速度收敛到最优解，这两个问题是矛盾的统一体。而在EC中这个矛盾就是如何在“探索”和“利用”之间建立一个平衡点也就是说既要使算法的搜索空间尽可能的大，以寻找那些可能存在最优解的解区间；同时充分利用群体内当前具有的有效信息，使得算法搜索的侧重点放在那些可能具有较高适应值的个体所在的区间内，从而以大概率收缩到全局最优解。而要使EC达到“探索”和“利用”的平衡就要解决如何较快地找到最优解并防止“早熟”收敛问题。根据以上的思路，黄樟灿博士[42]设计出一种改进的演化算法。其思想已用于对函数优化[47]，TSP 问题[48]，多目标优化[49]的研究当中。算法的基本思想是用相似性对搜索空间进行分割。首先，根据适应度的相似性对父代群体进行分割，让优秀的个体带领它的邻居进行搜索空间的开采，其它的个体带领它的邻居进行搜索空间的勘探，最后在分割的区域利用子代与父代信息用传统的方法实现加速。其进化过程见图3-1。下面对算法实现过程中的相似性、群体分级、加速操作。

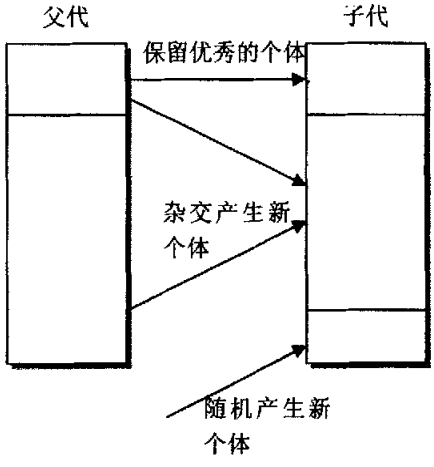

图3-1 进化过程

3.3.1 相似性的度量

相似性是衡量对象之间相似程度的指标，它可以对特征空间进行分割，它一般通过计算对象在特征空间中的距离获得。

定义1 设对象(实例) $x=(x_1,x_2,\ldots,x_n)$， $x_i(1\le i\le n)$是它的特征值。x是n维特征空间$D=(D_1\times D_2\times\cdots\times D_n)$上的一点。设$x,y\in D$，则$x,y$的关于特征$F$的

距离(或者说x,y在特征空间D上的距离)为

$$d_F(x,y)=[\sum_{i\in F}\omega_i\sigma(x,y)^r]^{\frac{1}{r}}$$

其中

$$\sigma(x_i,y_i)=\begin{cases}|x_i-y_i|,\text{如果}D_i\text{是连续}\\0\quad,\text{如果}x_i=y_i\\1\quad,\text{其它}\end{cases}$$

r的取值由用户定义,例如，$r=2$时，$d_F(x,y)$为欧拉距离。除了常用于处理连续属性和多值属性的欧拉距离、曼哈顿距离和无限模距离外，其它一些距离函数如Mahalanobis，Minkowsky，Hausdorff等，也在各种属性和应用中采用。F是特征空间D的子空间，$F\subseteq D,x_i,y_i\in F$。特别地，当$F=D$时，$d_F(x,y)$记为$d_D(x,y)$。$\omega_i$是特征$x_i$的权重，$\omega_i\geq 0$，$\omega_i$是为了增加不同属性之间距离的可比性，

定义2　特征空间D的直径为：

$$did_D=\max\{d_D(x,y)|x,y\in D\}。$$

定义3　对象x,y关于F的相似性为

$$s_F(x,y)=\frac{1}{d_F(x,y)}。$$

表达式说明相似性与对象之间的距离成反比，事实上任何单调下降的函数都可以用来刻画对象之间的相似程度。

定义4 给定常数d_0，对象$x,y\in D$，若$d_D(x,y)<d_0$，则称x,y是在D中的邻居。

3.3.2 群体的分级

对于复杂的优化问题，求解过程一般有两个特点，

(1) 采用局部搜索算法效率好效果差。

(2) 采用全局搜索算法效果好效率差。

如果设计一种算法能融合局部搜索和全局搜索两方面的本质属性，就有可能实现算法效率与效果的平衡。本文采取了群体分级的策略，根据适应值的相似性对个体进行分级。分级的原则是把适应值相似的个体分配在同一级，它们的

相似性是根据适应值之间的距离来度量。分级的目的是让优秀的个体进行局部极小值的搜索，让较差的个体进行搜索空间的探索，以发现新的局部极小值。产生新个体的规则是由它在搜索解空间的任务所确定，也就是说它与个体所在的级别相关联，即个体在父代中级别次序确定产生新个体的搜索范围和它们所生成的个体数目。具体有：

(1) 在父代中级别较高的个体的搜索范围较小，生成的个体数目较少；

(2) 父代中级别较低的个体的搜索范围较大，它生成的个体数目较多。

3.3.3 算法框架

Jim Gray [45]在文章 A dozen information-technology research goals 中指出信息技术的长远目标应考虑以下五个关键因素：Understandable, Challenging, Useful, Testable, Incremental。据此构造的一种混合优化算法框架的结构如图 3-2 所示，使具有以下几个特性：

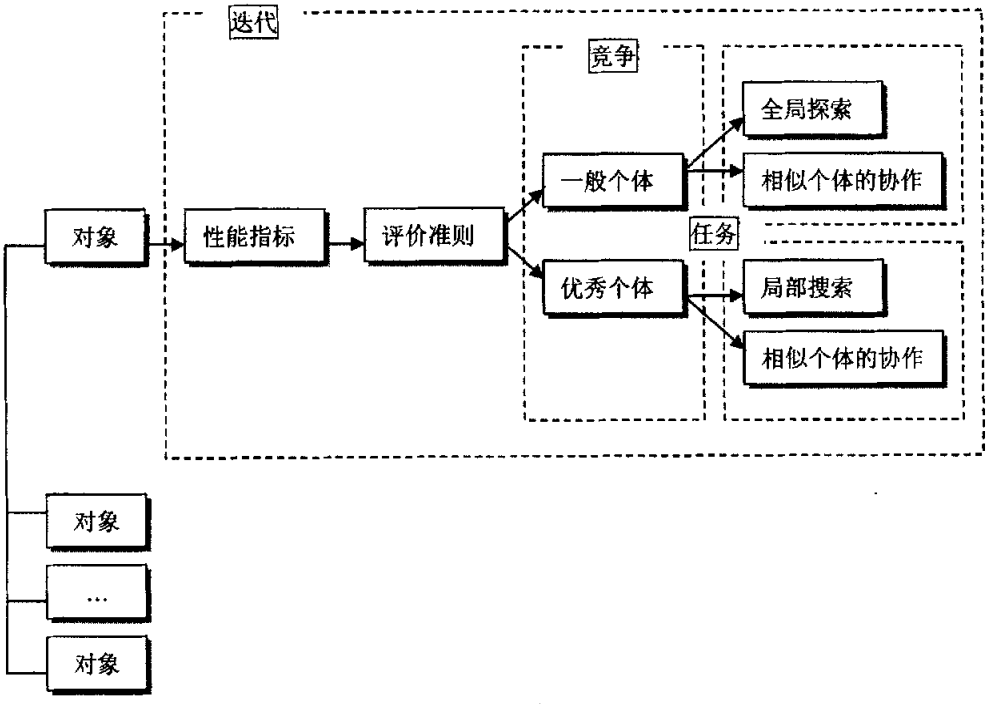

图 3-2 优化框架

1. 通用性： 可以求解不同的优化问题并可将求解具体问题所需要的优化方法纳入该框架而不需要重新设计；

2. 智能性： 框架能够指导搜索策略的设计，自适应调节各种参数；
3. 平衡性： 搜索效果和效率的平衡，反映到算法设计中效果往往是指解的精度，效率是指算法的复杂度。
4. 并行性：适合大规模并行

下面描述其演化算法步骤：

步骤 1：随机产生一组初始个体构成初始种群。

步骤 2：评价每个个体的适应值，并按适应值相似程度将个体分为 k 级。

步骤 3：利用个体相似性按相似性的程度和级别在搜索空间内分割相应的子空间(邻居)。

步骤 4：对每一级里的各个个体，分别在它们的邻居中产生新个体 S'，它们一起构成了子种群。对子种群进行加速找到较好的解集 S''，并将 S'' 加入到子种群中。对子种群进行选择运算。

步骤 5：将各子种群合并成新种群，对种群进行选择运算。

步骤 6：判断是否满足终止条件。若满足，则输出结果；否则转到步骤 2。

3.4 求解作业车间调度的禁忌演化算法

将禁忌搜索和演化计算相结合，设计求解作业车间调度的禁忌演化算法（Evolutionary-Tabu Algorithm，ETA)。下面详细介绍 ETA 实现的过程。

3.4.1 ETA 个体编码方式

算法采取 2.5.1 节介绍的基于操作的编码方式。如图 3-3 所示。加工工件，加工时间，操作机器是每个体所代表的三个维度，分别以 J，T，O 表示，在图中表示为 A1=(J1,T1,O1)。这种编码方式，从左至右扫描操作顺序，使得编码的表达已是一种可行解。以 4 工件 4 机器为例，其中个体有多种表示，其中两个可能的个体为

X1=(A1,A2,A3,A4,B1,B2,B3,B4,C1,C2,C3,C4,D1,D2,D3,D4)

也可表示为

X2=(A1,B1,B2,A2,C1,D1,B3,D2,C2,B4,C3,A3,C4,A4,D3,D4)

如果对于一个实际问题，需要考虑更多的维度，如操作者，以 W 表示。则 A1

可表示为 A1 = (J1,T1,O1,W1)。

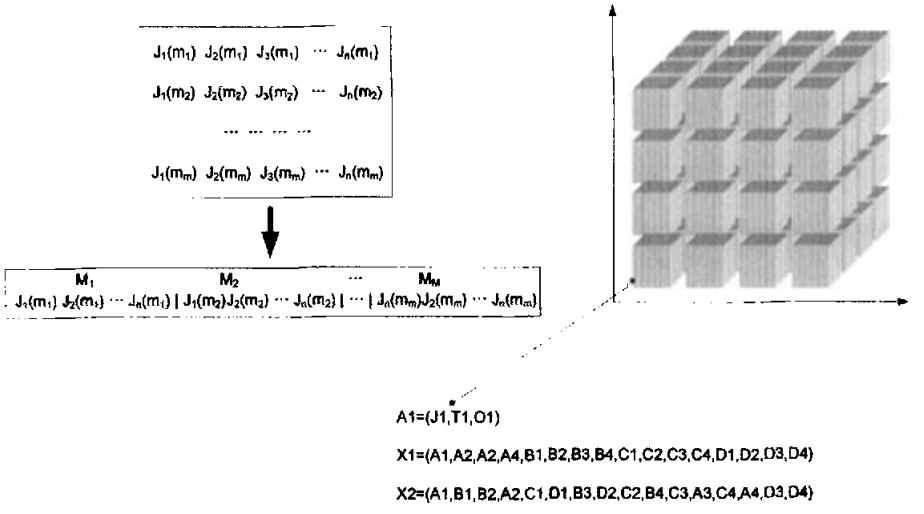

图 3-3 编码方式示例

3.4.2 ETA 群体分级方式

按 3.3.2 节介绍的分级方法，JSSP 问题的分级就是按照适应值大小进行，将群体按适应值大小降序排列，然后依据最好个体的适应值和最差个体的适应值之差动态分级。假设要将群体分成k级，群体为$X=(X_1,X_2,\cdots,X_N)$，其中$X_i(1\le i\le N)$为 JSSP 问题的一个个体。则具体操作步骤如下：

（1）将群体中个体的最小适应值和最大适应值之差分别赋值给a。即$a=\max f(X_i)-\min f(X_i)$。

（2）设$b=\frac{a}{k}$。

（3）如果$\min f(X_i)+(j-1)*b\le f(X_i)\le \min f(X_i)+j*b$，其中$1\le j\le k$。

则将个体X_i归到第j级。按照这个规则将群体N分为k级。

这里k可以是常数或者是动态变化，变化的趋势应该是越来越小。一旦对群体进行分级完后，就要产生新个体。

3.4.3 ETA 新解产生的方式

分级的原则是将适应值相似的个体分配在同一级。分级的目的是让优秀的个体进行局部极小值的开采，并且给它们较少的数量；让较差的个体进行搜索空间的探索，以发现新的局部极小值。

产生新个体的规则与个体所在的级别相关联，即产生新个体的操作和新个体数目由父个体所在的级别确定。规则就是父个体在父代中的级别次序确定了产生新个体的操作和它们所生成的子个体的数目，在父代中级别较好的个体，所采用的操作应能利用父个体的优良性能，它的搜索范围应该较小，产生的个体数目较小，反之，父代中级别较低的个体应扩大搜索范围，产生较多的个体。

对于 JSSP 问题，级别较高的个体采用禁忌搜索方式，级别较差的个体采用基于工件的次序交叉操作和互换操作。基于工件的次序交叉操作（Job-based Order Crossover, JOX）来自经典的次序交叉操作（OX）。本文在算法中采用了 JOX 交叉方式。首先它从基因池中随机选择工件数量和工件序号。选择工件序号，从 P_1 复制到 C_1 中，并保持他们的绝对位置不变。C_1 中空余的位置从 P_2 中复制，根据他们的在 P_2 中顺序。如图 3-4 所示。算法流程见图 3-5，图 3-6。

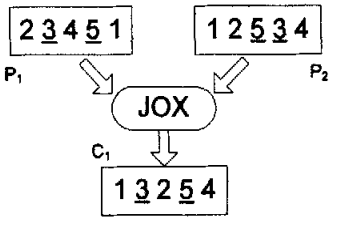

图 3-4 JOX 交叉操作

3.4.4 ETA 框架

适应值函数为最大完成时间，该算法的参数有：群体规模 n，分成 k 级，t 为代数，群体为 $p(t)$。

```
procedure ETA
begin
    t:=0
    GenerationInitialPopulation();    //  随机初始化群体
    EvalutePopulation();              //  计算 p(0)中个体的适应值
    Classification();                 //  将 p(0)进行分成 k 级
    repeat
        t = t +1;
        TSNewPopulation();            //  高级别个体采用禁忌操作
        ECNewPopulation();            //  低级别个体采用遗传操作
        EvaluatePopulation();         //  计算 p(t)中个体的适应值
        Classification();             //  分级
    until (stop_criteria) or t=MAX
end
```

3.5 本章小结

生产调度问题的求解方法很多，分精确算法和近似算法。包括分支定界，优先分配规则，移动瓶颈过程，模拟退火，禁忌搜索，进化计算等。每种方法都各有特点。本章分别介绍了禁忌搜索算法，演化算法及一种基于相似性的演化算法。本文主要针对作业车间调度问题采用禁忌搜索与演化算法相结合的混合算法进行研究。结合禁忌搜索算法局部寻优的能力，利用演化算法全局寻优的能力。两者特点的结合，以达到求解过程效率与效果平衡。针对演化算法在求解 JSSP 问题时，存在着收敛速度慢或易陷入局部最优解的缺点，本文提出了一种将禁忌搜索与演化算法相结合的禁忌演化算法，并描述了算法求解 JSSP 问题的流程。

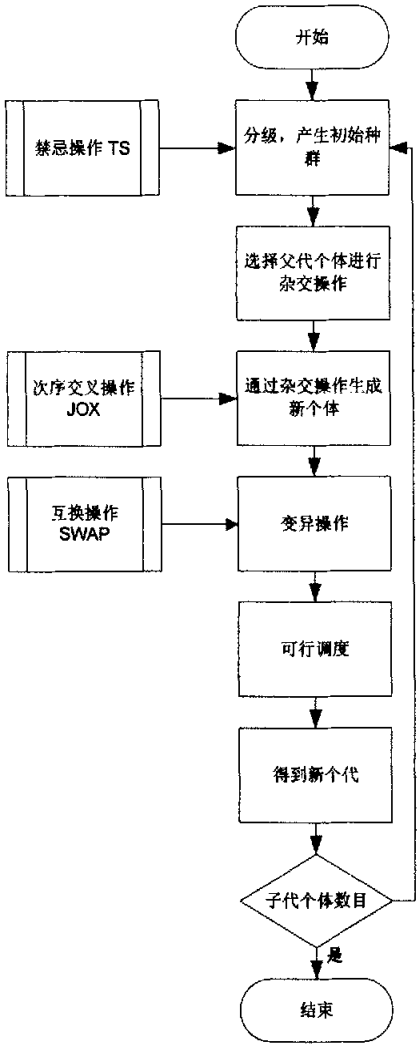

图 3-5 禁忌演化算法流程图

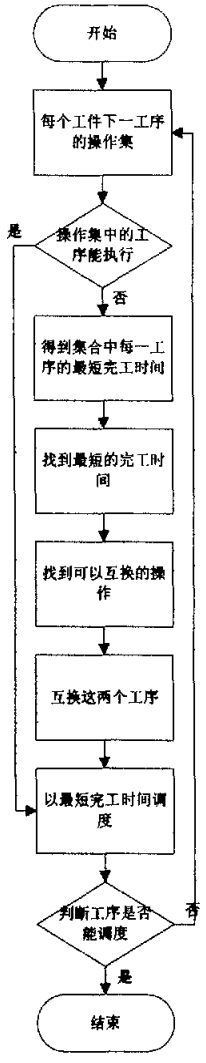

图 3-6 SWAP 操作流程图

第四章 用禁忌演化算法求解 JSSP 问题

4.1 算例

4.1.1 Benchmark 测试集

为了比较不同算法的性能，许多学者提供了公开测试的 Benchmark 算例[5],[7],[51]。下面分别介绍。

- FT 类

由 Fish 和 Thompson 设计（1963）。包括三个问题 FT06，FT10，FT20。其中 FT10，10×10 JSSP 是最常用的一个算例，从 1963 年提出后直到 26 年后才由 Carlier 和 Pinson 求得最优 Makespan。

- ABZ 类

由 Adams 设计（1988）。包括五个问题 ABZ5，ABZ6，ABZ7，ABZ8，ABZ9。其中 7，8，9 问题规模 20×15 难以求解。

- YN 类

由 Yamada 和 Nakano 设计（1992）。包括四个问题 YN1，YN2，YN3，YN4。问题规模 20×20。

- ORB 类

由 Applegate 和 Cook 设计（1991）。包括十个问题 ORB1 – ORB10。问题规模 10×10。

- SWV 类

由 Storer 设计（1992）。包括四个不同规模的二十个问题 SWV1–SWV20。问题规模分别为 20×10，20×10，50×10 和 50×10。

- LA 类

由 Lawrence 设计（1984）。包括八个不同规模的四十个问题 LA1 – LA40。问题规模分别为 10×5，15×5，20×5，10×10，15×10，20×10，30×10 和 15×15。

- TD 类

由 Taillard 设计（1993）。包括八个不同规模的八十个问题 TD1 – TD80。问题规模分别为从 15×15，20×15，20×20，30×15，30×20，50×15，50×20 和 100×20。

题规模分别为从 15×15，20×15，20×20，30×15，30×20，50×15，50×20 和 100×20。

- DMU 类

由 Demirkol 设计(1998)。包括八个不同规模的八十个问题 DMU1 – DMU80。问题规模分别为从 20x15，20x20，30x15，30x20，40x15，40x20，50x15 和 50x20。

在下面的仿真实验中，算法测试了 FT 类，LA 类，ABZ 类问题。

4.1.2 仿真结果与比较

为了检验 ETA 算法的性能，测试著名的 FT 类问题。以典型的 10 工件，10 机器的 10x10JSSP 问题 FT10 为例说明。在附录 1 中给出了 FT 类算例数据，对于第 k 行数据表示第 k 号工件的操作按加工先后顺序对应的机器号（工艺约束）及相应的加工时间，其中机器编号由 0 至 $m-1$。比如 FT06 第 1 行数据表示工件 1 在机器 2 上加工 1 单位时间，然后在机器 0 上加工 3 单位时间，最后在机器 4 上加工 6 单位时间。将算例分解为加工工序表和加工时间表，见表 4-1，表 4-2。

表 4-1：FT10 的加工工序

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
J_1	1	2	3	4	5	6	7	8	9	10
J_2	1	6	2	5	3	8	7	9	10	4
J_3	2	1	4	3	10	6	8	7	5	9
J_4	3	1	2	8	4	10	5	7	6	9
J_5	2	3	1	5	6	4	10	8	7	9
J_6	7	2	1	4	9	3	8	10	5	6
J_7	2	1	4	3	10	6	5	9	8	7
J_8	2	3	1	10	5	4	6	9	7	8
J_9	1	2	5	3	9	4	7	8	10	6
J_{10}	2	1	3	8	9	7	4	10	5	6

表 4-2：FT10 的加工时间

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
J_1	29	78	9	36	49	11	62	56	44	21
J_2	43	28	90	69	75	46	46	72	30	11
J_3	85	91	74	39	33	10	89	12	90	45
J_4	71	81	95	98	99	43	9	85	52	22
J_5	6	22	14	26	69	61	53	49	21	72
J_6	47	2	84	95	6	52	65	25	48	72
J_7	37	46	13	61	55	21	32	30	89	32
J_8	86	46	31	79	32	74	88	36	19	48
J_9	76	69	85	76	26	51	40	89	74	11
J_{10}	13	85	61	52	90	47	7	45	64	76

算法参数设置如下：20（种群大小），2（种群级数），0.7（杂交概率），0.1（变异概率），2000(终止代数)。$C_{max}=930$ 为最优排序。最优调度结果见表 4-3。

表 4-3：ETA 求解 FT10 的完成时间

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
J_1	9	523	532	568	617	628	707	777	836	908
J_2	72	665	314	637	430	799	753	885	915	441
J_3	493	399	606	532	920	709	842	721	699	887
J_4	256	81	179	735	355	842	364	501	416	788
J_5	262	308	224	396	499	369	895	579	520	665
J_6	408	86	84	233	505	138	492	530	281	353
J_7	185	132	327	294	753	448	427	698	609	480
J_8	361	445	210	892	554	522	645	813	718	766
J_9	148	286	506	370	694	421	557	668	792	517
J_{10}	275	217	388	787	877	675	395	**930**	480	593

FT10 的甘特图表示如图 4-1 所示。为了比较禁忌演化算法与其它算法的求解性能。分别用遗传算法和禁忌搜索算法测试 FT10 算例。其中通过 GA 求得 $C_{max}=1075$，通过 TS 求得 $C_{max}=1053$。可见 ETA 与这两种算法相比性能有一定的提升。

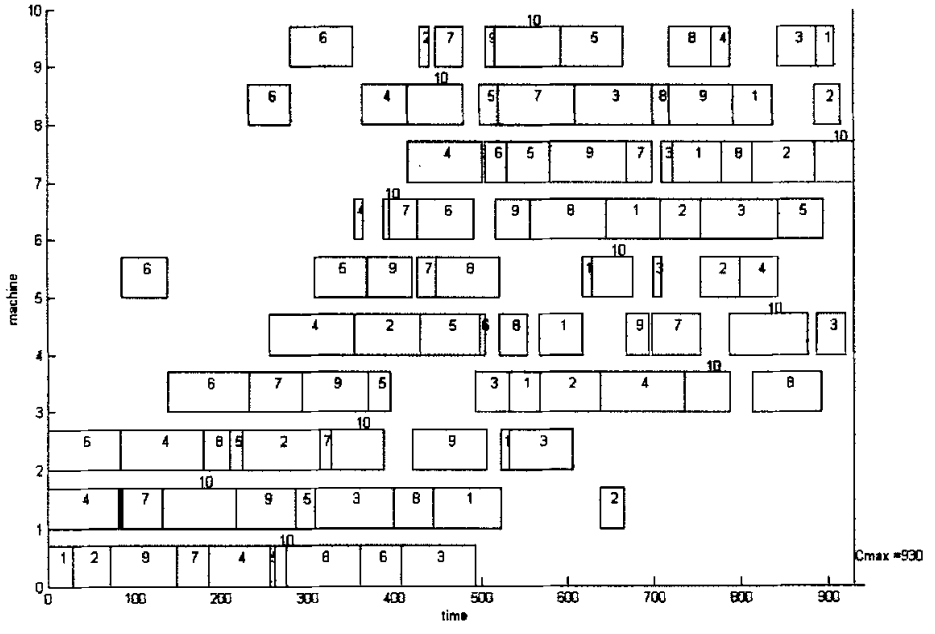

图 4-1 ETA 求解 FT10 完工时间甘特图

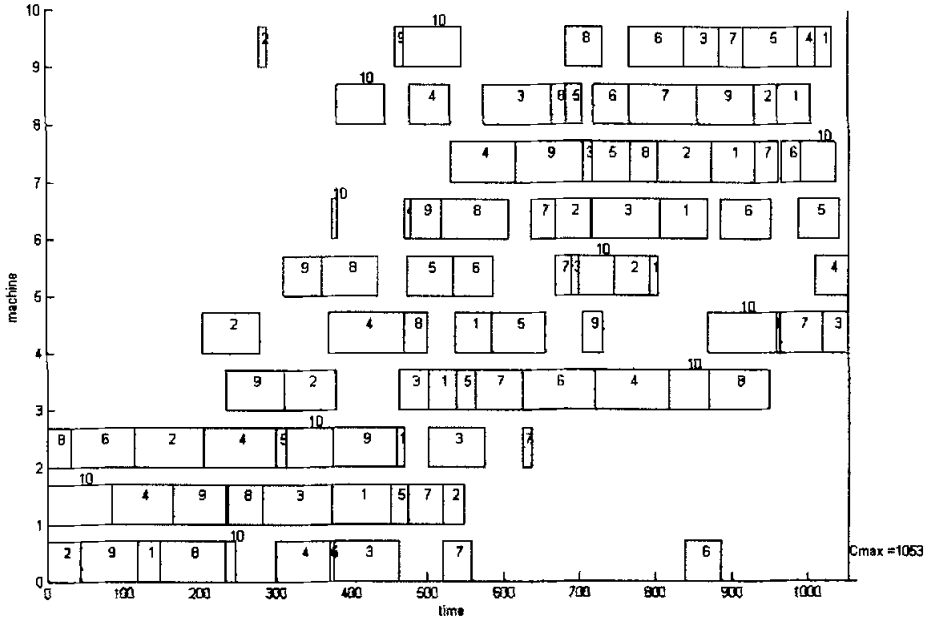

图 4-2 TS 求解 FT10 完工时间甘特图

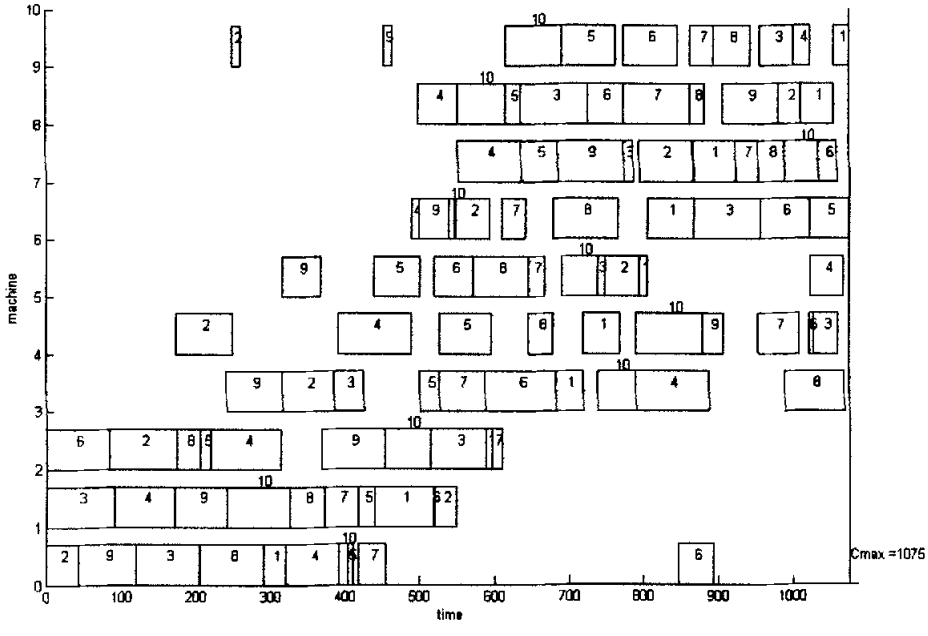

图 4-3 GA 求解 FT10 完工时间甘特图

对 FT 类其它问题 FT06，FT20 分别给出其运算结果，见图 4-4，图 4-5 和表 4-4。

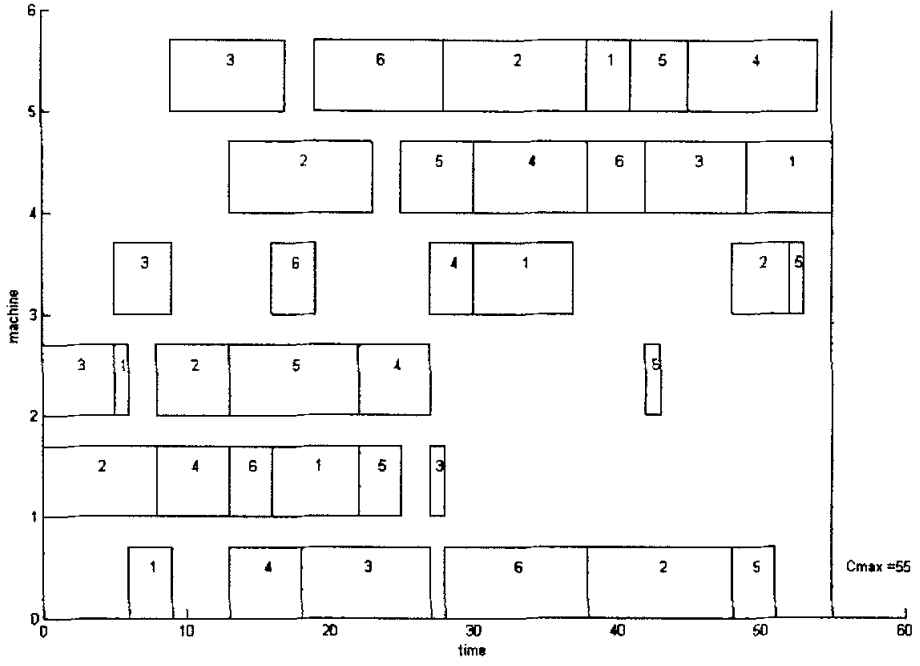

图 4-4 ETA 求解 FT06 完工时间甘特图

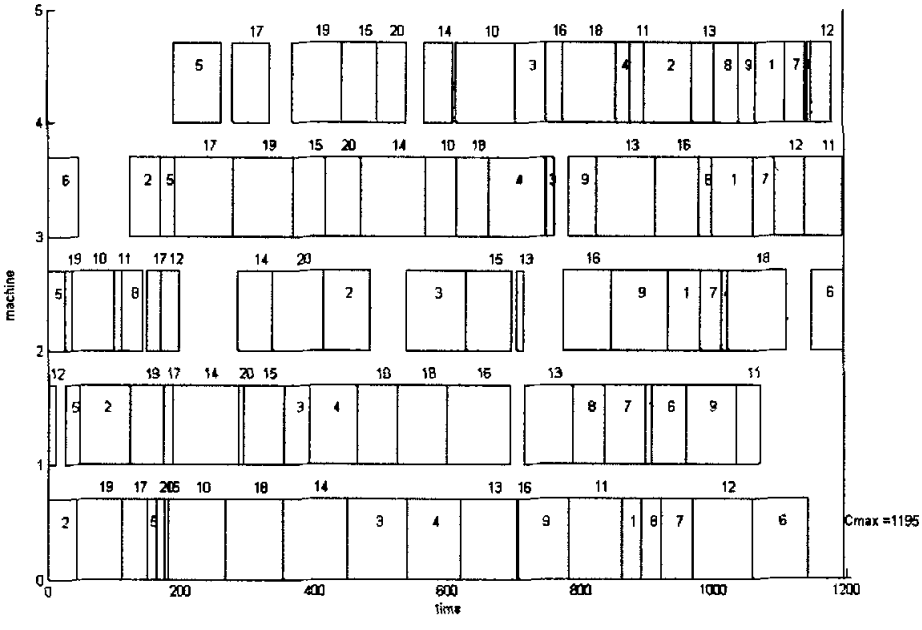

图 4-5 ETA 求解 FT20 完工时间甘特图

表 4-4 FT 类问题研究

n	*m*	问题	最优解	首先找到最优解的研究者[51]	ETA
6	6	FT06	55	Balas(1969)	55
10	10	FT10	930	Lageweg(1984)	930
20	5	FT20	1165	McMahon and Florian(1979)	1195

4.1.3 性能比较

接下来，我们测试了更的 Benchmark 问题，应用 GA ，TS，ETA 算法分别测试了不同规模大小的问题。结果见表 4-5。

表 4-5 ETA 算法测试及与文献结果比较

实例名称	规模及最优解				GA[13]	TS[13]	ETA
	n	*m*	工序数	最优解			
LA02	10	5	50	655	655	655	655
FT10	10	10	100	930	930	930	930
LA19	10	10	100	842	842	842	842
LA21	15	10	150	1046	1047	1048	1051
LA24	15	10	150	935	938	942	938
LA25	15	10	150	977	977	977	977
LA27	20	10	200	1235	1235	1255	1235
LA29	20	10	200	(1153)	1157	1167	1221
LA36	15	15	225	1268	1268	1268	1279
LA37	15	15	225	1397	1403	1415	1424
LA38	15	15	225	1196	1201	1199	1210
LA39	15	15	225	1233	1233	1233	1233
LA40	15	15	225	1226	1231	1229	1243
ABZ07	20	15	300	(656)	658	666	667
ABZ08	20	15	300	(669)	670	680	678
ABZ09	20	15	300	(679)	682	688	692

4.2 本章小结

本章介绍了作业车间调度问题的 Benchmark 测试集，以 FT 类问题及不同规模的 JSSP 问题对算法进行了测试。结果表明算法有较好的寻优性能。

第五章 禁忌演化算法的应用

5.1 旅行商问题

旅行商问题(Traveling Salesman Problem, TSP)与 JSSP 一样是组合优化中研究最多的问题之一，它是优化领域里的研究热点[48]。TSP 可描述如下一销售商从 n 个城市中的某一城市出发，不重复地走完其余个城市并回到原出发点，在所有可能的路径中求出路径长度最短的一条。其数学模型为：

距离之和最小

$$\min \sum_{i=j} d_{i,j} x_{i,j} \tag{5-1}$$

商人从城市 i 出来一次

$$\sum_{j=1}^{n} x_{i,j} = 1, \quad i = 1,2,\cdots n \tag{5-2}$$

商人走入城市 j 只有一次

$$\sum_{i=1}^{n} x_{i,j} = 1, \quad j = 1,2,\cdots n \tag{5-3}$$

每个城市经过一次

$$\sum_{i,j \in S} x_{x,j} \le |S| - 1 \qquad 2 \le |S| \le n-2,\ S \subset \{1,2,\cdots,n\} \tag{5-4}$$
$$x_{i,j} \in \{0,1\} \qquad i,j = 1,2,\cdots,n \qquad i \ne j$$

其中 $x_{i,j} = 1$，表示商人行走的路线包含城市 i 到城市 j 的路径；$x_{i,j} = 0$，表示商人没有选择该路径。$|S|$ 表示集合 S 中元素个数。

由于 TSP 编码方式与 JSSP 编码方式的相似性，采用 ETA 算法，设定群体规模采用 50，分为 4 级。以 50 个城市的标准算例[17]进行计算。求解过程如图 5-1，5-2，5-3，5-4 所示。在第 150 代求得最优解为 551.314。

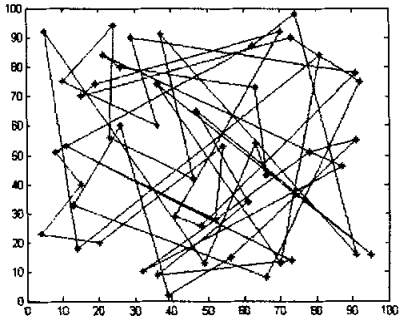
图 5-1 第 1 代

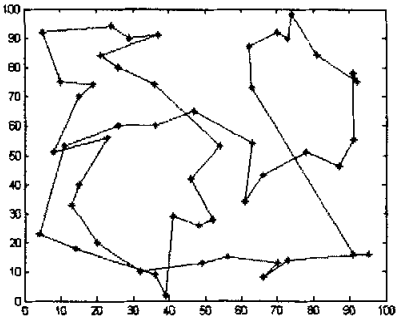
图 5-2 第 50 代

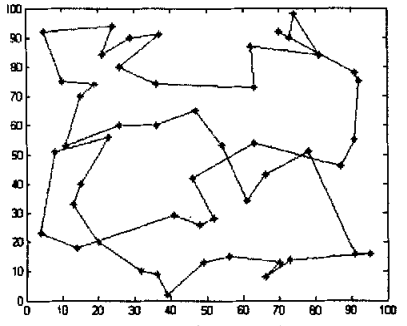
图 5-3 第 100 代

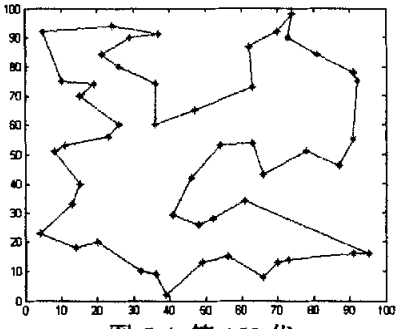
图 5-4 第 150 代

5.2 生产排产系统

从 20 世纪 80 年代初开始，人们就一直在尝试并致力于解决实决实际调度问题。调度研究从理论走向实际这是必然的一步。我们结合具体的实际项目，来描述如何将 ETA 植入生产执行调度系统框架之中。项目来源于浙江某汽车零部件有限公司的排产系统(Master Production System, MPS)，在其生产管理子模块部分中，需要实现车间作业调度功能。

该企业主要生产转向泵配件，转向机配件，轿车脚踏板总成为主的冲压件。不仅有企业制造系统典型过程中共性因素：约束信息、目标信息和设备运行状况、原材料供应情况、能源供应情况、生产、供应、销售部门遇到的突发事件情况、企业外协信息，其生产又有多品种，小批量的特点，非常适合将作业车间调度问题理论算法应用于企业信息系统，实现对企业生产需求的数据挖掘。

企业采用传统的手工方式编制生产作业计划，这种静态的、工作量极大的

计划通常需要二到三个工作日，其调整一般只能通过生产调度员的经验来进行。某一生产条件的变化甚至导致整个计划无法进行。这种操作是盲目的。制造企业 ERP 系统的投入运行，提高了效率，但由于制造企业的海量基础数据和计算复杂性，一个可行的生产计划难以实现。我们意识到对生产计划的优化更是一个长期过程。

那么如何将禁忌演化算法这样的智能算法应用于排产系统中呢？首先我们应该认识到智能算法的求解的问题并不依赖于问题所表征的物理意义，只要问题能表达成形如 $Min\ \{F_i(X)\},\ subject\ to\ G_i(x)<0,\ i = 1, \cdots; n.$ 这样的模型，智能算法就可以求解。既然如此，那么为什么从人工智能技术发展自今，理论算法层出不穷（如本文设计的 ETA 算法）能够投入到实际运行中的技术却很少呢？这里有两个主要原因：

第一，计算机求解速度慢，理论模型放宽了约束，用演化计算这样的全局搜索方法依然效率低，更何况企业的海量数据。

第二，生产约束难以量化。专家系统在调度中的不成功就表明新的约束的增加可能导致系统模型需要重新定义，或者收缩了可行域空间甚至无解。

因此，我们相信设计新的生产调度框架是非常有意义的。

5.2.1 系统输入

生产计划生成的功能可描述如下：根据要货计划，针对总成品，查询 BOM 得到组成零件。对每个零件，在满足约束条件下，调度可用车间资源（加工工序顺序、工人数量、机器数量、加工日产量、工序要求加工精度、机器加工精度、工人等级），将加工任务安排到相应生产部门各加工等级的工人及其负责的机器。针对零件，直接安排生产计划，当其与总成品组成零件相同时，合并两者的数量。系统输入为根据客户需求制定的要货计划，。

- 客户需求

客户的需求，是生产中最重要的一环，它决定需要生产的产品和完成的时间。在该系统中客户的需求可分为两类，总成件和工件。每个总成件由一系列工件组成。在 MPS 中，总成件的关系在物料清单(BOM)中体现。每个工件都必须在专用的机器上由相应工种的工人按指定的工艺流程操作。在生产计划排产之前需要载入相应工件的相关工序。通常情况下，工件处理受到严格的工艺路线约束，各道工序的先后关系不能颠倒。同时，由于设备的处理能力有限，工

件必须按顺序在机床上处理。

- 要货计划

产品决定了要生产的工件。通过要货计划定义了成品，生产数量，生产截止日期。在客户需求和要货计划之间有明显的相似性，其不同之外在于要货计划不用反映客户需求，也不用与某一需求订单直接关联。要货计划只用于生产，满足库存需要。复杂的产品，如汽车，其组成不仅包括总成件，工件。总成件里又有小总成。但最终加工的是一个个工件。

需事先指定的约束，见表 5-1。

表 5-1 约束

约束	类型
机器约束	1、对于同类型的机器，优先安排到已安装好夹具的机器； 2、对于加工周期超过一个月的工件，将计划安排到下月直到任务排完
工序转序约束	1、到达总数量的几分之一 2、现场工件堆放的数量 3、标准工时
生产计划优先级约束	1、优先 2、正常
车间指定约束	1、可以将生产总计划及生产进度控制计划下到不同的车间； 2、只能将生产总计划及生产进度控制计划下到指定的车间； 3、根据需要，将上述两种情况组合。

5.2.2 生产动态约束处理

本文的讨论的是静态调度问题，它其如下假设条件：(1)被调度的工件集合是确定的；(2)工件的加工时间是确定的，并且在安排计划时全部工件都已到达；(3)加工工件的机器是连续可用的。而在实际生产中，会不断的面临新情况、新

变化，如工件随机到达、加工机器出现故障等随机事件，使得事先确定的调度方案不能正常执行，这就需要安排重调度，因此调度方案的执行是个动态的过程。引起调度环境变化从而需要进行动态调度的事件称为动态事件，也称重调度因子。生产过程的随机性、不确定性往往要求不断进行重新调度，应避免追求全局最优化带来的巨大计算量。离散化整个过程，即可将动态调度转化为多次的静态调态调度。这就需要对生产过程的约束进行量化和建模。

表 5-2 生产调度中的动态事件

工件	机器	工序	其它
1、工件随机到达	1、机器损坏	1、工序延误	1、操作人员不在场
2、工件加工事件不确定	2、负载有限	2、质量否决	2、原材料延期到达
3、交货期变化	3、机器阻塞/死锁	3、生产不稳定	3、原材料有缺陷
4、动态优先级	4、生产能力冲突		4、动态加工路线
5、定单变化			

表 5-3 导致生产计划延误的动态约束

延误	策略
机器坏	1、技术人员或机器生产厂家抢修机器（特别是瓶颈设备）
	2、将任务转移到同类型的空闲机器
	3、购买新机器
员工生病、请假、辞职	1、请临时工替代
	2、要求同工种员工加班
紧急订单	1、将任务插入还空闲的生产资源上
	2、要求工人加班
	3、根据订单的优先级（从低到高）停止某个或多个零件的加工
外协	1、安排新任务（新订单）
	2、安排临时的加班任务
	3、员工放假

随着约束条件的增加，计算求解的复杂性随之增加，就可能导致解空间的收缩，甚至无解。动态事件的种类有很多，将动态事件分成以下四类，见表 5-2。

在实际的动态约束条件与企业的生产管理方式相关，设计生产排产系统之前，我们进行详细的需要分析，将各种可能导致生产计划改变的情况见表 5-3。由于企业自动化程度的限制，排产系统还需要人的参与。对于需要向上反映的情况，反映到生产部长，由其选择对应策略。当需要变动生产计划时，需保存原始的和改动的生产总计划及生产进度控制计划。

5.2.3 生产排产系统的目标

根据企业的需求，生产排产系统的目标定为，(1)结果正确；(2)操作方便；(3)调度时间短，从人工排产的 2 天减少到 2 到 10 分钟；(4)产生经济效益。

5.2.4 系统示例

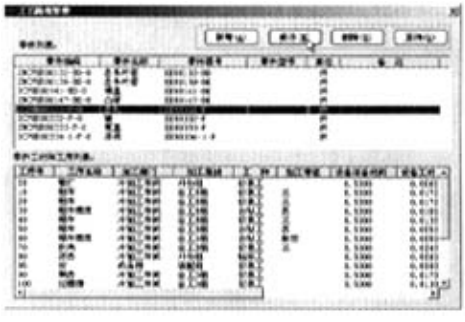

图 5-5　工艺路线管理

图 5-6 工序管理

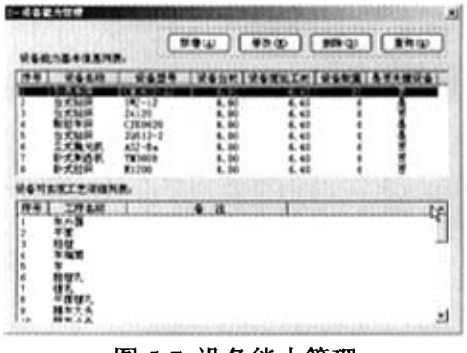

图 5-7 设备能力管理

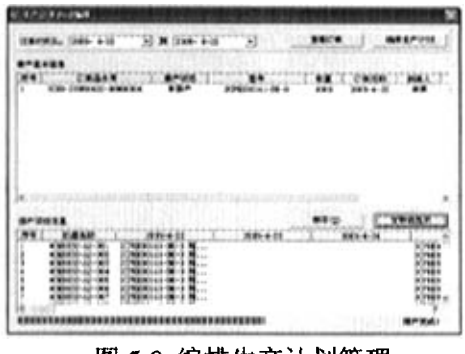

图 5-8 编排生产计划管理

生产排产系统编排生产计划时，通过查寻企业资源能力，工件的工艺路线（图5-5），生产工序（图5-6），生产设备（图5-7），进行后台计算得到排产结果（图5-8）并最终生成EXCEL生产计划表。

5.3 智能调度框架

算法实现的一个重要目的是指导企业对现有资源进行合理配置和充分利用。好的调度要求快速准确，它能(1)分析资源之间的相互影响，解决冲突；(2)提供指导，算法理论研究追求最优解，实验应用算法追求有效的可行调度。这时可行调度可能有多种，对于管理者作出决策提供的指导。纳入管理者，人的经验，亦是算法成功的一个表现；(3)分析与预测，实际生产过程中会出现很多故障和特殊情况，算法的智能性将对此提供基础判断，有利于工程技术人员较短时间解决问题。

现实中，能够执行的生产调度计划才是有效的方案。基于此，生产执行调度系统框架由数据库、调度器、生产监控三部分组成。见图5-9。

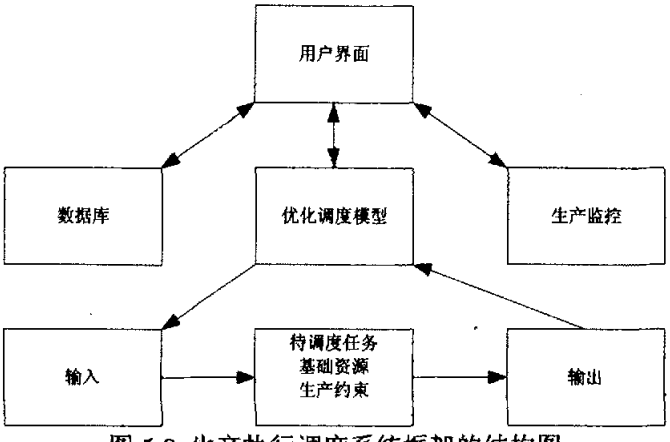

图5-9 生产执行调度系统框架的结构图

有很多算法可用于求解作业车间调度问题，那么当设计出一种性能表现好的算法是否在生产调度系统中只采用这一种算法呢？首先，根据NO FREE LUNCH（NFL）定理[46]可知，每种算法都有其一定的求解局限，它并不能解决所有的问题，即使在理论上也不能。其次，实际生产中存在大量不确定性因素，单一算法计算求解的稳定性无法保证。以上的问题，在调度器中可以通过决策的模

式处理。处理过程是让多种优化算法同时进行运算，比较每种算法的结果，选择最好的结果作为调度方案，见表 5-6。以表中星期五为例说明，在这次调度决策中，TS、SA(Simulated Annealing)、CH(Climbing Hill)、ETA 均得出可行解，其中 TS 胜出，GA 无法求出可行解。

表 5-6 调度器的决策模式

		星期日	星期一	星期二	星期三	星期四	星期五	星期六
算法 1	GA	○		○			-	
算法 2	TS						○	○
算法 3	SA		○					
算法 4	CH	-		-	○			
算法 5	ETA	-				○		

注：其中○表示得出当日该次调度方案的算法，-表示算法在该次调度中失效

当多种算法在系统后台共同参与决策时，计算的速度如何保证，在普通计算机上一种优化算法的求解时间也很长。这里应该指出，由于企业的海量数据及计算的复杂，普通性能的工作站难以实现系统的快速优化，近年来网格计算的发展，生产执行调度系统的开发应结合网格计算。在网格计算中，资源的调度优化和服务执行是一个关键技术问题，它主要包括请求的调度优化、资源的调度优化和资源的服务执行。请求的调度优化要对用户资源请求与可用资源进行匹配，当众多用户和应用请求同时到达，就必须统筹优化安排多个请求的资源需求。所以，选择这样的调度器时，并行性和大规模计算是需事先考虑的。

5.4 本章小结

本章从算法应用的角度，测试了与 JSSP 类似的组合优化问题 TSP 问题，并阐述将禁忌演化算法植入实际调度系统，生产排产系统的开发。通过分析智能算法应用于实际调度系统中失败的原因，以此为基础提出一种智能调度框架，该框架的特点在于生产反馈信息的收集和调度器多种算法共同决策的模式。

第六章 结论与展望

6.1 结论

在前面几章中，本文针对作业车间调度问题进行研究，介绍了生产调度问题的分类和国内外研究进展，以及演化算法的研究发展动态；介绍了静态的作业车间调度问题和遗传算法求解 JSSP 问题的主要操作；提出将禁忌搜索与演化计算相结合的禁忌演化算法；给出了用禁忌演化算法求解 JSSP Benchmark 算例的仿真结果；将禁忌演化算法应用在旅行商问题和生产排产系统中。本文的主要工作和具体研究成果总结如下：

(1) 首先系统总结了生产调度问题的国内外研究现状，分析了目前该研究领域存在的问题及发展趋势，为论文的研究奠定了理论基础。

(2) 对作业车间调度问题进行了研究，给出了它的数学模型及其混合优化算法求解技术，模型包括优化目标和约束。

(3) 将禁忌搜索和演化计算，局部搜索和全局搜索相给合设计了禁忌演化算法。在新解的产生过程中引入了分级策略，对种群按照适应值进行划分，对高级别的个体采用禁忌操作，处于低级别的个体采用遗传操作，避免了新解产生的盲目性，从而使得子代较好继承了父代的优势特征。并以该算法测试典型作业车间算例，结果表明新算法有较好的寻优性能。

(4) 将禁忌演化算法测试与 JSSP 类似的组合优化问题旅行商问题，并阐述将算法植入实际调度系统，生产排产系统中的开发过程。

6.2 展望

(1) 生产调度问题智能优化算法的计算机形式化描述。生产计划与调度集成具有全局优化的特征，在先进制造模式的环境下，提高生产系统的柔性。

(2) 基本上可以认为对特定企业的项目是一种偏见性研究，调度存在于现代社会的各个层面，普适性研究似乎可以将问题都纳入目标约束之中。*Min* $\{F_i(X)\}$, *subject to* $G_i(x)<0, i = 1, \cdots, n.$ 不确定因素的量化，中间件是研究的

目的。

(3) 任何以一种算法决定一个系统的项目都不会太成功，应以多种算法共同决策，进化计算的方法必然会在其中。

(4) 一些应观注的概念，Agent，Petri，网格这些方法与现行调度方法的结合可能改善系统的性能。

参考文献

[1]. 徐俊刚，戴国忠，王宏安. 生产调度理论和方法研究综述[J]. 计算机研究与发展，2004, 41(2):257~267.

[2]. 邓锋，孙树栋. 混合型企业生产调度问题综述 [J]. 制造业自动化, 2004, 26(2):1~4.

[3]. Zbigniew Michalewicz, Martin Schmdit, Matthew Michalewicz, Constantin Chiriac. Prediction, Distribution, and Transportation: A Case Study [C]. First International Symposium on Intelligence Computation & Applications (ISICA' 2005), China University of Geosciences, Wuhan, April, 2005, 545~557.

[4]. Graves S. A review of production scheduling [J]. Operations Research. 1981,29(4):646~675.

[5]. 王凌. 车间调度及其遗传算法[M]. 清华大学出版社,2003.

[6]. 玄光男，程润伟.遗传算法与工程优化[M]. 北京：清华大学出版社, 2004.

[7]. Dirk Christian Mattfeld. Evolutionary Search and the Job Shop: Investigations on Genetic Algorithms for Production Scheduling [OL]. Springer / Physica. PhD thesis, 1996. http://www-public.tu-bs.de:8080/~dcmatt/papers.htm Accessed Jan. 2005.

[8]. Serol Bulkan. A genetic algorithm approach to job shop scheduling. PhD thesis.1999

[9]. 熊禾根. 模具企业动态车间作业计划研究：[博士学位论文]. 武汉：华中科技大学材料加工工程, 2005.

[10].JSSP Software. LiSA[OL]. http://lisa.math.uni-magdeburg.de. 2003

[11].JSSP Software. HeuristicLab [EB/OL]. http://www.heuristiclab.com. Accessed December, 2005.

[12].José Fernando Gonçalves, Jorge José de Magalhães Mendes and Maurício G. C. Resende. A hybrid genetic algorithm for the job shop scheduling problem [J]. European Journal of Operational Research, 2005. 167(1): 77~95.

[13].A. MORAGLIO, H.M.M. TEN EIKELDER, R. TADEI. Genetic Local Search for Job Shop Scheduling Problem [OL]. http://privatewww.essex.ac.uk/~amoragn. Accessed March, 2005.

[14].Feng-Tse Lin. Constructing A Job-Shop Scheduling Model Based on Imprecise Data [C]. The IEEE International Conference on Fuzzy Systems. pp.1374-1379, 2003.

[15].黄明，闰淑娟，梁旭. 遗传算法和禁忌搜索算法在车间调度中的研究进展[J]. 工业控制

计算机, 2004, 17(2):4~5.

[16].Yan, L. Solving combinatorial optimization problems with line-up competition algorithm [J]. Computers & Chemical Engineering, 2003, 27(2), 251~258.

[17].鄢烈祥. 用列队竞争算法解旅行商问题 [J]. 运筹与管理, 1999, 8(3): 24~30

[18].彭志刚，吴广宇，杨艳丽，徐心和. 一机两流的连铸生产计划模型与算法[J]. 东北大学学报. 2000, 21(3): 244~246.

[19].余琦玮. 基于遗传算法的作业车间调度问题研究: [硕士学位论文].浙江: 浙江大学机械与能源工程学院, 2004.

[20].陈东升. 基于遗传算法的模糊车间作业调度问题的研究: [硕士学位论文]. 大连: 大连理工大学机械电子工程系, 2005.

[21].万芳. 基于遗传算法的车间作业调度问题研究与应用: [硕士学位论文]. 江西: 南昌大学计算机软件与理论, 2005.

[22].潘全科，孙志峻，朱剑英. 基于遗传算法的作业车间调度优化[J]. 机械科学与技术. 2002, 21(6):998~1000.

[23].姜思杰，张付亮，王孔茂. 基于遗传和禁忌算法求解一类车间调度问题[J]. 计算机集成制造系统-CIMS. 2003, 9(11):984~988.

[24].姜思杰，徐晓飞，李全龙. 基于遗传优化算法求解作业车间调度问题[J]. 计算机集成制造系统-CIMS. 2002, 8(3):229~230.

[25].陈雄，李海刚，吴启迪. 基于遗传算法的Job-shop调度问题研究[J]. 同济大学学报, 2002, 30(1):88~91.

[26].谢胜利，董金祥，黄强. 基于遗传算法的车间作业调度问题求解[J]. 计算机工程与应用. 2002, (10):79~82.

[27].吴云高，王万良. 基于遗传算法的混合 Flowshop 调度[J]. 计算机工程与应用. 2002, (12):82~84.

[28].徐本强. 车间调度的 MAS 智能决策技术研究与应用: [硕士学位论文]. 大连: 大连海事大学计算机学院, 2004.

[29].赵巍. 基于多智能体的生产调度方法及其应用: [硕士学位论文]. 浙江: 浙江工业大学, 2004.

[30].李进. 面向快速制造的车间调度策略研究: [硕士学位论文]. 南京: 南京航空航天大学机械工程, 2004.

[31].梁书锋. 柔性车间管理系统的设计与实现: [硕士学位论文]. 武汉: 武汉理工大学机电

学院, 2005.

[32].吴培栋. 实际生产系统中考虑任务相关性的作业计划规则调度算法研究与实现: [硕士学位论文]. 武汉: 武汉科技大学机械自动化学院, 2005.

[33].朱红. 用 DNA 算法求解车间调度问题的研究: [硕士学位论文]. 黑龙江: 哈尔滨理工大学计算机与控制学院, 2005.

[34].R.M. Aiex, S. Binato, M.G.C. Resende. Parallel GRASP with path-relinking for job shop scheduling [OL]. http://www.optimization-online.org. Accessed March, 2005.

[35].李霄峰, 邵惠鹤, 任德祥. 求解混合 Flow shop 调度问题的简化禁忌搜索方案 [J]. 上海交通大学学报, 2003, 37(4):516~519.

[36].童刚, 李光泉, 刘宝坤. 一种用于 Job-Shop 调度问题的改进禁忌搜索算法[J]. 系统工程理论与实践, 2001, 21(9):48~52.

[37].T. Baeck, D. Fogel, and Z. Michalewicz, Evolutionary Computation: Advanced Algorithms and Operators [M]. Institute of Physics, London, 2000

[38].Zbigniew Michalewicz, David B. Fogel, 如何求解问题—现代启发式方法[M], 北京: 中国水利水电出版社, 2003.

[39].Darwin and Natural Selection. http://anthro.palomar.edu/evolve/evolve_2.htm. Accessed December, 2004.

[40].Michalewicz, Z., Genetic Algorithms + Data Structures = Evolution Programs[M]. Springer-Verlag, 1994.

[41].潘正君, 康立山, 陈毓屏. 演化计算 [M]. 北京: 清华大学出版社 (第 2 版), 2000.

[42].黄樟灿. 演化计算的搜索策略研究: [博士学位论文]. 武汉: 武汉大学计算机学院, 2005.

[43].Box J E P. "Evolutionary operation: a method for increasing industrial productivity", Appl Statistics,1957, 6(2): 81~101

[44].The genetic algorithms archive. Washington DC, USA: Naval Research Laboratory [OL]. http://www.aic.nrl.navy.mil/galist, Accessed April, 2004

[45].Jim Gray. What Next? A dozen information-technology research goals [J]. Journal of the ACM, Vol.50, No.1, January 2003.

[46].D. H. Wolpert, W. G. Macready. No Free Lunch theorems for optimization. IEEE Transactions on Evolutionary Computation, 1997, 1(1):67~82.

[47].HUANG Zhangcan, WANG Zongyue, CHENG Hao. Similitude Frame Evolutionary Algorithm for Multi-modal Function Optimization [C]. In: Progress in Intelligence

Computation and Applications, Wuhan, China, 2005, 262~267.

[48].HUANG Zhangcan, LIN Zhiyi, CHENG Hao, WANG Zongyue. Similitude Frame Evolutionary Algorithm for TSP [C]. In: Progress in Intelligence Computation and Applications, Wuhan, China, 2005, 241~246.

[49].HU Yefa, CHENG Hao, HU Xiaolin, HUANG Zhangcan. A New Method for Solving the Continuous Multi-Object Optimization Problems [C]. In: Progress in Intelligence Computation and Applications, Wuhan, China, 2005, 268~274.

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

[51].A.S. Jain, S. Meeran. Deterministic Job Shop Scheduling: Past, Present and Future [J]. European Journal of Operational Research. 1999,113 390~434

[52].Suresh, V., Chaudhuri, D., 1993. Dynamic scheduling – A review International [J]. Journal of Production Economics 32, 53~63.

[53].De Jong. K. A. "An Analysis of Behavior of a Class of Genetic Adaptive Systems", [Doctoral dissertation], University of Michigan, 1993

致 谢

三年的硕士生阶段学习与研究生活已接近尾声。在本论文即将完成之际，对所有在课题研究和论文撰写过程中给予过我帮助的老师、同学、朋友和家人表示深深的谢意。

首先，感谢我的导师胡业发教授，胡老师渊博的知识、严谨认真的治学态度以及在本文开题、研究到论文撰写的过程中对我的悉心指导都给我以巨大的帮助。我很幸运从师于这样的导师。

感谢吴波教授，丁毓峰副教授，吴华春博士，在课题研究过程中各位老师对我耐心指导，热心帮助，他们的宝贵意见和建议使我深受启发。

感谢理学院数学系黄樟灿教授，化学工程系胡萍副教授几年来对我不断的鼓励，支持和指引。

感谢硕士生雷波，高小明，石国新，武广州，在系统开发过程中，和各位同学的合作，使系统有效快速实现。

感谢计算机学院硕士生王宗跃，林志毅，数学系硕士生李炜，焉炳艳，严商，香港中文大学博士生胡晓林，与大家的在不同学科领域的讨论和交流，激发我很多灵感。

感谢黄文华，陈雷，蒋丹珺，谢谢各位的友谊和关心。

最后衷心地感谢我的家人，是他们一如既往的关怀、支持和培养，使我在研究和生活中一直保持乐观的心态和前进的动力。

成浩

2006 年 05 月于武汉马房山

附录 1 典型 JSSP 测试算例

FT06	2 1 0 3 1 6 3 7 5 3 4 6 1 8 2 5 4 10 5 10 0 10 3 4 2 5 3 4 5 8 0 9 1 1 4 7 1 5 0 5 2 5 3 3 4 8 5 9 2 9 1 3 4 5 5 4 0 3 3 1 1 3 3 3 5 9 0 10 4 4 2 1
FT10	0 29 1 78 2 9 3 36 4 49 5 11 6 62 7 56 8 44 9 21 0 43 2 90 4 75 9 11 3 69 1 28 6 46 5 46 7 72 8 30 1 91 0 85 3 39 2 74 8 90 5 10 7 12 6 89 9 45 4 33 1 81 2 95 0 71 4 99 6 9 8 52 7 85 3 98 9 22 5 43 2 14 0 6 1 22 5 61 3 26 4 69 8 21 7 49 9 72 6 53 2 84 1 2 5 52 3 95 8 48 9 72 0 47 6 65 4 6 7 25 1 46 0 37 3 61 2 13 6 32 5 21 9 32 8 89 7 30 4 55 2 31 0 86 1 46 5 74 4 32 6 88 8 19 9 48 7 36 3 79 0 76 1 69 3 76 5 51 2 85 9 11 6 40 7 89 4 26 8 74 1 85 0 13 2 61 6 7 8 64 9 76 5 47 3 52 4 90 7 45
FT20	0 29 1 9 2 49 3 62 4 44 0 43 1 75 3 69 2 46 4 72 1 91 0 39 2 90 4 12 3 45 1 81 0 71 4 9 2 85 3 22 2 14 1 22 0 26 3 21 4 72 2 84 1 52 4 48 0 47 3 6 1 46 0 61 2 32 3 32 4 30 2 31 1 46 0 32 3 19 4 36 0 76 3 76 2 85 1 40 4 26 1 85 2 61 0 64 3 47 4 90 1 78 3 36 0 11 4 56 2 21 2 90 0 11 1 28 3 46 4 30 0 85 2 74 1 10 3 89 4 33 2 95 0 99 1 52 3 98 4 43 0 6 1 61 4 69 2 49 3 53 1 2 0 95 3 72 4 65 2 25 0 37 2 13 1 21 3 89 4 55 0 86 1 74 4 88 2 48 3 79 1 69 2 51 0 11 3 89 4 74 0 13 1 7 2 76 3 52 4 45

附录 2 攻读硕士期间发表的论文

HU Yefa, **CHENG Hao**, HU Xiaolin, HUANG Zhangcan. A New Method for Solving the Continuous Multi-Object Optimization Problems. In: Progress in Intelligence Computation and Applications, Wuhan, China, 2005, 268-274. (ISTP 收录. ISIP: 000231083700043)