分类号 密级

U D C 编号

CENTRAL SOUTH UNIVERSITY

硕士学位论文

论 文 题 目 正交遗传算法及其应用研究

学科、专业 控制理论与控制工程

研究生姓名 江 中 央

导师姓名及
专业技术职称 蔡自兴 教授

日 期 **2008 年 4 月 25 日**

MS THESIS

Research on Orthogonal Genetic Algorithm and Its Applications

Specialty： Control Theory and Control Engineering

Master Degree Candidate： Zhongyang Jiang

Supervisor： Prof. Zixing Cai

School of Information Science and Engineering

Central South University

Changsha Hunan P.R.China

摘 要

在科学、工程和商业等领域中，对很多实际问题进行数学建模后都可以转化为各类函数的优化问题。遗传算法是一种借鉴生物界自然选择和自然遗传机制的随机优化搜索算法，是解决各类函数优化问题的一种有效算法。然而，大量研究表明，传统的遗传算法也存在许多的不足和缺陷，如早熟收敛、计算量大和局部搜索能力弱。为了有效克服以上缺点，本文将正交试验设计方法和局部搜索技术引入到遗传算法中。提出了两种分别用于全局优化和约束优化的新的算法。最后将新的全局优化算法用到了新型PID免疫控制器的参数优化中。本文主要工作如下：

(1)提出了一种求解全局优化的混合自适应正交遗传算法。新算法根据父代个体的相似度，自适应地调整用于安排交叉操作的正交表的因素个数和对父代个体进行因素分割的位置，然后根据所选定的正交表重新设计交叉算子，提出了一种自适应的正交交叉算子。同时为了进一步提高传统遗传算法学习和利用搜索空间局部信息的能力，提高其收敛速度，引入了局部搜索策略，提出了一种新的基于种群分割和单形交叉的聚类局部搜索策略。对 14 个高维的 Benchmark 函数的测试结果表明，该算法在性能上显著优于其它算法。

(2)提出了一种新的基于正交试验设计的约束优化算法。在搜索机制方面，利用正交实验设计的方法来安排多个父代个体的交叉操作，提出了一种新的多父体正交交叉算子。此外，我们利用单行交叉算子对父代种群进行并行搜索，来协调算法的勘探和开采能力。在约束处理技术上，本文引入了一个衡量个体的优、劣的新比较准则。并通过 13 个标准的测试函数验证了算法的通用性和有效性。

(3)针对 P 型免疫反馈控制器不能克服动态干扰和消除静态误差的问题，利用模糊控制系统的非线性逼近能力，提出了一种将 P 型免疫反馈控制器同常规 PID 控制器进行混合联结的模糊免疫 PID 控制器的设计方法,并利用(1)中提出的混合自适应正交遗传算法对其控制器参数进行优化。仿真研究结果表明，该控制器较常规控制器具有更好的动、静态特性。

关键字 正交遗传算法，局部搜索，免疫控制，全局优化，约束优化

ABSTRACT

Many problems can be converted into all kinds of optimization problems in almost every field of science, engineering, business, etc.. Genetic Algorithms (GAs) are random optimization searching method based on the mechanics of natural selection and genetic theory, and are effective algorithms for solving all kinds of optimization problems. However, the results of a large number of researches indicated that the traditional genetic algorithms have many deficiencies and limitations, such as they are prone to premature convergence, high computational cost and weak local search abilities. To overcome the disadvantages mentioned above, this paper combines orthogonal experimental design methods and local search scheme to genetic algorithm, and two new algorithms are proposed to solve global optimization problems and constrained optimization problems separately. At last, the new global optimization algorithm is used for optimizing the parameters of the new PID immune feedback controller proposed in this paper. The main contribution and work are described as following:

(1) Proposed a hybrid self-adaptive orthogonal genetic algorithm (HSOGA). The new algorithm can self-adaptively adjusts the numbers and location of orthogonal array's factors when the two parent chromosome are divided, according to the similarity of the two parent chromosomes, and the selected orthogonal array is used to redesign the crossover operator and a new self-adaptive orthogonal crossover operator (SOC) is proposed. In addional, in the purpose of enhancing the local search ability and speeding up the convergence of the algorithm, a clusting local search scheme, whish is based on population splitting and simplex crossover is proposed. The new algorithm is test on fourteen benchmark functions, and the result demonstrates that the performance of the new algorithm is supper to the other algorithms.

(2) Proposed a new constrained optimization algorithm based on orthogonal design method. In search scheme, the orthogonal design method is used to arrange the crossover operation of multi-parent, and a new

multi-parent othogonal crossover is proposed. In addition, the simplex crossover is used to enrich the exploratory and exploitive abilities of the algorithm proposed. In constraint-handing technique, a new comparison criterion of comparing and selecting individuals is incorporated to the new algorithm. The new approach is tested on 13 well-known benchmark functions, and the empirical evidence demonstrates that the new approach is generic and effective.

(3) Considering the P-type immunity feedback controller can't get over dynamical disturbance and eliminate static error, according to the universal approximation capability of the fuzzy control system, a new design of fuzzy immune PID controller based on the mixed connection of conventional PID and P-type immunity feedback controller is put forward in the paper. In order to gain optimal controller parameters, the parameters of the controller are offline optimized by HSOGA. The simulation results demonstrate that the control performance of the this Immune controller is superior to that of conventional controller and this control method with higher dynamic and static performance

Key Words orthogonal genetic algorithm, local search scheme, immune control, global optimization, constrained optimization

目　录

第一章 绪论

1.1 遗传算法概述

遗传算法(Genetic Algorithm, GA)是一类借鉴生物界自然选择和自然遗传机制的随机优化搜索算法，由美国 J.Holland 教授于上个世纪 70 年代提出并发展起来的[1]，其主要特点是群体搜索策略和群体中的个体之间的信息交换，搜索不依赖梯度信息，操作对象是一组个体，而非单个个体，具有多条搜索轨迹，因而具有隐性并行性。遗传算法提供一种求解复杂系统优化问题的通用算法框架，该框架不依赖问题的的种类，是一类具有较强鲁棒性的优化算法，特别是对一些大型、复查非线性系统，它更表现出了比其他传统优化方法更加独特和优越的性能。它已被广泛地应用于计算机科学、工程技术、管理科学和社会科学等许多重要领域，成为 21 世纪计算智能科学中相当活跃的关键技术之一。

遗传算法是一种基于空间搜索的算法，它通过自然选择、遗传、变异等操作以及达尔文适者生存的理论，模拟自然进化过程来寻找所求问题的解答。因此，遗传算法的求解过程也可看作最优化过程并具有以下特点[2]：

(1) 遗传算法是对参数集合的编码而非针对参数本身进行进化；

(2) 遗传算法是从问题解的编码组开始而非从单个解开始搜索；

(3) 遗传算法利用目标函数的适应度这一信息而非利用导数或其它辅助信息来指导搜索；

(4) 遗传算法利用选择、交叉、变异等算子而不是利用确定性规则进行随机操作。

遗传算法利用简单的编码技术和繁殖机制来表现复杂的现象，从而解决非常困难的问题。它不受搜索空间的限制性假设的约束，不必要求诸如连续性、导数存在和单峰等假设，能从离散的、多极值的、含有噪音的高维问题中以很大的概率找到全局最优解。

1.2 遗传算法的发展历史

遗传算法的早期工作始于二十世纪 60 年代。1967 年，Holland 的学生 Bagley 在他的博士论文中首次提出了“遗传算法”这一术语，并采用双倍体编码，发展了与目前类似的复制、交叉、变异、显性和倒位等基因操作，他还敏锐地察觉到了防止早熟收敛的机理，并发展了自组织遗传算法的概念。与此同时，Rosberg 在他的博士论文中进行了单细胞生物群体的计算机仿真研究，对以后函数优化的研究颇有启发，并发展了自适应交叉策略。这些思想对于后来遗传算法的发展所

起的作用是十分明显的。

70 年代初，Holland 教授提出了遗传算法的基本定理—模式定理(Schema Theorem)，为遗传算法的研究奠定了数学基础。1975 年，Holland 出版了第一本系统论述遗传算法和人工自适应系统的专著《自然系统和人工系统的自适应性(Adaptation in Natural and Artificial Systems)》[1]。同年，De Jong 的博士文《遗传自适应系统的行为分析(An Analysis of the Behavior of a Class of Genetic Adaptive System)》完成，他结合 Holland 的模式定理进行了大量的纯数值实验，树立了遗传算法的工作框架，得到了一些重要的具有指导意义的结论，他推荐了在大多数优化问题中都比较适用的遗传算法参数，还建立了著名的 De Jong 五函数测试平台，定义了评价遗传算法性能的在线指标和离线指标，为遗传算法的深入研究奠定了坚实基础。

80 年代，遗传算法成为人工智能研究的一个热点。Holland 教授实现了第一个基于遗传算法的机器学习系统—分类器系统(Classifier Systems, 简称 CS)，开创了遗传算法的机器学习的新概念，为分类器系统够造出了一个完整的框架。1989 年，David Goldberg 出版了专著《搜索、优化和机器学习中的遗传算法(Genetic Algorithm in Search,Optimization and Machine Learning)》一书，该书系统地总结了遗传算法的主要研究成果，全面而完整地论述了遗传算法的基本原理及应用，奠定了现代遗传算法的科学基础。

进入 90 年代，遗传算法因其能有效地求解 NP 难问题以及非线性、多峰函数优化和多目标优化问题，得到了众多学科的高度重视，同时也极大地推动了遗传算法理论研究和实际应用的不断深入与发展。1991 年，Davis 编辑出版了《遗传算法手册(Handbook of Genetic Algorithm)》一书，书中包括了在科学计算、工程技术和社会经济中的大量应用实例[3]，同一时期，国内也有一些有关书籍相继出版[4,5]，为推广和普及遗传算法的应用起到了重要的指导作用。一系列以遗传算法为主题的国际会议变得十分活跃。从 1985 年开始，国际遗传算法会议，即 ICGA(International Conference on Genetic Algorithm)每两年举行一次。90 年代开始，《Evolutionary Computation》(MIT Press) 《IEEE Transactions on Evolutionary Computation》 (IEEE Press), 《SoftComputing》 (Springer Verlag),《Applied Soft Computing》 (Elsevier Science)和《Evolutionary Optimization-An International Journal on the Internet》等专业学术期刊相继创刊，更是推动着遗传算法实质性的进展。

1.3 遗传算法的研究现状

1.3.1 遗传算法的理论研究现状

Holland 的模式定理奠定了遗传算法的数学基础，它解析了遗传算法的搜索行为，为遗传算法的应用研究提供了初步理论的依据。迄今为止，遗传算法在全局收敛性分析方面取得了突破，运用的数学工具是马尔可夫链。Goldberg[6]采用马尔可夫链对标准遗传算法进行分析，并提出了最佳保留遗传算法，Eiben[7]应用马尔可夫链证明了一类基于精英保留机制的遗传算法的全局收敛性，Folgel[8]分析了没有变异算子的遗传算法的渐近行为，Rudolph[9]和 Qi[10]等对分别对二进制编码和浮点数编码的遗传算法进行了全局收敛性分析。在国内方面，梁艳春等[11]研究了基于扩展串的等价遗传算法的收敛性。张讲社等[12]提出了一类齐次、保证收敛且容易判断是否收敛的新型遗传算法，并证明了全局收敛的充要条件。郭观七[13]提出了独立于表示的遗传算法全局收敛性分析方法。

1.3.2 遗传算法的应用研究现状

遗传算法的应用研究比理论研究更为丰富，已渗透到许多学科。遗传算法的应用按其方式可分为三大部分，即基于遗传的优化计算、基于遗传的优化编程、基于遗传的机器学习，分别简称为遗传计算(Genetic Computation),遗传编程(Genetic Programming)、遗传学习(Genetic Learning)。

(1)遗传计算

遗传计算是 GA 最直接、最简单的应用，其面也最广。在自动控制学科中，陈根社[14]运用 GA 进行了 Riccati 方程求解。Murdock 等[15]用 GA 分析了控制系统的鲁棒稳定问题。Potte 等[16]运用 GA 研究了数字 PID 控制器的调节。Kristinsson 和 Dumant[17]深入研究了连续和离散的系统的参数辨识问题，用 GA 寻找零极点。Maclay 等[18]研究也显示了基于 GA 的参数辨识的潜力。Freeman 等[19]用 GA 精调漠糊逻辑控制器。Park 等[20]研究了一种新的基于遗传的模糊推理系统，用于产生优化参数集，获得了良好的性能。遗传算法的兴起伴随着神经网络的复活，令人新奇的是神经网络已成为 GA 应用最为活跃的领域。神经网络的应用面临着两大问题[21]：神经网络拓扑结构的优化设计与高效的学习算法。遗传算法已成为解决该两大问题的有力工具，用于优化神经网络的结构权重和学习规则。Yao[22]给出了该方面的详细综述，该领域的研究都显示了良好的性能和潜在的应用。遗传算法已渗透到了许多学科，如工程结构优化、计算数学、制造系统、航空航天、交通、计算机科学、电力、材料科学等等。

(2)遗传编程：遗传算法除了优化计算外还可以应用于更为复杂的情况，而要求强有力的编码表达是最关键的。Kaza，DJong 等发展了遗传编程的概念。

Kaza 运用 LISP 编程语言来编码，LISP 符号串表示树，串中由特定问题域的各种函数和终端组成，群体进行复杂超平面的搜索。Kaza 成功地把遗传编程用于人工智能、机器学习、符号处理等。遗传编码是遗传算法应用的深人，目前还不成熟，许多问题有待解决。

(3)遗传学习:遗传学习系统是由 GA 为内核构成的增强式学习系统,一般地，群体由产生式规则组成，利用和环境的接触来学习、完成给定任务。Holland 奠定了基于遗传的机器学习的框架，Holland 和 Reitman 发展了第一个遗传学习系统，称为认知系统一号(CS=I)，CSI 成为 GL 后继研究的模板，又称为分类器系统 (Classifier System)。在此基础上，Smith 发展了 LS-1，Schaffer 发展了 LS-2。遗传学习已被应用于许多领域，在机器人学中，Wilson 运用分类器系统进行了传感器—马达的协调研究，然后发展了 ANIMAT 系统，进行模拟人工生物在环境中自主适应的研究，Dorigo 结合遗传学习和基于行为的机器人学进行了简单的、模拟生物在变化的环境中学习趋光和避热两种行为模式的研究。

1.3.3 遗传算法的研究焦点

目前的研究焦点主要在以下几方面:

(1)算法的数学基础。包括算法的收敛性理论，早熟现象与欺骗问题，交叉算子的数学意义与统计解释，参数设置对算法的影响等。

(2)遗传算法和神经网络、模糊推理及混沌理论等其他智能计算方法相互渗透和结合。

(3)算法的改进与深化。根据实际应用不断改进与完善算法，编码策略、基因操作方法、参数选择等，而不是停留在仅应用遗传算法解决一般问题的低层次上。

(4)算法的并行化研究。遗传算法本质上具有很好的并行特性，而且效率很高，这也是遗传算法能够有效搜索的根本原因之所在。

(5)遗传算法和进化规划以及进化策略等进化理论的结合。

1.4 研究内容及论文结构

1.4.1 研究内容

虽然遗传算法在许多优化问题中都有成功的应用，但大量的研究表明，传统的遗传算法也存在许多的不足和缺陷，如早熟收敛、计算量大和收敛速度慢。遗传算法早熟问题的存在主要是种群多样性的缺失；计算量大，主要是算子本身的运算过程并不能很好地反映进化的方向，没有考虑产生的个体是否具有代表性；收敛速度慢，主要是因为算法的局部搜索能力差差，不能充分利用局部搜索空间

的信息。

针对这些问题，本文在吸取已有研究成果的基础上做了如下研究工作：

(1)利用正交试验设计方法设计用于两个父代个体之间的交叉算子，提出了一种新的自适应正交交叉算子(Self-adaptive Othogonal Crossover, SOC)。利用新算子产生的子代个体均匀分布在父代个体所确定的可行解空间，并根据所确定的空间大小调节子代个体繁殖的数量，以维持群体的多样性和减少计算开销。

(2) 为了充分利用多个父代个体之间的携带的有效信息，产生具有代表性的子代个体，利用正交试验设计方法设计了一种新的多父代正交交叉算子(Multi-parent orthogonal crossover, MOC)，并利用该算子来对群体中的多个父代个体进行交叉操作。

(3) 提出了基于种群分割和单形搜索的局部搜索策略(Local Search Schem, LSS)。将种群分割为互不相交的局部邻域，以提高单形搜索的局部收敛速度，同时对不相交的局部邻域进行并行搜索，以实现快速的全局搜索。

(4) 将 SOC 算子和 LSS 有机结合起来，提出了一种求解全局优化的混合自适应正交遗传算法(Hybrid Self-adaptive Orthogonal Genetic Algorithm,HSOGA)。HSOGA 算法在求解高维和超高维的全局优化中表现出了非常好的全局搜索能力和较高的搜索效率及搜索精度。

(5)把 MOC 算子和相应约束处理技术结合起来，提出了一种基于正交试验设计的约束优化算法（a novel constrained optimization algorithm based on orthogonal design ，NCOA/OD）。在求解大量的等式和不等式约束优化问题中，表现出了良好的搜索精度及搜索效率。

(6) 针对 P 型免疫反馈控制器不能克服动态干扰和消除静态误差的问题，利用模糊控制系统的非线性逼近能力，提出了一种将 P 型免疫反馈控制器同常规 PID 控制器进行混合联结的模糊免疫 PID 控制器的设计方法，并利用 HSOGA 算法对其控制器参数进行离线优化。

1.4.2 论文结构

在国家自然科学基金项目(项目号：60404021)“免疫计算的测不准有限计算模型与鲁棒性分析”和国家基础研究项目(项目号:A1420060159) “异质多移动体协同工作基础理论研究”的资助下，本文展开了正交遗传算法及其应用研究。本文分为六章，论文结构的具体安排如下：

第一章，绪论。总结了遗传算法的发展历史和目前国内外的研究现状。概括了本论文主要研究内容、结构安排。

第二章，遗传算法。对遗传算法进行了简单的描述和介绍了遗传算法实现方法。

第三章，一种用于求解全局优化的混合自适应正交遗传算法。首先对正交试验设计方法进行了介绍，然后分析了父代个体的相似度同它们确定空间区域的关系，并根据父代个体的相似度设计交叉算子(SOC)，接着详细地描述了基于种群分割和单形搜索的局部搜索策略。对 14 个标准的高维 Benchmark 函数的实验结果表明，该算法在鲁棒性，通用性，优化效率方面显著优于其它的全局优化算法。

第四章，一种新的基于正交试验设计的约束优化算法。在搜索策略上，首先利用正交试验设计方法设计多父体交叉算子(MOC)，然后将 MOC 算子和单形交叉算子做为重组算子用来产生新的子代个体。在约束处理技术上，引入了一个衡量个体的优、劣的新比较准则，以实现直接的比较。通过 13 个标准的测试函数验证了算法的通用性和有效性

第五章，一种改进的模糊免疫反馈 PID 控制器及优化。针对 P 型免疫反馈控制器不能克服动态干扰和消除静态误差的问题，利用模糊控制系统的非线性逼近能力，提出了一种将 P 型免疫反馈控制器同常规 PID 控制器进行混合联结的模糊免疫 PID 控制器的设计方法。该控制器通过免疫反馈控制规律和模糊控制规则在线调整控制器的参数。为了选择一组较优的控制器参数，用 HSOGA 算法在全局范围内对控制参数进行离线优化。仿真研究结果表明，该控制器较常规控制器具有更好的动、静态特性

第六章，结束语。对于本论文的研究进行了总结，并对有待进一步研究的问题进行了分析和展望。

第二章 遗传算法

基于生物界自然选择和生物遗传机理的模仿，针对不同的问题，到目前为止，已设计出了许多各种不同的编码方法来表示问题的可行解，开发出了许多不同的遗传操作算法来模拟不同情况下的生物遗传特性。因此，由不同的编码方法和不同的遗传算子就产生了各种不同的遗传算法。但这些遗传算法都有共同的特点，即通过对生物遗传和进化过程中选择、交叉、变异机理的模仿，来完成对问题最优解的自适应搜索过程。与传统搜索算法不同，遗传算法从一组随机产生的初始解，称为群体，开始搜索过程。群体中的每个个体是问题的一个解，称为染色体。通过交叉、变异、选择遗传运算，这此染色体在后续迭代中不断进化。交叉或变异运算生成下一代染色体，称为子代。染色体的优劣用适应度来衡量，根据适应度值的大小和一定的选择策略，从父代和子代中选择一定数量的个体，作为下一代群体，再继续进化，这样经过若干次迭代之后，算法收敛于最好的染色体，它很可能就是问题的最优解或次优解。遗传算法中使用适应度这个概念来度量群体中的个体在优化问题中可能到达最优解的程度。度量个体适应度的函数称为适应度函数。适应度函数的定义一般与具体求解问题有关。

2.1 遗传算法的描述

2.1.1 遗传算法的构成要素

(1)染色体的编码方法。

许多的待求解的优化问题都很复查，但可以表示为简单的位串形式的编码。遗传算法最常用的编码是二进制编码，用二进制符号串来表示群体中的个体，其等位基因是由二进制符号集{0,1}所组成的。如: $X=101110001010101$ 就可以表示一个染色体，该个体的长度 $n=15$。

二进制编码虽然简单易行，但不便于反映所求问题的结构特征，对于一些连续函数的优化问题等，也由于遗传运算的特性而使其局部搜索能力差。因此，对很多优化问题适合用其他编码方法。其他的编码方式主要有：浮点数编码、格雷码、符号编码、多参数编码等。

浮点数编码方法是指个体的每个染色体用给定区间范围内的一个浮点数来表示，个体的编码长度等于其问题变量的个数。因为这种编码方法使用的是变量的真实值，所以浮点数编码方法也叫做真值编码方法。浮点数编码方法适于求解多维、高精度要求的连续函数优化问题。

格雷码是其连续的两个整数所对应的编码值之间只有一个码位是不相同的，

其余码位都完全相同。例如十进制数 7 和 8 的格雷码分别为 0100 和 1100，而二进制编码分别为 0111 和 1000。

符号编码方法是指个体染色体编码串中的基因值取自一个无数值含义、而只有代码含义的符号集。这个符号集可以是一个字母表，如{A，B，C，D，…}；也可以是一个数字序号表，如{1，2，3，4，5，…}；还可以是一个代码表，如{x_1，x_2，x_3，x_4，x_5，…}等等。

(2)个体适应度评价

遗传算法在进化搜索中基本不用外部信息，仅用目标函数即适应度函数为依据。遗传算法的目标函数不受连续可微的束缚且定义域可以为任意集合。对目标函数的唯一的要求是，针对输入可计算出能加以比较的非负结果。这样根据不同种类的问题，必须确定好由目标函数值到个体适应度之间的转换规则。

(3)遗传算子

遗传算法遗传操作包括以下三个基本遗传算子(genetic operator)：1)选择(selection)；2)交叉(crossover)；3)变异(mutation)。这三个遗传算子有如下特点：

①这三个遗传算子的操作都是在扰动情况下进行的。换句话说，遗传操作是随机化操作，因此群体中个体向最优解迁移的规则是随机的。需要强调的是，这种随机化操作和传统的随机搜索方法是有区别的。遗传操作进行的是高效有向搜索而不是一般随机搜索方法进行的无向搜索。

②遗传操作的效果和上述三个遗传算子所使用的操作概率，编码方法，群体大小，初始群体以及适应度函数的设定是密切相关的。

③三个基本遗传算子的操作方法或操作策略随具体求解问题不同而异，与个体的编码方式直接有关。

2.1.2 遗传算法的基本框架

采用遗传算法求解优化问题的一般步骤如下：

(1) 随机生成初始种群；

(2) 计算群体中每一个体的适应度值；

(3) 若满足某种停止条件，则计算结束；

(4) 根据一定的选择策略，当前群体经选择操作后作为基因操作对象；

(5) 对所选择的解进行基因操作(交叉和变异)，生成新的种群，转到 2)；

算法的停止条件最简单有如下两种：

(1) 完成了预先给定的进化代数则停止；

(2) 群体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时停止。

遗传算法实现过程的描述见图 2-1。

Procedure GA
Begin
初始化：
{
选择遗传操作的类型，确定所有的参数值;
设置进化代数 $t=0$;
随机生成初始种群 $P(0)=\{x_1,x_2,\ldots,x_n\}$, n 为种群规模;
}
适应度评价：计算初始种群 $P(0)$ 中每个个体的适应度值;
While （终止条件不满足）do
{
交叉：对 $P(t)$ 进行交叉操作生成种群 $P'(t+1)$
变异：对 $P'(t+1)$ 进行变异操作生成种群 $P''(t+1)$
适应度评价：计算种群 $P''(t+1)$ 中每个个体的适应度值;
选择：对($Q\bigcup P''(t+1)$)进行选择操作生成新种群 $P(t+1)$,
其中 Q 代表 $P(t)$ 的某个子集或空集.
$t=t+1$;
}
end

图 2-1 遗传算法实现的基本过程

2.2 遗传算法的实现

2.2.1 个体适应度评价

遗传算法使用个体的适应值函数对解的质量进行评价，个体的适应值越高，相应解的质量越好，该个体被选择参与繁殖子代个体的概率就越大。由于标准遗传算法使用按适应值比例复制的选择策略，因此，必须将目标函数的值转换为正的适应值。设个体 x 的目标函数为 $f(x)$，适应值函数为 $F(x)$。对于最大化问题，其转换公式为

$$F(x)=f(x)+C_{\min} \tag{2-1}$$

对于最小化问题，其映射公式为

$$F(x)=C_{\max}-f(x) \tag{2-2}$$

常数 $C_{\min}(C_{\max})$ 通常取使 $F(x)>0$ 的相对较小(较大)的正数。

为了有效地控制选择压，一些研究者提出了不同的适应值变换技术。设原始的适应值函数为 $F(x)$，变换后新的适应值函数为 $F'(x)$，常用的适应值变换技术有：

线性静态变换

$$F'(x)=aF(x)+b \tag{2-3}$$

常数a,b需要满足：1) 使变换后种群的平均适应值应等于变换前的平均适应值，即$F'_{avg}=F_{avg}$；2）变换后种群中的最大适应值等于变换前平均适应值的指定倍数，即$F'_{\max}=C\cdot F_{avg}$，这里C为最佳个体的期望数，一般取$C=1.2-2$。

线性动态变换

$$F'(x)=aF(x)-\min\{F(x_i)\mid x_i\in P(t-w)\} \tag{2-4}$$

a为常数，t为进化代数，$P(t)$表示第t代种群，w称为适应值变换窗口，一般取 0 到 5 之间的整数。

对数变换

$$F'(x)=b-\log F(x) \tag{2-5}$$

常数b应满足$b>\log F(x)$。

乘幂变换

$$F'(x)=F^{\alpha}(x) \tag{2-6}$$

Michalewicz 根据对种群的统计测度动态地改变α的值，以使适应值满足一定的缩放要求。

δ截断

$$F'(x)=F(x)-(\overline{F(x)}-\delta(P(t))) \tag{2-7}$$

$\delta(P(t))$表示种群$P(t)$的标准差。该方法利用种群的平均适应值和标准差信息对适应值进行变换，目的在于有效地保证变换后的适应值不会出现负值。

目前，关于适应值变换技术和选择算子的交互作用尚无任何理论研究成果，在实际应用时，采用哪一种变换技术被认为是一种“黑色艺术”，完全取决于应用者的经验和偏爱。

2.2.2 选择算子

选择指从父代种群中挑选个体进入下一代种群的操作，选择仅与个体的适应值有关。标准遗传算法采用概率生存规则，适应值较高的个体能以较大的概率生存，并参与繁殖子代个体的变异和重组等遗传操作。选择是使进化算法实现定向搜索的唯一方法，个体之间的选择概率差即选择压强烈地影响算法的搜索性能，强选择压易导致早熟收敛，弱选择压引起缓慢的收敛速度，Goldberg[23]研究了多种选择方法的选择压对遗传算法搜索性能的影响。Potts 概括了大约 20 多种选择方法，并提出微观进化结构和人工选择等技术来改进遗传算法的早熟收敛现象。Back[24]将进化计算中最常用的选择方法归纳为：

轮盘赌选择

这是一种最基本、最常用的选择机制。在这种选择机制中，个体每次被选中的概率与其在整个种群中的相对适应度成正比，故又称适应度比例选择法。设种群规模为μ（下同），个体i的适应度值为F_i，则个体i被选择的概率p_i为：

$$p_i = \frac{F_i}{\sum_{j=1}^{\mu} F_j} \tag{2-8}$$

虽然比例选择是遗传算法的标准选择方法，但存在以下缺点：1）因选择概率正比于个体的适应值，适应值必须为正值；2)在算法运行的前期阶段，当某个体的适应值大大地高于种群的平均适应值时，易产生早熟收敛现象；3)在运行的后期阶段，当全体个体的适应值接近种群的平均适应值时，搜索过程易陷入停滞不前的状态。因此，为了提供合适的选择压，需要进行适应值变换。

排序选择

该方法首先将种群中的全体个体按其适应值大小排序，然后将事先确定的选择概率按序分配给各个体。常用的选择概率分配方法为线性排序方法，个体i的选择概率按公式(2-9)计算。

$$p_i = \frac{1}{\mu}\left(\eta^+ - \left(\eta^+ - \eta^-\right)\cdot\frac{i-1}{\mu-1}\right) \quad s.t\sum_{i=1}^{\mu} p_i = 1 \ \text{ and } p_i \geq 0 \tag{2-9}$$

其中$\eta^+(\eta^-)$最佳(差)个体的最大(小)期望数，且满足$1 \leq \eta^+ \leq 2, \eta^- = 2 - \eta^+$。通过调节最大期望数可实现选择压的控制。线性排序具有以下优点：1)无需适应值变换；2)便于控制选择压；3)可改善搜索性能，尤其是加快后期的收敛速度。

联赛选择

该方法从种群中随机地选择q个个体(称为联赛规模)，其中适应值最高的个体进入下一代种群，这一过程重复μ次，直到填满下一代种群。Backt[34]推导出联赛选择的选择概率计算公式为

$$p_i = \frac{1}{\mu^q}\left(\left(\mu - i + 1\right)^q - \left(u - i\right)^q\right) \tag{2-10}$$

改变联赛规模可调节联赛选择的选择压，联赛规模越大，选择压越高。选择适当的最大期望数和联赛规模，联赛选择完全等价于排序选择。

(μ/λ)选择和$(\mu+\lambda)$选择

这是一类在进化策略中普遍采用的选择方法。(μ/λ)选择先从μ个父代个体中随机地选择$\lambda(\lambda \geq \mu)$个个体参与重组和变异操作以生成临时种群，再从临时种群中确定性地选择前μ个适应值较高的个体进入下一代种群。$(\mu+\lambda)$选择允许μ个父代个体和λ个子代个体共同竞争，确定性地选择μ个适应值较高的个体进入下一代种群。显然，通过调节μ和λ的比值可实现选择压的控制。与其它选择方法相比，$(\mu+\lambda)$选择不但可提供较强的选择压，且能确保父代种群中的最佳个体进入下一代种群，是一种自然的精英保留(elitist reservation)选择方法。

2.2.3 交叉算子

有性繁殖是自然生物进化的普遍现象，同源染色体通过交叉实现基因重组，从而生成新的个体或物种。模拟这一自然进化现象，遗传算法使用交叉算子生成新的个体。交叉操作通过组合不同个体的遗传信息(基因)以生成可能的优良个体，是遗传算法生成新个体的主要方法。交叉算子的设计一般与所求解的问题和表示方案有关，但总的设计原则是保证父代个体的优良胜状能在子代个体中得到遗传和继承。文献中已有大量的交叉算子的设计报道，对于二进制编码和实数编码方案，Back 将进化计算中最常用的交叉算子归纳为一点交叉、多点交叉和均匀交叉，离散交叉和算术交叉等。

一点交叉

在染色体编码中随机地设定一个交叉点，将该点之前或之后的两个染色体的部分结构进行互换，并生成两个新的个体。当二进制字符串编码的长度为l位时，表示交叉点位置的随机数$r \in \{1,2,\ldots,l-1\}$。设父代个体

$$a = \{a_1,\ldots,a_{k-1},a_k,a_{k+1},a_l\}$$

$$b = \{b_1,\ldots,b_{k-1},b_k,b_{k+1},b_l\}$$

经一点交叉操作后生成的两个子代个体为

$$a' = \{b_1,\ldots,b_{k-1},a_k,a_{k+1},a_l\}$$

$$b' = \{a_1,\ldots,a_{k-1},a_k,b_{k+1},b_l\}$$

多点交叉

与一点交叉不同的是，多点交叉允许随机地生成多个交叉位置，例如，$k_1,\ldots,k_z \in \{1,2,\ldots,l-1\}$，如果 z 为奇数，增加一个增加一个交叉位置$k_z = 1$。父代个体a和b经:点交叉后生成的子代个体a'和b'分别为：

$$a_i' = \begin{cases} a_i, \forall i(k_k < i < k_{k+1}), k \le z, k\text{为奇数} \\ b_i, \text{ 其它} \end{cases}$$

$$b_i' = \begin{cases} b_i, \forall i(k_k < i < k_{k+1}), k \le z, k\text{为奇数} \\ a_i, \text{ 其它} \end{cases}$$

在实质应用中，使用较多的为两点交叉。

均匀交叉

均匀交叉实际上是多点交叉的扩展，两个父代个体的每一个基因是否交换取决于随机整数$k \in \{0,1\}$的取值。例如，对于任意基因座位置，如果$k = 1$，交换该位置上的基因，否则，保持原有基因不变。

离散重组

实数编码下的离散重组与二进制编码下的交叉操作相似，因为实数编码下的基因直接对应于目标变量的分量，所以交换的不是二进制位或子串，而是目标变量的一个或多个分量。

算术重组

在实数编码下，算术重组指子代个体的基因(目标变量的分量)由父代个体的相应位置上的基因的线性组合而形成。

多父代重组

近年来，组合多个父代个体的遗传信息已成为一种常用的重组技术，多父代重组算子在某些应用中产生了一定的性能改进。通过父代数量的扩展，Eiben 将传统的二父代交叉或重组算子推广为广义多父代交叉算子。

虽然自然进化的有性繁殖是进化算法应用重组算子的动机，但重组算子的搜索效率和应用的合理性一直是有争议的问题。重组算子的作用机制和理论分析是进化算法理论研究的最薄弱环节之一。遗传算法学派基于模式定理和积木块假设强调重组算子的优点，进化策略学派则基于遗传修复假设而应用重组算子。由于缺乏重组算子的理论成果，对于特定的目标函数，很难先验地评估应用重组算子的优点或缺点，当然也很难确定最优重组算子的设计和参数设置。

重组算子的设计与所求解的问题和表示方法有关，针对不同的问题，可以提出各种重组方法。刘勇、陈国良等介绍了面向特定问题和表示方法的不同重组算子设计实例，这些设计实例对于面向问题的重组算子设计具有一定的参考价值。

2.2.4 变异算子

在标准遗传算法中，变异算子是作为一个“背景算子”而提出的，其主要的作用是为了在种群中引入缺失的基因，以实现问题可行域空间的全局搜索。变异算子的设计应满足编码表示的数学性质。对于二进制编码表示，变异算子以很小的概率 $p_m \in [0.001, 0.01]$ 或 $p_m = 1/l$ 对每个二进制位进行逆转操作。个体 $a = (a_1, a_2, \ldots, a_l) \in IB^l$ 经变异操作后生成的子代个体 a' 的各个二进制位为

$$a_i' = \begin{cases} a_i & ,if\ k_i > p_m \\ 1 - a_i & ,if\ k_i < p_m \end{cases} \tag{2-11}$$

此处，$k_i \in [0,1]$ 表示对每个二进制位均重新采样的服从均匀分布的随机变量。对于实数编码表示，则常采用算术变异和 Gaussian 变异。位逆转变异的全局探索能力强，但较大的变异概率易使遗传算法退化为纯随机搜索算法，且对于编码中的每一个二进制位都需要附加的随机数计算成本。Gaussian 变异的局部搜索能力强，通过改变步长参数或相关变异可实现自适应变异操作。

变异概率或变异步长参数对遗传算法的搜索性能具有重要的影响，越来越多的研究表明，采用较大的初始变异概率或步长值，随着进化过程逐渐地减少变异概率或步长值的自适应变异有利于改善遗传算法的全局收敛可靠性和收敛速度。

2.3 本章小结

本章主要介绍了遗传算法的基本框架和实现方法。在本论文中，我们在遗传算法的基本框架下引进正交试验设计方法和局部搜索技术，分别提出了两种分别用于全局优化和约束优化的新算法。本文在接下来的两章将对新的算法进行详细的描述和验证其有效性。

第三章　混合自适应正交遗传算法在全局优化中的应用

在本章中，我们首先对正交试验设计方法在遗传算法中的应用进行简单的回顾，然后介绍正交试验设计的特点。最后我们提出并详细介绍一种新的基于正交试验设计方法和局部搜索技术的混合自适应正交遗传算法。

3.1 引言

全局优化问题是工程应用领域经常会遇到的一类数学规划问题，一般的全局优化问题可以描述为：

$$\underset{l\le x\le u}{\text{minimize}} \quad f(x) \tag{3-1}$$

其中 $x=(x_1,x_2,\cdots,x_N)$，$l=(l_1,l_2,\cdots,l_N)$，$u=(u_1,u_2,\cdots,u_N)$，$[l,u]$ 为 N 维的可行解空间，$f(x)$ 为目标函数。遗传算法是模仿生物遗传学和自然选择机理，通过人工方式构造的一类优化搜索算法，被广泛地用于求解全局优化问题。虽然遗传算法在许多优化问题中都有成功的应用，但大量的研究表明[3,4]，传统的遗传算法也存在许多的不足和缺陷，如早熟收敛，计算量大和局部搜索能力差。为了有效地克服以上缺点，近来不少学者将正交试验设计方法引入到遗传算法中，来处理各种函数优化问题[25-28]。Leung 和 Zhang[28]认为遗传算法的一些步骤可以从实验的角度去考虑，例如，交叉算子从选定的父代个体的附近随机产生新的子代个体，这个操作可以看作是采样实验，因此他们将正交试验设计的方法引入到遗传算法中，从而使算法的性能更加稳健。Leung 和 Wang[25]利用正交表来初始化种群和安排交叉操作提出了一种量化的正交遗传算法(OGA/Q)来求解高维单目标的数值优化问题。曾三友等[26]把正交试验方法和相应的统计优化方法相结合，提出了一种基于正交设计的多目标演化算法求解多目标优化问题。Wang 等[27]把正交设计和约束处理技术相结合提出了基于正交设计的约束优化算法来处理约束优化问题。这些方法都取得了很好的效果。

3.2 正交试验设计

在实际的实验场合中，一般实验系统有 F 个因素，而每一个因素有 Q 个水平时，则有 Q^F 个组合，若进行全面组合试验，则要做 Q^F 组实验。但是当 Q 和 F 很大时，不可能做 Q^F 组实验。正交试验设计是一种解决多因素、多水平试验问题的有效方法，它利用正交表 $L_M(Q^F)$ 安排少数次试验，就能找到最好或者较好的试验条件，因此它被广泛地用于寻优。$L_M(Q^F)$ 表示一个具有 F 个因素和 Q 个水平的正交表，其中 L 表示拉丁方，M 表示水平组合数。$L_M(Q^F)$ 有 M 行，每一行表示一个水平的组合。应用正交表 $L_M(Q^F)$，只需要选择 M 个组合去做实验，

这里 M 一般远小于 Q^F。我们以正交表 $L_9(3^4)$ 为例，如式(3-2)，对 4 因素、3 水平的问题而言，若依据正交表 $L_9(3^4)$ 来进行正交试验设计，则需做 9 次试验，但若进行全面组合试验，则需 $3^4=81$ 次试验。可见正交试验设计大大减少了试验次数，且因素和水平越大，该方法的优越性越明显。

$$L_9(3^4)=\begin{bmatrix}1&1&1&1\\1&2&2&2\\1&3&3&3\\2&1&2&3\\2&2&3&1\\2&3&1&2\\3&1&3&2\\3&2&1&3\\3&3&2&1\end{bmatrix} \tag{3-2}$$

为了方便，记 $L_M(Q^F)=[a_{i,j}]_{M\times F}$，其中第 i 个组合的第 j 个因素的水平值为 $a_{i,j}$，$a_{i,j}\in\{1,2,\cdots,Q\}$。设正交表 $[a_{i,j}]_{M\times F}$ 第 j 列为 a_j，若 $j=1,2,(Q^3-1)/(Q-1)+1,\cdots,(Q^{J-1}-1)/(Q-1)+1$，则 a_j 称为基本列，其它的列称为非基本列。文献[25,26]给出了创建正交表 $L_M(Q^F)$ 的算法，它首先创建基本的列，然后创建非基本的列，详细描述见表 3-1。其中 Q 为素数，且 $M=Q^J$，J 满足式(3-3)，因此，当 Q 给定时，

表 3-1 构建正交数组 $L_M(Q^F)$ 的伪代码

Step1: select the smallest J fulfilling $(Q^J-1)/(Q-1)\geq F$;

Step2: if $(Q^J-1)/(Q-1)=F$, then $F'=F$ else $F'=(Q^J-1)/(Q-1)$;

Step3: construct the basic columns as follows:

for $k=1$ to J **do**

$j=\dfrac{Q^{k-1}-1}{Q-1}+1;$

for $i=1$ to Q^J **do**

$a_{i,j}=\left\lfloor\dfrac{i-1}{Q^{J-1}}\right\rfloor \bmod Q;$

end for

end for

Step4: construct the non-basic columns as follows:

for $k=2$ to J **do**

$j=\dfrac{Q^{k-1}-1}{Q-1}+1;$

for $s=1$ to $j-1$ **do**

for $t=1$ to $Q-1$ **do**

$a_{j+(s-1)(q-1)+t}=(a_s\times t+a_j)\bmod Q;$

end for

end for

Step5: increment $a_{i,j}$ by one for all $1\leq i\leq M$ and $1\leq j\leq F'$;

Step6: delete the last $F'-F$ columns of $L_{Q^J}(Q^{F'})$ to get $L_M(Q^F)$ where $M=Q^J$;

正交表 $L_M(Q^F)$ 因素个数 F 越大，即正交表的维数越大，水平组合数 M 则越大。

$$F=\frac{Q^J-1}{Q-1} \tag{3-3}$$

3.3 混合自适应正交遗传算法

在群体进化的过程中，文献[25,27]使用一张固定的正交表来安排父代个体的交叉操作，且因素分割的位置是随机产生的，但当参与交叉的父代个体之间的空间距离很近即相似度很高时，若选用一张固定的正交表来安排交叉操作，则会产生很多冗余个体，使子代群体的多样性变差，这种现象在群体进化后期更为明显。而在对父代个体进行交叉操作的时候，应尽可能使新产生的子代个体均匀分布在父代个体所确定的可行解空间，并根据所确定的空间大小调节子代个体繁殖的数量，以维持群体的多样性和减少计算开销。为了尽可能实现这个目的，本章根据父代个体的相似度来自适应地选择正交表的因素个数和调整对父代个体进行因素分割的位置，提出了自适应正交交叉算子(Self-adaptive Orthogonal Crossover, 记 SOC)。本章利用 SOC 算子在整个定义域空间进行全局搜索，同时为了进一步提高算法学习和利用搜索空间局部信息的能力，提高其收敛速度，引入了局部搜索策略。因此，本章将自适应正交交叉算子和局部搜索策略结合起来提出了混合自适应正交遗传算法(Hybrid Self-adaptive Orthogonal Genetic Algorithm，记 HSOGA)。

3.3.1 自适应正交交叉算子

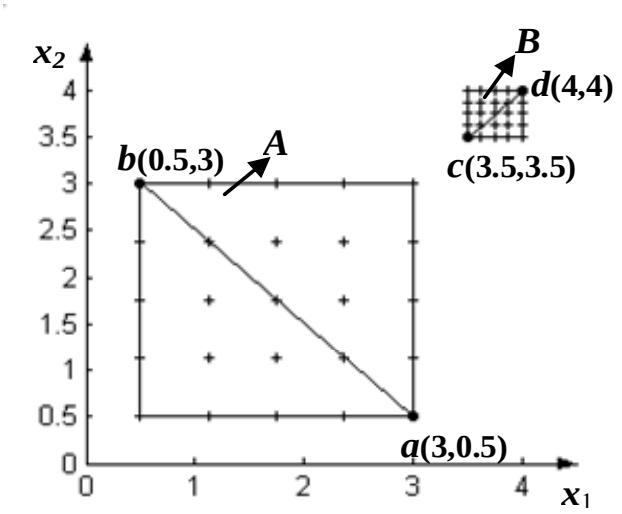

图 3-1　利用正交表安排父代个体交叉操作产生的子代个体分布图

设有两对父代个体，a 和 b，c 和 d，其中 $a=(3,0.5)$，$b=(0.5,3)$，$c=(3.5,3.5)$，$d=(4,4)$，个体 a 和 b 确定的空间区域 $[(0.5,0.5),(3,3)]$ 如图 3-1 中的区域 A 所示，c 和 d 确定的空间区域 $[(3.5,3.5),(4,4)]$ 如图 3-1 中的区域 B 所示。

若选用正交表$L_{25}(5^2)$，即$Q=5,F=2,M=25$，对这两对父代个体安排交叉操作，它们各自将产生M个后代个体，个体的分布如图 2-1 所示。由第 3.2 节的分析可知，当正交表$L_M(Q^F)$的水平数Q一定时，我们可以通过调节正交表的因素个数即F的大小，来调节水平组合数M的大小即调节子代个体的数目。从图 3-1 可以看出，个体c和d的距离近，它们所确定的空间区域B相对较小，若不减少用于安排交叉操作的正交表的因素个数，则产生过多相似的后代个体。因此，应根据父代个体所确定的区域大小来动态地选择一张合适的正交表，安排父代个体的交叉操作。

父代个体的空间距离越近，他们的相似度就越高，由这两个父代个体所确定的空间区域则越小，在对两者利用正交表安排交叉操作的时候，如果所选择的正交表的因素个数不变，不但增加函数的评价次数，而且使群体的多样性变差，这种现象在群体进化的后期更为明显。在群体进化的中后期，所有的个体不断地向适应度高的方向聚集，个体之间的空间距离越来越近。因此适当地减少相似父代个体繁殖子代个体的数目，能够减少计算量，抑制部分个体过度繁殖。

设$p_1=(p_{1,1},p_{1,2},\cdots,p_{1,N})$, $p_2=(p_{2,1},p_{2,2},\cdots,p_{2,N})$为参与交叉操作的一对父代个体，由$p_1,p_2$所确定的决策空间为$[l_{parent},u_{parent}]$。为了描述方便，作如下定义：

定义 1：令$\delta_i=\left|p_{1i}-p_{2i}\right|$，其中$i=1,2,\cdots,N$，$\delta_i\geq 0$。则定义$\delta_i$为$p_1,p_2$的第$i$维的相似度值。$p_{1i}$与$p_{2i}$数值越接近，相似度值$\delta_i$越小，则$p_1,p_2$相似度越高。

定义 2：令$\delta_i=\left|p_{1i}-p_{2i}\right|$，$\delta_0$是给定的接近 0 的正实数，若$\delta_i>\delta_0$，则$p_1,p_2$的第$i$维相似度低。

个体p_1,p_2之间的空间距离为：

$$d=\sqrt{\sum_{i=1}^{N}(p_{1i}-p_{2i})^2} \tag{3-4}$$

由定义 1 可得，$\left|p_{1i}-p_{2i}\right|=\delta_i$，当$p_{1i}$和$p_{2i}$的相似度很高时，$\delta_i$值接近于 0 即$\left|p_{1i}-p_{2i}\right|$的数值很小。因此，$p_{1i},p_{2i}$对$d$的影响很弱，即个体$p_1,p_2$的第$i$维对区域$[l_{parent},u_{parent}]$的大小影响很少，可以认为解空间$[l_{parent},u_{parent}]$的大小主要决定于$p_1,p_2$中那些相似度低的分量$j$，$j\in J$，其中，

$$J=\left\{i\,|\,\forall i\in\{1,2,\cdots,N\},\left|p_{1i}-p_{2i}\right|>\delta_0\right\} \tag{3-5}$$

由定义 2 可知，集合J的元素值为父代个体p_1,p_2相似度低的分量在个体p_1,p_2中的位置。

不妨设p_1,p_2的前t个分量相似度低，后$N-t$个分量有较高的相似度。则p_1,p_2前t分量将组成一个t维的多面体，我们将它记为Z。由前面的分析可以知道，p_1,p_2所确定空间$[l_{parent},u_{parent}]$的大小主要取决于p_1，p_2中相似度低的分量，即区域Z的体积大小。当p_1,p_2的后$N-t$个分量相似度值为 0，即后$N-t$个分量

完全相等时，区域$[l_{parent},u_{parent}]$的大小就等于区域Z的大小。所以通过对p_1,p_2相似度低的分量所确定的空间Z进行有效的搜索，来减少盲目搜索，提高了对区域$[l_{parent},u_{parent}]$的搜索能力。

例如，设参与交叉操作的两个父代个体$p_1=(2,4,5,7)$，$p_2=(1,3,5,7)$，则p_1,p_2所确定的可行解空间$[l_{parents},u_{parents}]=[(1,3,5,7),\ (2,4,5,7)]$，取$\delta_0=0.05$，则$t=2$。$p_1,p_2$前$t$维相似度低，它们确定的空间$Z=[(1,3),\ (2,4)]$。取水平数$Q=2$，构造正交表$L_M(Q^F)$，即$L_4(2^2)$。

$$L_4(2^2)=\begin{bmatrix}1&1\\1&2\\2&1\\2&2\end{bmatrix} \tag{3-6}$$

若将p_1,p_2的第 1 维和第 2 维分别作为一个因素，记为 factor_1和 factor_2, 将p_1,p_2的第 3 维和第 4 个维放入与之相邻的因素 factor_2中，这样就将父代个体分割成 2 个因素，因素分割的位置如式(3-7)中的虚线所示。然后将 factor_1和 factor_2分割成 2 个水平。这样就把对p_1,p_2交叉操作转化为 2 因素、2 水平的试验问题。最后利用正交表$L_4(2^2)$安排正交试验设计，产生子代种群P^1，如式(3-7)所示。

$$\begin{cases}p_1:(2,\ 4,\ 5,\ 7)\\ p_2:(1,\ 3,\ 5,\ 7)\end{cases}\longrightarrow\quad P^1=\begin{cases}(2,\ 4,\ 5,\ 7)\\(2,\ 3,\ 5,\ 7)\\(1,\ 4,\ 5,\ 7)\\(1,\ 3,\ 5,\ 7)\end{cases} \tag{3-7}$$

若将p_1,p_2的第 1 维和第 2 维一起作为一个因素，记为 factor_1，将p_1,p_2的第 3 维和第 4 维一起作为一个因素，记为 factor_2，因素分割的位置如式(8)中的虚线所示。然后将 factor_1和 factor_2分割成 2 个水平。这样同样把对p_1,p_2交叉操作转化为 2 因素、2 水平的试验问题。最后利用正交表$L_4(2^2)$安排正交试验设计，产生子代种群P^2，如式(3-8)所示。

$$\begin{cases}p_1:(2,\ 4,\ 5,\ 7)\\ p_2:(1,\ 3,\ 5,\ 7)\end{cases}\longrightarrow\quad P^2=\begin{cases}(2,\ 4,\ 5,\ 7)\\(2,\ 4,\ 5,\ 7)\\(1,\ 3,\ 5,\ 7)\\(1,\ 3,\ 5,\ 7)\end{cases} \tag{3-8}$$

对种群P^1中的个体而言，这些个体的前t个分量在空间$Z=[(1,3),\ (2,4)]$是均匀分布的。与种群P^2相比，种群P^1中的个体分布更均匀且群体的多样性更好。所以正交表的因素个数确定后，对父代个体进行因素分割的位置对子代种群的影响非常重要。由前面分析可知，p_1,p_2中相度低的维数是影响p_1,p_2所确定的可行解空间$[l_{parent},u_{parent}]$大小的主要因素，因此将这些维分别置于正交试验设计的不同因素中，使新产生的个体在空间Z中均匀分布，以充分利用这些维数所包含的空间信息，来对空间Z进行有效地搜索，从而提高子代种群分布的均匀性、多样

性，以提高对可行解空间$[l_{parent}, u_{parent}]$的搜索效率。

当p_1, p_2的所有相似度值大于δ_0的分量即相似度低的分量不相邻时，设相似度值大于δ_0的分量的个数为t，将每一个相似度值大于δ_0的分量作为一个因素，那么这些分量在p_1, p_2中的位置就是进行因素分割的位置，p_1, p_2中剩余的其它分量加入到与之位置相邻的因素中，因素分割的具体方法见算法 1 的 Step2，然后将每一个因素分割成Q个水平。因此，这样就将N维的两个父代个体的交叉操作转化为t因素、Q水平试验问题，然后构建正交表$L_M(Q^F)$安排正交设计试验，产生M子代个体，其中$F = t$。自适应正交交叉算子的实现步骤如算法 1 所示。

当水平个数$Q = 2$时，自适应正交交叉算子则变成多点交叉算子，对父代个体进行因素分割的位置就是进行交叉操作的位置，且当$t = 2$时，自适应交叉算子则变成单点交叉算子。在多点交叉中，其交叉组合有多种方式存在，随着交叉点的增多，组合方式的数量将会急剧增长。自适应正交交叉算子通过分析父代个体的相似度和其确定的空间区域信息，自适应地调整交叉点的个数和交叉操作的位置，交叉点的个数即为正交表的因素个数，交叉点的位置即对父代个体进行因素分割的位置，然后构造正交表，进行正交试验设计产生具有代表性的组合来生成子代个体，极大地提高了算法的搜索效率。

算法 1：自适应正交交叉 (Self-adaptive Orthogonal Crossover , SOC)

Step1：设$p_1 = (p_{1,1}, p_{1,2}, \cdots, p_{1,N})$, $p_2 = (p_{2,1}, p_{2,2}, \cdots, p_{2,N})$为参与交叉操作的两个父代个体，由$p_1$和$p_2$所确定的决策空间为$[l_{parent}, u_{parent}]$，其中，

$$\begin{cases} l_{parent} = [\min(p_{1,1}, p_{2,1}), \min(p_{1,2}, p_{2,2}), \cdots, \min(p_{1,N}, p_{2,N})] \\ u_{parent} = [\max(p_{1,1}, p_{2,1}), \max(p_{1,2}, p_{2,2}), \cdots, \max(p_{1,N}, p_{2,N})] \end{cases} \tag{3-9}$$

然后把空间$[l_{parent}, u_{parent}]$中的第i维离散化为Q水平，即：$\beta_{i,1}, \beta_{i,2}, \cdots, \beta_{i,Q}, i \in \{1, 2, \cdots, N\}$,记$\beta_i = (\beta_{i,1}, \beta_{i,2}, \cdots, \beta_{i,Q})$,其中,

$$\beta_{ij} = \begin{cases} \min(p_{1,i}, p_{2,i}), & j = 1 \\ \min(p_{1,i}, p_{2,i}) + (j-1)(\dfrac{|p_{1,j} - p_{2,j}|}{Q-1}), & 2 \le j \le Q-1 \\ \max(p_{1,i}, p_{2,i}) & j = Q \end{cases} \tag{3-10}$$

Step2：令向量$k = [k_1, k_2, \cdots, k_t]$，并且满足：$k_j \in J$且$1 \le k_1 < k_2 < \cdots < k_t \le N$，$j = 1, 2, \cdots, t$，集合$J$的定义见式(3-5),$t$为$p_1, p_2$中相似度低的分量的个数,其中$\delta_0$是给定的接近于 0 的正实数。向量$k$保存了相似度低的分量在$p_1, p_2$中的位置，即进行因素分割的位置。设$x$为$p_1, p_2$中的任一个体，把个体$x = (x_1, x_2, \cdots, x_N)$分成$t$份，其中每一份表示个体$x$的一个因素：

$$\begin{cases} f_1 = (x_1, \cdots, x_{k_1}) \\ f_2 = (x_{k_1+1}, \cdots, x_{k_2}) \\ \cdots \\ f_t = (x_{k_{t-1}+1}, \cdots, x_N) \end{cases} \tag{3-11}$$

令 $k_0 = 0$，则第 i 个因素 f_i 的 Q 个水平可以表示为：

$$\begin{cases} f_i(1) = (\beta_{k_{i-1}+1,1}, \beta_{k_{i-1}+2,1}, \cdots, \beta_{k_i,1}) \\ f_i(2) = (\beta_{k_{i-1}+1,2}, \beta_{k_{i-1}+2,2}, \cdots, \beta_{k_i,2}) \\ \cdots \\ f_i(Q) = (\beta_{k_{i-1}+1,Q}, \beta_{k_{i-1}+2,Q}, \cdots, \beta_{k_i,Q}) \end{cases} \tag{3-12}$$

Step3: 根据表 1 构造正交表 $L_M(Q^F) = [b_{i,j}]_{M \times F}$，其中 $F = t$。利用正交 $L_M(Q^F)$ 来对式(3-11)中确定的 t 个因素和式(3-12)中确定每一个因素所对应的 Q 个水平进行正交试验设计，将产生 M 个子代个体：

$$\begin{cases} (f_1(b_{1,1}), f_2(b_{1,2}), \cdots, f_F(b_{1,F}) \\ (f_1(b_{2,1}), f_2(b_{2,2}), \cdots, f_F(b_{2,F}) \\ \cdots \\ (f_1(b_{M,1}), f_2(b_{M,2}), \cdots, f_F(b_{M,F}) \end{cases} \tag{3-13}$$

我们以个体 $p_1 = (2,1,6,4,2,2)$，$p_1 = (0,3,8,4,2,2)$ 为例，取水平个数 $Q = 3$。执行算法 1，对个体 p_1, p_2 进行自适应正交交叉操作，其详细过程如下：

Step1：根据式(3-9)计算 p_1, p_2 所确定解空间 $[l_{parent}, u_{parent}] = [(0,1,6,4,2,2),(2,3,,8,4,2,2)]$，然后把空间 $[l_{parent}, u_{parent}]$ 的第 i 维离散化为 β_i：

$$\begin{cases} \beta_1 = (0,1,2) \\ \beta_2 = (1,2,3) \\ \beta_3 = (6,7,8) \\ \beta_4 = (4,4,4) \\ \beta_5 = (2,2,2) \\ \beta_6 = (2,2,2) \end{cases} \tag{3-14}$$

Step2: 取 $\delta_0 = 0.05$，由公式(5)得集合 $J = \{1,2,3\}$，$t = 3$，$F = 3$ 则 $k = [1,2,3]$。根据公式(3-11), 将 $x = (x_1, x_2, x_3, x_4, x_5, x_6)$ 分割成 3 个因素：$f_1 = (x_1), f_2 = (x_2), f_3 = (x_3, x_4, x_5, x_6)$。

Step3: 构造正交表 $L_9(3^3)$，进行正交试验，产生 9 个子代个体：

$$\begin{cases}(0,\,1,\,6,\,4,\,2,\,2)\\(0,\,2,\,7,\,4,\,2,\,2)\\(0,\,3,\,8,\,4,\,2,\,2)\\(1,\,1,\,7,\,4,\,2,\,2)\\(1,\,2,\,8,\,4,\,2,\,2)\\(1,\,3,\,6,\,4,\,2,\,2)\\(2,\,1,\,8,\,4,\,2,\,2)\\(2,\,2,\,6,\,4,\,2,\,2)\\(2,\,3,\,7,\,4,\,2,\,2)\end{cases} \tag{3-15}$$

3.3.2 种群初始化

在对一些高维多模态的函数进行全局优化时，函数本身具有多个极点，而函数全局最优点的位置是未知的，因此在群体初始化过程中，应尽可能使初始种群均匀地覆盖整个可行域。正交试验设计是研究多因素、多水平的一种数学试验方法，它是根据正交表，从全面试验中挑选出部分具有代表性的点进行试验，这些点具备了“均匀分散，齐整可比”的特点，所以本章选择采用正交试验法生成初始种群。初始种群分布的均匀性，保证了初始群体的多样性和较丰富的模式，从而使算法能在全局范围内以较快的速度收敛。采用正交试验方法初始化种群时，当可行解空间$[l,u]$较大的时候，为了提高搜索效率和精度，将可行解空间$[l,u]$分割成S个子空间，分割方法见本节算法2，然后构造正交表$L_M(Q_0^F)$，根据正交表$L_M(Q_0^F)$对每一个子空间利用SOC算子进行交叉操作，生成初始种群。

算法 2：子空间分割

Step1：选择可行解空间的第s维，s满足下式：

$$u_s - l_s = \max_{1\le i\le N}\{u_i - l_i\} \tag{3-16}$$

Step2：将可行解空间$[l,u]$在第s维处分割成S个子空间$[l(1),u(1)],[l(2),u(2)],\cdots,[l(S),u(S)]$。

$$\begin{cases} l(i) = l + (i-1)(\dfrac{u_s - l_s}{S})l_s \\ u(i) = u - (S-1)(\dfrac{u_s - l_s}{S})l_s \end{cases} \quad i = 1,2,\cdots,S \tag{3-17}$$

3.3.3 聚类局部搜索策略

为了增强算法学习和利用搜索空间局部信息的能力，提高其收敛速度，本章还引入了局部搜索策略。先对群体进行聚类，将种群分割为互不相交的局部邻域，使每个子种群仅覆盖搜索空间一个较小的邻域。对每个子种群，即局部邻域，利用单形交叉算子(SPX)[29,30]进行局部搜索。将种群分割为互不相交的局部邻域，以提高单形搜索的局部收敛速度，同时对不相交的局部邻域进行并行搜索，以实

现快速的全局搜索。种群分割方法使该算法具有搜索空间结构自学习能力，能充分地利用搜索空间的局部信息。

设 N 维最优问题的可行解空间为 $[l,u]$，种群 P 的规模为 n，Φ 表示空集，o 表示可行解空间中随机选择的一个参考点，种群 P 经局部搜索后生成新的种群 P'，局部搜索的执行过程如下。首先，将 $P=\{p_1,p_2,\cdots,p_n\}$ 中的 n 个点，按照相似个体聚类的规则分解成若干个子种群，每个子种群由 m 个个体组成，本章 m 取3。接着，利用SPX[7,8]算子对每一个子种群中的 m 个个体(即父代群体,记为 S)进行交叉操作，产生新的 g 子代个体(即子代群体,记为 S')，本章 g 取10，最后把每一个子代种群加入到群体 P' 中。其算法描述见算法3，其中 $\lfloor\bullet\rfloor$ 表示取整。

算法 3：局部搜索

$o=(x_1,x_2,\cdots,x_N),k=1,P'=\Phi;$

while($k<\lfloor n/m\rfloor$)

$S=\Phi,S'=\Phi;$

确定群体 P 中离参考点 o 最近的一点 p_j, $j\in\{1,2,\cdots,n_k\}$;

$S=\{p_j\}\cup\{P$ 中离 p_j 最近的 $m-1$ 个点$\}$;

$P=P-S$,此时 $P=\{p_1,p_2,\cdots,p_{n_k}\}$;

$S'=\mathrm{SPX}(S)$;

$P'=P'\cup S'$;

$k=k+1$;

End

3.3.4 算法步骤

Step1: 种群初始化

将可行解空间 $[l,u]$ 分割成 S 个子空间，分割方法见算法 2。取正交表的水平个数为 Q_0，利用 SOC 算子对每一个子空间进行交叉操作，产生新的群体 P，计算其适应度值，从群体 P 中 选择适应度值最好的 n 个个体生成初始种群 P_0。

Step2: 生成临时种群 P'_{gen}

设 gen 为进化代数，群体 P_{gen} 中的每一个个体，以概率 p_c 被选择进入临时群体 P'_{gen}。若群体 P'_{gen} 中个体的数目为奇数时，再从 P_{gen} 中随机选择一个个体加入到种群 P'_{gen} 中。

Step3: 交叉操作

对群体 P'_{gen} 的个体进行随机配对，每一对个体经过 SOC 交叉操作后，产生新的后代个体，从新产生的个体中选择一个适应度值最好的个体加入到群体 C_{gen} 中。其中，取正交表的水平个数为 Q。

Step4：局部搜索

种群 P'_{gen} 经局部搜索后，生成新的种群 L_{gen}。局部搜索的方法见算法 3。

Step5：变异操作

种群 P'_{gen} 的任一个体 $p_i=(p_{i,1},p_{i,2},\cdots,p_{i,N}),i\in\{1,2,\ldots,n\}$，以概率 p_m 参与变异操作。具体操作是：①产生一个随机整数 $j\in[1,N]$，和一个随机数实数 $r\in[0,1]$；②令 $p_{i,j}=l_j+r(u_j-l_j)$。群体 P_{gen} 经变异后生成的新种群记 G_{gen}

Step6:选择操作

为了保持群体的多样性，从种群（$P_{gen}+C_{gen}+L_{gen}+G_{gen}$）中选择适应度值最好的 $\lfloor n*70\%\rfloor$ 个个体进入下一代种群 P_{gen+1}，再从种群（$P_{gen}+C_{gen}+L_{gen}+G_{gen}$）剩余的个体中，随机选择 $n-\lfloor n*70\%\rfloor$ 个个体进入到下一代种群 P_{gen+1}。

Step7: 终止条件判断：若达到规定的代数或得到满意的结果，则结束并输出结果，否则转Step2。

3.4 实验结果与分析

为了测试本章提出的混合自适应正交遗传算法(HSOGA)的性能，我们选取了 14 个高维的 Benchmark 函数作为测试标准，并与其它 2 种已有的比较理想的算法进行了实验结果的比较。对函数 f_1-f_6 及函数 $f_{10}-f_{14}$，N 取 30 维，对函数 f_7-f_9，N 取 100 维函数。函数 f_1-f_8 属多峰函数，每一个函数拥有多个局部极值点，其中函数 f_7 拥有100!=9.33×10^{157} 个局部极值点，搜索过程容易陷入局部最优，这些函数最能检验算法的多峰搜索能力。

$$f_1=\sum_{i=1}^{N}(-x_i\sin(\sqrt{|x_i|})),\quad -500\le x_i\le 500$$

$$f_2=\sum_{i=1}^{N}(x_i^2-10\cos(2\pi x_i)+10),\quad 5.12\le x_i\le 5.12$$

$$f_3=-20\exp\left(-0.2\sqrt{\frac{1}{N}\sum_{i=1}^{N}x_i^2}\right)-\exp\left(\frac{1}{N}\sum_{i=1}^{N}\cos(2\pi x_i)\right)+20+\exp(1),\quad -32\le x_i\le 32$$

$$f_4=\frac{1}{4000}\sum_{i=1}^{N}x_i^2-\prod_{i=1}^{N}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1,\quad -600\le x_i\le 600$$

$$f_5=\frac{\pi}{N}\left\{10\sin^2(\pi y_1)+\sum_{i=1}^{N-1}(y_i-1)^2\left[1+10\sin^2(\pi y_{i+1})\right]+(y_N-1)^2\right\}+\sum_{i=1}^{N}u(x_i,10,100,4),$$
$$-5.12\le x_i\le 5.12$$

其中，$y_i=1+\frac{1}{4}(x_i+1)$，$u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, & x_i>a.\\ 0, & -a\le x_i\le a\\ k(-x_i-a)^m, & x_i<a\end{cases}$

$$f_6=\frac{1}{10}\left\{\sin^2(3\pi x_1)+\sum_{i=1}^{N-1}(x_i-1)^2\left[1+\sin^2(3\pi x_{i+1})\right]+(x_N-1)^2\left[1+\sin^2(2\pi x_N)\right]\right\}$$
$$+\sum_{i=1}^{N}u(x_i,5,100,4),\quad -50\le x_i\le 50$$

$$f_7=-\sum_{i=1}^{N}\sin(x_i)\sin^{20}\left(\frac{i\times x_i^2}{\pi}\right),\quad 0\le x_i\le\pi$$

$$f_8=\frac{1}{N}\sum_{i=1}^{N}\left(x_i^4-16x_i^2+5x_i\right),\quad -5\le x_i\le 5$$

$$f_9=\sum_{j=1}^{N-1}\left[100\left(x_j^2-x_{j+1}\right)^2+\left(x_j-1\right)^2\right],\quad -5\le x_i\le 10$$

$$f_{10}=\sum_{j=1}^{N}x_i^2,\quad -100\le x_i\le 100$$

$$f_{11}=\sum_{i=1}^{N}x_i^4+\text{random}[0,1),\quad -1.28\le x_i\le 1.28$$

$$f_{12}=\sum_{i=1}^{N}|x_i|+\prod_{i=1}^{N}|x_i|,\quad -10\le x_i\le 10$$

$$f_{13}=\sum_{i=1}^{N}\left(\sum_{j=1}^{i}x_j\right)^2,\quad -100\le x_i\le 100$$

$$f_{14}=\max\left\{|x_i|,\quad i=1,2,\cdots,N\right\},\quad -100\le x_i\le 100$$

数值实验在 MATLAB 中完成，对所有的测试函数，种群的规模 n 取 200，$Q_0=N-1$，$Q=2$，$S=5$，P_c=0.6，P_m=0.1，δ_0=0.05，当 HSOGA 算法找到最优解或运行 $gen=120$ 代，则终止运行。对于每一个问题，我们在相同的条件下独立运行 50 次，记录其平均函数评价次数(M-num-fun)，最优平均值(M-best)和标准差(St.dev)。为了对比实验结果，我们将本章算法(HSOGA)与经典算法—OGA/Q[25]和较新算法—LEA[31]进行了比较。从表 3-2 可知，新算法(HSOGA)对于函数 f_5-f_9 找到了接近全局最优的解，对其它函都找到了全局最优解。对函数 f_7，函数评价 1500,000 次，我们找到的全局最优解 x^* 的函数值为 $f_7(x^*)$=-99.618，x^*的值见表 3，而文献[25,31-33]公布的全局最优解的函数值为-99.278。表 3-2 列出了实验中 3 种算法在 14 个测试函数上的函数平均计算次数，最优平均值及标准方差比较的结果。与 OGA/Q 算法比，HSOGA 算法在 $f_1, f_3-f_5, f_8, f_9, f_{11}$ 函数的最优平均值、平均函数评价次数、标准差方面明显优于 OGA/Q 算法，对函数 $f_2, f_{10}, f_{12}-f_{14}$，HSOGA 算法和 OGA/Q 算法都同时找到了最优解。对函数 f_6, f_7, HSOGA 算法得到的标准差大于 OGA/Q 算法，但找到的最优解显著优于 OGA/Q 算法。与 LEA 算法相比，HSOGA 算法在函数 $f_1-f_3, f_5-f_6, f_8-f_{14}$ 的最优平均值、平均函数评价次数、标准差方面明显优于 LEA 算法。对 f_7 函数，虽然 HSOGA 算法得到的标准差大于 LEA 算法，但找到

表 3-2　3 个算法(OGA/Q[25],LEA[31], HSOGA)对 14 个测试函数的实验结果比较

Function/ optimal	status	Algorithm		
		OGA/Q	LEA	HSOGA
f_1/ -12569.5	M-num-fun	302,116	287,365	101,151
	M-best	-12569.4537	-12569.4542	-12569.4866
	St.dev	6.447×10^{-4}	4.831×10^{-4}	3.168×10^{-5}
f_2/ 0	M-num-fun	224,710	223,803	8,420
	M-best	0	2.103×10^{-18}	0
	St.dev	0	3.0359×10^{-18}	0
f_3/ 0	M-num-fun	105,926	112,421	8,420
	M-best	3.274×10^{-16}	4.440×10^{-16}	0
	St.dev	3.001×10^{-17}	3.989×10^{-17}	0
f_4/ 0	M-num-fun	130,498	134,000	8,420
	M-best	6.104×10^{-16}	0	0
	St.dev	2.513×10^{-17}	0	0
f_5/ 0	M-num-fun	134,556	132,642	98,745
	M-best	6.019×10^{-6}	2.482×10^{-6}	2.0808×10^{-11}
	St.dev	1.159×10^{-6}	2.276×10^{-6}	1.2544×10^{-10}
f_6/ 0	M-num-fun	134,143	130,213	105,518
	M-best	1.869×10^{-4}	1.734×10^{-4}	4.1316×10^{-5}
	St.dev	2.615×10^{-5}	1.205×10^{-4}	3.0987×10^{-5}
f_7/ -99.619	M-num-fun	302,773	289,863	236,867
	M-best	-92.83	-93.01	-98.0987
	St.dev	0.02626	0.02314	0.27125
f_8/ -78.33236	M-num-fun	245,930	243,895	161,147
	M-best	-78.3000296	-78.310	-78.332331
	St.dev	6.288×10^{-3}	6.127×10^{-3}	2.356×10^{-7}
f_9/ 0	M-num-fun	167,863	168,910	167,374
	M-best	0.7520	0.5609	5.941×10^{-5}
	St.dev	0.1140	0.1078	4.0216×10^{-4}
f_{10}/ 0	M-num-fun	112,559	110,674	8,240
	M-best	0	4.727×10^{-16}	0
	St.dev	0	6.218×10^{-17}	0
f_{11}/ 0	M-num-fun	112,652	111,093	8,240
	M-best	6.301×10^{-3}	5.136×10^{-3}	0
	St.dev	4.069×10^{-4}	4.432×10^{-4}	0
f_{12}/ 0	M-num-fun	112,612	110,031	8,240
	M-best	0	4.247×10^{-19}	0
	St.dev	0	4.236×10^{-19}	0
f_{13}/ 0	M-num-fun	112,576	110,604	8,240
	M-best	0	6.783×10^{-18}	0
	St.dev	0	5.429×10^{-18}	0
f_{14}/ 0	M-num-fun	112,893	111,105	8,240
	M-best	0	2.683×10^{-16}	0
	St.dev	0	6.257×10^{-17}	0

表 **3-3** HSOGA 算法找到函数 f_7 的最优解 x^*

x^* =(2.202905547653280,1.570796214486428,1.284991573407600,1.923058388896578,1.720469806895425, 1.570796348243437,1.454414249822540,1.756086529133708,1.655717410008324,1.570796326414277, 1.497728801123797,1.696616301332517,1.630075723629949,1.570796327164349,1.517546144807838, 1.666065026898027,1.616328712211932,1.570796788345188,1.528907225843681,1.647456357417006, 1.607757297131768,1.570796326390791,1.536272526517082,1.634931506943574,1.601901830669802, 1.570796270305114,1.541435141842966,1.625925976137474,1.597647948096007,1.570796299378235, 1.545254595484606,1.619138222286553,1.594417596438591,1.570796379934915,1.548194567190928, 1.526546902649965,1.591881034428863,1.570796313343089,1.550527822715220,1.609586107340208, 1.589836453853538,1.570796327219784,1.552424882718390,1.606098595613659,1.588153300004406, 1.570796326128234,1.553996272464775,1.537723975503683,1.586743574514364,1.570796413824375, 1.555320400458290,1.540293110285201,1.585545197006467,1.570796327374178,1.556451164688094, 1.598599762777015,1.584515257093225,1.570795754967596,1.557427799413277,1.596761313060471, 1.583619196955086,1.570796326921408,1.558279959922572,1.546058142407977,1.534119410225983, 1.570796305000000,1.559030026953491,1.593727644730032,1.536269493892730,1.570795943255299, 1.603638560303743,1.592463298428713,1.538181406064810,1.570796326907612,1.560289298990284, 1.591328444080860,1.580962812901610,1.570796259054305,1.560822996552620,1.590309524812736, 1.580462784142368,1.570794483397516,1.561305093133228,1.589386150344377,1.580009236001596, 1.570796317098385,1.561742730786400,1.588545120286697,1.579595952974376,1.570796319851633, 1.562141789372054,1.587778362586528,1.579218838519324,1.570796322547901,1.562507164309846, 1.587073730676870,1.578872386011957,1.570796365608036,1.562842930466590,1.586429358220735), f_7 (x^*)=-99.618006161436

的最优解显著优于 LEA 算法，对函数 f_4，HSOGA 算法和 LEA 算法都同时找到了最优解。

由上述的实验结果可以看出，将本章提出来的自适应正交交叉算子和局部搜索策略有机结合起来而形成的混合自适应正交遗传算法(HSOGA)，在求解复杂的高维函数优化中，显示出了良好的性能。

3.5 本章小结

在本章中，我们提出了一种基于正交试验设计和局部搜索技术的混合自适应正交遗传算法，并对其进行了详细的介绍。新算法的主要特点根据父代个体的相似度值来判断父代个体所确定空间大小，并根据空间大小自动调整用于安排交叉操作的正交的因素个数和因素分割的位置，产生适量的具有代表性的个体。同时把群体分割成许多局部邻域，用 SPX 算子对每一个局部邻域进行并行搜索，从而实现有效的全局搜索。

第四章 基于正交实验设计的约束优化算法

在本章中，我们首先对实验设计方法在进化算法中的应用及一些常用的约束处理技术进行简单地介绍，然后利用正交试验设计方法设计多父体交叉算子，并对其进行详细地描述，接着提出引入了一个衡量个体的优、劣的新比较准则，以实现直接的比较，最后通过 13 个标准的测试函数对算法的性能进行了验证。

4.1 引言

在科学、工程和商业等诸多领域中，很多实际问题通过数学建模后都可以转化为约束优化问题(Constrained Optimization Problems, COPs)。遗传算法(Genetic Algorithm, GA)是一种借鉴生物界自然选择和自然遗传机制的随机搜索算法，是解决约束优化问题的一种有效方法，目前已被广泛地用于求解约束优化问题。但大量的研究表明，传统的遗传算法也存在许多的不足和缺陷[3,4]，如早熟收敛，计算量大和局部搜索能力差等。近来不少的研究者将优化试验设计方法(正交试验设计、均匀设计、佳点集)引入到遗传算法中，一定程度上有效克服了传统遗传算法的上述缺点，表现了良好的搜索性能。Leung 和 Wang[25]利用正交表来初始化种群和安排两个父代个体的交叉操作，提出了一种量化的正交遗传算法(OGA/Q)，来求解高维的全局优化问题。Wang 和 Dang[34]利用均匀表来设计用于两个父代个体的交叉算子来产生子代个体，提出了一种基于水平集进化和拉丁方的进化算法（LEA）。张铃和张钹[35]利用数论中的佳点集理论和方法，对 GA 中的交叉算子进行了重新设计，提出了佳点集遗传算法，该算法在求解全局优化问题、SAT 问题、TSP 问题和背包问题取得了满意的效果。在用 GA 求解约束优化问题时，除了算法本身的搜索能力以外，如何处理约束条件也是得到好的优化结果的关键。目前常见的约束处理技术有：惩罚函数法[36-38]，多目标优化法[39-41]以及其它方法。惩罚函数法是进化算法处理约束优化问题的常用方法，它允许群体中的个体在一定程度上违反约束条件，但必须对该个体依其违反约束条件的程度进行惩罚以减小它被选择的概率，但是惩罚参数的选取比较困难，而且算法的性能强烈地依赖于所选择的参数；多目标优化法的主要思想是将约束优化问题转换为多目标优化问题后，利用多目标技术来处理约束优化问题，目前取得了很好的实验效果，但缺点是需要计算 Pareto 前沿，在一定程度上，增加了算法的计算的开销。

考虑到算法的搜索能力和约束处理技术对处理约束优化问题的性能影响，本章在搜索策略上，为了充分利用多个父代个体之间所携带的有效信息，产生具有

代表性的子代个体，利用正交试验设计方法设计了一种新的多父体正交交叉算子(Multi-parent Orthogonal Crossover, MOC)，并利用该算子来对群体中的父代个体进行交叉操作。与此同时，我们利用单形交叉算子(Simplex Crossover, SPX)对父代种群进行并行搜索，来协调算法的勘探和开采能力。在约束处理技术上，本章将整个群体的进化过程分为三个阶段，并根据群体在三个不同阶段的特点采取不同的选择策略，来有效引导群体不断地向最优解逼近。数值实验验证了算法的通用性和有效性。

4.2 约束优化问题的描述

不失一般性，一般的约束优化问题(Constrained Optimization Problems, COPs)可以描述为：

$$\begin{aligned} &\text{Minimize} \quad f(\vec{x}) \qquad \vec{x}=(x_1,x_2,\cdots,x_N)\in\Re^N \\ &\text{Subject to} \quad g_j(\vec{x})\le 0, \quad j=1,\cdots,l \\ &\qquad\qquad\quad h_j(\vec{x})=0, \quad j=l+1,\cdots,m \end{aligned} \tag{4-1}$$

这里，$\vec{x}\in\Omega\subseteq S$ 为决策向量，Ω 为可行域，S 为决策空间。一般地，S 为 $\Re^N$ 中的 N 维长方体：$l(i)\le x_i\le u(i)$，$l(i)$，$u(i)$ 为常数，$i=1,\cdots,N$。$f(\vec{x})$，$g_j(\vec{x})$，$h_j(\vec{x})$ 均为 $\Re^N$ 上的 N 元函数，$f(\vec{x})$ 为目标函数，$g_j(\vec{x})\le 0$ 为第 j 个不等式约束条件，$h_j(\vec{x})=0$ 为第 j 个等式约束条件，l 表示不等式约束条件的个数，$m-l$ 表示等式约束条件的个数。

4.3 基于正交试验设计的约束优化算法

4.3.1 多父体正交交叉算子

通过对多个父代个体进行重组，来交换多个父代个体的基因信息，以引入更多的启发式信息来对解空间进行有效地搜索，增加子代个体的采样范围，使种群在进化的过程中保持好的多样性，从而避免早熟收敛。目前在实数编码遗传算法中，广泛使用的交叉算子有单形交叉，模拟二进制交叉，单峰正态分布交叉等。但是这些算子在子代个体的样本空间中，随机产生新的子代个，搜索具有一定盲目性，在一定程度上降低了算法的搜索效率。因此，本章利用正交实验设计方法

表 4-1 p_1,p_2,p_3 的因素和水平表

因素 / 水平	f_1	f_2	f_3
$\beta_1(\vec{p}_1)$	p_{11} p_{12}	p_{13} p_{14}	p_{15} p_{16}
$\beta_2(\vec{p}_2)$	p_{21} p_{22}	p_{23} p_{24}	p_{25} p_{26}
$\beta_3(\vec{p}_3)$	p_{31} p_{32}	p_{33} p_{34}	p_{35} p_{36}

在子代个体的样本空间中产生具有代表性的组合来生成子代个体，以提高搜索效

率。正交试验设计方法的描述和性质见 3.2 节。

例如，设父代个体 $\vec{p}_1=(p_{11},p_{12},p_{13},p_{14},p_{15},p_{16})$, $\vec{p}_2=(p_{21},p_{22},p_{23},p_{24},p_{25},p_{26})$，$\vec{p}_3=(p_{31},p_{32},p_{33},p_{34},p_{35},p_{46})$，对 p_1,p_2,p_3 进行重组，将有 3^6 种可能的组合，即子代个体的样本空间大小为 3^6。我们利用正交试验设计的方法，从子代个体的样本空间中产生具有代表性的组合来生成子代个体，从而提高算法的搜索效率。对 $\vec{p}_1,\vec{p}_2,\vec{p}_3$ 而言，利用正交试验设计安排交叉操作的具体步骤是：把参与交叉操作的父代个体 $\vec{p}_1,\vec{p}_2,\vec{p}_3$，分别作为正交试验的一个水平，分别记为 $\vec{\beta}_1,\vec{\beta}_2,\vec{\beta}_3$。然后将每一个个体分成 F 个片段，例如把 $\vec{p}_1,\vec{p}_2,\vec{p}_3$ 分成 3 个片段，分片的位置如表 4-1 所示，将每一片段作为正交试验设计的一个因素，分别记为 f_1,f_2,f_3。这样就将对 $\vec{p}_1,\vec{p}_2,\vec{p}_3$ 的交叉操作转化为 3 因素、3 水平的试验问题。例如，因素 f_1 对应的 3 个水平分别为：$f_1(1)=p_{11}\ p_{12}, f_1(2)=p_{21}\ p_{22}, f_1(3)=p_{31}\ p_{32}$。最后构造正交表 $L_9(3^3)$ 来安排正交实验设计，如表 4-2 所示，将产生 9 个子代个体。

表 4-2　对 $\vec{p}_1,\vec{p}_2,\vec{p}_3$ 进行正交试验设计的方案和结果

试验号 \ 因素 列号	f_1	f_2	f_2	子代个体
	1	2	3	
1	$1(p_{11}\ p_{12})$	$1(p_{13}\ p_{14})$	$1(p_{15}\ p_{16})$	$p_{11}\ p_{12}\ p_{13}\ p_{14}\ p_{15}\ p_{16}$
2	$1(p_{11}\ p_{12})$	$2(p_{23}\ p_{24})$	$2(p_{25}\ p_{26})$	$p_{11}\ p_{12}\ p_{23}\ p_{24}\ p_{25}\ p_{26}$
3	$1(p_{11}\ p_{12})$	$3(p_{33}\ p_{34})$	$3(p_{35}\ p_{36})$	$p_{11}\ p_{12}\ p_{33}\ p_{34}\ p_{35}\ p_{36}$
4	$2(p_{21}\ p_{22})$	$1(p_{13}\ p_{14})$	$2(p_{25}\ p_{26})$	$p_{21}\ p_{22}\ p_{13}\ p_{14}\ p_{25}\ p_{26}$
5	$2(p_{21}\ p_{22})$	$2(p_{23}\ p_{24})$	$3(p_{35}\ p_{36})$	$p_{21}\ p_{22}\ p_{23}\ p_{24}\ p_{35}\ p_{36}$
6	$2(p_{21}\ p_{22})$	$3(p_{33}\ p_{34})$	$1(p_{15}\ p_{16})$	$p_{21}\ p_{22}\ p_{33}\ p_{34}\ p_{15}\ p_{16}$
7	$3(p_{31}\ p_{32})$	$1(p_{13}\ p_{14})$	$3(p_{35}\ p_{36})$	$p_{31}\ p_{32}\ p_{13}\ p_{14}\ p_{35}\ p_{36}$
8	$3(p_{31}\ p_{32})$	$2(p_{23}\ p_{24})$	$1(p_{15}\ p_{16})$	$p_{31}\ p_{32}p_{23}\ p_{24}\ p_{15}\ p_{16}$
9	$3(p_{31}\ p_{32})$	$3(p_{33}\ p_{34})$	$2(p_{25}\ p_{26})$	$p_{31}\ p_{32}\ p_{33}\ p_{34}\ p_{25}\ p_{26}$

在 N 维的实数空间中，设参与重组的 Q 个父代个体为 $\vec{p}_1,\vec{p}_2,\ldots,\vec{p}_Q$，其中 $\vec{p}_i=(p_{i1},p_{i2},\ldots,p_{iN})$，且 $i\in\{1,\ldots,Q\}$。将每一个参与重组的父代个体看作是正交试验设计的一个水平，即 Q 个水平；然后将每一个父代个体分成 F 个片段，每一个片段作为正交试验设计的一个因素，即 F 个因素。这样就将对 Q 个父代个体的重组问题转化为 Q 水平、F 因素的试验设计问题。然后构造正交表 $L_M(Q^F)$，安排正交试验设计将产生 M 个子代个体，其具体的操作过程见算法 1。

算法 1 多父体正交交叉

Step1: 将参与重组的 Q 个父代个体分别作为正交试验设计的一个水平，记第 i 个水平为 $\vec{\beta}_i$，$i\in\{1,\ldots,Q\}$，则 $\vec{\beta}_i=\vec{p}_i$。

Step2: 设 $\vec{x}=(x_1,x_2,\ldots,x_N)$ 为参与重组的任一，将个体 $\vec{x}$ 分成 F 个片段，将每一个片段作为正交实验设计的一个因素，具体的分片操作方法如下。首先随机

产生 F-1 个整数，即 $k_1,k_2,\ldots, k_{F-1}$，且满足 $1<k_1<k_2<\ldots<k_{F-1}<N$；然后把个体 $\vec{x}=(x_1,x_2,\ldots,x_N)$ 分成 F 个片段，其中每一个片段表示个体 x 的一个因素：

$$\begin{cases} f_1=(x_1,\cdots,x_{k_1}) \\ f_2=(x_{k_1+1},\cdots,x_{k_2}) \\ \cdots \\ f_F=(x_{k_{F-1}},\cdots,x_N) \end{cases} \tag{4-2}$$

令 $k_0=0$， $k_F=N$，因此第 i 个因素 f_i 的 Q 个水平可以表示为：

$$\begin{cases} f_j(1)=(\beta_{1,k_{j-1}+1},\beta_{1,k_{j-1}+2},\cdots,\beta_{1,k_j}) \\ f_j(2)=(\beta_{2,k_{j-1}+1},\beta_{2,k_{j-1}+2},\cdots,\beta_{2,k_j}) \\ \cdots \\ f_j(Q)=(\beta_{Q,k_{j-1}+1},\beta_{Q,k_{j-1}+2}\cdots,\beta_{Q,k_j}) \end{cases} \tag{4-3}$$

Step3：构造正交表 $L_M(Q^F)=[a_{i,j}]_{M\times F}$，根据正交表 $L_M(Q^F)$，安排实验，产生 M 个子代个体：

$$\begin{cases} (f_1(a_{1,1}),f_2(a_{1,2}),\cdots,f_F(a_{1,F})) \\ (f_1(a_{2,1}),f_2(a_{2,2}),\cdots,f_F(a_{2,F})) \\ \cdots \\ (f_1(a_{M,1}),f_2(a_{M,2}),\cdots,f_F(a_{M,F})) \end{cases} \tag{4-4}$$

4.3.2 单形交叉算子

单形交叉算子(Simplex Crossover, SPX)基于均匀分布来产生后代个体且单形交叉算子运算前后，群体中个体向量的均值应保持不变。m 个独立的父体向量 $(\vec{x}_i,i=1,\ldots,m)$ 行成了一个单形，子代个体的产生按照以下步骤进行:1)按一定的比率将单形向各个方向 $(\vec{x}_i-\vec{o}$,这里 $\vec{o}$ 是 m 个向量的中心即 $\vec{o}=\frac{1}{m}\sum_{i=1}^{m}\vec{x}_i$，$\varepsilon$ 为扩张比)进行扩张得到一个新的单形,2)从新的单形中随机选择一个点作为后代个体。例如,在两维空间中由三个点 $\vec{x}_1$, $\vec{x}_2$ 和 $\vec{x}_3$ 构成的一个单形。将这个单形以 $(1+\varepsilon)(\varepsilon\geq 0)$ 的比例向各个方向进行扩张,令 $\vec{o}=\frac{1}{3}\sum_{i=1}^{3}\vec{x}_i,\vec{y}_i=(1+\varepsilon)(\vec{x}_i-\vec{o})$,则由 $\vec{y}_1,\vec{y}_2$ 和 $\vec{y}_3$ 形成了一个新的单形。在新单形中随机取一点 $\vec{z}$,则 $\vec{z}=k_1\vec{y}_1+k_2\vec{y}_2+k_3\vec{y}_3+\vec{o}$,其中 k_1,k_2, 和 k_3 为 $[0,1]$ 中均匀分布的随机数且满足 $k_1+k_2+k_3=1$。

4.3.3 个体优劣的新比较准则

类似文献[53]，将种群的进化过程分为以下三个阶段：种群中只有不可行解；种群中既有可行解又有不可行解；种群中只有可行解。①当种群中只有不可行解的时候，应使整个种群尽快地向可行域边界逼近，而个体违反约束程度反映个体

到可行域边界的距离，因此我们根据个体违反约束程度的大小来对个体进行选择操作。选择违犯约束程度小的个体，进入下一代种群。② Runarsson 和 Yao[16]认为，通常情况下平衡可行解与不可行解的比例是约束处理技术的关键问题之一。文献[17]使违反约束程度最小的不可行解以一定的概率继续生存，以此来增加群体的多样性，实验结果表明这种多样性机制对提高算法的性能是有效的。事实上，增加群体中违反约束条件最小的不可行解的生存概率，不仅可以提高群体的多样性，而且还可以引导群体中其它不可行解迅速地逼近可行域边界。因此，当种群中既有可行解又有不可行解时，使可行解和不可行解保持一定的比列，同时增加违反约束条件最小的不可行解的生存概率，可以维持种群的多样性，有利于群体向最优解逼近。③当种群中只有可行解时，适应度值大小反映个体同最优解的距离，为了使种群以较快的速度收敛到最优解，我们应根据个体适应度值的大小来进行选择操作。基于以上目的，当种群处于不同进化阶段，遵循以下原则：

1)种群中只有不可行解时，只比较个体违反约束程度的大小，违反约束条件小的个体占优。

2)种群中既有可行解又有不可行解时，为了使可行解与不可行解保持一定的比列，以更好地搜索可行的最优解，同时为了对群体中的可行解和不可行解进行简单直接比较，我们采取了以下处理方式。

$$令\ G_j(\vec{x})=\begin{cases}\max\{0,g_j(x)\} & 1\le j\le l\\ |h_j(x)| & l+1\le j\le m\end{cases} \tag{4-5}$$

表示个体 $\vec{x}$ 与第 j 个约束条件的距离，则

$$G(\vec{x})=\sum_{j=1}^{m}G_j(\vec{x}) \tag{4-6}$$

表示个体 $\vec{x}$ 到可行域边界的距离，也反映了个体 $\vec{x}$ 违法约束条件的程度。

设 $\vec{x}_{\min}$，$\vec{x}_{\max}$ 分别为当前群体可行解函数值最小和最大的个体，$\vec{x}_{uf}$ 为当前群体不可行解中违反约束程度最小的个体

$$f'(\vec{x})=\begin{cases}f(\vec{x}) & \vec{x}为可行解\\ \max\{f(\vec{x}_{\min})+\eta*(f(\vec{x}_{\max})-f(\vec{x}_{\min})),f(\vec{x})\} & \vec{x}为不可行解\end{cases} \tag{4-7}$$

其中，η 为当前群体中不可行解的百分比。

$$G'(\vec{x})=\begin{cases}0 & x为可行解\\ \left|G(\vec{x})-G(\vec{x}_{uf})\right| & x为不可行解\end{cases} \tag{4-8}$$

$$F(\vec{x})=f'(\vec{x})+G'(\vec{x}) \tag{4-9}$$

$F(\vec{x})$ 为转化后的函数适应度值，对求最小值问题来说，$F(\vec{x})$ 数值小的个体占优。当群体中不可行解的比例较小时，由式(4-8)知，不可行解转换后的适应度值小，被选择进入下一代的概率增加，反之不可行解被选择概率减少。因此通过

这种自适应的动态调节机制来有效地调节群体中可行解与不可行解的比例。此外，由式(4-8)得知，取消了对群体中违反约束程度最小的不可行解的惩罚，进一步增加了它进入下一代种群的概率，来引导其它的不可行解快速地逼近可行域。

3)种群中只有可行解时，比较个体适应度函数值的大小。对求最小值问题来说，函数值小的个体占优。

4.3.4 算法步骤

设种群P的规模为n，t为进化代数，采用实数编码，$\lfloor w \rfloor$表示取实数w的整数部分。整个算法实现的具体步骤如下：

Step1:生成初始种群

随机生成初始种群$P_0=\{x_1,\ldots,x_n\}$，P_0中的每一个个体满足$x_i \in S, i \in \{1,\ldots,n\}$，令$t=0$；

Step2: 交叉操作

Step2.1 把群体P_t中的n个体随机分成$\lfloor n/\lambda \rfloor$子类，每一个子类中有$\lambda$个个体，$P_t$剩余的个体加入到种群$P_t^1$。利用多父体正交交叉算子以$p_0$的概率对每一个子类中的个体进行交叉操作产生新的子代个体。然后再从每一个子类的父代个体和子代个体中，根据 4.3.3 节的个体优劣比较准则，选择λ个最好的个体进入种群P_t^1；

Step2.2 把群体P_t^1中的n个体随机分成$\lfloor n/u \rfloor$子类，每一个子类中有u个个体，P_t^1剩余的个体加入到种群P_t^2。利用 SPX 算子以p_1的概率对每一个子类中的个体进行交叉操作产生新的子代个体，然后再从每一个子类的父代个体和子代个体中，根据 4.3.3 节的个体优劣比较准则，选择u个最好的个体进入种群P_t^2；

Step3:变异操作

种群P_t^2的任一个体$p_i=(p_{i,1},p_{i,2},\cdots,p_{i,N})$，以概率$p_m$参与变异操作。具体操作是：①产生一个随机整数$j \in [1,N]$，和一个随机数实数$r \in [0,1]$；②令$p_{i,j}=l_j+r(u_j-l_j)$。群体$P_t^2$经变异后生成的新的子代种群记$P_t^3$；

Step4:种群更新

从P_t^2和P_t^3中，根据 4.3.3 节的个体优劣比较准则，选择n个最好的个体进入到下一代种群P_{t+1}中；

Step5：终止条件判断

判断是否满足终止条件，若满足则输出最优解，否则$t=t+1$，同时转 Step2。

4.4 数值实验

4.4.1 测试函数

g01

Minimize: $f(\vec{x}) = 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$

subject to:

$$g_1(\vec{x}) = 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \le 0,$$
$$g_2(\vec{x}) = 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \le 0,,$$
$$g_3(\vec{x}) = 2x_2 + 2x_3 + x_{11} + x_{12} - 10 \le 0,$$
$$g_4(\vec{x}) = -8x_1 + x_{10} \le 0,$$
$$g_5(\vec{x}) = -8x_2 + x_{11} \le 0,$$
$$g_6(\vec{x}) = -8x_3 + x_{12} \le 0,$$
$$g_7(\vec{x}) = -2x_4 - x_5 + x_{10} \le 0,$$
$$g_8(\vec{x}) = -2x_6 - x_7 + x_{11} \le 0,$$
$$g_9(\vec{x}) = -2x_8 - x_9 + x_{12} \le 0,,$$
$$0 \le x_i \le 1,(i=1,\cdots,9), 0 \le x_i \le 100,(i=10,11,12), 0 \le x_{13} \le 1。$$

已知最优解为 $\vec{x}^*$=(1,1,1,1,1,1,1,1,1,3,3,3,1), $f(\vec{x}^*)$=15.约束条件 g_1, g_2, g_3, g_7, g_8 和 g_9 活跃。

g02

Maximize: $f(\vec{x}) = \left| \dfrac{\sum_{i=1}^{n} \cos^4(x_i) - 2\prod_{i=1}^{n} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} i x_i^2}} \right|$

subject to:

$$g_1(\vec{x}) = 0.75 - \prod_{i=1}^{n} x_i \le 0,$$
$$g_2(\vec{x}) = \sum_{i=1}^{n} x_i - 7.5n \le 0,$$
$$0 \le x_i \le 10,(i=1,\cdots,n), n = 20.$$

全局最优解未知,目前公布的最好结果为 $f(\vec{x}^*) = 0.803619$.约束条件 g_1 几乎活跃 $(g_1 = -10^{-8})$。

g03

Maximize: $f(\vec{x}) = (\sqrt{n})^n \prod_{i=1}^{n} x_i$

subject to:

$$h(\vec{x}) = \sum_{i=1}^{n} x_i^2 - 1 = 0,$$
$$0 \le x_i \le 1,(i=1,\cdots,n), n = 10。$$

已知最优解为 $x_i^* = 1/\sqrt{n},\ (i=1,\cdots,n),\ f(\vec{x}^*) = 1.$

g04

Minimize: $f(\vec{x}) = 5.3578547x_3^2 + 0.8356891x_1x_5 + +37.293239x_1 - 40792.141$

subject to:

$$g_1(\vec{x}) = 85.334407 + 0.0056858x_2x_5 + 0.0006262x_1x_4 + 0.0022053x_3x_6 - 92 \le 0,$$
$$g_2(\vec{x}) = -85.334407 - 0.0056858x_2x_5 - 0.0006262x_1x_4 + 0.0022053x_3x_6 \le 0,$$
$$g_3(\vec{x}) = 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 - 110 \le 0,$$
$$g_4(\vec{x}) = -80.51249 - 0.0071317x_2x_5 - 0.0029955x_1x_2 - 0.0021813x_3^2 + 90 \le 0,$$
$$g_5(\vec{x}) = 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 - 25 \le 0,$$
$$g_6(\vec{x}) = -9.300961 - 0.0047026x_3x_5 - 0.0012547x_1x_3 - 0.0019085x_3x_4 + 20 \le 0,$$

$$78 \le x_1 \le 102, 33 \le x_2 \le 45, 27 \le x_i \le 45, (i = 3,4,5)$$

已知最优解为 $\vec{x}^*$= (78,33,29.995256,45, 36.775812905788), $f(\vec{x}^*)$ = -30665.539. 约束条件 g_1 和 g_6 活跃.

g05

Minimize: $f(\vec{x}) = 3x_1 + 0.000001x_1^3 + 2x_2 + (0.000002/3)x_2^3$

subject to:

$$g_1(\vec{x}) = -x_4 + x_3 - 0.55 \le 0,$$
$$g_2(\vec{x}) = -x_3 + x_4 - 0.55 \le 0,$$
$$h_3(\vec{x}) = 1000\sin(-x_3 - 0.25) + 1000\sin(-x_4 - 0.25) + 894.8 - x_1 = 0,$$
$$h_4(\vec{x}) = 1000\sin(x_3 - 0.25) + 1000\sin(x_3 - x_4 - 0.25) + 894.8 - x_2 = 0$$
$$h_5(\vec{x}) = 1000\sin(x_4 - 0.25) + 1000\sin(x_4 - x_3 - 0.25) + 1294.8 = 0$$
$$0 \le x_1 \le 1200, 0 \le x_2 \le 1200, -0.55 \le x_3 \le 0.55, -0.55 \le x_4 \le 0.55.$$

已知最优解为 $\vec{x}^*$ = (679.9453,1026.067,0.11887, -0.3962336), $f(\vec{x}^*) = 5126.4981$.

g06

Minimize: $f(\vec{x}) = (x_1 - 10)^3 + (x_x - 20)^3$

subject to:

$$g_1(\vec{x}) = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \le 0,$$
$$g_2(\vec{x}) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \le 0,$$
$$13 \le x_1 \le 100, \; 0 \le x_2 \le 100.$$

已知最优解为 $\vec{x}^* = (14.095, 0.84296)$, $f(\vec{x}^*)$ = -6961.81388.两个约束条件均活跃.

g07

Minimize: $f(\vec{x}) = x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2$
$$+2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45$$

subject to:

$$g_1(x) = -105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \le 0,$$
$$g_2(x) = 10x_1 - 8x_2 - 17x_7 + 2x_8 \le 0,$$
$$g_3(x) = -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \le 0,$$
$$g_4(x) = 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \le 0,$$
$$g_5(x) = 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \le 0,$$
$$g_6(x) = x_1^2 + 2(x_2 - 2)^2 - 2x_1x_2 + 14x_5 - 6x_6 \le 0,$$
$$g_7(x) = 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30 \le 0,$$
$$g_8(x) = -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \le 0,$$
$$-10 \le x_i \le 10, (i = 1, \cdots, 10).$$

已知最优解为 $\vec{x}^*$ = (2.171996,2.363683,8.773926,5.095984, 0.9906548, 1.430574, 1.321644,9.828726,8.280092,8.375927), $f(\vec{x}^*) = 24.3062091$.约束条件 g_1, g_2， g_3, g_4, g_5 和 g_6 活跃.

g08

Minimize: $f(\vec{x}) = \dfrac{\sin^3(2\pi x_1)\sin(2\pi x_2)}{x_1^3(x_1 + x_2)}$

subject to:

$g_1(\vec{x}) = x_1^2 - x_2 + 1 \le 0$,

$g_2(\vec{x}) = 1 - x_1 + (x_2 - 4)^2 \le 0$,

$0 \le x_1 \le 10, 0 \le x_2 \le 10$。

已知最优解为 $\vec{x}^* = (1.2279713, 4.2453733)$, $f(\vec{x}^*) =$ 0.095825.最优解位于可行域内.

g09

Minimize: $f(\vec{x}) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7$

subject to:

$g_1(\vec{x}) = -127 + 2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5 \le 0$,

$g_2(\vec{x}) = -282 + 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 \le 0$,

$g_3(\vec{x}) = -196 + 23x_1 + x_2^2 + 6x_6^2 - 8x_7 \le 0$,

$g_4(\vec{x}) = 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \le 0$,

$-10 \le x_i \le 10, (i = 1, \cdots, 7)$.

已知最优解为 $\vec{x}^* =$ (2.330499,1.951372,-0.4775414, 4.365726,-0.6244870, 1.1038131,1.594227), $f(\vec{x}^*) = 680.600573$.约束条件 g_1 和 g_4 活跃.

g10

Minimize: $f(\vec{x}) = x_1 + x_2 + x_3$

subject to:

$g_1(\vec{x}) = -1 + 0.0025(x_4 + x_6) \le 0$,

$g_2(\vec{x}) = -1 + 0.0025(x_5 + x_7 - x_4) \le 0$,

$g_3(\vec{x}) = -1 + 0.01(x_8 - x_5) \le 0$,

$g_4(\vec{x}) = -x_1x_6 + 833.33252x_4 + 100x_1 - 83333.333 \le 0$,

$g_5(\vec{x}) = -x_2x_7 + 1250x_5 + x_2x_4 - 1250x_4 \le 0$,

$g_6(\vec{x}) = -x_3x_8 + 1250000 + x_3x_5 - 2500x_5 \le 0$,

$100 \le x_1 \le 10000, 1000 \le x_i \le 10000, (i = 2, 3), 10 \le x_i \le 1000, (i = 4, \cdots, 8)$.

已知最优解为 $\vec{x}^* =$ (579.3167,1359.943,5110.071,182.0174, 295.5985, 217.9799, 286.4162,395.5979), $f(\vec{x}^*)$=7049.3307.约束条件 g_1, g_2 和 g_3 活跃.

g11

Minimize: $f(\vec{x}) = x_1^2 + (x_2 - 1)^2$

subject to:

$h(\vec{x}) = x_2 - x_1^2 = 0$,

$-1 \le x_1 \le 1, -1 \le x_2 \le 1$.

已知最优解为 $\vec{x}^* = (\pm 1/\sqrt{2}, 1/2)$, $f(\vec{x}^*) = 0.75$.

g12

Maximize: $f(\vec{x}) = (100 - (x_1 - 5)^2 - (x_2 - 5)^2 - (x_3 - 5)^2)/100$

subject to:

$g(\vec{x}) = (x_1 - p)^2 + (x_2 - q)^2 + (x_3 - r)^2 - 0.0625 \le 0$,

$0 \le x_i \le 10, (i = 1, 2, 3), p, q, r = 1, 2, \cdots, 9$.

可行域由 9^3 个离散的球体组成.向量 (x_1, x_2, x_3) 可行当且仅当存在 p, q, r 满足上述

约束条件.已知最优解为 $\vec{x}^*=(5,5,5)$，$f(\vec{x}^*)=1$.最优解位于可行域内.

g13

Minimize: $f(\vec{x})=e^{x_1x_2x_3x_4x_5}$

subject to:

$$h_1(\vec{x})=x_1^2+x_2^2+x_3^2+x_4^2+x_5^2-10=0,$$

$$h_2(\vec{x})=x_2x_3-5x_4x_5=0,$$

$$h_2(\vec{x})=x_1^3+x_2^3+1=0,$$

$$-2.3\le x_i\le 2.3,(i=1,2),-3.2\le x_i\le 3.2,(i=3,4,5).$$

已知最优解为 $\vec{x}^*=$(-1.717143,1.595709,-1.827247, 0.7636413,-0.763645)，$f(\vec{x}^*)=$ 0.0539498.

4.4.2 实验结果

为了评估新算法(NCOA/OED)的性能，我们选取文献[42-45]中采用的 13 个复杂的测试函数进行实验研究，测试函数如 4.4.1 节所示。函数 g02 属于高维多峰函数，主要用来考察算法的搜索能力。函数 g03,g05,g11,g13 主要考察算法的约束处理处理能力，其它几个函数主要用来考察算法的综合能力。实验中，种群规模 $n=100$，$\lambda=3$，$u=3$，多父体正交交叉操作的概率 p_0=0.1，单行交叉操作的概率 $p_1=0.8$，扩张比 $\varepsilon=6$，变异概率 $p_m=0.1$。适应度函数评价次数为 300,000 次。对等式约束优化问题，我们采用文献[52]中提出的新方法。该方法将等式约束条件 $h(\vec{x})=0$ 转化为不等式约束条件 $|h(\vec{x})-e|<0$ 来处理，其中偏差值 e 随着种群代数的递增逐渐下降，其表达式如下： $e(t+1)=e(t)/C$，其中 C 为常数，t 为进化代数。本章对所有等式约束优化问题取 $C=1.0165$，$e(0)=2$。所有实验都在 MATLAB 中完成的，对于每一个测试函数，在相同的条件下独立运行 30 次，记录其最好的结果(best)、中间结果(median)、平均结果(mean)和解的标准差(std.dev)，表 4-3 列出了新算法 NCOA/OED 对 13 个测试函数实验的结果。

为了对比，我们将 NCOA/OED 的运行结果与另外三个国际上较好的约束进化算法进行了比较，他们分别是文献[42]中的 SAFF 算法，文献[43]中的 SMES 算法，文献[44]中的 RY 算法。表 4-4 列出了 4 种算法比较的结果，这些实验结果均来自相关算法的参考文献，NA 表示原文中没有可获得的结果。

Farmani 和 Wright [42]提出的 SAFF 算法实质是一种自适应惩罚函数法，它根据群体中目标函数值最大的个体和群体中最好的个体违反约束条件的程度来调节对群体中最差的不可行解的惩罚大小。SMES[43]算法使群体中最好的不可行解以一定的概率进入到下一代种群中，这对维持种群的多样性具有很重要的作用，特别是对于最优解位于可行域的边界时。Runarsson 和 Yao[44]提出的随机排序法是目前最经典的约束优化算法之一，它通过定义概率 p_f 来协调目标函数和惩罚函数对个体适应度的影响，并采用冒泡排序的方法来实现选择操作。

4.4.3 实验结果分析

从表 4-3 可以看出，除 g10 外，新算法（NCOA/OED）在其它的 12 个测试函数中达到了最优解。从表 3 可以看出，对于高维多峰测试函数 g02, 其它三种算法都不能找到精确的最优解。同 SAFF 算法相比，NCOA/OED 算法在函数

表 4-3 NCOA/OED 算法对 13 个标准测试函数独立运行 30 次的结果

function	optimal	best	median	mean	worst	st. dev
g01	-15.000	-15.000	-15.000	-15.000	-15.000	0
g02	0.803619	-0.803619	-0.792607	-0.792028	-0.771748	7.2E-03
g03	1.00	1.00	1.00	1.00	1.00	4.1E-06
g04	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539	1.1E-11
g05	5126.498	5126.498	5126.498	5126.498	5126.498	8.3E-13
g06	-6961.814	-6961.814	-6961.814	-6961.814	-6961.814	4.5E-12
g07	24.306	24.306	24.306	24.306	24.306	3.9E-06
g08	0.095825	-0.095825	0.095825	0.095825	0.095825	1.5E-17
g09	680.630	680.630	680.630	680.630	680.630	3.0E-12
g10	7049.248	7049. 372	7068.087	7074.198	7161.773	2.5E+01
g11	0.75	0.75	0.75	0.75	0.75	1.9E-08
g12	-1.000	-1.000	-1.000	-1.000	-1.000	0
g13	0.0539498	0.0539498	0.0539498	0.1180986	0.4388463	1.5E-01

g02,g04,g05,g06,g07,g09,g10 的最优结果、平均结果和最差结果方面均优于 SAFF 算法，对于函数 g01,g03,g08,g11,g12，NCOA/OED 算法取得了与 SAFF 算法相似的结果。同 SMES 算法相比，NCOA/OED 算法在函数 g02, g05, g07,g09,g10，g13 的最优结果、平均结果和最差结果方面均优于同 SMES 算法，对于函数 g01,g03,g04,g06,g08,g11,g12 , NCOA/OED 算法取得了与 SMES 算法相似的结果。同 RY 算法相比，NCOA/OED 算法在函数 g02, g07, g10 的最优结果、平均结果和最差结果方面均优于 RY 算法，对于函数 g01,g03, g04,g06,g08,g11,g12 , NCOA/OED 算法取得了与 RY 算法相似的结果。上述实验比较结果表明新算法是一种通用性强且具有较好稳定性的算法。

表 4-4 4 个算法（SAFF[42], SMES[43], RY[44] and NCOA/OED）对 13 个标准测试函数实验结果比较. NA 表示原文没有提供实验结果

Function/ optimal	status	methods			
		SAFF[42]	SMES[43]	RY[44]	NCOA/OED
g01/ -15.000	best	-15.000	-15.000	-15.000	-15.000
	mean	-15.000	-15.000	-15.000	-15.000
	worst	-15.000	-15.000	-15.000	-15.000
	st. dev	0.0E+00	0.0E+00	0.0E+00	0.0E+00
g02/ -0.803619	best	-0.80297	-0.803601	-0.803515	-0.803619
	mean	-0.79010	-0.785238	-0.781975	-0.792028
	worst	-0.76043	-0.751322	-0.726288	-0.771748
	st. dev	1.2E-02	1.7E-02	2.0E-02	7.2E-03

g03/ -1.000	best	-1.000	-1.000	-1.000	-1.000
	mean	-1.000	-1.000	-1.000	-1.000
	worst	-1.000	-1.000	-1.000	-1.000
	st. dev	7.5E-05	2.1E-04	1.9E-04	4.1E-06
g04/ -30665.539	best	-30665.50	-30665.539	-30665.539	-30665.539
	mean	-30665.20	-30665.539	-30665.539	-30665.539
	worst	-30663.30	-30665.539	-30665.539	-30665.539
	st. dev	4.9E-01	0	2.0E-05	1.1E-11
g05/ 5126.498	best	5126.989	5126.599	5126.497	5126.498
	mean	5432.080	5174.492	5128.881	5126.498
	worst	6089.430	5304.167	5142.472	5126.498
	st. dev	3.9E+03	5.0E+01	3.5E+00	8.3E-13
g06/ -6961.814	best	-6961.800	-6961.814	-6961.814	-6961.814
	mean	-6961.800	-6961.284	-6875.940	-6961.814
	worst	-6961.800	-6952.482	-6350.262	-6961.814
	st. dev	0	1.9E+00	1.6E+02	4.5E-12
g07/ 24.306	best	24.48	24.327	24.307	24.306
	mean	26.58	24.475	24.374	24.306
	worst	28.40	24.843	24.642	24.306
	st. dev	1.1E+00	1.3E-01	6.6E-02	3.9E-06
g08/ -0.095825	best	-0.095825	-0.095825	-0.095825	-0.095825
	mean	-0.095825	-0.095825	-0.095825	-0.095825
	worst	-0.095825	-0.095825	-0.095825	-0.095825
	st. dev	0	0	2.6E-17	1.5E-17
g09/ 680.630	best	680.64	680.632	680.630	680.630
	mean	680.72	680.643	680.656	680.630
	worst	680.87	680.719	680.763	680.630
	st. dev	5.9E-02	1.6E-02	3.4E-02	3.0E-12
g10/ 7049.248	best	7061.34	7051.903	7054.316	7049. 372
	mean	7627.89	7253.047	7559.192	7074.198
	worst	8288.79	7638.366	8835.655	7161.773
	st. dev	3.7E+02	1.4E+02	5.3E+02	2.5E+01
g11/ 0.75	best	0.750	0.75	0.750	0.75
	mean	0.750	0.75	0.750	0.75
	worst	0.750	0.75	0.750	0.75
	st. dev	0	1.5E-04	8.0E-05	1.9E-08
g12/ -1.000	best	-1.000	-1.000	-1.000	-1.000
	mean	-1.000	-1.000	-1.000	-1.000
	worst	-1.000	-1.000	-1.000	-1.000
	st. dev	0	0	0.0E+00	0
g13/ 0.0539498	best	NA	0.053986	0.053957	0.0539490
	mean	NA	0.166385	0.067543	0.1180986
	worst	NA	0.468294	0.216915	0.4388463
	st. dev	NA	1.8E-01	3.1E-02	1.5E-01

4.5 本章小结

全局搜索能力和约束处理技术是利用 GA 求解约束优化问题最为关键的两个问题。针对这些问题，本章提出了一种新的基于正交实验设计的约束优化算法。新算法具有以下优点：①利用正交实验设计方法设计用于多个父代个体的交叉操作的算子，新的交叉算子能够有效利用多个父代个体所携带的信息产生新的具有代表性的子代个体，以更好地搜索解空间和维持种群的多样性，使种群在进化的过程中尽可能收敛到最优解。②针对群体进化过程中所处不同阶段的特点，利用不同的比较个体优、劣的准则来指导选择操作，使群体逐步地向可行的最优解逼近。③在处理不同类型的约束优化问题中，如等式和不等式约束优化问题，表现了良好的寻优能力。④针对 13 个测试函数的优化结果表明，针对不同类型的约束优化问题，新算法参数比较固定，不需要针对不同的优化问题来过多地调整实验参数，有利于算法的推广和应用。对该算法进行完善和推广是进一步的工作。

第五章　基于正交遗传算法的改进型免疫 PID 控制器的优化

本章中我们首先介绍免疫反馈的基本概念和免疫反馈 PID 的基本结构，并分析了目前常用的免疫反馈 PID 控制器的本质特征及其存在的问题，提出了一种改进的模糊反馈 PID 控制器的结构，并利用第三章提出的混合自适应正交遗传算法对其控制器参数进行离线优化，最后用仿真实验验证了算法的有效性。

5.1 引言

生物免疫系统通过免疫机制中的抗原识别、抗原记忆和抗体的抑制、促进等，通常情况下，能在动态变化的环境中维持其自身系统的稳定性，是一个具有极强鲁棒性的自适应系统，能处理各种干扰和不确定性，这种免疫调节机制可用来有效改进控制系统的性能。Takahashi等[46]已把这种免疫反馈机制应用到控制系统中，提出了免疫反馈控制的策略。

文献[46]基于分析 B 细胞、T 细胞相互作用及 T 细胞的免疫反馈机理,设计了一种 P 型免疫反馈控制系统；文献[47-49]针对 P 型免疫控制器抗扰性能较差、难于克服稳态偏差等不足,结合模糊控制原理,将常规的 PID 控制与 P 型的免疫反馈控制器进行串联，设计了一种模糊自调整免疫反馈控制系统。但控制器的比例增益，积分增益和微分增益三个参数的变化对系统性能的影响是不同的[50]，将 P 型免疫控制器与常规 PID 串联不是最好的处理方法。因此本章提出了一种将 P 型免疫控制器分别与积分控制器、微分控制器进行混合联接的方法，有效地消除静态偏差，提高了系统的响应速度，改善了控制品质。

5.2 模糊免疫反馈 PID 控制器的改进

5.2.1 免疫反馈控制的概念

.免疫P控制器是借鉴生物系统的免疫机理而设计出的一种非线性控制器，免疫系统调节原理如图5-1所示，其中，APC为抗原呈递细胞,T_H为辅助细胞，T_S为抑制细胞[51]。

当外界物质(包括抗原、被病毒感染的自身细胞等) 的信息被抗原呈递细胞(APC) 捕获后，APC将信息传递给T细胞,即传递给T_H细胞和T_S细胞,然后刺激B细胞,B细胞产生抗体以消除抗原。当抗原较多时,机体内的T_H细胞也较多,而T_S细胞却较少,从而产生的B细胞会多些。随着抗原的减少,体内T_S细胞增多，它抑制了T_H细胞的产生,则B细胞也随着减少，经过一段时间后，免疫反馈系统便趋于平衡。

基于上述T细胞反馈调节机理，假设第 k 代的抗原数量为 $\varepsilon(k)$,受到抗原刺激的来自 T_H 细胞的输出定义为 $T_H(k)$, T_S 细胞对B细胞的作用为 $T_S(k)$，则B细胞接

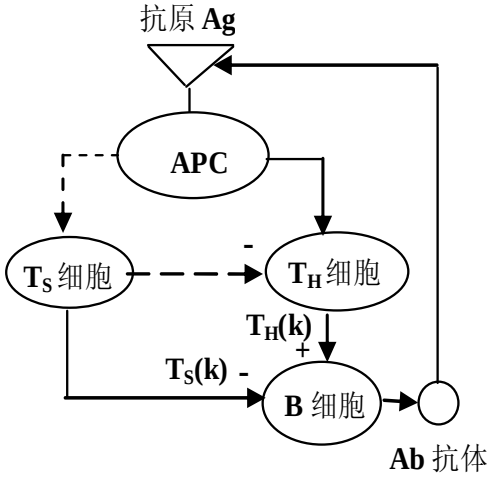

图 5-1 免疫反馈原理图

受的总激励为：

$$S(k) = T_H(k) - T_S(k) \tag{5-1}$$

$$\Delta S(k) = S(k) - S(k-1) \tag{5-2}$$

式中 $T_H(k) = k_1\varepsilon(k), T_S(k) = k_2 f[s(k), \Delta s(k)]\varepsilon(k)$，其中 k_1 是刺激因子，k_2 是抑制因子，$\Delta S(k)$ 是B细胞接收的刺激的误差的变化量，若以抗原的数量 $\varepsilon(k)$ 作为偏差 $e(k)$，B细胞接收的刺激 $S(k)$ 作为控制输入 $u(k)$，则有如下反馈控制规律

$$\begin{aligned} u(k) &= K\{1-\eta f[u(k), \Delta u(k)]\}e(k) \\ &= \overline{K}_p e(k) \end{aligned} \tag{5-3}$$

其中，$K = k_1, \eta = k_1 / k_2$，$f(\bullet)$ 是一个关于 $\Delta u(k)$ 的非线性函数，参数 K 控制反应速度，参数 η 控制稳定效果，P 型免疫控制器 IMF 方框图如图 5-2 所示。

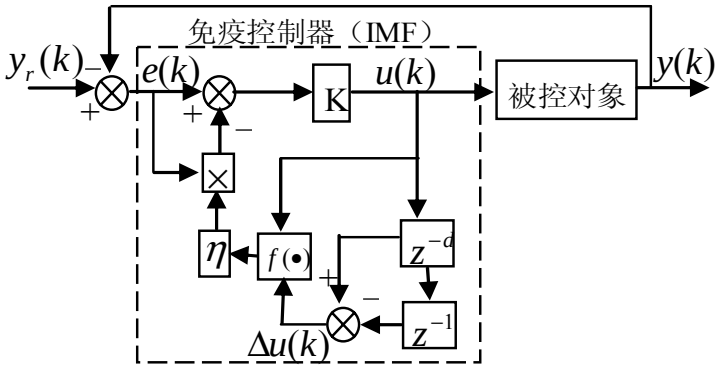

图 5-2 免疫反馈控制框图

常规的 P 控制器的控制算法为：

$$u(k) = K_p e(k) \tag{5-4}$$

将式(5-3)与(5-4)比较可知：基于免疫反馈机理的反馈控制器 IMF 是一个非线性的 P 控制器，其比例增益为

$$\overline{K}_p = K\{1-\eta f[u(k),\Delta u(k)]\} \tag{5-5}$$

其中，比例增益 $\overline{K}_p$ 随着控制器的输出变化而变化，控制器的性能在很大程度上依赖于参数 K, η 和非线性函数 $f(\bullet)$ 的选取。

5.2.2 免疫 PID 控制器的改进

由 5.2.1 可以看出，免疫反馈控制器是一个非线性 P 控制器，由 P 控制器的作用可以知道，它不能补偿噪声或非线性干扰引起的误差，因此一般不单独使用，而常规 PID 控制是将偏差的过去、现在和未来的信息都进行了综合考虑，因此将免疫反馈控制与常规 PID 控制相结合，可以相互取长补短，以进一步提高系统的控制性能。

考虑常规 PID 控制算法的离散形式为

$$u(k) = (K_p + \frac{K_i}{z-1} + K_d\frac{z-1}{z})e(k) \tag{5-6}$$

若将免疫控制器和常规 PID 控制器串联，则由式(5-3)和(5-6)可得免疫 PID 控制器的输出如式(5-7)所示。

$$u(k) = K\{1-\eta f[u(k),\Delta u(k)]\}\times(K_p + \frac{K_i}{z-1} + K_d\frac{z-1}{z})e(k) \tag{5-7}$$

即：$$u(k) = [\overline{K}_p + \overline{K}_i\frac{1}{z-1} + \overline{K}_d\frac{z-1}{z}]e(k) \tag{5-8}$$

其中,

$$\overline{K}_p = KK_p\left\{1-\eta f\left[u(k),\Delta u(k)\right]\right\}$$

$$\overline{K}_i = KK_i\left\{1-\eta f\left[u(k),\Delta u(k)\right]\right\}$$

$$\overline{K}_d = KK_d\left\{1-\eta f\left[u(k),\Delta u(k)\right]\right\}$$

分别为可变的比例增益、积分增益和微分增益。由 $\overline{K}_p,\overline{K}_i,\overline{K}_d$ 的表达式可以知道，$\overline{K}_p,\overline{K}_i,\overline{K}_d$ 同时增大或者同时缩小相同的倍数，但我们知道比例增益、积分增益和微分增益三个参数对系统性能的影响是不同的，所以将免疫控制器与常规 PID 串联不是最好的处理方法，本章将免疫控制器与积分控制器和微分控制器进行串并联，以消除静态偏差，提高响应速度，改善控制品质，免疫 PID 控制器的结构如图 5-3 所示。

免疫PID控制算法的输出形式如式子(5-9)所示。

$$u(k) = [\overline{K}_p' + \overline{K}_i'\frac{1}{z-1} + \overline{K}_d'\frac{z-1}{z}]e(k) \tag{5-9}$$

其中,$\overline{K}_p' = K_1\left\{1-\eta_1 f\left[u(k),\Delta u(k)\right]\right\} = K_p'\left\{1-\eta_1 f\left[u(k),\Delta u(k)\right]\right\}$

$$\overline{K}_i' = K_iK_2\left\{1-\eta_2 f\left[u(k),\Delta u(k)\right]\right\} = K_i'\left\{1-\eta_2 f\left[u(k),\Delta u(k)\right]\right\}$$

$$\overline{K}_d' = K_dK_3\left\{1-\eta_3 f\left[u(k),\Delta u(k)\right]\right\} = K_d'\left\{1-\eta_3 f\left[u(k),\Delta u(k)\right]\right\}$$

分别为可变的比例增益、积分增益和微分增益。免疫控制器(5-9)是一个非线性的 PID 控制器，根据所控对象的不同情况，$\overline{K}_p'$、$\overline{K}_i'$ 和 $\overline{K}_d'$ 中的参数 K_p', K_i', K_d', η_1, η_2, η_3 和非线性函数 $f(\bullet)$ 也可以不同，从而也可以获取随工作状况而自动调节的非线性PID控制器。

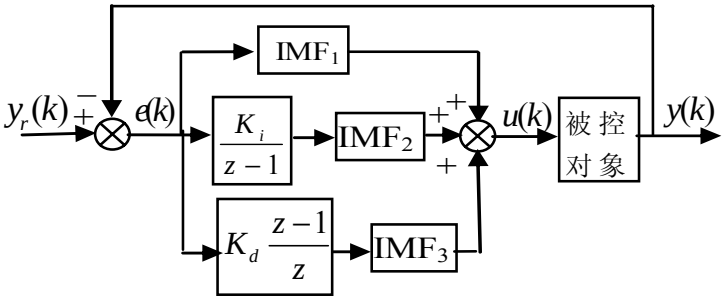

图 5-3 免疫 PID 控制器结构框图

5.2.3 免疫反馈规律的模糊自适应调整

利用模糊控制器可逼近非线性函数 $f(\bullet)$。本系统中的采用的模糊控制器结构与文献[49]中的相同，即每个输入变量被二个模糊集模糊化，分别是 " 正 " (P) 和"负" (N)；输出变量被3个模糊集模糊化，分别是"正" (P)、"零" (Z) 和" 负"(N)。以上隶属度函数都定义在整个 $(-\infty,+\infty)$ 区间。按“细胞接受的刺激越大，则抑制能力越小 ”及“细胞接受的刺激越小，则抑制能力越大”的原则，可采用以下4条模糊规则：

(1) *If* u is P and Δu is P then $f(u,\Delta u)$ is N (1)

(2) *If* u is P and Δu is N then $f(u,\Delta u)$ is Z(1)

(3) *If* u is N and Δu is P then $f(u,\Delta u)$ is Z(1)

(4) *If* u is N and Δu is N then $f(u,\Delta u)$ is P(1)

各规则中, 使用Zadeh的模糊逻辑AND操作,并采用常用的“centroid”反模糊化方法得到模糊控制器的输出 $f(\bullet)$，由(5-8)式可得增量式免疫PID控制器的输出为：

$$u(k)=u(k-1)+\overline{K}_p'[e(k)-e(k-1)]+\overline{K}_i'e(k)Ts+\overline{K}_d'[e(k)-2e(k-1)+e(k-2)]/Ts \quad (5\text{-}10)$$

其中 Ts 为采样时间。

5.3 模糊免疫 PID 控制器的优化

由(5-9)式和(5-10)式可以看出，在设计PID型免疫控制器时，由于存在着非线性项，因此解析确定参数 K_p', K_i', K_d', η_1, η_2, η_3 的值比较困难，遗传算法是一种并行分布式随机搜索的全局优化方法，它不需要任何初始信息，只根据适应度函数值，通过遗传算子进行全局的搜索，就能得到全局最优解，第三章中的自适应正交遗传算法(HSOGA)已表现出良好的搜索性能。因此，将其应用于模糊免疫

PID控制器参数的优化是可行的。

利用 HSOGA 算法优化模糊免疫 PID 控制器参数的步骤如下：

Step1：对要优化的参数编码，确定每个参数的大致范围和编码长度。为了避免参数选取范围过大，可以先按经验选取一组参数，然后再在这组参数的附近利用免疫算法进行设计，从而大大减少初始寻优的盲目性，减少计算量。本章根据 Ziegler-Nichols 设计法得到的 PID 参数 $K_{p-zn}, K_{i-zn}, K_{d-zn}$，选择参数范围为 $K_p^{'} \in (1-\alpha_1, 1+\alpha_1)K_{p-zn}, K_i^{'} \in (1-\alpha_2, 1+\alpha_2)K_{i-zn}$，$K_d^{'} \in (1-\alpha_3, 1+\alpha_3)K_{d-zn}$，$\eta_1$，$\eta_2$，$\eta_3 \in$ （0.5-α_4，0.5+α_4），采用实数编码。

Step2：根据具体的控制性能要求确定出最优指标函数 J,则适应度函数可取为：F=1/J。将待优化的参数 $K_p^{'}$、$K_i^{'}$、$K_d^{'}$、η_1、η_2、η_3 作为个体的各分量，在指定的约束范围内，初始化种群。

为获取满意的过渡过程的动态性能，采用误差 $e(k)$ 绝对值的时间积分性能指标作为参数选择的最小目标函数 J。

$$J = \sum_{k=0}^{\infty} k^* |e(k)| * Ts \tag{5-11}$$

式中 $e(k) = y_r - y(k)$ 为系统误差，为了避免超调，一旦误差的变化率大于给定的 σ，将误差的最大变化率作为目标函数的一个项，此时的目标函数为：

$if \quad \max(ec) > \delta$

$$J = \left(\sum_{k=0}^{\infty} k^* |e(k)| * Ts \right) \left(1 + \max(ec) \Big/ \delta \right) \tag{5-12}$$

式中 $ec(k) = e(k) - e(k-1), (k>1)$，$ec = (ec(2), ec(3), ec(4), \cdots, ec(\infty))$，$\delta$ 是用来控制超调量的大小。

Step3：根据目标函数 J 和个体各分量的范围，利用第三章的混合自适应正交遗传算法对待优化的模糊免疫控制器参数进行寻优。HSOGA 详细描述见第三章。

5.4 模糊免疫反馈 PID 控制器的仿真

本章以一阶延时系统

$$G_1(s) = \frac{1.5}{100s+1} e^{-100s} \tag{5-13}$$

为被控制对象模型，采样时间 Ts=20s，输入为单位阶跃信号。

最大进化代数为50,种群 P 的规模为80，采用实数编码,长度6位。由 Ziegler-Nichols 设计法得到的 PID 参数 K_{p-zn}=0.9166, K_{i-zn}=0.0060, K_{d-zn}=33.5195，取 $\delta = 0.003$，$\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = 0.8$。通过10代进化，获得的模糊免疫PID优化参数：$K_p^{'}$=0.5224, $K_i^{'}$=0.0048, $K_d^{'}$=10.9853, η_1=0.3769, η_2=0.5563, η_3=0.3416。

仿真结果如图 5-4 所示,其中曲线 1 为采用常规 PID 控制算法时的系统输出，曲线 2 为采用 Smith 预估控制算法时的系统输出，曲线 3 为采用本章提出的控制算法时的系统输出，曲线 4 为将常规 PID 与免疫控制器串联后的系统输出(η=0.5563)。由图 5-4 可以看出，与常规的 PID 控制算法和 Smith 预估控制算法相比，本章提出的控制算法的快速性能好，无超调，比将常规 PID 同免疫控制器串联后的快速性能稍好.

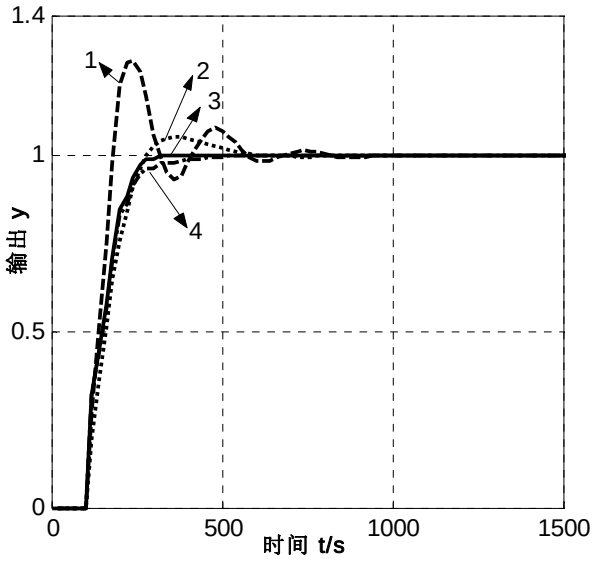

图5-4　四种控制器控制性能的比较

5.5 本章小结

本章针对 P 型免疫反馈控制器存在的不足，提出一种改进型的基于免疫反馈机理的模糊免疫 PID 控制器的结构。根据 P 型免疫反馈控制器和常规 PID 控制器的特点，该控制器由免 P 型疫反馈控制器分别先和积分控制器、微分控制器串联后再与其并联组成。最后利用第三章的提出的混合自适应正交遗传算法对改进后的控制器参数进行离线优化。实验结果表明，它有很好的动态性能和稳定性能，具有一定的实用价值。

第六章 总结和展望

本论文针对正交遗传算法其及在函数优化中的应用进行了深入细致的研究，研究的重点及相应的特色主要体现在以下方面：

1、根据父代个体的相似度，利用正交试验设计方法设计交叉算子。新的交叉算子的主要特点是根据父代个体所确定的空间大小，自适应地调整繁殖子代个体的数目，以有效抑制相似个体过度繁殖；同时新的交叉算子能产生均匀地分布在父代个体所确定的解空间中的子代个体，以维持子代种群的多样性和较丰富的模式。

2、提出基于种群分割和单形搜索的局部搜索策略。通过对多个不同的局部领域进行并行搜索从而实现有效的全局搜索。

3、为了有效继承多个父代个体携带的基因信息，利用正交试验设计方法从多个父代个体所确定的子代个体的样本空间中，产生具有代表性的组合来生成子代个体。

4、提出了一种新的约束处理技术。同目前常用的惩罚函数法和多目标优化法相比，新的约束处理技术有以下优点：①同惩罚函数法相比，新方法不需要调整任何参数②同多目标方法想比，新的方法不需要计算 Parato 前沿，计算代价少，且实现简单，不需要过多的背景知识③能自动平衡群体中不可行解和可行节的比例，以维持群体的多样性。

5、提出了一种新的免疫控制器的结构，并利用第三章提出的混合自适应正交遗传算法对其控制器参数进行离线优化。新的控制器能够根据干扰的大小自适应地调整控制器参数来抑制外界的扰动。

随着计算智能地不断发展和各交叉学科的融合不断地加深，为遗传算法及优化技术的发展不断地注入新的血液和提供了广阔的发展空间。因此，本论文还可以从以下几个方面进行改进和提高：

1、把本文提出的两种正交交叉算子进一步推广到多目标优化问题和超高维的全局优化问题中，值得进一步深入研究。

2、引进简单而高效的局部搜索技术和加强邻居个体之间的信息交换，来充分挖掘群体中的局部信息以提高算法的局部搜索能力和搜索精度，从而实现有效的全局搜索。

3、目前，在用遗传算法求解全局和约束优化问题时，问题本身的信息并没有得到充分的利用，例如，函数的梯度和可微的信息。因此，如何对函数本身所携带的信息进行简单的近似处理来引导整个群体或子群体进化

的方向有待更进一步研究。

4、通过已有的数学方法，利用群体中的部分个体的数学特征，来建立优化问题的概率模型，有待深入研究。

5、好的重组算子(交叉、变异)对算法优化性能的影响至关重要，但选择和替换策略直接影响这些算子的作用的发挥，因此如何选择和设计好的选择和替换策略，使群体中的个体保持一定的空间距离，以维持群体的多样性和防止群体收敛到局部最优，是我们需要进行的下一步工作。

6、如何设计合理的目标函数，同时把多目标技术引入到 PID 控制器的设计中，来进行多目标鲁棒控制器的设计，这样可以得到一组 Pareto 最优解，供决策者根据控制指标和控制性能的需求选择一组合适的控制器参数，有待进一步深入研究。

参考文献

[1] Holland, J H. Adaptation in nature and artificial systems[M]. MIT Press,1975.

[2] 蔡自兴, 徐光祐. 人工智能及应用[M]. 北京:清华大学出版社,2004.

[3] Davis L D. Handbook of genetic algorithms[M]. Van Nortrand Reinhold, 1991.

[4] 陈国良等. 遗传算法及其应用[M]. 北京:人了邮电出版社, 1996.

[5] 周明, 孙树栋. 遗传算法原理与应用[M]. 北京:国防工业出版社, 1999.

[6] Goldberg D E, Segrest P.Finite markov chain analysis of genetic algorithms[C]. In Proc of the 2nd Int Conf on Genetic Algorithms, Cambridge, MA: Lawrence Erlbaum, 1987: 69-76.

[7] Eiben A E, Aarts E H L, Van Hee K. Global convergence of genetic algorithms: An infinite Markov chain analysis[C]. In Parallel Problem Solving from Nature. Berlin: Springer Verlag,1991.

[8] Fogel D B. Asymptotic convergence properties of genetic algorithms and evolutionary programming: Analysis and experiments[M]. Cybernetics and Systems, 1994: 25(1):389-407.

[9] Rudolph G. Convergence analysis of canonical genetic algorithms[J]. IEEE Transanction on Neutral Networks,1994,5(1):96-101.

[10] Qi X, Palmieri F. Theoretical analysis of evolutionary algorithms with an infinite population size in continues space[J]. IEEE Transactions on Neural Networks, 1994, 5(1): 102-119.

[11] 梁艳春，周春光，王在中，基于扩展串的等价遗传算法的收敛性[J]. 计算机学报, 20(8), 1997:686-694.

[12] 张讲社, 徐宗本, 梁怡, 整体退火遗传算法极其收敛充要条件[J]. 中国科学(E), 27(2), 1997:154-164.

[13] Guo G Q, Yu S Y. The unified method analyzing the convergence of genetic algorithms[J]. 控制理论与应用，18(3), 2001: 443-446.

[14] 陈根社, 陈新海. 应用遗传算法设计自动交会控制器[J]. 西北工大学报,1994,11(2):247-252.

[15] Mudock T M. Use of a genetic algorithm to analyze robust stability problems[C]. Proc. American Control Conf., Boston, 1997,886-889.

[16] Potter B, Jones A H. Genetic tuning of digital PID controllers[J]. Electronic Letters, 1992, 28(9):834-844.

[17] Kristinsson K, Dument G A. System identification and control using genetic algorithms[J]. IEEE Transactions SMC,1992,22(5):1033-1046.

[18] Maclay D. Applying genetic search techniques to driver-train modeling[J]. IEEE Control Systems Magazine, 1993,3:50-55.

[19] Freeman I M. Tuning fuzzy logic controller using genetic algorithms Aerospace Applications[C]. Proc. AAAIC, Dayton, 1990,351-358.

[20] Park D, Kandel A. Langholz, G. .Genetic-Based new fuzzy reasoning models with application to fuzzy control[J]. lEEE Transactions SMC,1994,24(1):39-47

[21] Zurada J. Introduction to artificial neural systems[J]. West Publishing compeny, 1992.

[22] Yao, X. A review of evolutionary artificial neural networks[J]. International Journal of Intelligent Systems, 1993, 8: 539-567.

[23] Goldberg D E, Deb K. A comparative analysis of selection schemes used in genetic algorithms[C]. In Foundations of Genetic Algorithms, San Mateo, CA: Morgan Kaufmann, 1991:69-93.

[24] Back T. Evolutionary algorithms in theory and practice[M]. New York:Oxford University Press,1996.

[25] Leung Y W, Wang Y P. An orthogonal genetic algorithm with quantization for global numerical optimization[J]. IEEE Transactions on Evolutionary Computation,2001,5(1):41-53.

[26] 曾三友,魏巍,康立三,等.基于正交设计的多目标演化算法[J].计算机学报,2005,28(7): 1153-1162.

[27] Wang Y, Liu H, Cai Z X. An orthogonal design based constrained optimization. Engineering Optimization, evolutionary algorithm.2007,39(6):715-736.

[28] Leung Y W and Zhang Q F. Evolutionary algorithms + experimental design methods: A hybrid approach for hard optimization and search problems[C]. Res.Grant Proposal, Hong Kong Baptist Univ., 1997.

[29] Tsutsui S, Yamamura M, Higuchi T. Multi-parent recombination with simplex crossover in real-coded genetic algorithms[C]. Proceedings of the GECCO-99, 1999:657-664.

[30] Cai Z X, Wang Y. A multiobjective optimization-based evolutionary algorithm for constrained optimization[J]. IEEE Transactions on Evolutionary Computation, 2006,10(6):658-675.

[31] Wang Y P, Dang C Y. An evolutionary algorithm for global optimization based on level-set evolution and latin squares[J]. IEEE Transactions on Evolutionary Computation, 2007,11(5):579-595.

[32] Tsai J T, Liu T K, Chou J H. Hybrid taguchi-genetic algorithm for global numerical optimization[J]. Transactions on Evolutionary Computation,2004,8(4): 365-377.

[33] Zhang Q F, Sun J J, Tsang E P K et al. Ford hybrid estimation of distribution

algorithm for global optimization, Engineering Computations, 2003,21(1):91-107.

[34] Wang Y P, Dang C Y. An evolu- tionary algorithm for global optimization based on level-set evolution and latin squares[J].IEEE Transactionson Evolutionary Computation Computation,2007,11(5):579-595.

[35] 张铃,张钹. 佳点集遗传算法[J].计算机学报, 2001,24(9):917-922.

[36] Hamida S B, Schoenauer M. ASCHEA: New results using adaptive segregational constraint handling[C]. In Proc of the Congress on Evolutionary Computation 2002 (CEC'2002), Piscataway, NJ: IEEE Press, 2002. 82–87.

[37] Farmani R., Wright J A. Self-adaptive fitness formulation for constrained optimization[J]. IEEE Transactions on Evolutionary Computation, 2003,7(5): 445–455.

[38] Yu J X, Yao X, Choi C, et al. Materialized view selection as constrained evolutionary optimization[J]. IEEE Transactions on Systems, Man and Cybernetics Part C, 2003,33(4): 458–467.

[39] 周育人,李元香,王勇,康立山. Pareto 强度值进化算法求解约束优化问题[J].软件学报,2003,14(7): 1243-1249.

[40] Cai Z X, Wang Y. A multiobjective optimization-based evolutionary algorithm for constrained optimization[J]. IEEE Transactions on Evolutionary Computation, 2006.10(6):658-675.

[41] Wang Y, Cai Z X, Guo G Q, et al. Multiobjective optimization and hybrid evolutionary algorithm to solve constrained optimization problems[J]. IEEE Transactions on Systems, Man, and Cybernetics. B, 2007,37(3):560-575.

[42] Farmani R, Wright J A. Self-adaptive fitness formulation for constrained optimization[J]. IEEE Transactions on Evolution Computation (S1089-778X). 2003, 7(5):445–455.

[43] Efrén M, Carlos A C C. A simple multimembered evolution strategy to solve constrained optimization problems[J]. IEEE Transactions on Evolution Computation (S1089-778X). 2005, 9(1):001-017.

[44] Runarsson T P, Yao X. Stochastic ranking for constrained evolutionary optimization[J]. IEEE Transactions on Evolutionary Computation, 2000,4(3): 284–294.

[45] Wang Y, Liu H, Cai Z X. An orthogonal design based constrained optimization[J]. Engineering Optimization, 2007, 39(6):715-736.

[46] Takahashi K, Yamada T. Application of an immune feedback mechanism to control systems[J]. JSME Int J ,Series C, 1998, 41 (2) : 184 -191.

[47] Ding Y S, Ren L H.Fuzzy Self-Tuning Immune Feedback Controller for Tissue Hyperthermia[C]. IEEE International Conference on Fuzzy Systems C. Piscataway: IEEE, 2000. 534-538.

[48] 谈英姿, 沈炯, 吕震中.免疫PID控制器在汽温控制系统中的应用研究[J].中国电机工程学报,2002, 22(10):148-152.

[49] SU Y X, LI X,et al. Fuzzy-Immune PID control for AMB systems[J]. WUHAN UNIVERSITY JOURNAL OF NATURAL SCIENCES,2006,11(3):637-641.

[50] 肖永利,张琛.位置伺服系统的一类非线性PID调节器设计[J].电气自动化,2000,22(1):20-22.

[51] 蔡自兴.智能控制原理与应用[M].北京:清华大学出版社,2007.

[52] Hamida S B, Schoenauer M. ASCHEA:New results using adaptive segregational constraint handling[C]. Congress on Evolutionary Computation, Hawaii, USA, 2002.

[53] Wang Y, Cai Z X, Zhou Y R, et al. An Adaptive Trade-off Model for Constrained Evolutionary Optimization[J]. IEEE Transactions on Evolutionary Computation. 2008,12(1): 80-92.

致 谢

三年的研究生学习和生活当中，得到了导师、家人、师兄弟、老师们在学习上和生活上帮助和关心，特此表示感谢！同时也感谢三年来曾经关心和帮助过我的同学和朋友们。

首先，我衷心地感谢我的导师蔡自兴教授在这三年对我无微不至的关怀和栽培，本文也是在蔡老师的督促与悉心指导下完成的。蔡老师治学严谨，知识渊博，认真负责，态度和蔼和对学生的真诚关怀都给我留下了至深的印象和深远的影响。而他超出常人的毅力，执着的科学追求，务实的工作作风是我一生学习的典范。

感谢师母翁老师对我学习和生活上的关怀和帮助。

感谢龚涛师兄和王勇师兄对我学术上的指导和帮助。

感谢夏洁师姐和贺阳建师兄对我生活中的关心和帮助。

感谢智能所得诸位老师，唐琎老师、肖晓明老师、陈爱斌老师、唐素勤老师、魏世勇老师、高平安老师、刘丽珏等老师对我学习上的指导和帮助。

感谢进化计算小组诸位同门的支持和照顾，他们是肖赤心，刘星宝，罗一丹，刘慧，封鹏成，在与他们的讨论中，我获得了丰富的灵感和创意，对我的研究帮助甚大。

此外还要感谢和我朝夕相处的亲密同学，他们是：宋嘉灿，王绍钰，肖常，阚晶，蒋莹，许海柱等。和他们相处的日子给我留下了非常美好的回忆！感谢好友左海洋，丁厉华，刘鹏，唐文杰，周建伟，高科对我的鼓励和帮助。

最后，感谢我的家人这二十多年来的对我无私的支持和关怀。

特别感谢国家基础研究项目（项目号：A1420060159）、国家自然科学基金项目(项目号：60404021)的资助！

江 中 央

2008 年 4 月于中南大学

攻读学位期间主要的研究成果

[1] 江中央,蔡自兴,龚涛.一种改进的模糊免疫反馈 PID 控制器.控制工程(已录用).

[2] 江中央,蔡自兴,王勇.一种用于全局优化的混合正交遗传算法.计算机工程(已录用)

[3] 江中央,蔡自兴,王勇.一种求解全局优化问题的混合自适应正交遗传算法.软件学报.(已投稿)

[4] 蔡自兴,江中央,王勇,罗一丹. 一种新的基于正交试验设计的约束优化算法.计算机学报(已投稿).