硕士学位论文

基于状态转移算法的多目标优化方法研究
Multiobjective optimization method based on state transition algorithm

学科专业　控制科学与工程

研究方向　智能控制与优化决策

作者姓名　周佳佳

指导教师　周晓君 副教授

中 南 大 学
2019 年 5 月

中图分类号 TP18　　学校代码 10533
UDC 621.3　　学位类别 学术学位

硕士学位论文

基于状态转移算法的多目标优化方法研究
Multiobjective optimization method based on state transition algorithm

作者姓名：　周佳佳

学科专业：　控制科学与工程

学科方向：　控制理论与控制工程

研究方向：　智能控制与优化决策

二级培养单位：　自动化学院

指导教师：　周晓君　副教授

论文答辩日期＿＿＿＿＿　　答辩委员会主席＿＿＿＿＿

中 南 大 学
2019 年 5 月

学位论文原创性声明

本人郑重声明，所呈交的学位论文是本人在指导教师指导下进行的研究工作及取得的研究成果。尽我所知，除了论文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得中南大学或其他教育机构的学位或证书而使用过的材料。与我共同工作的同志对本研究所作的贡献均已在论文中作了明确的说明。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

作者签名：_____ 日期：_____年___月___日

学位论文版权使用授权书

本学位论文作者和指导教师完全了解中南大学有关保留、使用学位论文的规定：即学校有权保留并向国家有关部门或机构送交学位论文的复印件和电子版；本人允许本学位论文被查阅和借阅；学校可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用复印、缩印或其它手段保存和汇编本学位论文。

保密论文待解密后适应本声明。

作者签名：_____ 指导教师签名_____

日期：_____年___月___日 日期：_____年___月___日

基于状态转移算法的多目标优化方法研究

摘要：多目标优化是指对多个相互矛盾的目标进行优化，以获得多个目标权衡下的最优解集。求解此类问题十分困难，基于智能优化算法的多目标优化方法易于实现，实用性强，因而得到了广泛的研究。状态转移算法作为一种新型的智能优化算法，在多目标优化领域的研究较少。本文基于状态转移算法对多目标优化方法进行研究，主要研究工作及贡献如下：

首先，为了获得状态转移算子的搜索性能对多目标优化算法收敛性的影响规律，基于 GD 指标，设计了相关实验，分析了算子单独使用和组合使用及其参数对算法收敛性的影响。研究发现，在选择策略一致的前提下，伸缩算子适合单独使用产生候选解集，平移算子适合与其他算子组合使用，才能使得算法具有较好的收敛性；旋转算子和坐标搜索算子无论是单独使用还是与其他算子组合使用产生候选解集，都对算法的收敛性具有积极的影响。

其次，提出了基于 Pareto 占优的多目标状态转移算法框架，MOSTA/P (Multi-objective state transition algorithm based on Pareto dominance)。针对该框架下，子代候选解与种群所有候选解进行 Pareto 占优比较时，存在没有推进候选解集朝着最优解集演化，反而消耗过多的计算资源及确定候选解前沿等级困难的问题，提出基于计算资源动态分配的高效非支配排序策略。实验表明，该基于该策略的 MOSTA/P 算法可以快速有效求解 MOP 和 F 测试问题。

最后，提出基于分解的多目标状态转移算法框架，MOSTA/D (Multi-objective state transition algorithm based on decomposition)。针对该框架下存在候选解与权重向量匹配不当的问题，提出了基于匹配度的修正切比雪夫聚合函数，并证明该聚合函数与传统切比雪夫聚合函数具有等价性。实验表明，该分解方法对于算法收敛性和解集的分布性都有一定程度的提升，该算法在求解 MOP 和 F 测试集函数上具有较好的性能。

论文共计图 41 幅，表 5 个，参考文献 80 篇。

关键词：多目标优化；状态转移算法；Pareto 占优；动态资源分配；高效非支配排序；匹配度；修正切比雪夫聚合函数

分类号：TP18

Multi-objective optimization method based on state transition algorithm

Abstract: Multi-objective optimization optimizes several confilict objectives to obtain optimal trade-off solution set. Multi-objective method based on intelligent optimization algorithms has been widely studied due to its tractablity and practicability. State transition algorithm which is a novel intelligent algorithm has less research in the field of multi-objective optimization. The paper studies state transition algorithm based multi-objective method and the main contributions of paper are shown as follows:

Firstly, in order to clarify how the search performance of state transition operators affect the convergence performance of multi-objective state transition algorithm, the related experiments are designed based on GD performance matrix. The effects of the single and combined use of the operators and the parameters on the convergence of the algorithm are analyzed. It is found that under the premise of the same selection strategy, expand transformation operator is suitable for gennerating candidate solution set alone and translation transformation operator is suitable for combination with other transformation operators to make the algorithm have better convergence. Rotation transformation operator and axision transformation operator have positive effects on the convergence of the algorithm, whether they are used alone or coordinate with other operators to generate candidate solutions set.

Secondly, a new multi-objective state transition algorithm based on Pareto dominance (MOSTA/P) is proposed. The algorithm considers that when offspring solutions compared with each other, some offspring solutions do not make population evolve towards the optimal solution set and it costs too much computing resources. Then, an efficient non-dominated sorting strategy based on dynamic computing resources allocation is proposed to solve this problem. The experimental results demonstrate that MOSTA/P is an effective algorithm for solving MOP and F series benchmark test functions.

Finally, a new multi-objective state transition algorithm based on decomposition is proposed. The algorithm considers the influence of match degree between candidate solutions and weight vectors. A new modified

Chebyshev aggeration function based on match degree is proposed and it is equivalent to the traditional Chebyshev decomposition method. The experimental results demonstrate that MOSTA/P is also an effective algorithm for solving MOP and F series benchmark test functions.

There are 41 figures, 5 tables and 80 references.

Keywords: multi-objective optimization；state transition algorithm；Pareto domination; dynamic computing resources allocation; efficient non-dominated sorting; matching degree; modified Chebyshev aggeration function

Classification: TP18

目　录

1 绪 论

在实际生活中，人们希望以较低的成本生产出较高质量的产品，或希望以较低的价格购买品质较优的商品。该类问题经过建模，可抽象为多目标优化问题。多目标优化问题包含多个目标函数，需要将其同时优化，使得决策变量在满足约束限制的条件下，使得这些目标达到最小（大）。与单目标优化问题不同，多目标优化问题的多个目标之间是相互冲突的，因而这些目标不能同时达到最优，即不存在一个候选解使得所有的目标达到最优[1]。一个候选解可能在其中一个目标函数上较好，而在另一个目标函数上较差，故候选解之间在全部的目标函数上难以比较[2]；同时，在比较过程中也会消耗大量的计算资源，致使求解时间过长。因此，快速求解多目标优化问题存在一定的困难。智能优化算法在求解多目标优化问题上，易于实现且操作性强，已经成为求解多目标优化问题的一种重要的方法。如何设计多目标智能优化算法，使其在较少的时间内快速求解多目标优化问题，并获得具有分布性和收敛性良好的帕累托最优解集，是多目标智能优化算法设计上的难点。此外，实际生活中存在很多的多目标优化问题，实际工程优化的需要推动着在多目标优化领域进行更加深入的研究和探讨[3]。

1.1 研究背景及意义

在电力传输中，希望电网传输的有功功率损失、碳排放量和总燃料成本，这三目标最小[4]。在电磁发射器设计上，希望设计出有用功率最大化。发射能量最大化的发射器[5]。在烯烃生产问题上，希望烯烃的产率和选择性最大化[6]；在镁粉生产中，希望镁粉产量和品位达到最大[7]。在污水处理过程中，希望在此过程中资源消耗尽可能降低，最终排放的污水质量指数尽可能小[8]。由此可以看出，实际工程中面临的优化问题是多目标优化问题。众多实际工程多目标优化问题的求解需要，使得研究一种可以快速有效求解多目标优化问题的多目标优化方法极为迫切[9-11]。

多目标优化问题需要优化多个目标，这些目标之间存在内在的矛盾，候选解对应的多个目标函数之间不能直接比较。传统的求解方法是根据经验，对每个目标设立 $(0,1)$ 之间的权重值，且所有目标的权重和为 1，将一个多目标优化问题转化为在一个固定权重向量下，多个目标的权重之和最小的单目标优化问题[12]。这样，通过求解该单目标优化问题，就能获得该权重向量下的一个最优解。然而，要想获得反映目标函数之间权衡的多个解，就必须改变目标函数的权重，得到多个不同的目标函数权重之和的优化问题，进行多次求解。近年来，随着智能计算的兴起，多种多目标智能优化算法已经被提出，并广泛应用于求解实际多目标优化问题[13]。利用智能优化算法求解多目标优化问题的优势在于求解一次多目标

优化问题，就可以得到针对所有目标相互权衡之下的一组最优解集，且对于所优化目标函数诸如可导，连续等数学特性没有明确的要求，成为求解多目标优化问题的一种主要方式[14-15]。

多目标智能优化算法的核心思想是通过设计智能启发式算子在决策空间内进行全局和局部搜索，产生候选解，设计合理的选择策略和终止条件，对候选解进行选择，最终获得在目标函数空间分布较为均匀，在各个目标上权衡的最优解集。在多目标优化算法的设计过程中，面临的难点如下：如何设计启发式算子，保证其全局搜索和局部搜索的能力，如何设计选择策略使得候选解朝着 Pareto 最优解方向进行演化，使得设计出的算法在求解多目标优化问题时，获得分布较为均匀，收敛性较好的最优权衡解集。

目前提出的多种智能优化算法最常见的三种框架分别是基于 Pareto 占优框架、基于分解框架和基于指标框架；代表性的算法分别有 NSGAII[16]，MOEA/D[17]，IBEA[18]等。基于 Pareto 占优框架的算法主要在候选解产生策略和时间复杂度及候选解多样性维护策略上进行改进和创新[19]。基于分解框架的多目标智能优化算法主要通过分析和改进权重向量的产生方式，产生候选解的方式，分解聚合函数，邻域划分方式等做改进和创新[20]，基于指标框架的算法主要对指标进行改进和创新[21]。

状态转移算法[22]是近些年提出的一种针对单目标优化问题、具有快速性、可控性、全局性和保证最优性的智能优化算法。状态转移算法将最优化问题的一个解看成是搜索过程的一个状态，将候选解更新过程看成是状态转移过程。状态转移算子的设计通过状态空间表达式实现了统一的框架性描述。通过设计合理的状态转移矩阵，产生多种具有启发性的状态转移算子，承担着在决策空间内进行全局搜索和局部搜索的任务。通过控制状态转移算子的参数，可以调节其搜索的空间大小及几何形状，并且保证算法至少可以获得一个局部最优解。同时，该算子在搜索空间进行均匀采样，且搜索过程具有一定的启发性，避免在决策空间内进行穷举而浪费大量的时间。该算法与遗传算法[23]、粒子群优化算法[24]、差分进化算法[25]、人工蜂群算法[26]等多种经典单目标优化算法相比，可以更加快速地找到优化问题的最优解[27]。近些年，很多学者也对状态转移算法在约束优化，离散优化等方面进行了改进[28-31]。但是，状态转移算法在多目标优化领域的研究还不多，对多目标智能优化算法框架研究和分析还不够全面，没有设计出求解复杂问题的基于不同框架的多目标状态转移算法，仅仅针对某些特殊的实际的多目标优化问题给出了基于状态转移算法的解决方案[32-33]，尚不具备求解更复杂多目标优化问题的能力；因此，在多目标优化领域，亟待对状态转移算法进行研究，提出不同框架下的多目标状态转移算法。

针对上述问题，本课题将对状态转移算法在多目标优化领域进行研究，其意义在于：

1) 分析了四种状态转移算子对多目标优化算法收敛性的影响，为多目标状态转移算法的设计提供经验和指导；
2) 提出了基于计算资源动态分配的高效的非支配排策略(ENS-DRA)，有效减少了候选解比较次数和算法的时间复杂度，并提出一种有效的基于 Pareto 的多目标状态转移算法（MOSTA/P）；
3) 定义了权重向量和候选解匹配度的概念，提出基于匹配度的修正切比雪夫分解聚合函数，并提出另一种有效的基于分解的多目标状态转移算法（MOSTA/D）。

1.2 多目标优化算法研究现状

本节以多目标智能优化算法的不同框架做为分类依据，将多目标优化算法分为三类，对多目标优化算法的研究现状进行归纳和整理，为后续研究提供基础和参考。

1) 基于 Pareto 占优的多目标优化算法

基于 Pareto 占优的多目标优化算法是利用智能优化算子产生候选解，利用 Pareto 占优的概念，对所有候选解进行比较，选择出非占优解。同时，还需要设计一定的多样性维护策略，将处于密集区域的解剔除，使得非占优解尽可能的分布在空间内。

许多算法对多目标优化算法的搜索过程进行研究，在多目标优化算法中，引入局部搜索，避免算法过早地收敛到 Pareto 前沿算法。IM-MOGLS 算法[34]是第一个多目标遗传局部算法，它将多个目标的加权和函数作为适应度函数，并依此选择一组父代候选解，利用交叉变异产生新的候选解，之后利用局部搜索产生候选解使的候选解朝着适应度函数最大的方向进化，并成功解决了某些 0-1 背包问题。C-MOGLS[35]结合了网格多目标遗传算法与局部搜索，利用多个目标的权重之和作为适应度函数。近些年来，NSLS[36]算法也被提出，该算法将非支配排序与局部搜索结合起来，该算法受参数的影响较小，相较于其他算法时间复杂度较低，但是对于某些问题容易陷入局部前沿面，因此还需要对局部搜索策略的设计方法进行更加深入的研究。

在候选解的比较过程中，基于该框架的多目标优化算法利用 Pareto 占优进行比较，为了在较小时间复杂度内尽快地确定非占优解，许多非占优排序机制被提出来。NSGA[37]是一种设计了非支配排序的机制，非占优排序根据候选解之间相互占优的情况，先选择出种群内的非占优解，作为第一前沿的解，再比较种群内

其他解，此时得到的非占优解为第二前沿的解，以此类推，为候选解指定其 Pareto 前沿等级。该机制的时间复杂度为$O(MN^3)$，空间复杂度为$O(N)$。此非支配排序方法计算量大，种群数量越大时，该支配方法的时间复杂度越大。为了减少比较次数，从而减少时间复杂度，在 NSGAII[16]算法中，提出了快速非支配排序的机制，使得算法复杂度复杂度降低至$O(MN^2)$。该策略通过记录候选解 p 被其他候选解占优的个数n_p和被 p 占优的解S_p，根据n_p对候选解指定对应的非支配等级，但该方法牺牲了空间复杂度，利用大量的空间存储变量，空间复杂度较高。Jensen 等人提出一种基于递归的 Jensen 排序方法[38]，该排序基于分治策略，对候选解进行非支配排序，对于两目标优化问题，时间复杂度降低至$O(MN\log N)$，对于多于两目标的优化问题，其求解的时间复杂度为$O(N\log^{M-1}N)$，该方法将随着目标函数个数的增多，时间复杂度呈指数性逐渐增长。Tang 等人提出了基于擂台赛法则的比较策略[39]，首先从种群中随机选择一个解作为冠军，其他候选解和冠军解进行比较，若其他候选解优于冠军解，则其他候选解作为冠军解，继续和其他候选解比较，该算法时间复杂度为$O(MN^2)$，空间复杂度为$O(N)$；整体说来，该方法优于前述的两种排序方法。此外，爬山排序及演绎排序[40]，快速排序[41]等非支配排序方法也相继被提出来减少候选解比较的时间复杂度。Zhang[42]等于 2015 年提出高效非支配排序 (ENS)方法，在最差情况下，该方法时间复杂度为$O(MN^2)$，空间复杂度仅为$O(1)$，大大降低了算法运行时间和变量存储消耗的资源空间。

基于 Pareto 占优框架的多目标优化还需设计多样性维护策略，来保证非占优解集的分布尽可能均匀。通过对个体是否处于密集区域进行判断，保留处于稀疏区域的非支配解。NSGA[36]通过共享小生境技术，设计共享函数，得到个体共享度和共享适应度函数值，判断出个体是否处于密集区域。SPEA[43]通过计算个体之间的欧式距离，对个体进行聚类，根据聚类结果选择具有代表性的解，实现解多样性的选择。上述多样性维护策略都需要指定参数，候选解选择结果受参数影响较大。NAGAII[16]算法设计了拥挤距离比较算子对候选解集的多样性进行维护。该算子定义了拥挤度的概念，将由某个个体的周围个体组成立方体的边长的平均值定义为该个体的拥挤度。个体的拥挤度越大，说明个体与其他个体的距离越远。该算法先利用非支配快速排序，再利用拥挤距离算子来选择候选解，先考虑候选解的前沿等级，再考虑候选解的拥挤度。该方法的优势是不包含人为参数，避免了参数大小的选择对候选解选择的影响。在文献[44]中，设计了分布性加强模块。首先将种群中全部个体映射在超平面上，对超平面进行均匀划分，将全部的候选解进行聚类，通过计算每一个分区中的聚类中心个数来判断当前种群的分布是否均匀。当前种群分布不均匀时，分布性加强模块被激活，利用聚类子种群中，拥

挤距离大的个体进行局部搜索，保证每一分区都有一定数量的个体。在文献[45]中，基于数据场的理论，利用数据场的势能对候选解的拥挤度进行判断，根据势能信息进行候选解的选择。

2) 基于分解的多目标优化算法

基于分解框架的多目标算法最早起源于 C-MOGA 算法[46]，Zhang 提出的 MOEA/D[47]继承了 C-MOGA 基于分解的框架，并在分解策略及选择策略上做了创新和改进，成为该框架下代表性算法。MOEA/D 算法主要利用均匀的权重向量，通过分解聚合函数将多目标优化问题分解为多个等价单目标优化子问题进行求解。候选解仅仅与其处于同一邻居关系的候选解比较，从而很好保证了种群的多样性和快速收敛性。常用分解方法有，权重和（Weighted Sum，WS）[48]，切比雪夫方法（Tchebycheff，TCH）[49]和基于惩罚的边界交叉（Penalty-based Boundary Intersection，PBI）[50]。其中切比雪夫方法是 MOEA/D 最常用的方法。后续基于分解框架的多目标优化算法主要在权重向量的产生方式，更新邻域确定方式，分解聚合函数等方面进行改进和创新，为了提高算法的计算效率，对算法的计算资源的分配进行研究。

权重向量的产生方式对得到的最优解集是否分布均匀有着较大的影响。在权重向量产生方式上，起初是使用 simplex-lattice 设计思想[51]。通过在目标函数空间内以固定的间隔进行均匀采样，来获得分布均匀的权重向量。然而该方式在求解三个及更多目标问题上产生的权重向量不是很均匀，且种群大小一般不能随意设定。文献[52]中，提出了基于方向角产生权重向量的方法，在 1/4 超球体上，利用向量夹角与半径和弦长的关系建立了权重向量选择模型，经过多次随机产生权重向量并选择更新，最终获得了分布均匀的方向向量。该方法虽然可以产生较为均匀的权重向量，但是时间复杂度很高。在 MOEA/AWA 算法中[53]，基于权重向量和其子问题的理想解之间的对应几何关系，提出 WS 变换，使得均匀采样的获得的权重向量映射到目标函数函数空间也是均匀的。

在分解聚合函数上，最初由在 MOEA/D 引入的权重和法，切比雪夫法和基于惩罚的边界交叉法，对于最小化多目标优化问题上，这些聚合函数采用的理想参考点都是各个目标最小值组成的。而在后续研究中发现，采用该点可以促进候选解朝着 Pareto 最优解的方向收敛，但是对于算法最终获得解集的广泛分布有一定的影响。采用各个目标最小值组成的理想向量，非占优解集中于中部，而边界较为稀疏，为此，专家学者采用种群候选解最大的目标函数组成理想参考点，距离该点越远，表明候选解越好，并提出翻转切比雪夫和翻转 PBI 分解方法[54]。更进一步，专家发现使用目标函数最大值作为理想参考点可以得到较多处于边界上的非支配解，而处于前沿中部的解相比之下有所减少。在算法 MOEA/D-MR[54]

中利用两者相互互补的特性，使用两个种群，将两种理想参考点结合使用，提出了基于多参考点的多种群协同进化的多目标优化算法。此外，还有专家学者使用在分解聚合函数中使用参考点集合而不是单一参考点，进行目标函数的分解。针对 PBI 聚合函数需要事先确定参数，学者又提出参数线性变化的自适应 PBI 分解方法[56]。后来，专家又提出 SPS 策略[57]，根据几个固定的权重向量，参数θ进行动态变化；马小亮等通过将切比雪夫分解函数的特性与最优解的几何特性关联起来，提出基于权重向量L_p范数约束的修正切比雪夫方法[58]。此外，角罚函数距离(APD)的分解方法在文献[59]中提出，其思想与 PBI 方法类似，但其通过计算候选解和理想点之间的角度对候选解的多样性进行量化，且其参数进行自适应变化，实验结果证明，该方法在求解多目标优化问题上鲁棒性较强。

考虑到某些实际问题的求解中，对于某一区域非占优解的搜索可能比其他非占优解困难，每个子问题消耗同等的计算资源就会十分浪费。为此，大量学者对基于分解框架的多目标的计算资源分配做了大量研究。张等人提出了动态资源分配的 MOEA/D-DRA[60]算法，通过定义一个特征函数来表征子问题求解的优化率，并在计算过程中，选择优化率较高的子问题进行进化，减少了算法盲目计算消耗的过多计算资源。更进一步，张青富提出 MOEA/D-AWA 算法[53]，该算法采用自适应权重向量调整策略，融合 MOEA/D-DRA 提出的动态资源分配的概念，先在预设的权重向量下进行搜索，待算法收敛到一定程度，开始第二阶段，采用自适应权重向量，移除处于密集区域的子问题，在稀疏区域添加子问题进行求解，在测试函数的求解上显示出了较为突出的性能。

此外，另外一种基于分解的多目标优化算法也逐渐受到学者的广泛关注。此类算法将多目标优化问题分解成为多个等价的多目标优化问题进行求解，其中每个多目标的寻优空间，都是原问题寻优空间的子集。该种思想，最先由 MOEA/D-M2M 算法中提出[61]。该算法将原问题分解成为多个多目标优化子问题，采用多种群策略，对子问题进行并行优化求解。在某些标准测试函数上，该算法的求解优于 MOEA/D-DE[51]和 NSGA-II 算法。

3) 基于指标的多目标优化算法

基于指标框架的多目标优化算法一类常见的算法。该类算法利用多目标优化中的性能指标作为候选解的适应度函数。基于该框架的算法不需要额外的多样性维护机制，同时还一定程度上反应了决策者的偏好。常用的指标为 HV 指标[61]，ε-指标[62]，S 矩阵[64]，代表性算法为 IBEA[63]，SMS-EMOA[65]等。基于指标的多目标算法在计算候选解的指标时，需要消耗大量的计算资源和时间资源，随着基于其他框架算法的不断改进，基于指标框架的优化算法研究热度较其他框架下的算法，研究热度已经减弱。

1.3 论文主要内容及安排

在国家自然科学基金项目课题“有色冶金级联反应器不确定动态优化方法”（61873285）、“基于分布式状态转移算法的有色冶金过程系统辨识方法研究”（61503416）、中南大学创新驱动计划“流程工业过程数据建模与分布式优化方法研究”（2018CX012）和中南大学研究生科研创新项目（2017zzts492）的资助下，开展了基于状态转移算法的多目标优化方法的研究工作，主要包括：首先，对目前已有的多目标优化方法进行大量的文献调研，主要以多目标智能优化算法为研究对象，以多目标智能优化算法框架为分类依据，对算法的研究进展进行了调研；其次，概括总结了多目标优化的相关概念及算法框架，及常用的标准测试函数及评价指标，常用的分解函数等，为后续研究奠定基础。

状态转移算法是由周晓君[22]等出的一种智能优化算法，它以控制理论的离散时间状态空间表达式为框架，利用状态转移算子在决策空间内进行全局和局部搜索，采用贪婪准则，最终获得单目标优化问题的最优解或者近似最优解，具有全局性，快速性，收敛性，可控性，最优性的特点。通过对状态转移算法原理的不断深入研究，与经典优化算法做对比，展现出良好的快速收敛性能。通过对其进一步分析，提出状态转移算法的改进版本[66]，相比与之前版本，更加快速有效地，在较少时间内完成对问题的求解。值得一提的是，其状态转移算法设计的状态变换算子具有可控性，可以通过控制算子参数来控制算子的搜索空间和几何形状。为了进一步深入状态转移算法在多目标智能优化领域的研究，本论文对状态转移算子的搜索性能做了研究，旨在探讨状态转移算子对多目标优化算法收敛性的影响，为多目标状态转移算法的设计提供搜索方面的研究经验和依据。此外，在两种现有的框架下，本论文提出基于 Pareto 占优和基于分解的多目标状态转移算法，融合了现有较新的研究成果并进行了创新。本论文共包括六章内容，各章节内容及关系如图 1-1 所示。

第一章为绪论，该章以现有多目标优化算法框架为分类依据，论述了不同框架下多目标优化算法的研究现状及进展。状态转移算法作为一种智能的，具有快速性、可控性及能够保证最优性的算法，尚未在多目标优化领域进行深入研究。本章基于目前智能优化算法在求解多目标优化问题的研究现状，给出了设计多目标状态转移算法的必要性和可行性。

第二章，多目标优化概述。此章给出了多目标优化基础概念及常见的算法框架、常用的测试函数性能评估矩阵；该章作为论文整体的研究基础，为其他章节的内容起到基础支撑作用。

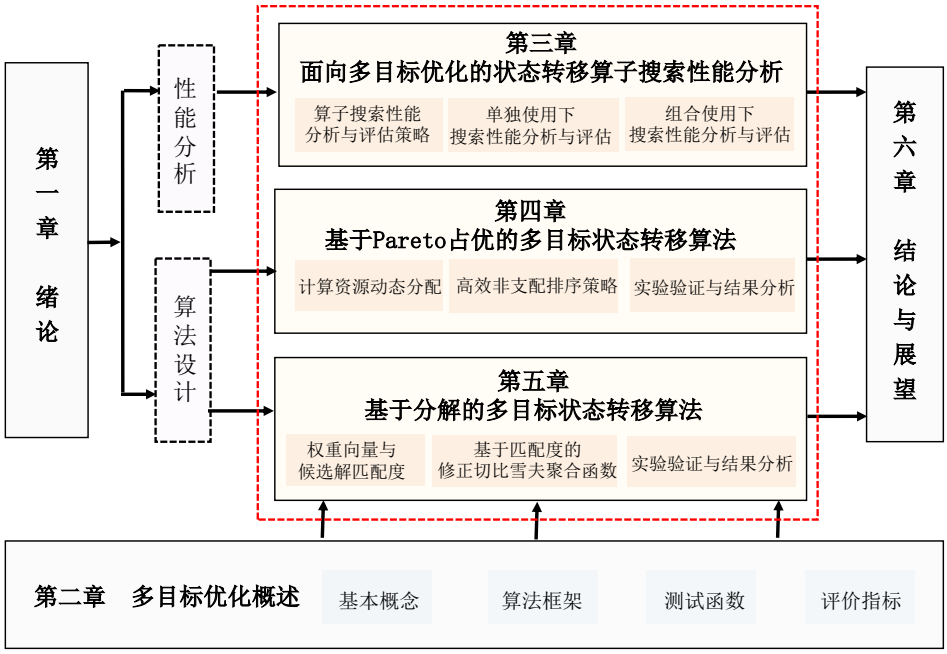


图 1-1 论文框架与结构

第三章，面向多目标优化的状态转移算子性能分析。本章内容主要是利用多目标性能矩阵，对状态转移算子的搜索性能进行定量分析，为接下来提出不同框架下的多目标状态转移算法做基础和铺垫。对于一个随机智能搜索算法而言，良好的搜索性能是算法找到最优解集的前提和保证。本章设计多组对比实验，通过固定初始解集和最大迭代次数，利用相同的选择策略对解集进行更新，通过计算每次迭代，性能矩阵的变化情况，来反映状态转移算子的搜索性能；在同样的选择方式下，分析该算子对多目标优化算法收敛性的影响，同时探究状态转移算子的最佳使用方式，为下一步设计算法在搜索过程提供经验和依据。

第四章，基于 Pareto 占优的多目标状态转移算法设计。本文基于第三章对状态转移算子搜索性能的分析的结果，基于 Pareto 占优算法的框架，提出了一种新型的基于 Pareto 占优的多目标状态转移算法。该算法采用 Pareto 占优为比较准则，采用动态资源分配的高效非支配排序策略为候选解的前沿等级进行划分；采用拥挤距离的多样性维护算子，剔除处于密集区域的候选解，保证解集分布的均匀性。经过测试函数和性能指标的验证，基于 Pareto 占优的多目标状态转移算法是有效的。

第五章，基于分解的多目标状态转移设计。本文同样基于第三章对状态转移算子搜索性能的分析的结果，基于分解的框架，提出了另外一种基于分解的多目标状态转移算法。基于分解框架的多目标优化算法一个重要核心是分解聚合函数的设计，然而现有分解聚合函数并未考虑权重向量和候选解的匹配程度，以此为出发点，本章在切比雪夫分解函数的基础上，通过利用权重向量，候选解目标函数值与理想点组成向量之间的夹角余弦值，表征候选解和权重向量的匹配关系，

提出一种基于匹配度的修正切比雪夫分解函数，可以证明，该函数与切比雪夫是等价的。此外，本章提出的算法还采用了动态邻域，强化搜索等策略，加快了算法更新过程。经过多种测试函数和性能指标的验证，表明了本章提出的分解方法的合理性和算法求解多目标优化问题的有效性。

第六章，总结与展望。本章论文的整体研究进行了整理。本论文在详细调研多目标优化算法的研究现状，多目标优化相关理论，对状态转移算子的搜索性能对算法收敛性的影响进行研究，对比了状态转移算子不同使用方式的搜索性能差异；更进一步，提出了两种不同框架下的多目标状态转移算法，具有一定的研究价值和意义。同时，对未来多目标状态转移算法的研究方向进行了展望。未来将在约束多目标优化等领域进行更多深入的研究。

2 多目标优化概述

2.1 基本概念

多目标优化问题的可以抽象为如(2-1)所示的数学形式：

$$\begin{aligned} &\min \ \ \boldsymbol{F}(\boldsymbol{x})=[f_1(\boldsymbol{x}), f_2(\boldsymbol{x}),..., f_M(\boldsymbol{x})] \\ &\text{s.t.} \quad \boldsymbol{x}\in\Omega\subseteq\mathbb{R}^n \end{aligned} \tag{2-1}$$

其中，$\boldsymbol{x}=(x_1,x_2,...x_n)$ 是 n 维决策变量，Ω 是决策变量的寻优范围。$f_1(\boldsymbol{x}), f_2(\boldsymbol{x}),..., f_M(\boldsymbol{x})$ 是 M 个待优化的目标函数，其中各个目标相互矛盾，一个候选解可能在某一目标上较好而在另外几个目标上较差，因而候选解无法直接比较。在优化过程中，在某些目标上朝着最优化方向的改善的同时，其他目标的最优性会被削弱，求解多目标优化问题最终得到的是优化目标之间相互权衡且相对最优的 Pareto 最优解集。为了更好地理解多目标优化，本节给出多目标优化几个重要且常见的基本定义[67-68]：

定义 1 (Pareto 支配). 对于任意的两个候选解 $\boldsymbol{x}_a,\boldsymbol{x}_b$，称 $\boldsymbol{x}_a$ 支配 $\boldsymbol{x}_b$，当且仅当式(2-2)成立，记作 $\boldsymbol{x}_a\prec\boldsymbol{x}_b$。

$$\{\forall i\in(1,2,...,M): f_i(\boldsymbol{x}_a)\le f_i(\boldsymbol{x}_b)\}\wedge\{\exists j\in(1,2,...,M): f_j(\boldsymbol{x}_a)< f_j(\boldsymbol{x}_b)\} \tag{2-2}$$

即候选解 $\boldsymbol{x}_a$ 的目标函数值全部小于等于 $\boldsymbol{x}_b$ 的目标函数值，且 $\boldsymbol{x}_a$ 至少存在一个目标函数值小于 $\boldsymbol{x}_b$ 的目标函数值。

定义 2 (Pareto 最优解). $\boldsymbol{x}*$ 为 Pareto 最优解(或非支配解)，当且仅当在决策空间内，不存在其他候选解 $\boldsymbol{x}$ 支配它，其数学表达式如(2-3)所示：

$$\neg\exists\boldsymbol{x}\in\Omega:\boldsymbol{x}\prec\boldsymbol{x}^* \tag{2-3}$$

定义 3 (Pareto 最优解集). 所有 Pareto 最优解 $\boldsymbol{x}*$ 构成的集合 $\boldsymbol{PS}$ 为 Pareto 最优解集，定义如式(2-4)：

$$\boldsymbol{PS}=\{\boldsymbol{x}^*\in\Omega\,|\,\neg\exists\boldsymbol{x}\in\Omega:\boldsymbol{x}\prec\boldsymbol{x}^*\} \tag{2-4}$$

定义 4 (Pareto 前沿面). Pareto 最优解集中所有 Pareto 最优解在目标空间的映射形成的解集，组成 Pareto 最优解的目标向量值集合。该集合为 Pareto 最优前沿，定义如式(2-5)：

$$\boldsymbol{PF}=\{\boldsymbol{F}(\boldsymbol{x})\,|\,\boldsymbol{x}\in\boldsymbol{PS}\} \tag{2-5}$$

简而言之，Pareto 最优解是最优候选解的集合，而 Pareto 最优前沿是 Pareto 最优解对应目标函数值组成的集合。

为了更加详细的解释多目标优化的相关概念，用如下图对 Pareto 占优等相关概念进行更加直观化的阐述。如图 2-1 所示。标记为 $A-H$（除 F）的圆圈代表 7 个候选解，图中表示了这些解在目标函数空间内的位置。以候选解 A 和 B 为例，可以看出候选解 A 的第一个目标函数值小于候选解 B 的，而其另一个目标函数大

于候选解 B 的，故 A,B 不能直接比较，A 和 B 是相互不占优的。针对候选解 C 和 G ,可以看出候选解 C 的两个目标函数君小于候选解 G 的，满足 Pareto 占优的概念，故候选解 C 是占优候选解 G，候选解 C 为非占优解；若在整个搜索空间内，候选解 A,B,C,D,E 对于其它候选解都占优，则候选解 A,B,C,D,E 称为 Pareto 最优解，他们共同组成了 Pareto 最优解集，其目标函数的集合 $F(A),F(B)$ $F(C),F(D)F(E)$ 形成了 Pareto 前沿。

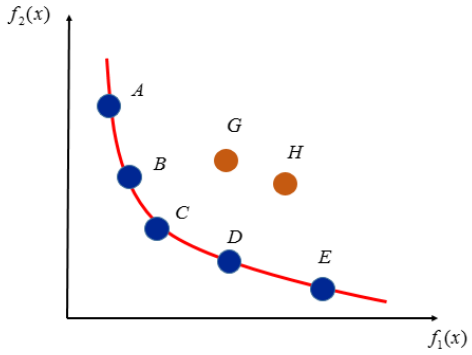


图 2-1 两个目标的 Pareto 最优前沿曲线

2.2 多目标优化算法的框架概述

本小节主要对目前常见的多目标优化算法框架进行简要介绍和说明，为后续研究提供必要的基础，其流程图如图 2-2 所示。

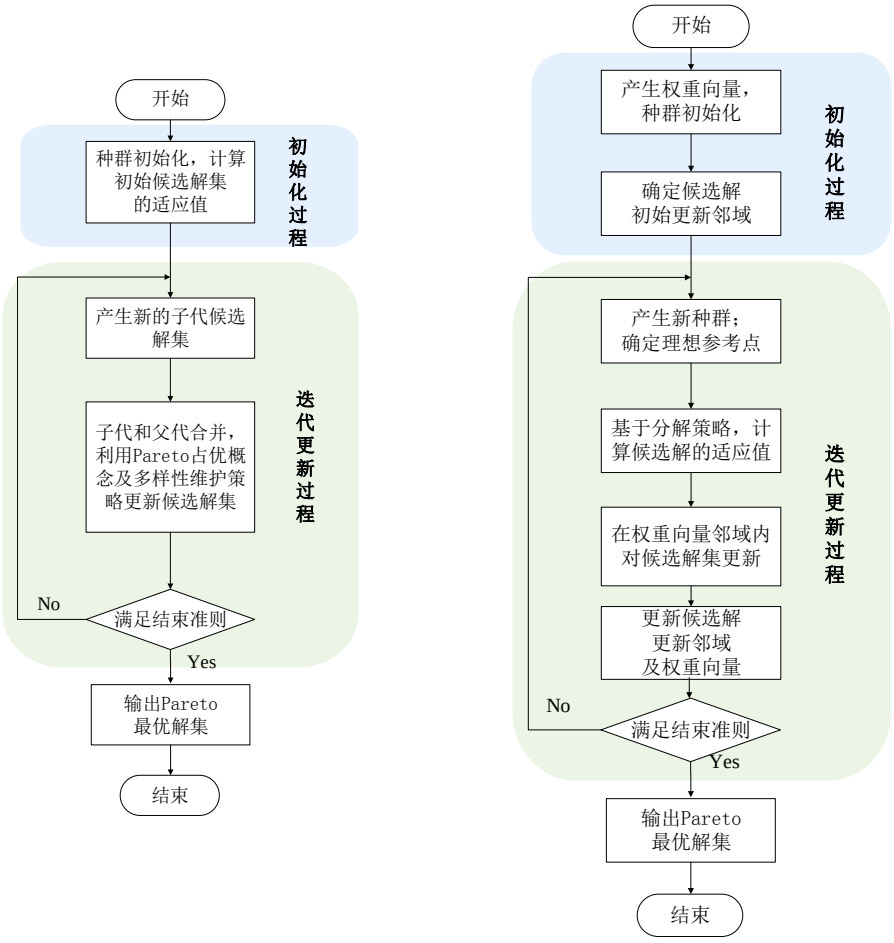
2.2.1 基于 Pareto 占优的多目标优化算法框架

基于 Pareto 占优的多目标优化算法流程可用图 2-2(a) 表示。该类算法以 Pareto 占优为比较准则的一类算法,其基本思想如下：首先通过随机算子产生候选解集，利用 Pareto 占优概念对候选解进行比较，选择出非占优的解，此外还需要设计一些多样性维护算子，从非占优解中剔除处于密集区域的解，保留分布较为均匀的解，使得候选解分布较为均匀。

2.2.2 基于分解的多目标优化算法框架

基于分解的多目标优化算法主要包括以下部分：权重向量生成，候选解集产生，理想参考点确定，分解函数设计这几个主要部分。该类算法首先采用一定的方式产生一定数量的权重向量，并根据权重向量几何关系等其他因素确定更新邻域范围；其次，利用智能随机搜索算子产生候选解集，根据候选解集的目标函数确定理想参考点；最后利用权重向量，理想参考点及分解方法，对每个候选解进行评估，评估较优的候选解不断在其更新邻域范围内替换评估较差的解，直至算法终止。权重向量在目标标函数空间是否均匀，更新邻域范围大小设置是否合适，

对最终候选解集中的解，能否均匀分布有着较大的影响。分解策略不同对于候选解的选择也有较大的影响，基于分解的多目标算法流程可用图 2-2(b)表示。



(a) 基于 Pareto 占优的多目标优化算法流程图

(b) 基于分解的多目标优化算法流程图

图 2-2 基于两种典型框架的多目标优化算法流程图

其中，分解聚合函数的设计是基于分解框架算法的核心，是权重向量 $\boldsymbol{\lambda}$，理想参考点 $\boldsymbol{z}^*$，候选解 $\boldsymbol{x}$ 的函数，其中，$\boldsymbol{\lambda}=(\lambda_1,\lambda_2,...\lambda_M)$，$\boldsymbol{z}^*=(z_1^*,z_2^*,...,z_M^*)$ 为已知候选解各个目标的最小值，最为典型的几种分解函数如下[47]：

1) 权重求和分解函数：

$$\max\ g^{ws}(\boldsymbol{x},\boldsymbol{\lambda})=\sum_{i=1}^{M}\lambda_i f_i(\boldsymbol{x}) \tag{2-6}$$

权重求和法是对每一个目标函数施加一个权重，对所有的目标函数求权重和，多组权重向量就构成多个单目标优化问题；权重求和方法具有明确的几何意义，其函数值等于候选解目标函数向量在权重方向上的投影。

2) 切比雪夫分解函数：

$$\min\ g^{tc}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)=\max_{1\le i\le M}\left\{\lambda_i\mid f_i(\boldsymbol{x})-z_i^*\mid\right\} \tag{2-7}$$

切比雪夫分解利用参考点和权重向量，通过最小化单个目标在权重下与理想点之间的差距，使得候选解不断进化。其它使用较多的切比雪夫分解方法变种表示如下：

$$\min_{x\in\Omega} g(\mathrm{F}(\boldsymbol{x})\mid\boldsymbol{\lambda},\boldsymbol{z}^*)=\max_{1\le i\le m}\left\{\frac{f_i(\boldsymbol{x})-z_i^*}{\lambda_i}\right\} \tag{2-8}$$

3) 惩罚边界法：

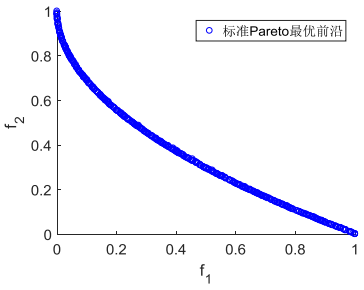
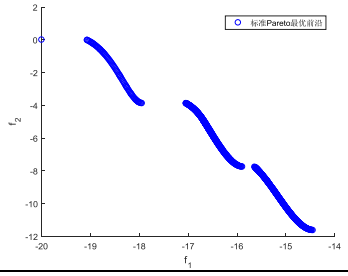
$$\min\ g^{\mathrm{PBI}}(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{z}^*)=d_1+\theta d_2 \tag{2-9}$$

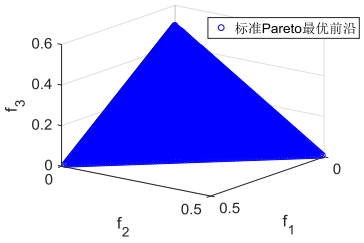
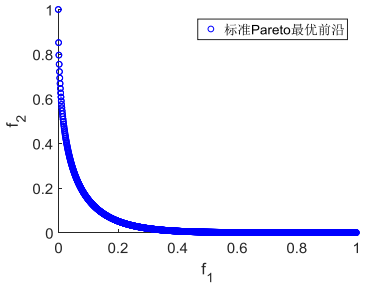
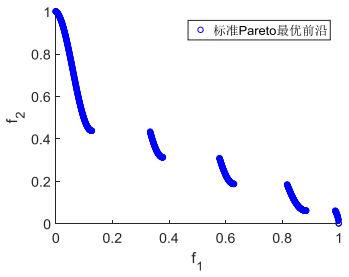
其中，$d_1=\left\|(\boldsymbol{z}^*-\boldsymbol{F}(\boldsymbol{x}))^T\boldsymbol{\lambda}\right\|/\left\|\boldsymbol{\lambda}\right\|$，$d_2=\left\|\boldsymbol{F}(\boldsymbol{x})-(\boldsymbol{z}^*-d_1\boldsymbol{\lambda})\right\|$，惩罚边界法包含一个参数$\theta$，$d_1$用来表征候选解的目标函数与理想参考点组成的向量在权重向量上的投影距离，d_1越小，表明候选解越接近 Pareto 前沿；d_2表示候选解目标函数组成的点，到权重与理想参考点组成直线的垂直距离，其值越小，说明该候选解与理想参考点组成的向量与权重向量的夹角越小；θ的取值决定算法更偏向收敛性还是更偏向分布性；一般而言，$\theta=5$较为合适。θ的取值对算法的收敛性和分布性有较大的影响，一些在专家学者提出自适应θ参数的 PBI 聚合函数分解策略。

2.3 标准测试函数

为了评估算法性能，学者设计出许多标准测试函数[69-73]，来测试多目标优化算法的分布性和收敛性，本小节对于本论文使用到的标准测试函数及其前沿进行归纳和整理。

表 2-1 测试函数及其标准前沿

测试函数	函数描述	最优前沿
ZDT1	$f_1(x)=x_1$ $f_2(x)=g(x)(1-\sqrt{f_1/g(x)})$ $g(x)=1+9\sum_{i=2}^{n}x_i/(n-1)$ $n=30;x_i\in[0,1];i=1,...,n$	
KUR	$f_1(x)=\sum_{i=1}^{n-1}(-10\exp(-0.2\sqrt{x_i^2+x_{i+1}^2}))$ $f_2(x)=\sum_{i=1}^{n-1}(\lvert x_i^2\rvert^{0.8}+5\sin x_i^3)$ $n=3;x_i\in[-5,5];i=1,...,n$	

DTLZ1	$f_1(x)=\frac{1}{2}x_1x_2(1+g(x))$ $f_2(x)=\frac{1}{2}x_1(1-x_2)(1+g(x))$ $f_3(x)=\frac{1}{2}(1-x_1)(1+g(x))$ $g(x)=100\left(5+\sum(x_i-0.5)^2)-h(x)\right)$ $h(x)=\cos(20\pi(x_i-0.5))$ $n=7;x_i\in[0,1];i=1,...,n$ $\text{PF}:\sum_{i=1}^{M}f_i=0.5$	
F1	$f_1(x)=(1+g(x))x_1$ $f_2(x)=(1+g(x))(1-\sqrt{x_1})^5$ $g(x)=2\sin(0.5\pi x_1)(\text{n}-1+h(x))$ $h(x)=\sum_{i=2}^{n}(y_i^2-\cos(2\pi y_i))$ where $y_{i=2:n}=x_i-\sin(0.5\pi x_i)$ $x_i\in[0,1],i=1,...n;\ n=30$ $\text{PF}:\ f_2=(1-\sqrt{f_1})^5$ $\text{PS}:\ x_i=\sin(0.5\pi x_j),\ j=2,...n$	
F2	$f_1(x)=(1+g(x))(1-x_1)$ $f_2(x)=\frac{1}{2}(1+g(x))t(x)$ $g(x)=2\sin(0.5\pi x_1)h(x)$ $h(x)=\text{n}-1+\sum_{i=2}^{n}(y_i^2-\cos(2\pi y_i))$ $t(x)=x_1+\sqrt{x_1}\cos^2(4\pi x_1)$ where $y_{i=2:n}=x_i-\sin(0.5\pi x_i),$ $x_i\in[0,1],i=1,...n;\ n=30$ $\text{PS}:\ x_j=\sin(0.5\pi x_j),\ j=2,...n$	

F3

$$f_1(x) = (1+g(x))x_1$$

$$f_2(x) = \frac{1}{2}(1+g(x))t(x)$$

$$t(x) = 1 - x_1^{0.1} + (1-\sqrt{x_1})^2 \cos^2(3\pi x_1)$$

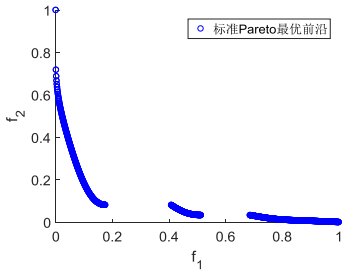
$$g(x) = 2\sin(0.5\pi x_1)h(x)$$

$$h(x) = \mathrm{n} - 1 + \sum_{i=2}^{n}(y_i^2 - \cos(2\pi y_i))$$

where $y_{i=2:n} = x_i - \sin(0.5\pi x_i)$

$x_i \in [0,1]; i = 1,...n;\ n = 30$

PS: $x_i = \sin(0.5\pi x_i),\ \forall x_i \in \mathrm{x_{II}}$



F4

$$f_1(x) = x_1(1+g(x))/\sqrt{x_2 x_3}$$

$$f_2(x) = x_2(1+g(x))/\sqrt{x_1 x_3}$$

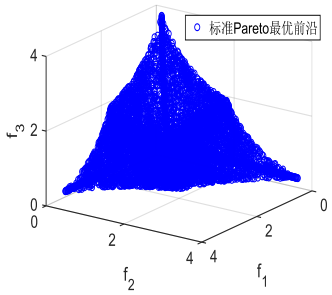
$$f_3(x) = x_3(1+g(x))/\sqrt{x_1 x_2}$$

$$g(x) = \sum_{i=4}^{n}(x_i - 2)^2$$

where $x_i \in [1,4], i = 1,...n;\ n = 30$

PF: $f_1 f_2 f_3 = 1$

PS: $x_j = 2,\ j = 3,...n$



F5

$$f_1(x) = ((1-x_1)x_2)(1+g(x))$$

$$f_2(x) = ((1-x_2)x_1)(1+g(x))$$

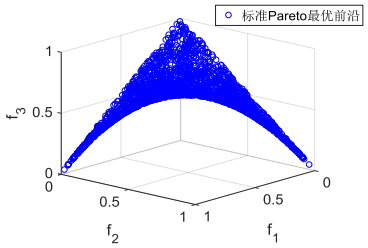
$$f_3(x) = (1-x_1-x_2+2x_1x_2)6x_3(1+g(x))$$

$$g(x) = \sum_{i=4}^{n}(x_i - 0.5)^2$$

where $x_i \in [1,4], i = 1,...n;\ n = 30$

PF: $f_3 = (1-f_1-f_2)^6$

PS: $x_j = 0.5,\ j = 3,...n$



F6

$$f_1(x) = \cos^4(0.5\pi x_1)\cos^4(0.5\pi x_2)$$

$$f_2(x) = \cos^4(0.5\pi x_1)\sin^4(0.5\pi x_2)$$

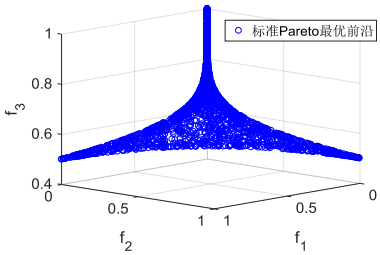
$$f_3(x) = \left(\frac{1+g(x)}{1+\cos^2(0.5\pi x_1)}\right)^{\frac{1}{1+g(x)}}$$

$$g(x) = \frac{1}{10}\sum_{i=3}^{n}(1+x_i^2 - \cos(2\pi x_i))$$

where $x_i \in [0,1], i = 1,...n; n = 30$

PF: $f_3(1+\sqrt{f_1}+\sqrt{f_2}) = 1$

PS: $x_j = 0,\ j = 3,...n$

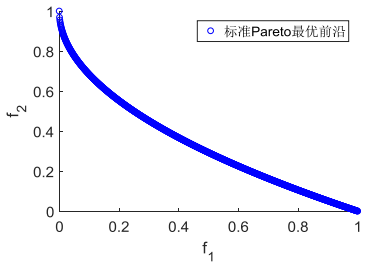


MOP1

$$f_1(x) = (1+g(x))x_1$$
$$f_2(x) = (1+g(x))(1-\sqrt{x_1})$$
$$g(x) = 2\sin(\pi x_1)\sum_{i=2}^{n}(-0.9t_i^2 + |t_i|^{0.6})$$
$$t_i = x_i - \sin(0.5\pi x_1)$$
where $x_i \in [0,1], i = 1,...n; n = 10$

PF: $f_2 = (1-\sqrt{f_1}), \quad 0 \le f_1 \le 1$

PS: $x_j = \sin(0.5\pi x_1); j = 2,...,n;$
$x_1 = 0,1.\}$

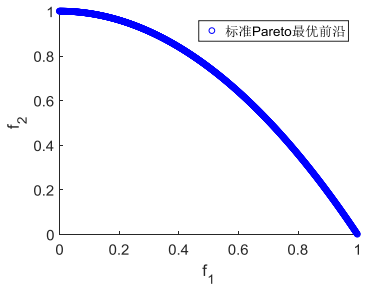


MOP2

$$f_1(x) = (1+g(x))x_1$$
$$f_2(x) = (1+g(x))(1-x_1^2)$$
$$g(x) = 10\sin(\pi x_1)\sum_{i=2}^{n}\frac{|t_i|}{1+e^{5|t_i|}}$$
$$t_i = x_i - \sin(0.5\pi x_1)$$
where $x_i \in [0,1], i = 1,...n; n = 10$

PF: $f_2 = 1 - f_1^2, \quad 0 \le f_1 \le 1$

PS: $x_j = \sin(0.5\pi x_1); j = 2,...,n;$
$x_1 = 0,1$

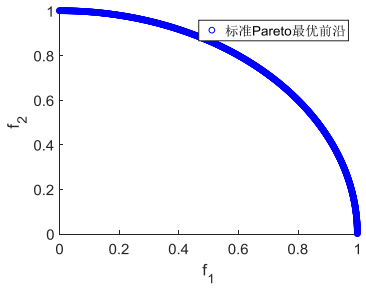


MOP3

$$f_1(x) = (1+g(x))\cos(\frac{\pi x_1}{2})$$
$$f_2(x) = (1+g(x))\sin(\frac{\pi x_1}{2})$$
$$g(x) = 10\sin\sin(\frac{\pi x_1}{2})\sum_{i=2}^{n}\frac{|t_i|}{1+e^{5|t_i|}}$$
$$t_i = x_i - \sin(0.5\pi x_1)$$
where $x_i \in [0,1], i = 1,...n; n = 10$

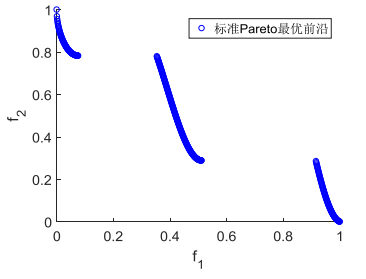
PF: $f_2 = \sqrt{1-f_1^2}, \quad 0 \le f_1 \le 1$

PS: $x_j = \sin(0.5\pi x_1); j = 2,...,n;$
$x_1 = 0,1$



MOP4

$$f_1(x) = (1+g(x))x_1$$
$$f_2(x) = (1+g(x))(1-x_1^{0.5}\cos^2(2\pi x_1))$$
$$g(x) = 10\sin(\pi x_1)\sum_{i=2}^{n}\frac{|t_i|}{1+e^{5|t_i|}}$$
$$t_i = x_i - \sin(0.5\pi x_1)$$
where $x_i \in [0,1], i = 1,...n; n = 10$



2.4 性能评价指标

多目标优化算法的设计目标是获得收敛至 Pareto 最优解且具有较好分布性的最优解集；然而，算法最终获得的解是否是 Pareto 最优解集中的解，其分布性是否良好，难以通过获得候选解集的目标函数判断，因此，众多专家学者设计了多种性能评价指标来表征算法获得解集的收敛性和分布性[75]。本节将常用的性能指标给予归纳和总结，这些指标也会在后续章节中用于算法之间的对比。假设 $\boldsymbol{P}^*$ 是理想 Pareto 参考解集， $\boldsymbol{A}$ 是算法获得的 Pareto 最优解集，$|\boldsymbol{A}|$表示解集 $\boldsymbol{A}$ 中解的数量，$\boldsymbol{p}$ 是参考解集 $\boldsymbol{P}^*$ 内的解，$\boldsymbol{x}$ 是解集 $\boldsymbol{A}$ 中的解, $d(\boldsymbol{p},\boldsymbol{x})$ 表示 $\boldsymbol{x}$ 与 $\boldsymbol{p}$ 的欧式几何距离。这些指标定义如下：

1) 世代距离（Generational Distance，GD）[76]: GD 指标可以表述如下：

$$\mathrm{GD}(\boldsymbol{A})=\frac{1}{|\boldsymbol{A}|}\sum_{\boldsymbol{x}\in\boldsymbol{A}}\min_{\boldsymbol{p}\in\boldsymbol{P}^*} d(\boldsymbol{p},\boldsymbol{x}) \tag{2-10}$$

GD 指标是计算解集 $\boldsymbol{A}$ 中的解与解集 $\boldsymbol{P}^*$ 中的解的最小距离的平均值，该值越小，表明两个解集越接近，算法收敛性越好。该指标适合算法获得的解集覆盖了全部前沿之后进行对比。

2) 反向世代距离 (Inverted Generational Distance，IGD)[77]:

IGD 可以表述如下：

$$\mathrm{IGD}(\boldsymbol{A})=\frac{1}{|\boldsymbol{P}^*|}\sum_{\boldsymbol{p}\in\boldsymbol{P}^*}\min_{\boldsymbol{x}\in\boldsymbol{A}} d(\boldsymbol{p},\boldsymbol{x}) \tag{2-11}$$

IGD 指标是计算参考解 $\boldsymbol{P}^*$ 中的解与解集 $\boldsymbol{A}$ 中的解 $\boldsymbol{x}$ 的最小距离，之后求取平均值得到。IGD 和 GD 统计的最小欧式几何距离的参考点不同，GD 以解集 $\boldsymbol{A}$ 为参考，而 IGD 以解集 $\boldsymbol{P}^*$ 为参考。IGD 可以综合反映解集 $\boldsymbol{A}$ 的分布性和收敛性，其值越小，解集 $\boldsymbol{A}$ 的收敛性和分布性越好。

3) 超体积指标（Hyper Volume, HV）[78]：

HV 指标需要设定一个参考点 $\boldsymbol{r}=(r_1,r_2,..,r_M)$，一般该向量的各个元素取值略大于解集 $\boldsymbol{A}$ 内所有候选解目标函数的最大值，即 $\boldsymbol{r}$ 可以被解集 $\boldsymbol{A}$ 中的解占优，解集 $\boldsymbol{A}$ 的 HV 可以表述如下：

$$HV(\boldsymbol{A},\boldsymbol{r})=volume(\bigcup_{\boldsymbol{x}\in\boldsymbol{A}}[f_1(\boldsymbol{x}),r_1]\times...[f_M(\boldsymbol{x}),r_m]) \tag{2-12}$$

HV 指标具有良好的几何特性，可以综合考虑算法的多样性和收敛性。该值越大，表明解集的分布性和收敛性越好。

4) 集合覆盖率（Set Coverage ， SC）：

假设 $\boldsymbol{A}$，$\boldsymbol{B}$ 分别是算法 1 和算法 2 获得获得的 Pareto 前沿面近似最优解集，集合覆盖率 $C(\boldsymbol{A},\boldsymbol{B})$ 表示解集 $\boldsymbol{B}$ 中的解被 $\boldsymbol{A}$ 中至少一个解支配的百分比，可以表

述如下；

$$\mathrm{C}(\boldsymbol{A},\boldsymbol{B})=\frac{|\{\boldsymbol{x}\in\boldsymbol{B}\,|\,\exists\boldsymbol{y}\in\boldsymbol{A}:\boldsymbol{y}\prec\boldsymbol{x}\}|}{|\boldsymbol{B}|} \tag{2-13}$$

其中， $\boldsymbol{y}$ 是解集 $\boldsymbol{A}$ 中的解， $\boldsymbol{x}$ 是解集 $\boldsymbol{B}$ 中的解；特别要说明的是， $C(\boldsymbol{A},\boldsymbol{B})$ 与 $C(\boldsymbol{B},\boldsymbol{A})$ 之和并不一定为 1。在解集 A 和解集 B 的比较中，如果 $C(\boldsymbol{A},\boldsymbol{B})$ 大于 $C(\boldsymbol{B},\boldsymbol{A})$，则解集 $\boldsymbol{A}$ 在某种程度上比解集 $\boldsymbol{B}$ 更好。该指标并不直接的反应解集分布性和收敛性，仅仅比较两个集合在 Pareto 占优概念上面的相对优劣。

2.5 本章小结

本章对多目标优化进行了概述和总结，详细阐述了多目标优化算法中 Pareto 占优框架和分解框架的主要流程。此外，本章还对常用的标准测试函数和性能评价指标做出了归纳和整理。在后续章节中，会利用本章涉及到的标准测试函数和性能评价指标对状态转移算子及所提出的多目标状态转移算法性能进行分析和比较。

3 面向多目标优化的状态转移算子搜索性能分析

在多目标智能优化算法的设计中，搜索算子搜索能力强弱，很大程度上决定了算法的整体运行效果。状态转移算法与其他智能优化算法相比，在求解单目标优化问题上显示出了较为优良的全局搜索能力。其中状态转移算子具有可控性，可以通过调节其参数，实现对其搜索空间大小的控制，是其他智能随机搜索算子不具备的特性之一。然而，状态转移算子在多目标优化领域的研究还不够深入。本章主要对状态转算子搜索性能进行分析，为后续多目标状态转移算法的设计提供支持和经验。

3.1 基本状态转移算法简介

状态转移算法[21]是近些年提出的一种新型智能随机型优化算法，该算法将优化问题的候选解作为一种状态，候选解进行更新的过程，认为是一种状态转移至另一种状态的过程。借助于状态空间的概念，其基本框架可描述如下：

$$\begin{cases} \boldsymbol{x}_{k+1} = \boldsymbol{A}_k \boldsymbol{x}_k + \boldsymbol{B}_k \boldsymbol{u}_k \\ y_{k+1} = f(\boldsymbol{x}_{k+1}) \end{cases} \tag{3-1}$$

其中，$\boldsymbol{x}_k \in R^n$ 对应着优化问题的一个解；$\boldsymbol{A}_k, \boldsymbol{B}_k \in R^{n\times n}$ 称为状态转移矩阵，$\boldsymbol{u}_k \in R^n$ 是关于当前候选解和历史候选解的函数；可以通过设计状态转移矩阵 $\boldsymbol{A}_k$，$\boldsymbol{B}_k$ 与 $\boldsymbol{u}_k$ 设计各种各样的状态变换算子；$f(\boldsymbol{x})$ 是代价函数或评价函数。状态转移算法的搜索依赖设计的状态转移算子实现，目前设计出的状态转移算子有：

1) 旋转变换 (Rotation Transformation)

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha \frac{1}{n\|\boldsymbol{x}_k\|_2} \boldsymbol{R}_r \boldsymbol{x}_k \tag{3-2}$$

其中，α 称作旋转因子，是一个可控的参数，它控制着旋转变换算子的搜索空间的大小；$\boldsymbol{R}_r \in \mathbb{R}^{n\times n}$ 是一个服从[-1,1]均匀分布随机矩阵，$\|\bullet\|_2$ 表示向量的二范数。该算子搜索邻域为半径为 α 的超球体；通过控制 α 的大小，可以控制该算子的寻优空间，α 越大，寻优空间越大，在 α 固定的情况下，旋转变换算子的搜索空间随着决策变量维数的增加而变小；在状态转移算法中，它常常用来进行局部搜索。

2) 平移变换 (Translation Transformation)

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \beta \boldsymbol{R}_t \frac{\boldsymbol{x}_k - \boldsymbol{x}_{k-1}}{\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|_2} \tag{3-3}$$

其中，β 称作平移因子，是一个可控的参数，它控制着平移变换算子的搜索力度，$\boldsymbol{R}_t \in \mathbb{R}$ 是一个元素服从[0,1]的均匀分布随机矩阵，该算子具有沿着从点 $\boldsymbol{x}_{k-1}$ 到点 $\boldsymbol{x}_k$ 的直线上的线搜索功能，其线搜索起点为 $\boldsymbol{x}_k$，最大的搜索长度为 β。在状态转移算法中，它常常应用于选择机制之后，选择出两个较好的候选解，进行强化

搜索。

3) 伸缩变换 (Expansion Transformation)

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \gamma \boldsymbol{R}_e \boldsymbol{x}_k \tag{3-4}$$

其中，γ 称作伸缩因子，是一个可控的参数，$\boldsymbol{R}_e \in \mathbb{R}^{n\times n}$ 是一个元素服从高斯分布的随机对角矩阵。伸缩变换具有使 $\boldsymbol{x}_k$ 中的每个元素伸缩变换到$[-\infty,+\infty]$的功能，在状态转移算法中，利用该算子实现在整个决策变量空间的全局搜索。

4) 坐标变换 (Axesion Transformation)

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \delta \boldsymbol{R}_a \boldsymbol{x}_k \tag{3-5}$$

其中，δ 称作坐标因子，是一个可控的参数，$\boldsymbol{R}_a \in \mathbb{R}^{n\times n}$ 是一个随机对角稀疏矩阵，矩阵内某些元素是非零的，且这些元素的值服从高斯分布。坐标搜索具有使 $\boldsymbol{x}_k$ 沿着坐标轴方向搜索的功能，δ 越大，搜索力度越大。在状态转移算法中，利用该算子在某一维度进行搜索，增强单维搜索能力。

在使用过程中，状态转移算子运用采样的概念，在决策空间进行采样来避免穷举造成的时间浪费。在单目标优化问题的求解中，它是一个基于个体的搜索算法，仅仅利用一个当前最优解，在状态转移算子的寻优空间采样 *SE* 次，产生 *SE* 个候选解。

3.2 状态转移算子对算法性能影响的评估

求解单目标优化问题时，希望产生的当前最优解序列收敛至最优解，且该最优解是唯一的。然而，求解多目标优化问题得到是一组权衡解集，对于一个候选解来说，并没有明确的收敛方向。因此，多目标优化的这些特性对智能搜索算子的全局搜索能力要求更高。为此，本节通过借助多目标算法性能评价指标，对状态转移算子的搜索性能进行分析，对状态转移算子单独使用和组合使用对多目标优化算法收敛性影响进行探究，为后续研究提供基础和经验。

本节主要阐述了状态转移算子对多目标优化算法收敛性作用的评估流程进行说明。在实验设计中，保证对比实验的单一变量原则，算法的初始候选解集和候选解的选择策略一致，以几种典型的标准测试函数为例，利用不同的状态转移算子产生候选解集，对候选解进行选择和更新后，解集相对于理想参考解集的性能评价指标，通过迭代过程中的性能指标的变化情况来反映状态转移算子的搜索能力对算法分布性和收敛性的影响情况。在本章及以后，用 θ 作为状态转移算子参数统一的表示。

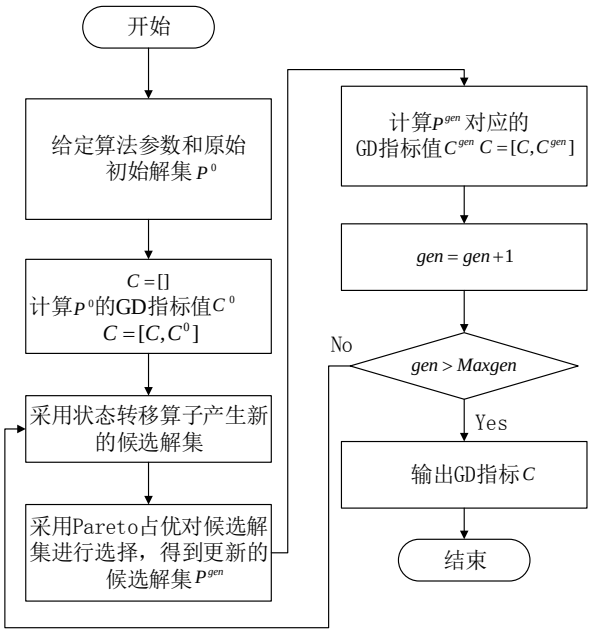


图 3-1 状态转移算子对算法性能影响的评估流程图

由第二章的中 2.4 节对多目标性能指标的分析可以得知，在多目标优化问题中，GD 指标可以反映多目标种群的收敛性；在本章实验中，由于选择策略是相同的，不同状态转移算子产生候选解下，GD 指标的变化仅仅与搜索方式有关。基于此，借助该指标，反映状态转移算子其搜索性能对于多目标优化算法的收敛性影响。更具体的，将搜索算子对算法收敛性影响作用的评估流程图如图 3-1 所示，将其表述如下：

首先，进行初始化，设置相关算法参数，最大迭代次数 $Maxgen$ 及种群大小 N，将迭代次数 gen 初始值设置为 1，对变量 C 进行初始化，C 用于储存迭代过程中的 GD 指标。接下来进行算法的迭代进化阶段，通过从候选解集中随机选择 1 个候选解作为初始解，对这个解进行状态转移变换，产生新的子代种群。子代种群与父代种群合并为一个新种群，利用 Pareto 占优的概念选择出 N 个占优等级较高的解作为当前迭代过程的最优解集 $\boldsymbol{P}^{gen+1}$，利用参考解集 $\boldsymbol{P}^*$，计算出 $\boldsymbol{P}^{gen+1}$ 对应的 GD 指标并存储在变量 C 中。重复上述过程直到迭代次数达到最大迭代次数 $Maxgen$。本实验中，$Maxgen=500$，$N=200$。

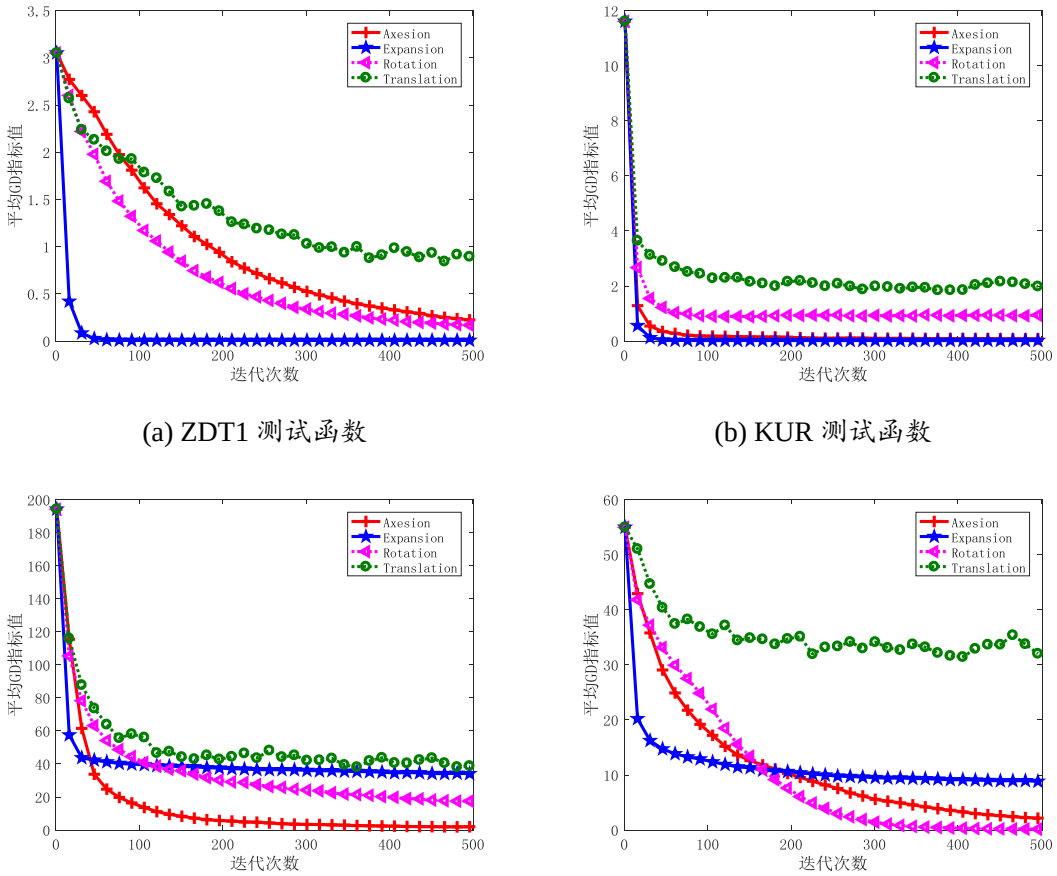
在此对该算法基于 Pareto 占优概念从合并的子代种群和父代种群中选择出 N 个解作为当前迭代过程的最优解集 $\boldsymbol{P}^{gen+1}$ 的流程进行进一步的解释。该选择策略基于非支配排序策略，根据 Pareto 占优概念，候选解之间进行相互比较，根据候选解之间的占优情况，将候选解划分为不同的前沿等级的解。选择前沿等级较前的解，若某一前沿的解及比其前沿等级高的解的总数量大于 N，则从该前沿等级中随机选择一定数量的解。

3.3 搜索算子单一使用下的搜索性能分析

本小节对状态变换算子单一使用时，其搜索性能对多目标优化算法的收敛性的影响；本实验中，算法的初始解集保持不变，算法独立运行 30 次，取 30 次运行中每一代的 GD 性能评估指标的平均值作为最终的实验结果。

3.3.1 算子搜索性能整体分析

本小节通过利用 ZDT1（两目标），KUR（两目标），DTLZ1（三目标），F4（三目标）这四个标准测试函数，利用不同的状态转移算子产生候选解集，对搜索算子对算法收敛性和分布性影响进行分析。其中，ZDT1 是一个凸函数；KUR 是分段不连续的函数；DTLZ1 包含众多的局部 Pareto 前沿面；F4 的前沿为不规则形状；这四个测试函数的 Pareto 前沿特性包含了大多数测试函数的 Pareto 前沿特性，可以通过使用这四个测试函数，较为充分地反映状态转移算子搜索性能对多目标算法的收敛性影响。在实验设计中，四个状态转移算子的参数一致，迭代终止条件一致，迭代 500 次后算法终止。



(a) ZDT1 测试函数　　(b) KUR 测试函数

(c) DTLZ1 函数　　(d) F4 测试函数

图 3-2 算子独立搜索下 GD 平均指标平均值迭代图

图 3-2 表示使用单个状态转移算子进行搜索求解四个不同测试问题的平均 GD 性能指标迭代曲线。由于给出的初始解集相同，因而对于每一个测试函数，四种状态转移算子的平均 GD 性能指标迭代曲线都从同一点出发。

在 ZDT1 测试问题上，从平均 GD 指标迭代图可以看出伸缩算子的 GD 指标可以较快下降至较低的位置。说明在搜索过程中，在同样的参数设定下，使用伸缩算子搜索，对算子的收敛性帮助较大。使用平移变换算子进行搜索时，在前期，指标值较快下降，为后期下降较为缓慢，其 GD 在迭代终止时的指标值是四个算子当中最大的。旋转变换算子和坐标变换算子迭代终止时 GD 指标值基本上一致，但是整体说来，旋转变换算子的指标值的下降速率均快于坐标变换算子。

在 KUR 测试问题上，GD 指标指标值先下降后趋于稳定，其中平移算子的指标值与其他状态转移算子相比，存在略微的波动。产生这种现象的一个原因是平移算子的变换需要两个初始解，相比于其他算子，受初始解的影响更大，且其迭代终止的最后值是四个状态转移算子之中最大的。在求解 KUR 测试问题中，前期 GD 指标的下降率，四个算子之间的差别不明显，在后期，平均 GD 指标稳定从高较低依次是平移算子，旋转算子，坐标算子和伸缩算子，说明在同样的参数下，伸缩算子的搜索性能更好。

在 DTLZ1 测试问题上，GD 指标由高到低的顺序依次是：平移变换，伸缩变换，旋转变换和坐标变换。坐标变换的 GD 指标值最低，其他算子得到的 GD 指标值与坐标变换指标值有很大的差距。此外，平移变化的 GD 迭代曲线一直处于波动之中。伸缩变换的 GD 迭代前期就基本稳定，说明此时已经陷入了局部前沿面中，而旋转变换和坐标变换保持一定的下降趋势，说明增大迭代次数，该问题有可能获得更好的效果。

在求解 F4 测试函数中，在迭代初期，按指标的下降速率看，最快的是伸缩变换算子，然后是坐标变换算子和旋转变换算子，最后是平移变换算子。从算法的终止最终的性能指标指标上看，有小到大依次是旋转变换算子，坐标变换算子，伸缩变换算子和平移变换算子。在求解 F4 优化问题中，平移变换算子的指标和其他变换算子的差距较大，下降速率最低，且最终的指标最大。伸缩变换算子虽然前期下降速率较快，但是其在迭代过程中过早的稳定，算法终止时的指标值也大；坐标变换算子和旋转变换算子，则一直处于下降趋势；迭代终止时，GD 指标值优于平移变换算子和伸缩变换算子。

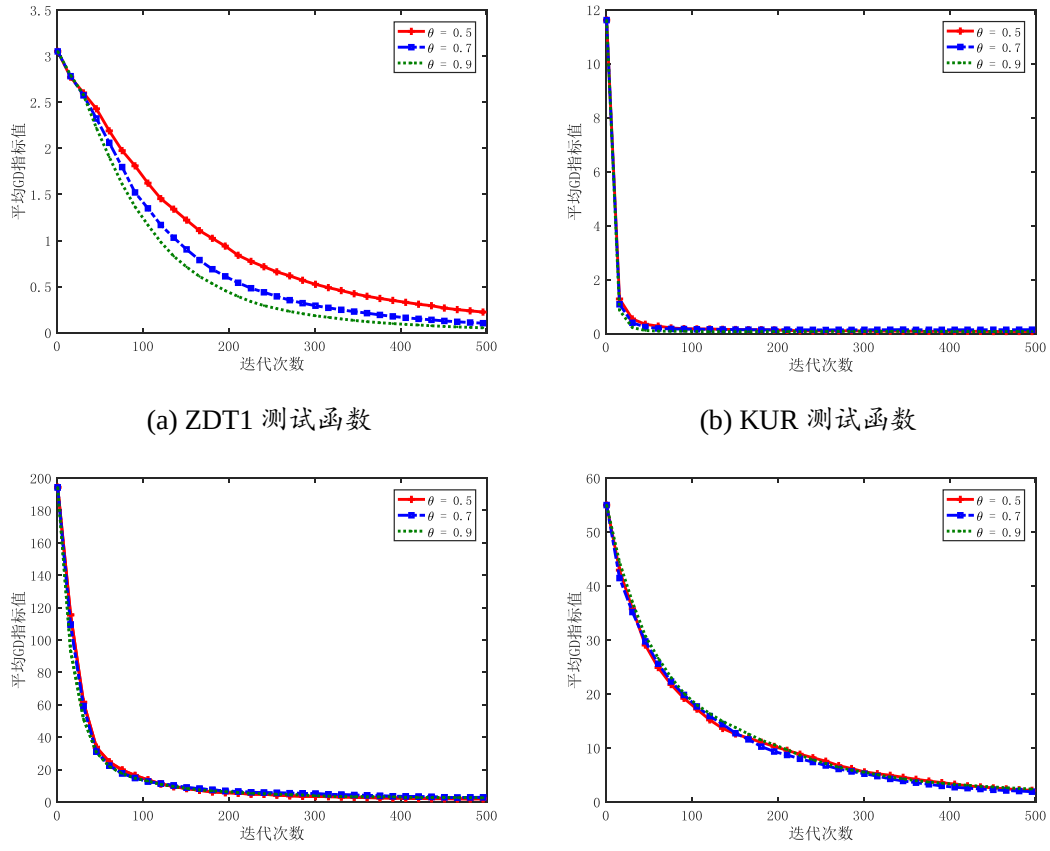
本小节通过使用 ZDT1，KUR，DTLZ1，F4 这四个测试函数，借助 GD 指标，对不同状态变换算子单独使用，产生候选解时，对算法收敛性的影响进行研究。其实验结果表明，当算法选择策略一定的情况下，仅仅使用平移变换算子产生候选解，相比于使用其他算子产生候选解，算法的收敛性更差。使用伸缩变换

算子，算法的收敛速度会增加但较为容易过早的陷入某一局部而算法最终的收敛性较差。旋转变换算子和坐标变换算子相比于上述两种算子，其收敛速度加快且其对于复杂的多目标优化测试问题上可以一直保持下降的趋势。综上所述，在设计算法中，综合考虑收敛性和收敛速度应使用使用旋转变换算子和伸缩变换算子，考虑收敛速度可以使用伸缩变换算子，平移变换算子相比于其他算子，不适合单独使用产生候选解。

3.3.2 算子参数对搜索性能影响分析

本小节通过改变状态算子的参数，分析了参数对于算法收敛性的影响。状态转移算子参数θ分别取值 0.5，0.7，0.9。

1) 坐标变换算子参数分析



(a) ZDT1 测试函数 (b) KUR 测试函数

(c) DTLZ1 测试函数 (d) F4 测试函数

图 3-3 坐标变换算子参数对 GD 指标影响作用分析图

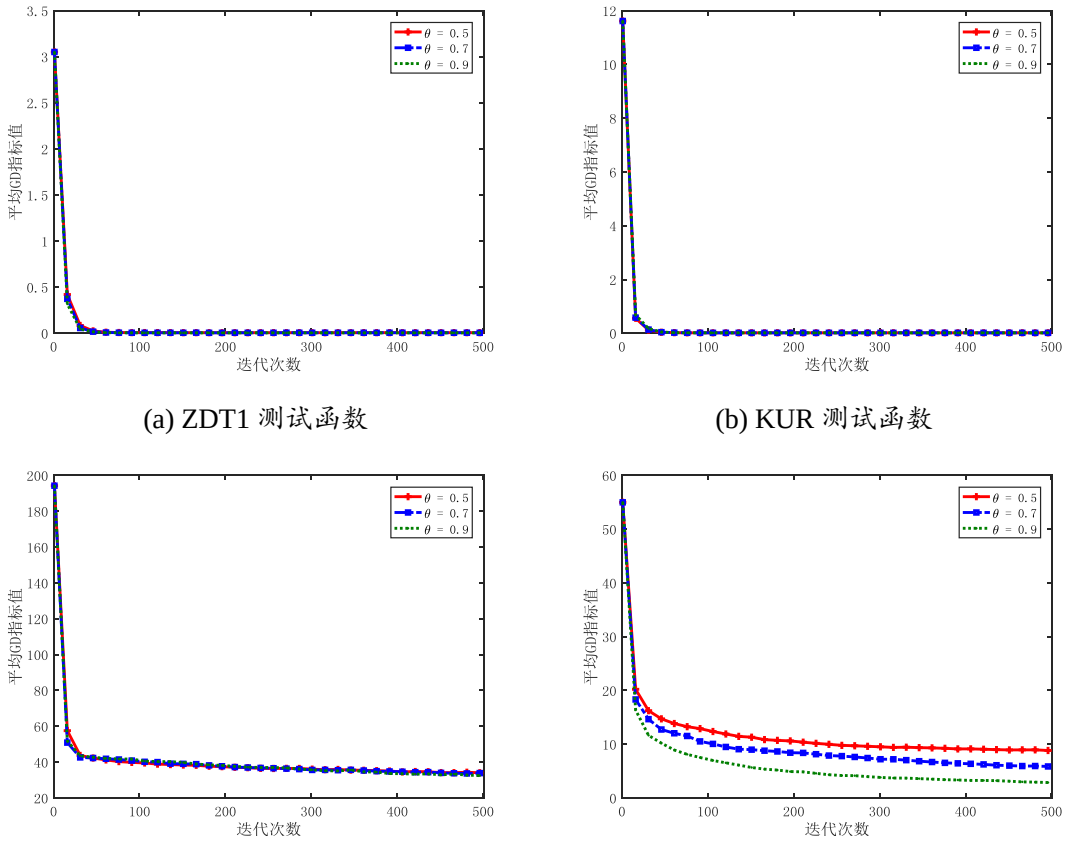
图 3-3 显示了坐标变换算子在不同参数下求解四种测试问题的 GD 指标平均迭代曲线。在求解 ZDT1 测试函数中，随着θ增加，在迭代终止时，GD 指标越来越小，说明在坐标变换算子中，参数越大，算法整体的收敛性越好；值得注意的是，在迭代最初期，参数不同对算法的收敛性影响很小，随着迭代的进行，在

迭代后期出现了明显的区别，参数越大，GD 指标越小，说明算法收敛性和分布性越好。将θ为 0.5 和θ为 0.7 的曲线进行对比时发现，GD 指标在迭代过程中都有了明显的下降，而将θ为 0.7 和θ为 0.9 的曲线进行对比时发现，指标下降的程度不如θ为 0.5 和θ为 0.7 的情形，说明当参数取得一定值后，通过增大参数来增强算法的收敛性的作用是比较有限的。

在求解 KUR 测试函数时，通过改变坐标变换算子的参数，由 GD 指标平均值的迭代图可以看出，当参数变化时，GD 指标迭代曲线几乎没有什么巨大的变化；当参数变大时，GD 指标平均值前期的下降率会有所增加，而后期，改变参数对其的影响并不是很大；因此，想通过增大参数来提高算法收敛性是不理想的。

在求解 DTLZ1 测试问题时，当参数θ逐渐变大时，平均 GD 指标在迭代过程中下降速率几乎没有什么变化，对于求解 F4 函数也是如此；而最终的终止至值有略微的差异。

2) 伸缩变换算子参数分析



(a) ZDT1 测试函数　　(b) KUR 测试函数

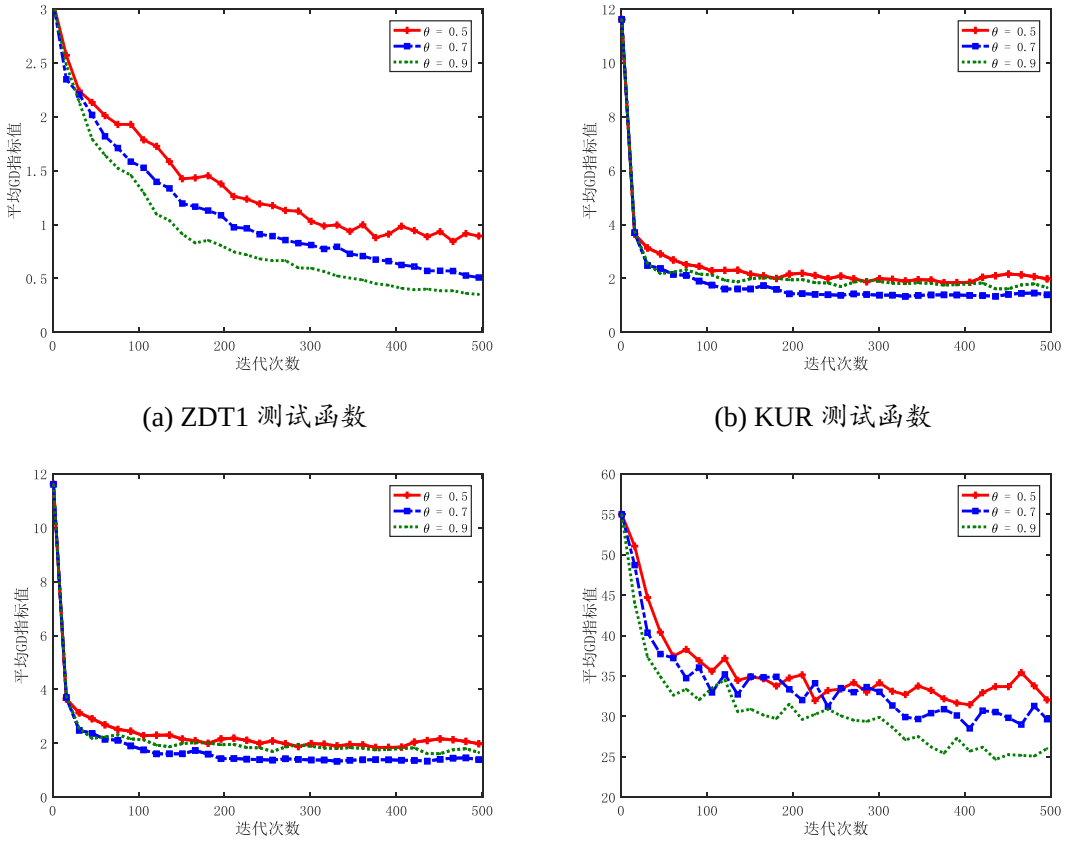
(c) DTLZ1 测试函数　　(d) F4 测试函数

图 3-4 伸缩变换算子参数对 GD 指标影响作用分析图

图 3-4 显示了伸缩变换算子在不同参数下求解四种测试问题的 GD 指标平均迭代曲线。当参数变化时，伸缩变换算子在 ZDT1 函数上，其性能指标几乎重叠，

没有什么变化，然而这并不能说明伸缩变换算子其搜索性能不受参数的影响，而是由于使用该算子比较有利于 ZDT1 问题的求解。求解 KUR 测试函数问题时，算子参数改变，GD 指标的迭代曲线没有显著的变化。整体说来，算子参数的改变对于求解 KUR 问题，没有显著影响。求解 DTLZ1 测试函数问题时，算子参数改变，GD 指标的迭代曲线没有显著的变化。随着算子参数的增大，其算法终止的 GD 指标值有略微的下降。当其参数发生变化求解测试函数 F4 时，随着参数的增大，在迭代前期，GD 指标的收敛率有着略微的提升。随着继续迭代，θ 为 0.9 时，算法终止时其 GD 指标最大。参数的改变对于迭代终值的指标值相比于在其他测试函数的情况，有较大的影响，参数越大，算法终止时的 GD 指标参数越小。

3) 平移变换算子参数分析



(a) ZDT1 测试函数　　(b) KUR 测试函数

(c) DTLZ1 测试函数　　(d) F4 测试函数

图 3-5 平移变换算子参数对 GD 指标影响作用分析图

图 3-5 显示了平移变换算子在不同参数下求解四种测试问题的 GD 指标平均迭代曲线。在求解 ZDT1 测试函数时，在迭代过程中，虽然不同参数下的平均 GD 指标差异不大，但是当θ为 0.9 时的迭代曲线还是略高于θ为 0.5 及 0.7 的迭代曲线。因此在算法初期，平移变换算子参数较小时，更加有利于候选解集朝着 Pareto

最优解集的方向进化。在迭代后期，指标出现了明显的区别，参数越大，其 GD 指标越小，说明算法收敛性越好。在参数θ为 0.5 和 0.7 的曲线进行对比时发现，GD 指标在迭代过程中都有了明显的下降，而将θ为 0.7 和θ为 0.9 的曲线进行对比时发现，指标下降也有了相当程度的下降。说明在一定程度上，通过增大参数来增强算法的收敛性是较为有效的。在求解 KUR 测试问题时，对于 GD 指标，在迭代最初期，参数变化对 GD 指标的影响不大。θ为 0.5，0.7 和 0.9 时，指标迭代曲线几乎重合。在后期，参数θ为 0.5 的 GD 指标值最大，θ为 0.7 的 GD 指标值最小，参数θ为 0.9 的处于三者中间。说明在求解 KUR 测试函数时，可以调节平移变换算子的参数θ来明显改善算法的收敛性效果。

在求解 DTLZ1 测试问题时，随着参数θ由小变大，GD 指标的下降率变快，在算法终止时的 GD 值中，θ为 0.9 的最小，θ为 0.7 的最大。当参数增大时，可以明显看到 GD 指标下降的趋势，参数由 0.5 和 0.7 对比与由 0.7 与 0.9 对比时，GD 指标下降程度都十分明显，说明增大参数平移算子参数对于算法收敛性的提高是有效的。

同样的，在求解 F4 函数时，当参数越大时，GD 指标的下降率变大，且算法终止时的 GD 指标数值也逐渐变小，相比于上述其他算子，平移算子参数的改变对算法收敛性的影响较大；增大平移算子的参数有利于算法收敛性的提高。

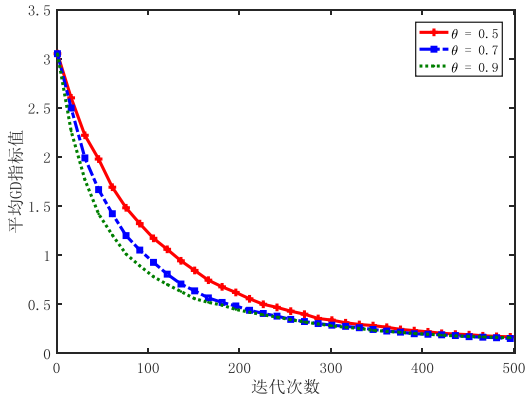
4) 旋转变换算子参数分析

图 3-6 显示了旋转变换算子在不同参数下求解四种测试问题的 GD 指标平均迭代曲线。与平移变换算子一致，在求解 ZDT1 测试问题中，在 GD 指标迭代图中，随着θ增加，在迭代终止时，平均 GD 指标的变化不大。说明在使用旋转变换算子中，参数越大，对于算法整体收敛性的影响不大。然而，值得注意的是，虽然最终的指标差距不大，但是随着θ的增加，在迭代前期，GD 性能指标变化速率是变大的，后期逐渐变慢并稳定。

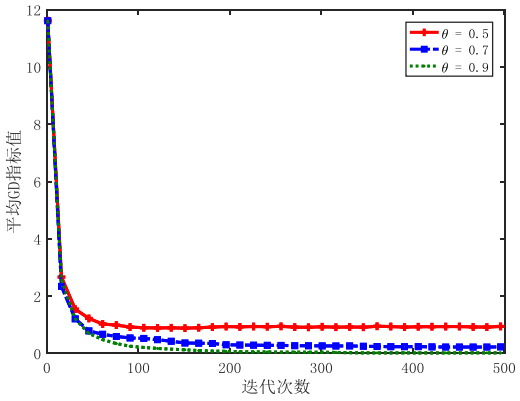
求解 KUR 测试问题时，参数变化对算法求解的收敛性都有较大的影响。从 GD 指标平均值的迭代曲线看，随着参数的增加，迭代后期 GD 的稳定值也随着下降。在迭代前期，三种参数的迭代曲线基本上重合，迭代下降速率的差别也不大，在后期，GD 指标值出现了明显变化。参数越大，对应的 GD 指标值越小，θ为 0.7 与θ为 0.5 的迭代终止时最终 GD 指标值的改善程度大于θ为 0.9 与θ为 0.7 的情况，说明增大旋转变换算子的参数虽然有利于算法收敛性和稳定性的提高，但也不是越大越好，参数到达某一数值后，再通过增大参数来增强算法性能，算法的性能也不会获得更大的提升。

在求解 DTLZ1 问题中，从 GD 指标平均值的迭代曲线上看，随着参数的增大，GD 指标的下降速率和迭代终止时的 GD 值都在减少，且 GD 指标的存在略

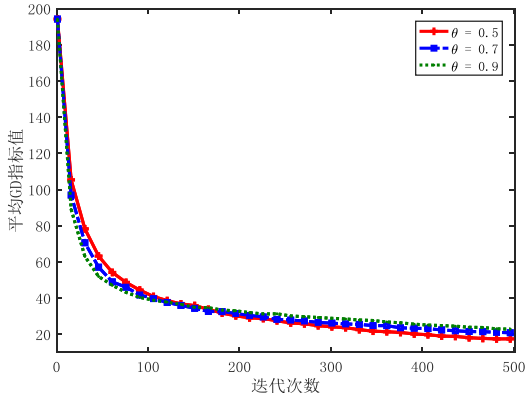
微的下降。增大算子参数在算法迭代前期有利于算法提升收敛速度。



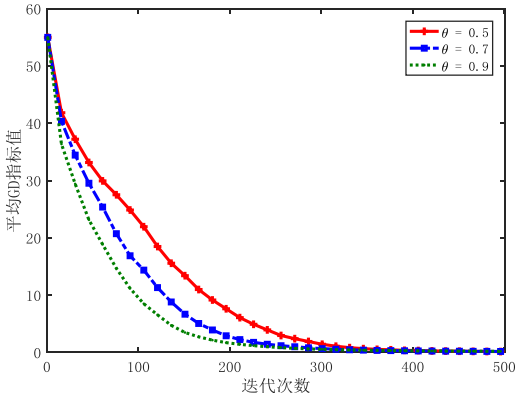
(a) ZDT1 测试函数



(b) KUR 测试函数



(c) DTLZ1 测试函数



(d) F4 测试函数

图 3-6 旋转变换算子参数对 GD 指标影响作用分析图

在求解 F4 测试函数时也可以看出，当参数增大时，GD 指标下降的收敛率增大，指标下降较快，而当迭代终止时，GD 指标没有基本变化。从 F4 函数看出，增大旋转变换算子的参数是对于提高算法的收敛速率是有利的。

本小节通过利用 ZDT1，KUR，DTLZ1，F4 四种不同的函数对四种状态转移算子的对多目标算法收敛性的影响进行了分析。当算法的选择策略一致时，使用伸缩变换算子产生候选解，可以加快算法的收敛速率，而单独使用平移变换算子，算法的收敛性和收敛速度都是四个算子中最差的。从四个算子单独使用求解 4 个测试问题上，旋转变换算子和伸缩变换算子不仅仅对算法收敛性帮助较大，同时也对收敛速度有着积极的作用，其作用受优化问题的影响较小。在算法设计中，应该合理使用这两种算子。同时，本小节也分析了参数对算法收敛性和收敛速率的影响。对于平移算子，增加其参数对于收敛速率和收敛性有着较大的作用。而对于旋转变换算子，增加其参数仅仅增加了算法的收敛速率，对于算法收敛到最优前沿的帮助不大，对于伸缩变换算子和坐标变换算子而言，参数的改变对其的

影响作用不明显。

3.4 算子组合使用下的性能分析

上述小节利用四个测试函数进行测试，使用不同状态转移算子搜索，对算法收敛性影响进行了分析，本节主要对算子综合使用的性能进行分析，即分析使用多种状态转移算子共同产生新种群对算法收敛性的影响。本节主要对两种状态转移算子组合使用对算法性能影响进行分析和研究。设种群大小为 N，本小节通过平移变换算子产生 $N/2$ 个候选解，其他状态转移算子 $N/2$ 个候选解，两种状态转移算子共同使用，产生 N 个候选解，形成新的种群。

3.4.1 平移变换算子与其他算子组合使用的性能分析

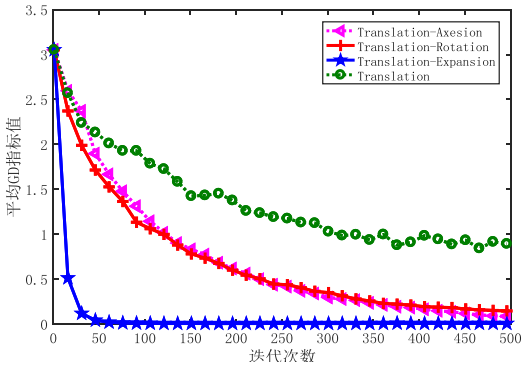
本小节通过比较平移变换算子与其他变换算子组合与仅使用平移变换算子产生候选解，这些不同产生候选解集的方式之间相互比较，探索不同搜索算子组合方式对算法收敛性和分布性的影响，实验结果图如图 3-7 所示。

在求解 ZDT1 问题上，从 GD 指标影响作用分析图上看，平移变换算子与其他变换算子组合使用，其 GD 指标都有较大的改善。其中平移变换算子-伸缩变换算子组合使用，GD 指标随着迭代次数下降较快且最终的 GD 指标值较小。平移变换算子-坐标变换算子组合使用与平移变换算子-旋转变换算子组合使用与单独使用平移变换算子相比，GD 指标变好程度相当，且这两种组合的 GD 指标迭代曲线变化一致。综上所述，在算法收敛性方面，其他状态转移变换算子与平移变换算子综合使用，相比于单独使用平移变换算子，可以有效地提高算法的收敛性。

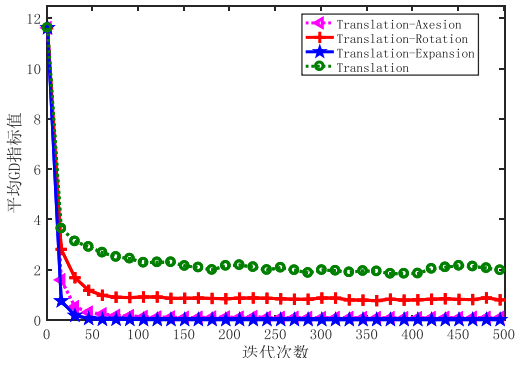
在求解 ZDT1 问题上，从 GD 指标影响作用分析图上看，平移变换算子与其他变换算子组合使用，其 GD 指标都有较大的改善。其中平移变换算子-伸缩变换算子组合使用，GD 指标随着迭代次数下降较快且最终的 GD 指标值较小。平移变换算子-坐标变换算子组合使用与平移变换算子-旋转变换算子组合使用与单独使用平移变换算子相比，GD 指标变好程度相当，且这两种组合的 GD 指标迭代曲线变化一致。综上所述，在算法收敛性方面，其他状态转移变换算子与平移变换算子综合使用，相比于单独使用平移变换算子，可以有效地提高算法的收敛性。

在求解 KUR 问题上，在 GD 指标迭代图中，算子的使用方式对 GD 指标的影响十分明显，平移变换算子与其他算子组合使用与平移变换算子单独使用，GD 指标大大减小，可以大大增强算法的收敛性。其中按照迭代终止 GD 指标值来看，提升效果明显程度由大到小，依次是：平移变换-伸缩变换，平移变换-坐标变换，平移变换-旋转变换。同样的，在迭代前期，GD 指标下降率从大到小也是该顺序。

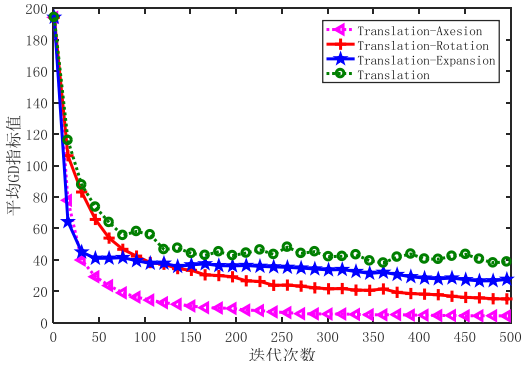
通过使用平移变换和其他变换算子组合使用与单独使用对比发现，算法的收敛性和收敛速率因算子组合而增强，算法获得解集的分布性会因算子组合而减弱。



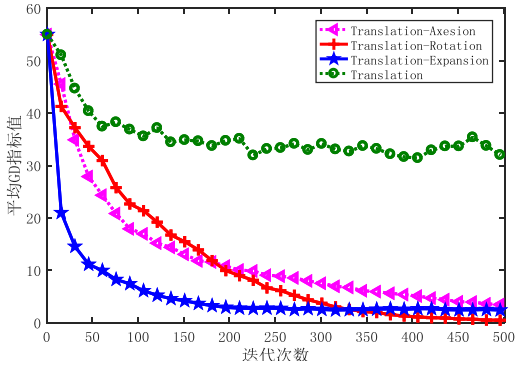
(a) ZDT1 测试函数



(b) KUR 测试函数



(c) DTLZ1 测试函数



(d) F4 测试函数

图 3-7 平移变换算子与其他算子组合使用对 GD 指标影响作用分析图

在求解 DTLZ1 测试问题时，发现平移变换算子和其他算子一起使用时，在迭代过程中，GD 指标都有所下降。其中平移变换-坐标变换的下降趋势最为明显。从 GD 指标的下降速率上进行分析，其中平移变换-旋转变换在迭代前期下降速率与仅使用平移变换算子相比，差距不大。但是在迭代后期，平移变换-旋转变换一直保持着下降的趋势。平移变换-伸缩变换和平移变换-坐标变换相比于仅使用平移变换，其下降速率有着较大的提升。其中平移变换-伸缩变换的下降率提升最大。随着迭代的进行，平移变换-伸缩变换下降速率逐渐减小，平移变换-坐标变换一直保持下降，且其算法终止时的 GD 指标是四种方式中最小的。

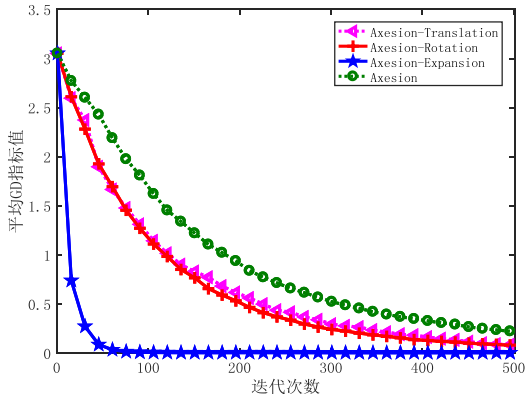
在求解 F4 测试函数中，平移变换算子和其他变换算子组合使用时，对算法的收敛性和收敛速率有着较大的提升。其中，平移变换-伸缩变换下降较快，之后是平移变换-坐标变换和旋转变换-伸缩变换。在迭代终止时，三种组合方式下的 GD 指标值相差不大。其中，平移变换-旋转变换这种组合方式下的 GD 指标最

小，且在整个迭代过程中一直保持着下降，平移变换-伸缩变换虽然在迭代前期下降较快，但是后期下降较慢，直至几乎稳定。

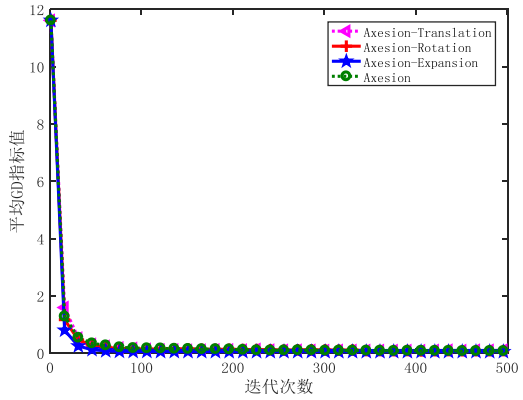
本小节通过比较单独使用平移变换算子搜索候选解，产生候选解集与其他状态转移算子共同产生候选解集求解 4 个标准测试函数，独立运行 30 次，观察 GD 指标的平均值的迭代曲线发现，平移变换算子和其他状态转移算子组合使用对算法的收敛性和收敛速度具有大大提升的作用。

3.4.2 坐标变换算子与其他算子组合使用的性能分析

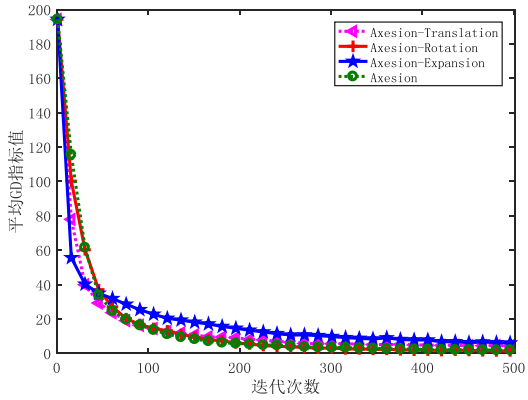
本小节通过比较坐标变换算子与其他变换算子组合与仅使用坐标变换算子产生候选解，这些不同产生候选解集的方式之间相互比较，探索针对坐标变换算子的合理算子组合方式及其对算法性能的影响，实验结果如图 3-8 所示。



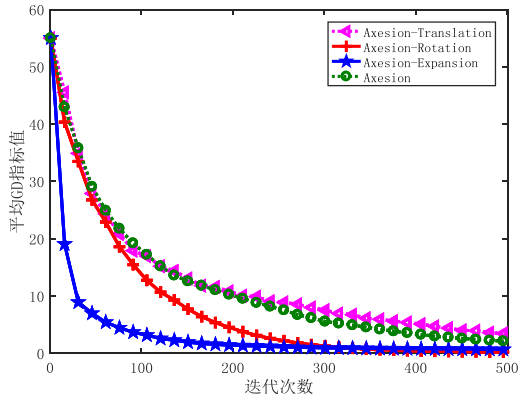
(a) ZDT1 测试函数



(b) KUR 测试函数



(c) DTLZ1 测试函数



(d) F4 测试函数

图 3-8 坐标变换算子与其他算子组合使用对 GD 指标影响作用分析图

在使用坐标变换算子与其他算子组合求解 ZDT1 测试函数时，从 GD 指标的迭代图上可以发现，与单独使用坐标变换算子相比，与其他算子的组合在 GD 指标上都有提升。具体说来，从 GD 指标的下降率来看，从慢到快依次是：单独使

用坐标变换算子，坐标变换-平移变换，坐标变换-旋转变换，坐标变换-伸缩变换，同样的，观察最终得到的 GD 指标的平均值发现，从大到小也是这个顺序。值得说明的是，坐标变换-平移变换，坐标变换-旋转变换在整个迭代过程之间的差距不大。综合说来，相比之下，综合使用状态转移算子可以一定程度上提高算法的收敛性。

在求解 KUR 问题，从 GD 指标迭代过程图可知，在迭代初期，四种产生候选解的方式 GD 指标迭代曲线重合度较高。迭代终止时的 GD 指标差距不大，其中，坐标变换-伸缩变换下降最快，最终的 GD 指标值最小；坐标变换-平移变换相对而言，其值最大。从整体上面看，坐标变换和其他变换算子组合相对于坐标变换单独使用，性能指标还是可以提高的。

在求解 DTLZ1 问题上，坐标变换算子和其他算子组合使用时，在迭代前期，GD 指标的下降率没有明显的变化。在后期，坐标变换-伸缩变换与单独使用坐标变换算子相比，算法收敛性的性能指标变大。说明这两种算子组合使用对于求解 DTLZ1 问题不利。整体来看，坐标变换算子与其他变换算子组合使用时，在 DTLZ1 问题上的求解，帮助不大，性能提升有限。

在求解 F4 测试函数中，坐标变换-旋转变换，坐标变换-伸缩变换对于算法的收敛性有着较大的提升，且在迭代前期，GD 指标的下降速率较快。坐标变换-平移变换组合使用时，在迭代后期，GD 指标略微大于单独使用坐标变换算子；同时坐标变换-平移变换相比于其他组合方式，迭代曲线的不够光滑，这是由于平移变换算子变换需要两个初始解，其影响较大。整体说来，在 F4 测试函数的求解上，状态变换算子之间组合使用，对性能还是有提升贡献的。

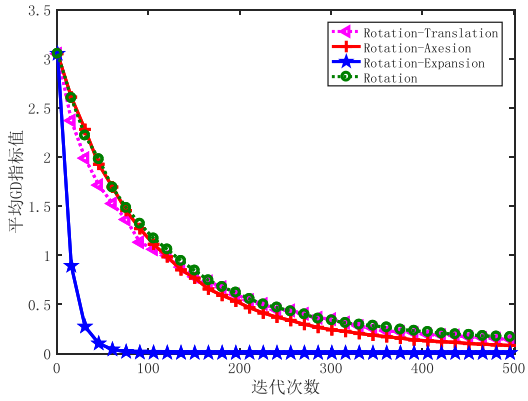
3.4.3 旋转变换算子与其他算子组合使用的性能分析

本小节通过比较旋转变换算子与其他变换算子组合与仅使用旋转变换算子产生候选解，这些不同产生候选解集的方式之间相互比较，探索针对旋转变换算子的合理算子组合方式及其对算法性能的影响。实验结果图如 3-9 所示。

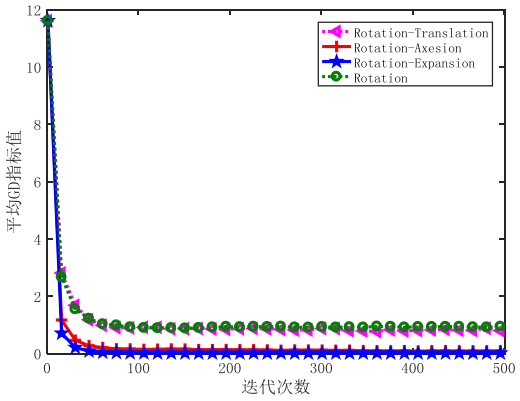
在求解 ZDT1 问题时，从 GD 指标的迭代图可以看出，旋转变换-伸缩变换的组合大大加强了算法的收敛性，GD 指标快速下降并稳定。旋转变换-坐标变换和旋转变换-平移变换的组合对收敛性有轻微的帮助。从最终的下降值看，旋转变换-坐标变换的贡献大于旋转变换-平移变换的组合。整体上说，旋转算子和伸缩算子的结合。可以最大程度的提高算法的收敛性和其所获解集的分布性。

在 KUR 问题求解中：从 GD 指标的迭代图看，旋转变换-平移变换和仅仅使用旋转变换的迭代曲线图基本重合。在迭代后期，略微下降，说明有一定的改善效果。而旋转变换-坐标变换，旋转变换-伸缩变换的结合使用使得的 GD 指标有了较大程度的下降。因此，旋转变换算子与伸缩变换和坐标变换搜索结合，有利

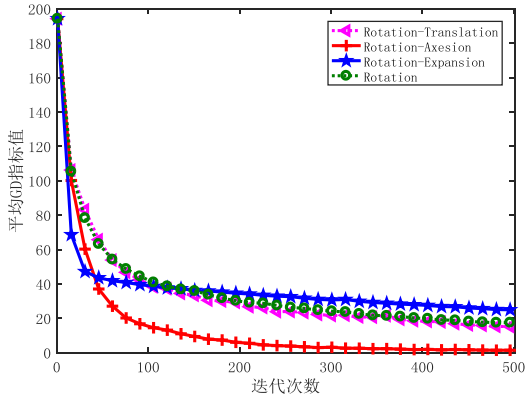
于算法收敛性和所获解集分布性性能的提高。



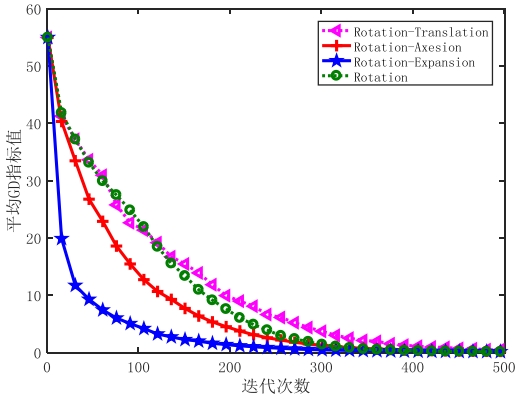
(a) ZDT1 测试函数



(b) KUR 测试函数



(c) DTLZ1 测试函数



(d) F4 测试函数

图 3-9 旋转变换算子与其他算子组合使用对 GD 指标影响作用分析图

在 DTLZ1 问题求解中：从 GD 指标的迭代图看，旋转变换-平移变换和仅仅使用旋转变换的迭代曲线图基本重合；而旋转变换-伸缩变换的结合使用相比于单独使用旋转变换，GD 指标的反而增加，说明这两个算子组合使用会削弱算法性能。旋转变换-坐标变换这两种进行组合，在迭代前期，GD 指标的下降率变化不大，但是一直保持着下降的趋势，且最终算法终止时，旋转变换-坐标变换的 GD 值是最小的。说明旋转变换-坐标变换的组合对于求解 DTLZ1 有着较大的帮助。

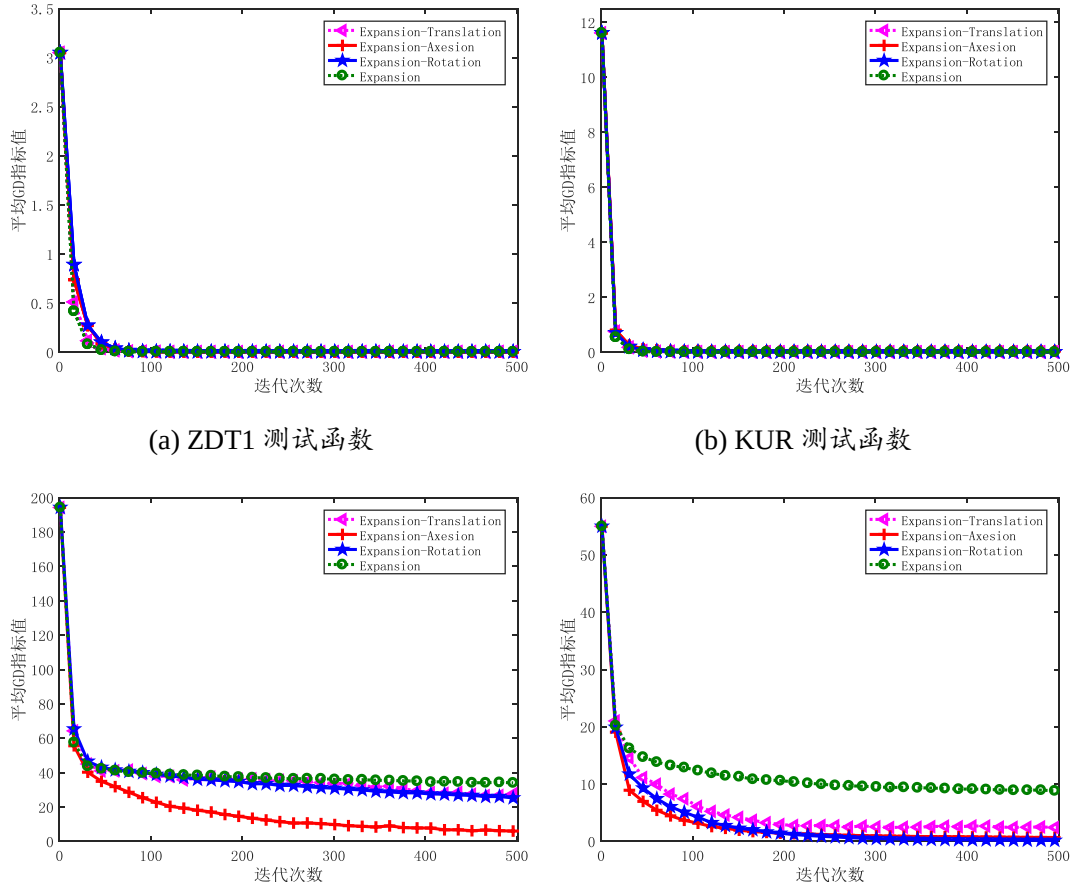
在求解 F4 测试函数中，旋转变换算子和其他变换算子共同使用时，相对而言算法的收敛性还是有一定的提高。在迭代前期，算子组合使用，GD 指标的下降加快，算法的收敛速率加快，然而随着迭代的进行，旋转变换-平移变换的组合下降逐渐变慢，且在算法终止时，其 GD 指标高于单独使用旋转变换算子。对于旋转变换-坐标变换和旋转变换-伸缩变换这两种组合方式来说，其在前期迭代速率加快，后期变慢，在算法终止时，四种产生候选解的方式其 GD 指标几乎相同。

整体来说，旋转变换算子和其他算子组合使用对算法收敛速率还是有提升作用的。

3.4.4 伸缩变换算子与其他算子组合使用的性能分析

本小节通过比较伸缩变换算子与其他变换算子组合与仅使用伸缩变换算子产生候选解，这些不同产生候选解集的方式之间相互比较，探索针对坐标变换算子的合理算子组合方式及其对算法性能的影响，其实验结果如图 3-10 所示。

在利用伸缩变换算子与其他算子组合产生候选解集时，求解 ZDT1 测试问题时发现，伸缩算子与其他算子的组合会削弱算法的收敛性和分布性。在 GD 指标迭代图中，发现伸缩变换-旋转变换，伸缩变换-坐标变换在收敛速率上有所减慢，而伸缩变换-平移变换在下降速率上与单独使用伸缩变换差异不大，而从迭代终止时的 GD 指标上看，最小的是单独使用伸缩变换算子产生候选解得到的。因此，整体说来，伸缩变换算子更适合于独立产生候选解集。



(a) ZDT1 测试函数　　(b) KUR 测试函数

(c) DTLZ1 测试函数　　(d) F4 测试函数

图 3-10 伸缩变换算子与其他算子组合使用对 GD 指标影响作用分析图

在求解 KUR 测试问题时，伸缩算子与其他算子结合使用的效果不理想。在 GD 指标的迭代图中看到，使用状态转移算子之间与伸缩变换组合会削弱收敛性，从迭代后期的 GD 指标上看，最理想的还是单独使用伸缩算子产生

候选解的情况。因此，伸缩算子适合单独使用产生候选解。

在求解 DTLZ1 测试函数时，使用伸缩算子-坐标算子组合时，算法的收敛性相比于单独使用伸缩算子或伸缩算子和其他变换算子组合使用时，有了较大的提升。算法终止时，GD 指标值是四个中最小的。对于伸缩变换-旋转变换，伸缩变换-平移变换这两种组合方式时，算法迭代终止时的 GD 指标也有一定的下降，且这两个方式对于算法的收敛性作用相当，这两种组合方式的 GD 迭代曲线基本重合。从 GD 指标迭代曲线可知，在求解 DTLZ1 问题时，采用伸缩算子-坐标算子更好。

在求解 F4 函数时，可以看出伸缩变换算子在于其他算子组合使用产生候选解集时，对算法的收敛性有较大的帮助；在四种产生候选解方式时，单独使用伸缩变换算子的 GD 指标值是最大的，其次是伸缩变换-平移变换这种方式；伸缩变换-旋转变换和伸缩变换-坐标变换的迭代曲线基本重合；值得说明的是，四种组合方式下，GD 指标下降率最大的是伸缩算子-坐标算子；因此，该种组合是较为具有潜力的产生候选解方式。

3.5 本章小结

在多目标优化算法的设计中，候选解的分布越广泛，对于算法的收敛性和分布性越有帮助。产生较优的候选解是后续选择的前提，也是算法最终获得在 Pareto 前沿均匀分布候选集解的基础。因此，本章针对状态转移算子的搜索能力设计了实验，对搜索算子对算法收敛性能的影响进行评估，为后续算法的设计提供一定的经验和指导。通过实验结果表明，对于不同的状态转移算子参数而言，增大平移变换算子的参数可以增加算法的收敛性和收敛速度。而对于旋转变换算子而言，增大参数对算法的收敛速度有所帮助，而对算法收敛性没有显著影响。平移状态变换算子不适合单独使用其产生候选解，该算子和其他状态变换算子组合，可以增强算法的收敛性。利用伸缩变换算子进行搜索，对于求解某些简单问题十分有效，但是面对复杂问题，可能效果不是十分理想；旋转变换和坐标变换相比较而言是最有潜力的搜索算子，无论是与其他状态转移算子组合使用还是单独使用，算法的收敛性都较为可观。

4 基于 Pareto 占优的多目标状态转移算法

状态转移算法目前集中于单目标优化领域及实际工程问题，在多目标优化领域的研究还较少。本章以第三章对于状态转移算子的分析为基础，基于 Pareto 占优框架，设计出了一种基于 Pareto 占优的多目标状态转移算法（MOSTA/P）。为了减少候选解比较的时间复杂度，在比较过程中，提出了基于计算资源动态分配的非支配排序策略。本章详细阐述基于 Pareto 占优框架的多目标状态转移算法的算法流程和设计思想，通过利用标准测试函数及相应的性能指标对本章提出的算法进行测试，并和其他经典算法进行对比，表明本章提出的算法可以快速有效地求解多目标优化问题，本算法得到的最优解集，具有良好的收敛性和分布性。

4.1 基于计算资源动态分配的高效非支配排序策略

基于 Pareto 占优框架的多目标优化算法中，一个重要的研究内容就是研究如何在候选解比较过程中，减少时间复杂度，快速得到候选解的 Pareto 占优等级及非占优解。在此方面，Zhang 等提出一种高效非支配排序策略（ENS）[42]，该策略的时间复杂度在最好情况下为 $O(MN\log N)$，最差情况下为 $O(MN^2)$，空间复杂度为 $O(1)$，比 NSGAII 中提出的快速非支配排序策略的时间复杂度和空间复杂度都有一定程度的减少。MOSTA/P 算法中，引进了 ENS 策略的同时，考虑了候选解比较过程中计算资源分配的情况，引入了 MOEA/D-DRA 中计算资源动态分配的思想[60]，提出基于计算资源动态分配的高效非支配排序策略(ENS-DRA)，进一步减少了非支配排序过程中的比较次数。

4.1.1 计算资源动态分配策略(DRA)

所有的智能随机优化算法，无论是单目标优化算法还是多目标算法，都存在一个共同的缺陷，且这个缺陷是智能随机优化算法设计思想本身决定的，是不可避免的。那就是智能随机优化算法的候选解都是通过算子随机搜索产生的，不能保证新产生的候选解就优于原来的候选解；在多目标优化算法中，这个缺陷的影响尤为严重。因为多目标优化算法的目标是获得优化问题的非支配解集，在比较过程中，就需要知道新产生的候选解和当前候选解集中所有候选解的支配关系，如果产生的新候选解目标函数都比当前候选解集中解的目标函数大，则与其他候选解比较与选择之后，并没有使候选解集朝向 Pareto 最优解集的方向进化，此时就造成了计算资源的浪费。为了尽可能减少诸如上述情况的存在，在本小节提出计算资源动态分配策略，在比较中尽可能的节约计算资源。

假设 $\boldsymbol{P}$ 是父代解集，解集内的解 $\boldsymbol{P}^i$ 通过随机搜索算子产生了一个候选解 $\boldsymbol{Q}^i$，新产生的候选解组成子代集合 $\boldsymbol{Q}$，对于每一个子代 $\boldsymbol{Q}^i$ 定义一个资源配置值 π^i，

其计算方式如下：

$$\pi^i = \begin{cases} 1 & \text{if } \Delta > 0.001 \\ (0.95 + 0.05\frac{\Delta^i}{0.0001})\pi^i & otherwise \end{cases} \tag{4-1}$$

其中，$\Delta^i = (\sum_{j=1}^{M} f^j(\boldsymbol{P}^i) - \sum_{j=1}^{M} f^j(\boldsymbol{Q}^i)) / \sum_{j=1}^{M} f^j(\boldsymbol{P}^i)$。

Δ^i 利用新产生候选解 $\boldsymbol{Q}^i$ 的目标函数之和与初始解 $\boldsymbol{P}^i$ 目标函数之和之间的关系，对候选解 $\boldsymbol{Q}^i$ 是否优于 $\boldsymbol{P}^i$ 进行初步判断，Δ^i 表示候选解目标函数之和的减小率。当 $\Delta^i > 0$ 时，表明新产生的候选解 $\boldsymbol{Q}^i$ 至少存在一个目标函数比 $\boldsymbol{P}^i$ 小，即 $\boldsymbol{Q}^i$ 占优；当 $\Delta^i > 0.001$ 时，Δ^i 越大，π^i 越大，表明新产生的 $\boldsymbol{Q}^i$ 资源配置值越高。在后续非支配排序过程中，选择资源配置值高的解，进行比较，即为资源配置值高的解，分配计算资源，这样就减少了候选解的比较次数。在初始化阶段，Δ^i 的初始值设置为 1。

4.1.2 高效非支配排序策略(ENS)

本小节主要介绍高效非支配排序策略。该策略通过深入分析候选解之间比较的各种情况，充分考虑了非支配排序过程中，存在的候选解重复排序的情况。利用比较过程中的信息，减少了候选解之间的比较次数。对于没有划分前沿等级的候选解，仅仅和已经分配前沿的解，进行比较，大大地减少了比较次数，提高了算法的执行效率。

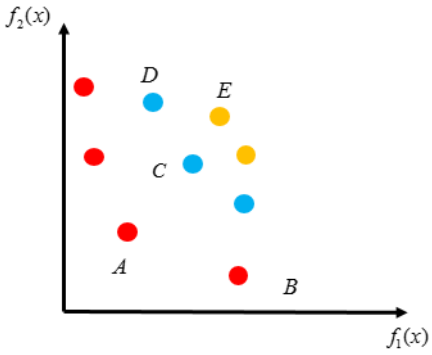


图 4-1 候选解比较情况示意图

候选解之间相互比较时可能存在的情况用图 4-1 所示简要说明。图中红色的点表示处于第一前沿等级的解，图中蓝色的点表示处于第二前沿等级的解，图中黄色的点表示处于第三前沿等级的解。由图 4-1，可见候选解之间比较时，存在如下几种情况：（1）候选解和候选解之间相互不占优，且处于同一前沿；如候选解 A 和候选解 B 所示；（2）候选解和候选解之间相互不占优，但是处于不同前沿，如候选解 B 和候选解 C 所示；（3）候选解和候选解之间存在占优关系，如候选解 A 和候选解 D 所示。情况（2）中，虽然候选解 B 和候选解 C 相互不占优，但是

与候选解 B 处于同一前沿的候选解 A，对于候选解 C 是占优的，候选解 B 和候选解 C 处于不同前沿。由此可得结论，如果一个解和另一个解互不支配，但是存在与另一个解处于同一前沿等级的解支配该解，则这两个解处于不同前沿，且无需在与该候选解其他前沿比较。ENS 策略充分利用上述结论，候选解与已知前沿等级的候选解进行比较，避免了重复与大量多次的比较。

算法 4.1： $\boldsymbol{F} \leftarrow \mathrm{ENS}(\boldsymbol{P}, \boldsymbol{F}, N)$

1. 输入：候选解 ，前沿集合 $\boldsymbol{F}$ ，种群大小 N
2. 输出：前沿集合 $\boldsymbol{F}$
3. $\boldsymbol{P} \leftarrow \mathrm{sort}(\boldsymbol{F}(\boldsymbol{P}))$ 依次按照单个目标函数值大小从小到大对候选解集进行 $\boldsymbol{P}$ 排序
4. $\tilde{N} = size(\boldsymbol{P})$
5. $k = 1$
6. **for** $i \leftarrow 1$ to $\tilde{N}$ **do**
7. $x \leftarrow \mathrm{size}(\boldsymbol{F})$ 记录已知前沿等级的个数
8. **while true do**
9. 将 $\boldsymbol{P}^i$ 与处于前沿集合 $\boldsymbol{F}^k$ 的所有解进行比较
10. **if** $\boldsymbol{F}^k$ 中的解与 $\boldsymbol{P}^i$ 相互不占优 **then**
11. $\boldsymbol{F}^i = k$ （ P^i 处于第 k 前沿中）
12. **else**
13. $k{+}{+}$
14. **if** $k > x$ **then**
15. $\boldsymbol{F}^i = x+1$ （ $\boldsymbol{F}^k$ 处于第 $x+1$ 前沿中）
16. **break**
17. **end if**
18. **end if**
19. **end while**
20. **if** $\sum_{i=1}^{k} |\boldsymbol{F}^i| \geq N$ **then**
21. **break**
22. **end if**
23. **end for**
24. **return** $\boldsymbol{F}$

ENS 策略中，最开始的一步也是较为关键的一步是按照单个目标函数从小到大对候选解依次排序，使得排在后面的解至少有一个目标函数大于或等于排在前面的解。因此，排在后面的解与排在前面的解只存在两种关系，（1）排在前面

的解其目标函数都小于排在后面的解，即排在前面的解占优排在后面的解。（2）排在前面的解其目标函数有的小于排在后面的解，排在前面的解其目标函数有的大于排在后面的解，即排在前面的解与排在后面的解相互不占优。鉴于此，候选解只用和排在前面的解进行比较，排序在后面的解只可能与排序在前面的解处于同一前沿等级或者处于当前前沿等级的下一等级。这样就可以尽可能的减少比较次数，就得知自身所处与的占优等级。当已知其处于前沿等级的候选解个数大于所需要的个数的时候，算法终止；其算法流程如算法 4.1 所示。

4.1.3 ENS-DRA 流程

本小节主要对计算资源动态分配的高效非支配排序策略 (ENS-DRA) 的主要流程进行阐述和分析；该策略首先通过计算候选解的资源配置值，将资源配置值由大到小排列，选择出处于前 R 个的候选解，基于 Pareto 占优的概念，利用高效非支配排序策略，得知每个候选解与其他候选解相比的占优情况和非占优情况，为每个候选解分配非支配等级。

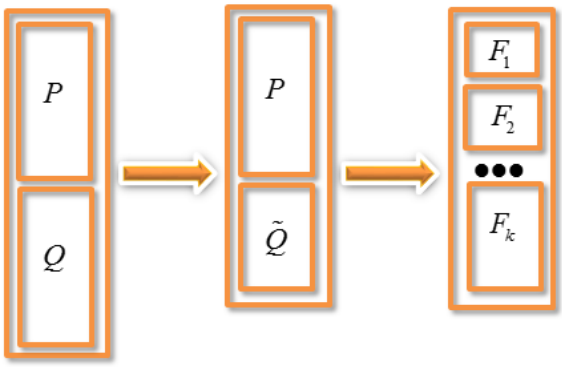


图 4-2 ENS-DRA 流程

算法 4.2： ENS-DRA $(\boldsymbol{P},\boldsymbol{Q},R,N)$

1. 输入：父代种群 $\boldsymbol{P}$，子代种群 $\boldsymbol{Q}$，分配计算资源候选解个数 R，种群大小 N
2. 输出：前沿集合 $\boldsymbol{F}$
3. $\tilde{N}=size(\boldsymbol{P})$
4. **for** $i\leftarrow 1$ to $\tilde{N}$ **do**
5. 计算每个 $\boldsymbol{Q}^i$ 的资源分配值 π^i
6. **end for**
7. $sort(\pi^i)$，π^i 由大到小进行排序
8. $\tilde{\boldsymbol{Q}}=\{\tilde{\boldsymbol{Q}}^{r_1},\tilde{\boldsymbol{Q}}^{r_2},...,\tilde{\boldsymbol{Q}}^{r_R}\}$ 其中 $\tilde{\boldsymbol{Q}}^{r_1},\tilde{\boldsymbol{Q}}^{r_2},...,\tilde{\boldsymbol{Q}}^{r_R}$ 是 π^i 最大的 R 个候选解
9. $\boldsymbol{P}\leftarrow\{\boldsymbol{P},\tilde{\boldsymbol{Q}}\}$
10. $\boldsymbol{F}\leftarrow \text{ENS}(\boldsymbol{P},\boldsymbol{F},N)$

以图 4-2 为例，较为形象的说明 ENS-DRA 流程，首先计算种群 $\boldsymbol{Q}$ 的资源分

配值，选择 R 个解组成候选解集 $\tilde{\boldsymbol{Q}}$，之后对 $\tilde{\boldsymbol{Q}}$ 和 $\boldsymbol{P}$ 内的所有解进行非支配排序，获取每一个解的前沿等级值。在 ENS-DRA 流程中，节约比较次数和比较时间的主要步骤在于两点。一是在算法 4.2 的第 8 行，通过计算资源分配值，选择了 R 个候选解和当前父代种群合并，而不是全部子代种群中的全部解都和父代种群合并，减少了比较的候选解数量，二是算法 4.2 的第 10 行，通过 ENS 策略，候选解仅仅和已经分配前沿等级的候选解进行比较，大大减少了比较次数，减少了时间复杂度。

4.2 MOSTA/P 算法描述

MOSTA/P 算法利用状态转移算子在决策空间内进行搜索，采用本章提出的基于计算资源动态分配的高效非支配排序策略（ENS-DRA）对候选解进行前沿等级划分，利用拥挤距离算子计算候选解的拥挤距离，根据候选解的前沿等级和拥挤距离选择出比较优的解，作为下一代的父代种群，并迭代上述过程直至达到算法终止条件。本小节对 MOSTA/P 的主要组成部分和流程进行详细阐述和说明。

4.2.1 候选解产生方式

本小节介绍在 MOSTA/P 算法中对候选解的产生方式进行说明。通过第三章对于状态转移算子搜索性能的研究，采用平移变换算子和坐标变换算子使用产生候选解集，候选解的产生方式可以用图 4-3 来说明，初始候选解集中的一个初始解，通过状态转移变换算子的作用，在决策空间在中进行采样，生成多个候选解，所有初始解的共同作用，最终生成候选解集。之后对候选解集进行修正，对于超过决策变量范围的解，进行一定的修正，使其映射到边界上，或者随机产生新的候选解，使得生成的候选解，完全在决策空间内。具体产生候选解的主要流程如算法 4.3 所示。

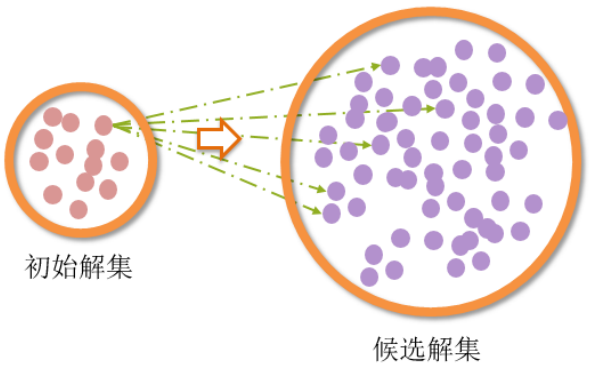


图 4-3 候选解产生方式

算法 4.3： $\boldsymbol{P} \leftarrow \text{GeneratePopulation}(\boldsymbol{X}, T, \beta, \delta)$

11. 输入：初始解集 $\boldsymbol{X}$，采样个数 T，状态转移算子参数 β, δ
12. 输出：候选解集 $\boldsymbol{P}$
13. **if** $rand() < r$ **do**
14. $\quad \boldsymbol{P} \leftarrow \text{axision}(\boldsymbol{X}, T, \beta)$
15. **else**
16. $\quad \{\boldsymbol{X}_1, \boldsymbol{X}_2\} \leftarrow \boldsymbol{X}$ 将 $\boldsymbol{X}$ 划分成候选解集数量相同的两个候选解子集
17. $\quad \boldsymbol{P}_1 \leftarrow \text{translation}(\boldsymbol{X}_1, T, \beta)$
18. $\quad \boldsymbol{P}_2 \leftarrow \text{axision}(\boldsymbol{X}_2, T, \beta)$
19. $\quad \boldsymbol{P} \leftarrow \{\boldsymbol{P}_1, \boldsymbol{P}_2\}$
20. **end if**
21. $\boldsymbol{P} \leftarrow \text{bound}(\boldsymbol{P}, X_{\min}, X_{\max})$
22. **return** $\boldsymbol{P}$

算法 4.4： $\boldsymbol{P} \leftarrow \textbf{bound}(\boldsymbol{P}, X_{\min}, X_{\max})$

1. 输入：候选解集 $\boldsymbol{P}$，决策变量下界 $X_{\min}$，决策变量上界 $X_{\max}$
2. 输出：修正后的候选解集 $\boldsymbol{P}$
3. **for** $i \leftarrow 1$ to **do**
4. $\quad$ **for** $j \leftarrow 1$ to M **do**
5. $\quad\quad$ **if** $P_j^i > X_{\min}$ or $P_j^i < X_{\max}$ **do**
6. $\quad\quad\quad$ **if** $rand() < r$ **do**
7. $\quad\quad\quad\quad$ 将超过决策变量范围的值映射到边界上
8. $\quad\quad\quad$ **else**
9. $\quad\quad\quad\quad$ 随机产生一个解替换 $\boldsymbol{P}^i$
10. $\quad\quad\quad$ **end if**
11. $\quad\quad$ **end if**
12. $\quad$ **end for**
13. **end for**
14. **return** $\boldsymbol{P}$

对于在搜索过程中，使用状态转移算子在决策空间搜索候选解时，决策变量的值可能超过优化问题对决策变量上下界的约束，此时就需要对候选解进行修正，算法 4.3 的第 11 行就是对候选解进行修正。当其某一维度的值超过上下界约束的时候，以一定概率将此维的值映射到上下边界上，或者放弃已经搜索到的候选解，在决策空间内随机产生一个候选解替代超过上下界约束的候选解。采用这样

的方式对候选解进行修正，首先是考虑到随机搜索的候选解一定要满足优化问题的上下界约束条件。其次是通过随机在决策空间内替换不满足上下界约束的候选解，增加了候选种群的多样性，避免了候选解集中的解在进化过程中过于相似，候选解在边界上的解较多等问题，从而陷入局部最优 Pareto 解集中；通过大量实验表明，当概率选择较为合理时，采用该种方式可以提高算法的的收敛性和分布性。

在候选解产生方式上，沿用了单目标状态转移算法中基于采样的思想，即一个初始解，通过状态转移算子变换，在其搜索空间内采样一定个数的解；由一个产生初始解产生多个候选解。因此，初始解集 X 的个数和采样个数 T 的乘积要等于候选解集 $\mathbf{P}$ 的个数 N 。即 $N = size(\mathbf{X}) \bullet T$ 。 r 是 0 到 1 的一个参数，取 0.5 比较适宜。

4.2.2 拥挤距离的多样性维护策略

基于 Pareto 占优框架的算法需要设计一定的多样性维护策略来保证候选解集在目标函数空间内的分布较为均匀。在 MOSTA/P 算法中是采用 NSGAII 提出的拥挤距离的多样性维护策略来保证解集分布的均匀性。该策略的的主要流程如算法 4.5 所示。

算法 4.5： $\boldsymbol{D} \leftarrow \text{Crowedistance}(\boldsymbol{P})$

1. 输入：候选解集 $\boldsymbol{P}$
2. 输出：拥挤距离 $\boldsymbol{D}$
3. $l = |\boldsymbol{P}|$
4. **for** $i \leftarrow 1$ to N **do**
5. $\quad \boldsymbol{I}(i) = 0$
6. **end for**
7. **for** $m \leftarrow 1$ to M **do**
8. $\quad \boldsymbol{P} = \text{sort}(\boldsymbol{P}, m)$
9. $\quad \boldsymbol{D}(1) = \boldsymbol{D}(l) = \infty$
10. $\quad$ **for** $i \leftarrow 2$ to $l-1$ **do**
11. $\quad\quad \boldsymbol{D}(i) = \boldsymbol{D}(i) + (\boldsymbol{D}(i+1)_m - \boldsymbol{D}(i-1)_m) / (f_m^{\max} - f_m^{\min})$
12. $\quad$ **end for**
13. **end for**
14. **return** $\boldsymbol{D}$

对于一个候选解，若其某个目标函数值是当前种群最大或者最小的值，则其相对拥挤距离设定为最大值∞。对于其他解而言，拥挤距离等于解在各个目标函数的方向上的前后两个目标函数差的绝对值并进行归一化。如果一个候选解与其邻近解在某个目标函数上相距较大，就会对总距离的大小有很大程度的影响。因此，在计算拥挤距离的时候，就需要对距离进行归一化。

4.2.3 算法更新方式

本小节主要对 MOSTA/P 算法的更新过程做出详细描述，算法更新过程综合使用上述基于计算资源动态分配的高效非支配排序策略与拥挤距离多样性维护策略，对候选解集进行更新，使得候选解集朝着 Pareto 最优解集进化。

在此，需要对计算资源动态分配的高效非支配排序策略（ENS-DRA）与拥挤距离多样性维护策略在选择候选解时的作用进一步说明。ENS-DRA 主要用于划分候选解集的前沿等级，拥挤距离多样性维护策略主要对候选解的分布性进行评估，若候选解处于较为密集的区域，其拥挤距离较小。相反地，若其处于较为稀疏的区域，其拥挤距离较大。在此，进行如下规定，若候选解 $\boldsymbol{x}^1$ 与候选解 $\boldsymbol{x}^2$ 处于同一前沿等级，则其根据拥挤距离判断候选解的优劣，拥挤距离大的解更优。若候选解 $\boldsymbol{x}^1$ 与候选解 $\boldsymbol{x}^2$ 处于不同同的前沿等级，则处于前沿等级高的候选解更优。基于上述规定，MOSTA/P 的更新方式如算法 4.6 所示：

算法 4.6： $\boldsymbol{P} \leftarrow \text{UpdatePopulation}(\boldsymbol{P}, \boldsymbol{Q}, R, N)$

1. 输入：父代解集 $\mathbf{P}$，子代解集 $\boldsymbol{Q}$，分配计算资源候选解个数 R，种群大小 N
2. 输出：已更新的解集 $\boldsymbol{P}$
3. $\boldsymbol{F} \leftarrow \text{ENS-DRA}(\boldsymbol{P}, \boldsymbol{Q}, R, N)$
4. $t \leftarrow size(\boldsymbol{F})$
5. $\boldsymbol{P} \leftarrow \{\boldsymbol{P}(\boldsymbol{F}^1), ..., \boldsymbol{P}(\boldsymbol{F}^{t-1})\}$ 将前沿等级 $t-1$ 的解放入 $\boldsymbol{P}_{new}$
6. $\mathbf{D} \leftarrow \text{Crowedistance}(\boldsymbol{P}(\boldsymbol{F}^t))$
7. $\text{sort}(\boldsymbol{D})$（按照拥挤距离从大到小排序）
8. $\boldsymbol{P}_{temp} \leftarrow \{\boldsymbol{P}(F_{r1}^t), \boldsymbol{P}(F_{r2}^t), ..., \boldsymbol{P}(F_{N-size(P)}^t)\}$（储存拥挤距离较大的前 $N - size(\boldsymbol{P}_{new})$ 个解）
9. $\boldsymbol{P} \leftarrow \{\boldsymbol{P}_{new}, \boldsymbol{P}_{temp}\}$
10. **return** $\boldsymbol{P}$

4.2.4 算法整体框架与流程

本小节综合上面对 MOSTA/P 算法的主要组成部分，详细讲述 MOSTA/P 算法的主要框架。首先，初始化候选解集，利用 ENS-DRA 策略为候选解集中的解划分前沿等级。然后，根据前沿等级及拥挤距离比较候选解，选择较优的一部分

解作为父代种群，利用平移变换和坐标变换产生新的候选解集，继续迭代，直至算法迭代终止条件满足，算法输出获得的最优解集。在迭代过程中，平移变换和坐标变换算子参数在迭代过程中一直处于动态变化中。这两算子分别以 fc_T，fc_A 的速率逐渐减少，到达参数值下界后，取参数上界并继续动态变化。算法整体流程描述如算法 4.7 所示，流程图如图 4-4 所示。

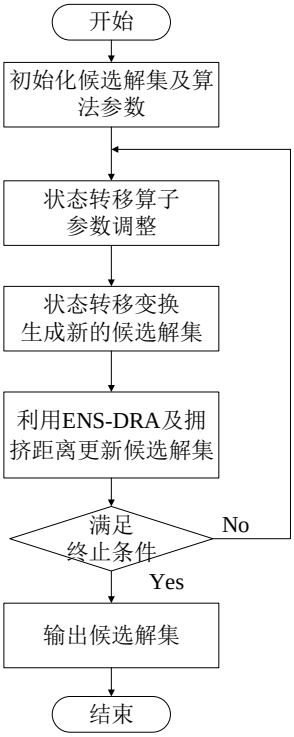


图 4-4 MOSTA/P 流程图

算法 4.7： MOSTA/P 算法框架

1. 种群大小 N，状态转移算子参数 β,δ，采样个数 T，资源分配个数 R
   状态转移算子参数上下界 $\beta_{max},\beta_{min},\delta_{min},\delta_{max}$
2. 初始化候选解种群 $\boldsymbol{P}\leftarrow\{\boldsymbol{x}^1,\boldsymbol{x}^2,...\boldsymbol{x}^N\}$ 计算其目标函数
   $\boldsymbol{FP}\leftarrow\{\boldsymbol{F}(\boldsymbol{x}^1),\boldsymbol{F}(\boldsymbol{x}^2),...\boldsymbol{F}(\boldsymbol{x}^N)\}$
3. **While** 终止条件不满足
4. **if** $\beta<\beta_{\min}$
5. $\beta=\beta_{\max}$
6. **end if**
7. **if** $\delta<\delta_{\min}$
8. $\delta=\delta_{\max}$
9. **end if**
10. 从 $\boldsymbol{P}$ 中随机选择一些解作为候选解产生的初始解集 $\boldsymbol{X}$
11. $\boldsymbol{P}\leftarrow$ GeneratePopulation$(\boldsymbol{X},T,\beta,\delta)$

12. $\boldsymbol{P} \leftarrow \text{UpdatePopulation}(\boldsymbol{P}, \boldsymbol{Q}, R, N)$
13. $\beta = \beta / fc_T$
14. $\delta = \delta / fc_A$
15. **end While**
16. **return** $\boldsymbol{P}$

4.3 对比实验结果及分析

为了说明本章提出的算法可以有效地解决多目标优化问题，本小节通过和 MOEA/D-DE，NSGAIII 及 NSLS 三种算法在 10 个测试函数上进行对比，采用 IGD 指标和 HV 指标对算法的收敛性和分布性进行评估；其中 MOEA/D-DE，NSGAIII 及 NSLS 算法的求解在 PlatEMO 算法平台上进行[79]。

4.3.1 实验设置

本章实验的对比三种算法在实验平台 PlatEMO 上进行[79]。PlatEMO 是由田野等学者在 Matlab 环境下，开发的多目标优化算法综合实验平台；该平台包含数十种多目标算法及百种多目标优化的测试问题，为多目标优化算法的对比实验操作提供了便利和支持。在本章算法实验对比时，采用 NSLS[36]，MOEA/D-DE[74]，NSGAIII[80]，这三种算法进行比较；其算法参数为 PlatEMO 实验平台的提供的默认参数。

在本次实验中，MOSTA/P 算法参数的设置如下：最大迭代次数 *Maxgen*=500 次，种群大小 N 为 200 个，采样个数 T 为 10，资源分配个数 R=3N/4，β 及 δ 初始值为 1，$\beta_{\max}$ 及 $\delta_{\max}$ 分别为 1 和 7，$\beta_{\min}$ 及 $\delta_{\min}$ 分别为 0.001 和 0.1，fc_T 及 fc_A 分别为 1.3 和 1.5。

本章所使用的测试函数为 MOP1-MOP4 及具有复杂前沿的 F1-F6 标准测试函数。不同算法在不同测试函数上，独立运行 30 次之后，利用已知的标准前沿，对多目标算法的 IGD 指标和 HV 指标进行统计，来表征算法获得最优解集的分布性和收敛性。统计结果包括最大值，最小值，平均值及标准差。其中，IGD 指标可以同时对收敛性和分布性进行评估，其值越小，表示算法的收敛性和分布性越好；HV 指标更加侧重于算法的分布性；算法获得前沿解分布越广泛，HV 值越大，其算法的分布性越好。本论文中，HV 指标参考点取标准前沿的 1.2 倍。

4.3.2 对比实验结果及分析

本小节对四种算法在 10 种测试问题上得到的最优解集的前沿图及 IGD 指标和 HV 指标的相关统计结果如图 4-5 至 4-14 所示；从前沿图与性能指标的统计结果看，MOSTA/P 是可以有效解决多目标优化问题，尤其面对低尾问题，与其

他算法想比，其收敛性和分布性的性能更优。

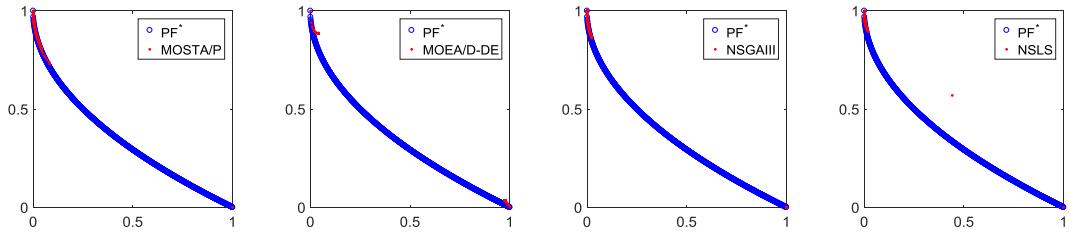


图 4-5 四种算法求解 MOP1 测试函数的前沿图

从 MOP1 的标准 Pareto 最优解来看，其决策变量的第一维为 0 或 1，决策变量的其他维为 $x_j = \sin(0.5\pi x_1)$；变量之间具有强烈的耦合，因此该问题是较难求解的。从四种算法求解 MOP1 测试问题获得前沿图看，四种算法在都没有完整获得 MOP1 测试问题的最优前沿；四种算法都找到了前沿的边界点 $(1,0)$ 和 $(0,1)$，其中 MOSTA/P 算法得到的前沿解的分布最广泛；就 IGD 指标的平均值来说，MOEA/D-DE 取得的平均值最优；而 IGD 指标的最大值看，MOSTA/P 最小，说明 MOSTA/P 可以一定程度上获得最优前沿，观察 IGD 指标的标准差及最大最小值来看，四种算法在求解 MOP1 问题都不稳定，这也是算法需要改进的地方；从 HV 指标上看 MOSTA/P 算法的平均值最大，说明平均来看 MOSTA/P 获得的最优解集的分布性最好。

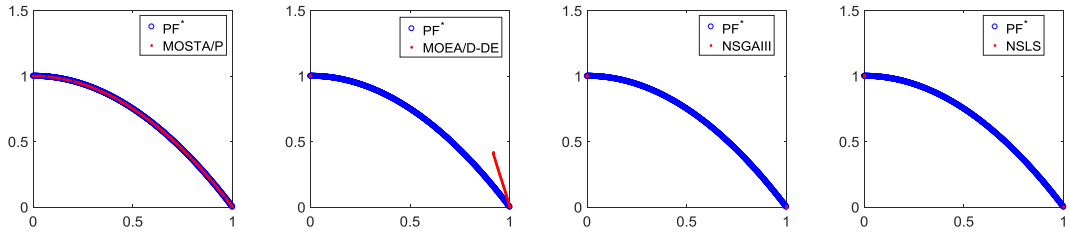


图 4-6 四种算法求解 MOP2 测试函数的前沿图

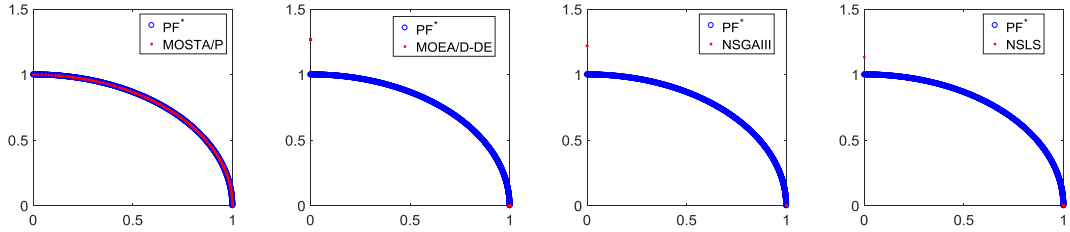


图 4-7 四种算法求解 MOP3 测试函数的前沿图

MOP2 测试问题的标准 Pareto 最优解与 MOP1 测试问题的最优解一致，因此也是较难求解的。从四种算法获得了 MOP2 测试问题前沿图来看，MOSTA/P 可以找到测试问题的全部前沿解，而 MOEA/D-DE 算法找到了边界 Pareto 最优

解，算法获得的其他解并未收敛至 Pareto 最优解集，其他两种算法仅仅找到了边界 Pareto 最优解。从 IGD 指标和 HV 指标的平均值看，MOSTA/P 与其他算法相比展现出了绝对的优势。但是相比之下，算法的稳定性还不足。

在求解 MOP3 测试问题时，MOSTA/P 获得的 Pareto 最优解集覆盖了整个标准前沿，其他算法获得的解都处于边界上或者并未收敛至 Pareto 最优解集中；从 IGD 和 HV 指标上看，MOSTA/P 无论是指标的最大值，平均值还是标准差，都是最好的，说明算法在求解 MOP3 测试问题稳定且最优。

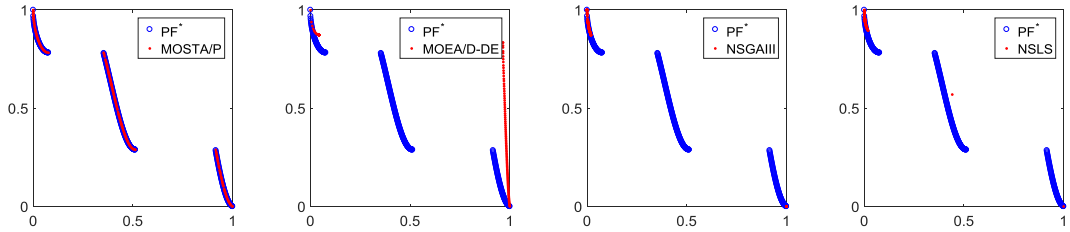


图 4-8 四种算法求解 MOP4 测试函数的前沿图

MOP4 测试问题的 Pareto 前沿是分 3 部分的，每部分目标函数的区间都较小，Pareto 前沿较为狭长。从四种算法求解 MOP4 问题的前沿图上看，MOSTA/P 算法获得的最优解集覆盖了整个 Pareto 前沿解，MOEA/D-DE 找到了一小部分解，大多数的解还未收敛到 Pareto 最优解集上。其他两个算法也仅仅找到了其中一下部分。从 IGD 和 HV 指标上看，MOSTA/P 算法的 IGD 指标最小，HV 指标最大，两个指标的标准差也小，说明 MOSTA/P 求解在 MOP4 问题较为稳定且是四个算法中最优的。

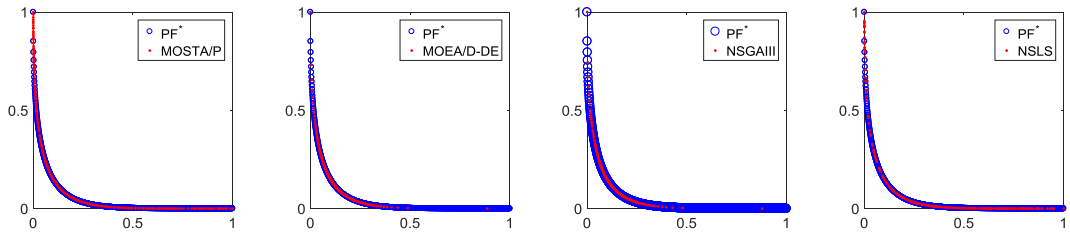


图 4-9 四种算法求解 F1 测试函数的前沿图

F1 测试问题是一种低尾问题，其标准 Pareto 前沿解之间的目标函数差距较小，Pareto 前沿整体变换较为平缓，且 Pareto 前沿的形状是非凸的；从四种算法求解 F1 测试问题所获得的前沿看，MOSTA/P 和 NSLS 获得的最优解集分布较为广泛，MOEA/D-DE 和 NSGAIII 获得的前沿解都没有完全覆盖最优前沿的尾部。从 IGD 和 HV 指标上看，MOSTA/P 的 IGD 平均值最小，HV 平均值最大，算法在求解 F1 问题时与其他算法相比，获得了较好的收敛性和分布性。

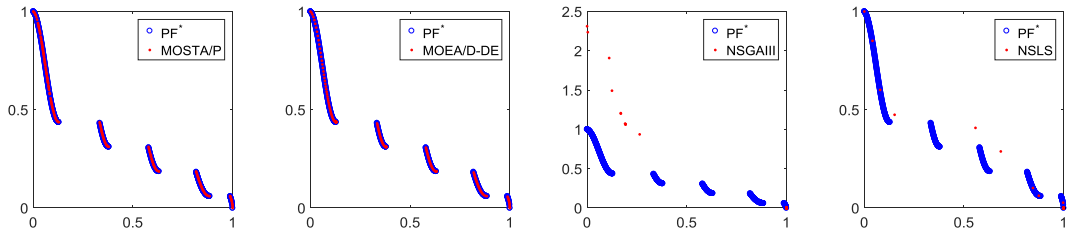


图 4-10 四种算法求解 F2 测试函数的前沿图

F2 测试问题的 Pareto 前沿是分段不连续的，且是多模态的一个测试问题。其前沿一共分为 5 段，且长度不等。从四个算法求解 F2 测试问题的前沿图来看，MOSTA/P 和 MOEA/D-DE 获得了较优的前沿曲线。NGSAIII 在算法终止时，其得到的解集并没有完全收敛到 Pareto 最优解上，NSLS 算法得到的最优解有一部分在 Pareto 标准前沿解上，分布在标准前沿上的解较少。从指标的统计结果上看上看，MOSTA/P 指标 IGD 指标和 HV 指标都是四个算法之中最优的。

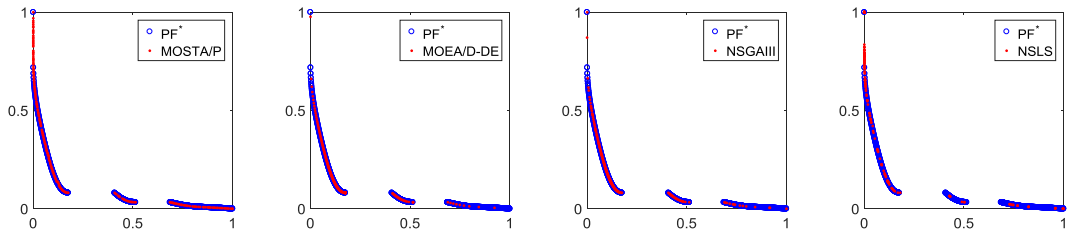


图 4-11 四种算法求解 F3 测试函数的前沿图

同样的，F3 测试函数的 Pareto 前沿也是多模态分段不连续的，与 F1 测试函数类似，从单个目标函数来看，其每一支 Pareto 前沿上的单个目标函数变化趋势也较为平稳。从四种算法得到的 Pareto 前沿图上看，除 MOSTA/P 算法外，其他函数获得的前沿与标准前沿相比，都有一定的损失。从指标上看，MOSTA/P 算法的 IGD 指标最小，HV 指标最大，是四个算法中获得最优解集分布性和收敛性最好的算法。

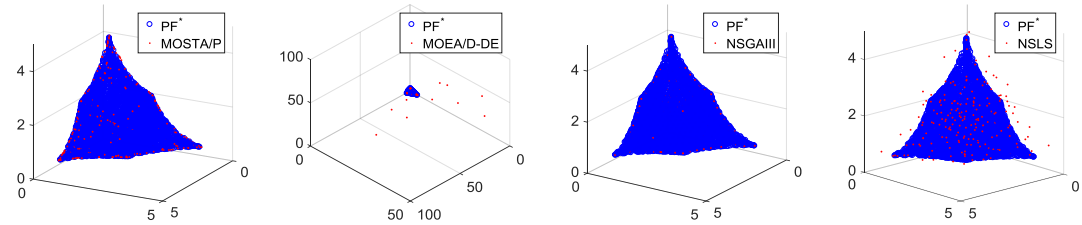


图 4-12 四种算法求解 F4 测试函数的前沿图

F4 测试函数是具有三个目标的非多模态测试问题，从四种算法求解该问题获得的前沿图上看，MOSTA/P 和 NSGAIII 得到的解都分布在标准 Pareto 前沿

上，NSLS 获得的解集并没有完全收敛至最优 Pareto 前沿解上，而 MOEA/D-DE 部分解与 Pareto 前沿解相差巨大。而通过指标来看，MOEA/D-DE 的 IGD 指标最小，表明其收敛性和分布性较好，而从获得的前沿图说明收敛性并不好，这两者是相互矛盾的；究其原因，是因为 IGD 指标的定义的局限性；由于 IGD 指标的定义为距离标准前沿值的平均最小距离，导致远离 Pareto 前沿的解的距离并未考虑进去，所以，MOEA/D-DE 虽然获得的前沿不是 Pareto 最优前沿，其 IGD 也较小的原因；对于 HV 指标，参考点设定为标准前沿解各个目标函数最大值的 1.2 倍，对于目标值大于参考点的，不进行计算，MOEA/D-DE 参与 HV 指标计算的解较少，这也是其 HV 指标最小的原因之一。整体说来，在求解 F4 测试问题上，NSLS 和 MOSTA/P 的都取得了不错的效果，这两种算法获得最优前沿的收敛性和分布性都较好。

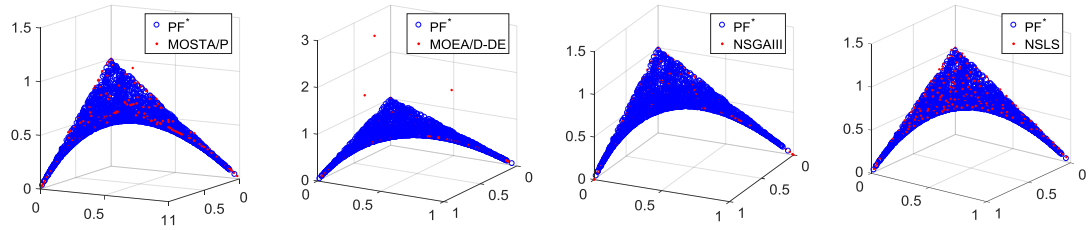


图 4-13 四种算法求解 F5 测试函数的前沿图

F5 测试函数是三目标优化问题，该问题的决策变量之间相互耦合，难以求解。从四种算法求解该问题获的的前沿图来看，MOEA/D-DE 算法获得的最优解集并未全部收敛至标准最优前沿上。MOSTA/P 有极个别点未收敛至 Pareto 前沿上，NSGAIII 和 NSLS 在求解该问题获的的前沿较好；从 IGD 指标和 HV 指标上看，最好的算法是 NSGAIII。

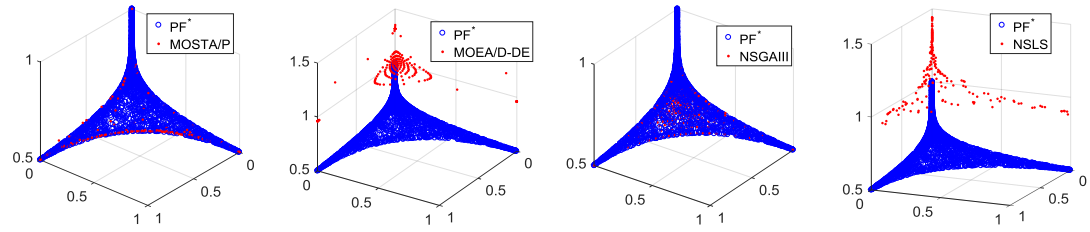


图 4-14 四种算法求解 F6 测试函数的前沿图

F6 测试函数是三目标的非多模态测试问题；从四种算法获得的最优前沿上看，NSGAIII 获得的前沿完全分布在标准 Pareto 前沿上，MOSTA/P 算法有少量解没有收敛至标准 Pareto 前沿上，MOEA/D-DE 获得解集收敛较差，NSLS 算法得到了标准 Pareto 前沿的形状，但是没有收敛至最优前沿。从指标上看，平均来

说，MOSTA/P 的 IGD 指标较小，HV 指标较大，算法获得最优前沿的收敛性和分布性较好。

表 4-1 四种算法求解测试函数的 IGD 统计结果

测试函数	算法	最大值	平均值	最小值	标准差
MOP1	MOSTA/P	**0.3062**	0.2648	0.2196	**0.0227**
	MOEA/D-DE	0.3523	**0.2600**	**0.1291**	0.0728
	NSGAIII	0.3540	0.3446	0.3357	0.0043
	NSLS	0.3591	0.2599	0.1774	0.0639
MOP2	MOSTA/P	**0.0027**	**0.0023**	**0.0021**	0.001
	MOEA/D-DE	0.3546	0.2503	0.1861	0.0521
	NSGAIII	0.3546	0.3546	0.3546	**5.5511e-17**
	NSLS	0.3546	0.3546	0.3546	5.6460e-17
MOP3	MOSTA/P	**0.0030**	**0.0024**	**0.0023**	**0.0002**
	MOEA/D-DE	0.5278	0.4569	0.2367	0.0511
	NSGAIII	0.5327	0.4795	0.4117	0.0309
	NSLS	0.4805	0.3809	0.2186	0.0684
MOP4	MOSTA/P	**0.1530**	**0.1529**	**0.1528**	**2.7852e-5**
	MOEA/D-DE	0.4035	0.2380	0.2021	0.0340
	NSGAIII	0.4217	0.4072	0.3802	0.0138
	NSLS	0.4197	0.4136	0.3924	0.0053
F1	MOSTA/P	**0.0032**	0.0028	0.0025	0.0002
	MOEA/D-DE	0.1327	0.0855	0.0473	0.0370
	NSGAIII	0.0507	0.0479	0.0292	0.0038
	NSLS	0.0034	**0.0026**	**0.0022**	**3.2828e-4**
F2	MOSTA/P	**0.0777**	**0.0776**	**0.0075**	**0.0001**
	MOEA/D-DE	0.6577	0.0983	0.0790	0.1056
	NSGAIII	0.5073	0.3632	0.2063	0.0865
	NSLS	0.2573	0.1452	0.1049	0.0371
F3	MOSTA/P	**0.0253**	**0.0251**	**0.0249**	0.0001
	MOEA/D-DE	0.0295	0.0292	0.0290	**9.9140e-5**
	NSGAIII	0.0322	0.0290	0.0279	9.7428e-4
	NSLS	0.0608	0.0373	0.0259	0.0091
F4	MOSTA/P	0.2402	0.1802	0.1547	0.0170
	MOEA/D-DE	0.1415	0.1372	0.1351	**0.0017**
	NSGAIII	**0.1132**	**0.1097**	**0.1043**	0.0018
	NSLS	0.2716	0.2197	0.1658	0.0297
F5	MOSTA/P	0.0362	0.0297	0.0238	0.0026
	MOEA/D-DE	0.0305	0.0301	0.0297	**2.0795e-4**
	NSGAIII	**0.0209**	**0.0201**	**0.0192**	3.5569e-4

	NSLS	0.0256	0.0239	0.0217	9.3525e-4
F6	MOSTA/P	**0.0331**	**0.0253**	**0.0180**	0.0034
	MOEA/D-DE	0.4536	0.3706	0.2218	0.0466
	NSGAIII	0.0390	0.0332	0.0313	**0.0013**
	NSLS	0.3623	0.3053	0.1899	0.0414

表 4-2 四种算法求解测试函数的 HV 统计结果

测试函数	算法	最大值	平均值	最小值	标准差
MOP1	MOSTA/P	0.7845	**0.7244**	0.6503	0.0391
	MOEA/D-DE	0.9400	0.7022	0.5403	0.1330
	NSGAIII	0.5869	0.5657	0.5442	0.0098
	NSLS	0.7975	0.6766	0.5285	0.0931
MOP2	MOSTA/P	**0.7708**	**0.7706**	**0.7699**	1.6425e-4
	MOEA/D-DE	0.5681	0.4980	0.4400	0.0416
	NSGAIII	0.4400	0.4400	0.4400	**2.7775e-16**
	NSLS	0.4400	0.4400	0.4400	2.8230e-16
MOP3	MOSTA/P	**0.6525**	**0.6523**	**0.6520**	**1.3214e-4**
	MOEA/D-DE	0.3905	0.2494	0.2400	0.0315
	NSGAIII	0.2991	0.2420	0.2400	0.0108
	NSLS	0.3061	0.2510	0.2400	0.0251
MOP4	MOSTA/P	**0.9569**	**0.9569**	**0.9567**	**4.8632e-5**
	MOEA/D-DE	0.6526	0.6078	0.5740	0.0207
	NSGAIII	0.6082	0.5781	0.5575	0.0114
	NSLS	0.5717	0.5599	0.5491	0.0059
F1	MOSTA/P	**1.3914**	**1.3912**	**1.3909**	**1.1832e-4**
	MOEA/D-DE	1.3901	1.3897	1.3893	3.5046e-4
	NSGAIII	1.3908	1.3901	1.3899	1.8628e-4
	NSLS	1.3911	1.3906	1.3898	3.0957e-4
F2	MOSTA/P	**1.1203**	**1.1202**	**1.1201**	**4.7343e-5**
	MOEA/D-DE	1.1200	1.0906	0.2400	0.1607
	NSGAIII	0.8283	0.4556	0.2400	0.1647
	NSLS	1.0086	1.0226	0.9286	0.0470
F3	MOSTA/P	**1.3588**	**1.3587**	**1.3586**	3.9926e-5
	MOEA/D-DE	1.3578	1.3575	1.3573	1.3452e-4
	NSGAIII	1.3583	1.3578	1.3575	**1.8546e-6**
	NSLS	1.3579	1.3560	1.3496	0.0017
F4	MOSTA/P	86.1943	85.5546	83.7451	0.4832
	MOEA/D-DE	85.7035	85.3821	84.9053	0.1949
	NSGAIII	**87.0399**	**87.0119**	**86.9814**	**0.0160**
	NSLS	83.5883	82.4864	80.9858	0.7654
F5	MOSTA/P	1.2612	1.2526	1.2408	0.0038

	MOEA/D-DE	1.2545	1.2528	1.2508	**7.9247e-4**
	NSGAIII	**1.2661**	**1.2633**	1.2610	0.0011
	NSLS	1.2521	1.2481	**1.2423**	0.0023
F6	MOSTA/P	0.9949	0.9924	0.9896	0.0012
	MOEA/D-DE	0.6018	0.3541	0.0025	0.0752
	NSGAIII	**0.9977**	**0.9976**	0.9975	**4.3259e-5**
	NSLS	0.6752	0.4600	0.0032	0.0692

4.4 本章小结

本章提出一种基于 Pareto 占优框架的多目标状态转移算法（MOSTA/P）用于求解具有箱型约束的多目标优化问题。在产生候选解集方面，该算法利用状态转移算子产生候选解集。在候选解比较方面，该算法充分考虑了候选解在进行 Pareto 占优比较所耗费的计算资源和时间资源，提出一种基于计算资源动态分配的高效非支配排序策略（ENS-DRA），候选解比较的次数和时间大大减小。同时，MOSTA/P 利用拥挤距离的多样性维护策略，剔除处于密集区域的解，使得算法获得的解集具有良好的分布性。利用 IGD 和 HV 这两种性能指标，和 MOEA/D-DE，NSLS 和 NSGAIII 四种算法在 10 种测试问题上进行比较，实验结果表明本章提出的 MOSTA/P 算法可以有效的解决多目标优化问题，该算法获得的候选解集具有良好的收敛性和分布性。

5 基于分解的多目标状态转移算法

基于 Pareto 占优框架的多目标优化算法是解决多目标优化问题的一种方式，除此框架外，众多学者还提出基于分解框架的多目标优化算法，将多目标优化问题转化成为多个等价的单目标优化子问题进行求解，每一个单目标优化子问题的最优解都对应原问题的一个 Pareto 最优解，最终获得多目标优化问题的 Pareto 最优解集。基于分解框架的多目标优化算法逐渐成为多目标智能优化算法领域热门的一个研究方向。本章基于分解框架，提出基于分解的多目标状态转移算法。

5.1 基于匹配度的修正切比雪夫聚合函数

根据聚合函数的目标函数值，对候选解进行比较，是基于分解框架的多目标优化算法在选择过程中的关键一步。聚合函数的目标函数值是否可以反映候选解的支配关系，决定了是否能选择出算法能否选择出非支配解，能否成功收敛于真实的前沿。本节通过分析现有聚合函数设计中存在的不足，提出一种基于匹配度的修正切比雪夫聚合函数；可以证明，本节提出的聚合函数分解方法与传统的切比雪夫聚合函数的最优值是一致的。

5.1.1 现有聚合分解函数的缺点分析

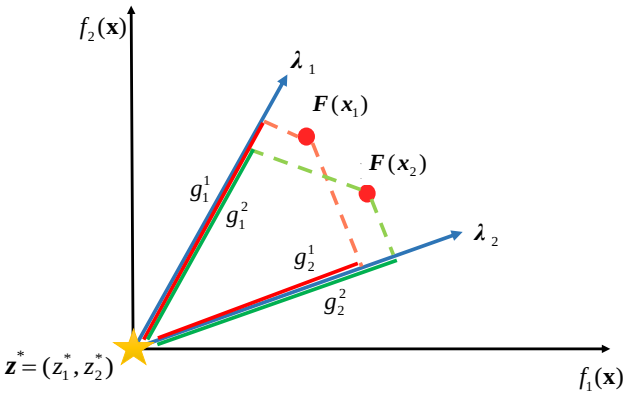


图 5-1 加权聚合函数选择过程示意图

加权聚合函数具有明显的几何意义，其目标函数就是候选解目标函数值与理想参考点组成的向量在权重向量上的投影长度。鉴于加权聚合函数这一突出的特点，本小节以加权聚合函数为例，对采用该函数进行候选解选择时的过程进行阐述，表明权重向量与候选解的匹配关系在设计聚合函数的重要性。

图 5-1 加权聚合函数选择过程示意图，其中 $\boldsymbol{x}_1$ 是原始种群的解之一，$\boldsymbol{x}_2$ 是新种群的解之一。$\boldsymbol{F}(\boldsymbol{x}_1)$ 和 $\boldsymbol{F}(\boldsymbol{x}_2)$ 分别是 $\boldsymbol{x}_1$ 和 $\boldsymbol{x}_2$ 的目标函数；$\boldsymbol{\lambda}_1$ 和 $\boldsymbol{\lambda}_2$ 是两个不同的权重向量。g_j^i 代表的是 $\boldsymbol{x}_j$ 与权重向量 $\boldsymbol{\lambda}_i$ 计算出的目标函数值；由于采用加权聚

合函数，故目标函数值的大小可以在图中明显的表示出来。图中，红线的长度表示为候选解 $\boldsymbol{x}_1$ 的聚合函数的目标值，绿线的长度表示候选解 $\boldsymbol{x}_2$ 的聚合函数的目标值。

从图 5-1 可以得出看出，如果 $\boldsymbol{x}_1$ 和 $\boldsymbol{x}_2$ 与权重向量 $\boldsymbol{\lambda}_1$ 匹配， g_1^1 大于 g_2^1 ，那么在更新过程中，$\boldsymbol{x}_1$ 优于 $\boldsymbol{x}_2$，$\boldsymbol{x}_1$ 不会被 $\boldsymbol{x}_2$ 替换；但是，$\boldsymbol{x}_1$ 和 $\boldsymbol{x}_2$ 与权重向量 $\boldsymbol{\lambda}_2$ 匹配，g_1^2 大于 g_2^2,优于 $\boldsymbol{x}_1$，那么在更新过程中，$\boldsymbol{x}_1$ 则被 $\boldsymbol{x}_2$ 替换。因此，候选解与不同的权向量进行匹配，对更新过程种群的进化有着很大的影响。同一个候选解与不同的权重向量匹配，更新过程的结果不同。

因此，在使用聚合函数对候选解进行选择的过程中，应仔细考虑解与权向量的匹配程度。在聚合函数设计中，应该考虑两者的匹配关系；然而，目前的聚合函数分解方法并没有明确强调权重向量和候选解之间的匹配关系，这在一定程度上影响了较优候选解在迭代过程中的保留。

5.1.2 基于匹配度的修正切比雪夫聚合函数

如前所述，在基于分解的算法中，权重向量与候选解的匹配程度是选择的关键。然而，目前的分解方法并没有明确强调权重向量和候选解之间的匹配关系。本文提出了一种基于匹配度的分解方法，综合考虑了切比雪夫方法、权重向量与，候选解的关系。可以证明，本节提出的聚合函数分解方法的获得的最优值与切比雪夫方法是等价的。

首先给出切比雪夫聚合函数相关定理。切比雪夫聚合函数表述如(5-1)所示，当权重向量，理想参考点确定之后，切比雪夫函数是以候选解得目标函数为自变量的函数，因目标函数的变化而变化。公式(5-1)中，$\boldsymbol{\lambda}=(\lambda_1,\lambda_2,...\lambda_M)$，$\boldsymbol{z}^*=(z_1^*,z_2^*,...,z_M^*)$ 为已知候选解各个目标的最小值

$$\min\ g^{\text{te}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)=\max_{1\le i\le M}\left\{\lambda_i\mid f_i(\boldsymbol{x})-z_i^*\mid\right\} \tag{5-1}$$

首先，关于切比雪夫聚合函数，有定理 1[53]：

定理 1 假设多目标优化问题的 Pareto 前沿是分段连续的。如果直线 L_1 与 Pareto 前沿存在一个交点，则该交点是标量问题 Q_1 的最优点；其中，Q_1 即为(5-1)所示的问题；直线 L_1 表达式如下：

$$\frac{f_1-z_1^*}{1/\lambda_1}=\frac{f_2-z_2^*}{1/\lambda_2}=...=\frac{f_M-z_M^*}{1/\lambda_M}(\lambda_i\neq 0,i=1,2,...,M) \tag{5-2}$$

在此之中，λ_i 是权重向量各个维度的数值。z_i^* 是第 i 维目标函数的理想值。

以两目标优化问题为例，通过如下所示的图 5-2，对上述定理给予解释。图中红色的线表示权重向量 $\boldsymbol{\lambda}=(\lambda_1,\lambda_2)$，该问题的直线 L_1 可表示为如(5-3)所示：

$$\frac{f_1-z_1^*}{1/\lambda_1}=\frac{f_2-z_2^*}{1/\lambda_2}(\lambda_i\neq 0) \tag{5-3}$$

与 Pareto 最优曲线的交点是权重向量 $\boldsymbol{\lambda}$ 下的切比雪夫函数的最优值，即图中红点表示权重向量 $\boldsymbol{\lambda}$ 聚合函数的最优值。

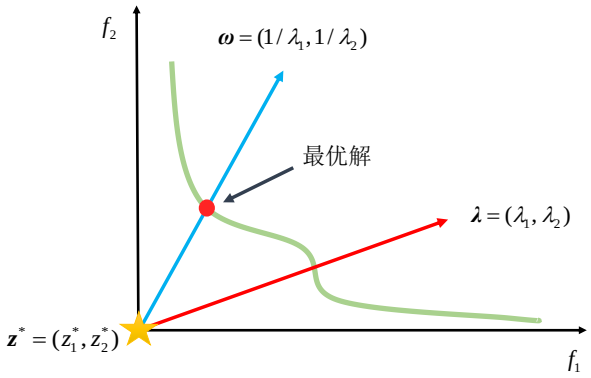


图 5-2 针对两目标优化问题的定理 1 示意图

定理 1 表明，一个解如果在直线 L_1 与 Pareto 前沿的交点上，那么该解一定是一个权重向量下的切比雪夫函数的最优值。定理 1 的逆否命题也成立，即一个解如果不是某个权重向量下的切比雪夫函数的最优值，则该点不是直线 L_1 与 Pareto 前沿的交点。那么，这个点的位置就要两种情况，一种是在直线上，另外一种不在直线上。判断该点的是否在直线上，可以引用数学中向量夹角余弦值的概念。

定义 1 （向量夹角余弦值）对于向量 $\boldsymbol{u}$ 和向量 $\boldsymbol{v}$，其夹角的余弦值 $\cos(\boldsymbol{u},\boldsymbol{v})$ 定义为：

$$\cos(\boldsymbol{u},\boldsymbol{v})=\frac{\boldsymbol{u}\bullet\boldsymbol{v}}{\|\boldsymbol{u}\|\bullet\|\boldsymbol{v}\|} \tag{5-4}$$

其中 $\|\bullet\|$ 表示一个向量的范数。若向量 $\boldsymbol{u}$ 和向量 $\boldsymbol{v}$ 处于一条直线上，则 $\cos(\boldsymbol{u},\boldsymbol{v})=\pm1$。若 $\cos(\boldsymbol{u},\boldsymbol{v})\neq\pm1$，则向量 $\boldsymbol{u}$ 和向量 $\boldsymbol{v}$ 不在一条直线上。

基于向量夹角余弦值的概念，在此，提出权重向量与候选解匹配度的概念。

定义 2 （匹配度）权重向量 $\boldsymbol{\lambda}$ 与候选解 $\boldsymbol{x}$ 匹配度 φ 定义为：

$$\varphi=\mathrm{h}(\boldsymbol{x}|\boldsymbol{\lambda},\boldsymbol{z}^*)=\left|\left\|\frac{(\boldsymbol{F}(\boldsymbol{x})-\boldsymbol{Z}^*)\boldsymbol{\omega}^T}{\|\boldsymbol{F}(\mathrm{x})-\boldsymbol{Z}^*\|\|\boldsymbol{\omega}\|}\right\|-1\right| \tag{5-5}$$

其中 $\boldsymbol{\omega}=(1/\lambda_1,1/\lambda_2,...,1/\lambda_M)$，$\boldsymbol{z}^*$为理想参考点。

从定义中可以看出，若候选解目标函数与理想参考点组成的向量与 $\boldsymbol{\omega}$ 方向一致，φ 为 0；候选解目标函数值与理想参考点组成的向量与权重向量的方向越接近，说明候选解与权重向量越匹配，φ 越小；反之，方向越远，两者越不匹配，φ 越大。因此，在设计聚合函数时，可以加入匹配度的信息，将候选解与权重向量的匹配程度考虑进去，使得选择过程更加的合理。

基于上述对于候选解与权重向量匹配度的分析，在本节提出基于匹配度的修正切比雪夫函数，作为基于分解的多目标状态转移算法中的分解方法。基于匹配度的修正切比雪夫函数表述如下：

$$\min\ g^{\text{tmd}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)=g^{\text{te}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)\bullet(1+h(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)) \tag{5-6}$$

其中，即一般定义下的切比雪夫聚合函数，$h(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)$ $\min\ g^{\text{tmd}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)=g^{\text{te}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)\bullet(1+h(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*))$ 即候选解和权重向量之间的匹配度。可以证明，基于匹配度的修正切比雪夫聚合函数与原切比雪夫聚 $g^{\text{te}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)$ 合函数函数的最优解是一致的。

定理 2 若 $g^{\text{te}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)$ 的最优解是 $\boldsymbol{x}_1^*$，$g^{\text{tmd}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)$ 的最优解是 $\boldsymbol{x}_2^*$，则 $\boldsymbol{x}_1^*=\boldsymbol{x}_2^*$。

证明：在 $\boldsymbol{\lambda},\boldsymbol{z}^*$ 一定时，$g^{\text{te}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)$ 的最优解是 $\boldsymbol{x}_1^*$，$g^{\text{tmd}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)$ 的最优解是 $\boldsymbol{x}_2^*$；那么由 $g^{\text{te}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)$ 和 $g^{\text{tmd}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)$ 的函数构造可知，$h(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)\geq 0$，故 $g^{\text{tmd}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)\geq g^{\text{te}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)$；所以 $g^{\text{tmd}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)$ 是有下界的；且当 $h(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)=0$ 时，$g^{\text{tmd}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)$ 函数取得最小值；$h(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)=0$ 时，说明候选解的目标函数与理想参考点组成的向量与 $\boldsymbol{\omega}=(1/\lambda_1,1/\lambda_2,...,1/\lambda_{\text{M}})$ 是共线的，由定理 1 知，此时 $\boldsymbol{x}=\boldsymbol{x}_1^*$；当 $\boldsymbol{x}=\boldsymbol{x}_1^*$ 时，$h(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)=0$，$g^{\text{tmd}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)=g^{\text{te}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)$，$g^{\text{tmd}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)$ 取得最小值，即 $\boldsymbol{x}_1^*$ 为 $g^{\text{tmd}}(\boldsymbol{x}\mid\boldsymbol{\lambda},\boldsymbol{z}^*)$ 的最优值，故 $\boldsymbol{x}_1^*=\boldsymbol{x}_2^*$。

由上述证明得知，本节提出的基于匹配度的修正切比雪夫聚合函数与原切比雪夫函数的最优值和最优解是一致等价的。

在采用基于匹配度的修正切比雪夫分解函数作为多目标优化的分解方式时，在选择过程中，不仅仅考虑了目标函数与权重向量分量之间的乘积大小，还考虑了候选解与权重向量的匹配关系，减少了因权重向量不匹配问题，造成的较好候选解没有被选择的不足。

5.2 MOSTA/D 算法描述

本小节对基于分解框架的多目标状态转移算法 MOSTA/D 的主要内容和流程进行阐述，MOSTA/D 算法利用状态转移算子在决策空间内进行搜索，采用本章提出的基于匹配度的修正切比雪夫函数，将多目标优化问题分解为若干个单目标优化问题进行优化，最终获得分布性较好的 Pareto 最优解集。

5.2.1 候选解产生方式

多目标优化算法要求最终获得的候选解具有较好的分布性，要均匀分布在最优 Pareto 前沿上，这对于算法的搜索能力相比于单目标优化算法而言，有了更高的要求。从基于分解的多目标状态转移算法实际设计过程中发现，产生的候选解是否均匀分布在整个搜索空间，对于实验结果有着较大的影响。现有的状态转移算子，如旋转变换算子，伸缩变换算子，平移变换算子产生的候选解与初始解相

差较大；而坐标变换算子新产生的候选解的与初始解仅有一维不同，而过于相似；在算法搜索过程中，即希望通过状态转移变换产生的新候选解于原候选解有一定的相似性，又希望候选解与初始解相比，有一定的不同，故基于状态转移算子产生框架，提出一种新的状态转移算子，同化状态转移算子，并利用此算子结合原有状态转移算子，完成基于分解的状态转移算法的搜索过程。

同化状态转移算子 （Assimilation Transformation）的描述如下：

$$\boldsymbol{x}_{k+1} = \boldsymbol{R}_s \boldsymbol{x}_k^1 + \boldsymbol{B}_s \boldsymbol{x}_k^2 \tag{5-7}$$

其中，$\boldsymbol{x}_k^1, \boldsymbol{x}_k^2$，是当前候选解集中的两个解，其维度为$n$，是一个$n\times1$的列向量；$\boldsymbol{R}_s$是一个$1\times n$的行向量，除第$r$个元素为 0 外，其他元素均为 1；$\boldsymbol{B}_s$也是一个$1\times n$的行向量，除第$r$个元素为 1 外，其他元素均为 0，$r$是 1 至$n$的任意一个值；通过同化状态转移算子的变换，新产生的候选解$\boldsymbol{x}_{k+1}$其$r-1$个元素和$\boldsymbol{x}_k^1$相同，其 1 个元素与$\boldsymbol{x}_k^2$相同，即$\boldsymbol{x}_{k+1}$是$\boldsymbol{x}_k^1$和$\boldsymbol{x}_k^2$同化的结果。

在 MOSTA/D 算法中，同化算子的使用方式表述如下：对于种群大小为N的初始解，首先从当前候选解集$\boldsymbol{P}$中随机选择t个解作为初始解集$\boldsymbol{X}$，$t = N/T$，T是人为设定的采样个数，对初始解集$\boldsymbol{X}$复制T个，使其维度与$\boldsymbol{P}$相同，初始解$\boldsymbol{X}$和当前种群$\boldsymbol{P}$中的每一个解在同化算子的作用下，产生了新的种群$\boldsymbol{P}'$。下面以候选解变量维数为 3 维的候选解，对同化算子的作用进行说明。

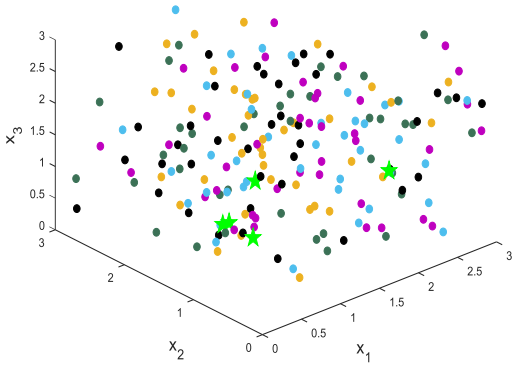


图 5-3 同化变换前的种群

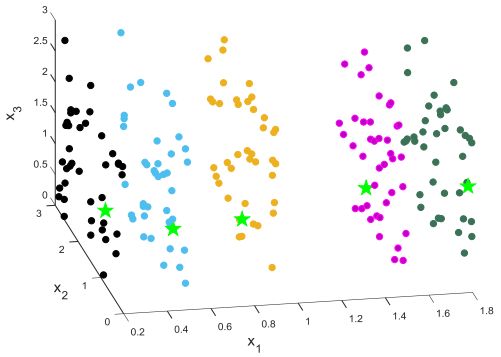


图 5-4 同化变换后的种群

图 5-3 和 5-4 显示了种群大小为 200 个的种群在同化变换算子前和同化变换算子后的比较示意图；图 5-3 中 5 个绿色五角星表示从当前候选解集中随机选取的 5 个解，采样个数$T=40$，其他原点表示当前候选解集；经过同化算子的变换，候选解集中的解某一维度与随机选取的 5 个解中的一个相同，新产生的候选解被 5 个候选解分为 5 组，在图 5-4 中用不同的颜色表示；原来无序的候选解集转换为较为有序的候选解集，这样使得在某一维度值固定下可以进行较为深度的搜索。在 MOSTA/D 算法中，同化算子的使用与迭代过程的进展有关，分别在迭代初期

和迭代后期使用；在迭代初期使用使得可以快速获得某一维度下的较优解集，在迭代后期使用使得候选解在某一维度进行更细致的搜索。在此之间，采用旋转变换算子进行搜索。除此之外，MOSTA/D 在产生候选解上，也使用了平移变换算子和坐标变换子共同使用，之后采用旋转变换算子进行修正的方式。

MOSTA/D 采用两种产生候选解集的方式，一种利用同化变换算子和旋转变换算子产生不同的解，另外一种是利用平移变换算子和坐标变换算子共同产生候选解集，最后使用旋转变换算子对每一个候选解进行修正；在算法实际使用中，两种方式交替使用，使得算法具有较强的搜索性能，其搜索空间更加广泛。

具体产生候选解的主要流程如下：

算法 5.1： $\boldsymbol{P} \leftarrow \text{GeneratePopulation1}(\boldsymbol{X}, T, \alpha, gen, X_{max}, X_{min})$

1. 输入：大小为 N 初始解集 $\boldsymbol{X}$，采样个数 T，状态转移算子参数 α，前迭代次数 gen 决策变量的下界 $X_{\min}$ 决策变量的上界 $X_{\max}$
2. 输出：候选解集 $\boldsymbol{P}$
3. 从 $\boldsymbol{X}$ 中随机选择 N/T 个候选解作为初始解集 $\boldsymbol{X}'$
4. **if** $gen < gen_1 \,\|\, gen > gen_2$ **do**
5. $\mathbf{P} \leftarrow \text{Assimilation}(\boldsymbol{X}', T, \boldsymbol{X})$
6. **else**
7. $\boldsymbol{P} \leftarrow \text{Rotation}(\boldsymbol{P}, T, \alpha)$
8. **end if**
9. $\boldsymbol{P} \leftarrow \text{bound}(P, X_{\min}, X_{\max})$
10. **return** $\boldsymbol{P}$

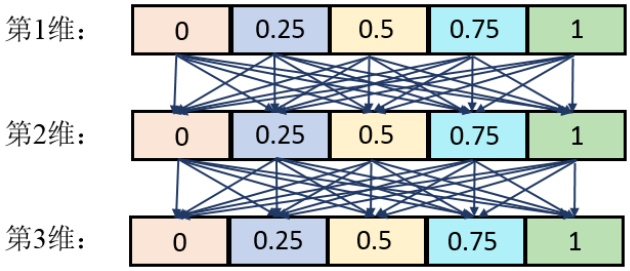
算法 5.2： $\boldsymbol{P} \leftarrow \text{GeneratePopulation2}(\boldsymbol{X}, T, \alpha, \beta, \delta, X_{max}, X_{min})$

1. 输入：初始解集 $\boldsymbol{X}$，采样个数 T，状态转移算子参数 α, β, δ，策变量的下界 $X_{\min}$，策变量的上界 $X_{\max}$
2. 输出：候选解集 $\boldsymbol{P}$
3. 从 $\boldsymbol{X}$ 中随机选择 N/T 个候选解作为初始解集 $\mathbf{X}'$
4. $\boldsymbol{P}_1 \leftarrow \text{Transation}(\boldsymbol{X}', T, \beta)$
5. $\boldsymbol{P}_2 \leftarrow \text{Axesion}(\boldsymbol{X}', T, \delta)$
6. $\boldsymbol{P} \leftarrow \{\boldsymbol{P}_1, \boldsymbol{P}_2\}$
7. $\boldsymbol{P} \leftarrow \text{Rotation}(\boldsymbol{P}, 1, \alpha)$
8. $\boldsymbol{P} \leftarrow \text{bound}(\boldsymbol{P}, X_{\min}, X_{\max})$
9. **return** P

5.2.2 权重向量产生方式与邻域确定

基于分解框架的多目标优化算法利用权重向量和分解函数将多目标的优化问题转化为多个单目标优化问题进行求解，其产生方式对后续候选解的选择和分布有着较大影响。权重向量邻域的范围与权重向量的分布有关，领域的确定对候选解的选择过程具有较大的影响。本小节主要介绍在 MOSTA/D 中，权重向量产生方式，及如何通过权重向量或其他方法确定权重向量邻域。

MOSTA/D 中，权重向量通过在单维超平面采样产生；当目标函数的个数为 m 时，需要在 $m-1$ 维的 $[0,1]$ 区间内以采样间隔 $1/H$ 进行采样，最终获得 $N=C_{H+m-1}^{m-1}$ 个权重向量。对于权重向量 $\boldsymbol{\lambda}=(\lambda_1,\lambda_2,...,\lambda_m)$ 而言，其每个元素 $\lambda_i, i\in\{1,2,...,m\}$ 其值在采样集合 $\{0,1/H,2/H,...,H/H\}$ 中，且各个元素之和要为 1。下面以 $H=4$ 为例，对权重向量的产生进行说明。



(0,0,1) (0,0.25,0.25) (0,0.5,0.5) (0,0.75,0.25) (0.25,0.25,0.5)
(0,1,0) (0.75,0,0.25) (0.5,0,0.5) (0.25,0,0.75) (0.25,0.5,0.25)
(1,0,0) (0.25,0.75,0) (0.5,0.5,0) (0.75,0.25,0) (0.5,0.25,0.25)

图 5-5 三维权重向量产生示意图

如图 5-5 所示，在区间 $[0,1]$ 内，每一维采样 $H=4$ 个点，则每一维度可以取值的集合为 $\{0,0.25,0.5,0.75,1\}$，权重向量 $\boldsymbol{\lambda}=(\lambda_1,\lambda_2,...,\lambda_m)$ 每一维度的取值是集合元素的组合，且满足元素之和为 1 的约束条件；通过计算，当 $H=4$，$m=3$ 时，共计得到 21 个权重向量，这些权重向量显示在图 5-5 中。

权重向量邻域大小是指一个候选解可以匹配权重向量的个数，设候选解 $\boldsymbol{x}$ 的更新邻域大小为 T，即候选解 $\boldsymbol{x}$ 至多和 T 个权重向量进行匹配；权重向量的初始邻域通过权重向量之间的欧式几何距离确定。具体来说，对于候选解 $\boldsymbol{x}$ 来说，对应一个权重向量 $\boldsymbol{\lambda}$，距离 $\boldsymbol{\lambda}$ 几何距离最近的 T 个权重向量组成权重向量 $\boldsymbol{\lambda}$ 的邻域，候选解 $\boldsymbol{x}$ 在选择过程中，和这 T 个权重向量进行匹配；MOSTA/D 算法过程中，采用动态权重向量邻域机制；随机产生一个随机数 r，当 $r<0.5$ 时，邻域确定方式如上所述；当 $r\geq 0.5$ 时，权重向量 $\boldsymbol{\lambda}$ 的邻域是随机从权重向量中选择 T 个权重向量组成的，使用动态权重向量邻域机制从全局加快了算法收敛进程，加快了算法收敛速度。

5.2.3 算法更新方式

MOSTA/D 算法中，候选解的比较采用本章提出的基于匹配度的且比雪夫函数，在选择的同时也加强了搜索过程。具体来说，MOSTA/D 的更新过程如算法 5.3 所示，通过事先根据权重向量邻域确定的候选解更新邻域 $\boldsymbol{B}$，在候选解更新邻域内通过利用 $g^{\text{tmd}}(\boldsymbol{x}\,|\,\boldsymbol{\lambda},\boldsymbol{z}^{*})$ 函数，对父代候选解集和子代候选解集进行比较时，若父代候选解劣于子代候选解，则利用父代候选解和子代候选解作为初始解，通过平移变换算子产生新的候选解，继续进行比较，根据比较结果，更新父代候选解，保证父代中的候选解是当前找到的最优的候选解。

算法 5.3： UpdatePopulation

```
1.  for i ← 1 to N do
2.      for j ← 1 to T do
3.          if g^tmd(x^{B(j)} | λ^{B(j)}, z*) > g^tmd(y^i | λ^{B(j)}, z*) then
4.              v ← translate(x^{E(j)}, y^i, β)
5.              if g^tmd(y^i | λ^{B(j)}, z*) > g^tmd(v | λ^{B(j)}, z*) then
6.                  x^{E(j)} ← v
7.              else
8.                  x^{E(j)} ← y^i
9.              end if
10.         end if
11.     end for
12.  end for
13. return P
```

5.2.4 算法整体框架

本小节对 MOSTA/D 算法的主要框架进行介绍；MOSTA/D 整个算法流程中，通过状态转移算子在整个决策空间内进行搜索，利用权重向量，采用基于匹配度的修正切比雪夫函数对候选解进行选择。最终获得优化问题的近似最优解集。

更详细地，MOSTA/D 算法首先进行初始化，设置算法相关参数，产生权重向量，并确定候选解的更新领域；之后是迭代更新过程，利用状态转移算子产生子代解集，在候选解更新邻域内利用基于匹配度的修正切比雪夫进行比较；值得注意的是，候选解更新邻域在算法中是动态变化的，使得候选解既可以在确定邻域更新，也可以与其他候选解进行比较更新，使得候选解集更新范围更广泛；此外，如果经过比较，发现了更优的解，则利用状态转移算子进行二次搜索。再进行选择过程。旋转变换算子及平移变换算子的参数在迭代过程中，一直处于有规

律的动态变化之中；这三个算子分别以 fc_R， fc_T 的速率逐渐减少，到达参数值下界后，取参数上界并继续动态变化。MOSTA/D 算法的算法流程框架及算法流程如算法 5.4 所示：

算法 5.4： MOSTA/D 算法框架

1. 初始化权重向量集合 $\boldsymbol{\lambda}=(\lambda_1,\lambda_2,...,\lambda_m)$ 及算法参数 N,T,α,β,δ，状态转移算子参数上下界 $\alpha_{max},\alpha_{min},\delta_{min},\delta_{max}$
2. 初始化候选解种群 $\mathbf{P}\leftarrow(\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_n)$ 计算其目标函数 $\boldsymbol{FP}\leftarrow(\boldsymbol{F}(\boldsymbol{x}_1),\boldsymbol{F}(\boldsymbol{x}_2),...,\boldsymbol{F}(\boldsymbol{x}_n))$
3. 初始化理想参考点 $\boldsymbol{z}^*\leftarrow\{z_1^*,z_2^*,...z_m^*\}$
4. **for** $i\leftarrow 1$ to N **do**
5. $\boldsymbol{B}^i\leftarrow\{i_1,i_2,...i_T\}$,其中 $\boldsymbol{\lambda}^{i_1},\boldsymbol{\lambda}^{i_2},...\boldsymbol{\lambda}^{i_T}$ 是距离 $\boldsymbol{\lambda}^i$ 欧式距离最近的 T 个权重向量
6. **end for**
7. **While** 终止条件不满足
17. **if** $\alpha<\alpha_{min}$
18. $\alpha=\alpha_{max}$
19. **end if**
20. **if** $\delta<\delta_{min}$
21. $\delta=\delta_{max}$
22. **end if**
8. 随机从 $\boldsymbol{P}$ 中选择 N/T 个解做为初始解集 $\boldsymbol{X}$
9. 交替使用 GeneratePopulation1 和 GeneratePopulation2 产生子代种群 $\boldsymbol{Q}$
10. **for** $i\leftarrow 1$ to N **do**
11. **if** $rand()<r$ **then**
12. $\boldsymbol{B}\leftarrow\{r_1,r_2,...r_T\}$ 其中 $r_1,r_2,...r_T$ 是 $\{1,2,...,N\}$ 中随机选择的 T 个数
13. **end if**
14. $\boldsymbol{z}^*\leftarrow\min(\boldsymbol{z}^*,\boldsymbol{F}(\boldsymbol{Q}))$
15. $\boldsymbol{P}\leftarrow\textbf{UpdatePopulation}(\boldsymbol{P},\boldsymbol{Q},\boldsymbol{z}^*,\boldsymbol{B},\boldsymbol{\lambda},\beta)$
16. **end for**
17. $\delta=\delta/\ fc_T$
18. $\alpha=\alpha/\ fc_R$
19. **end While**
20. **return** $\boldsymbol{P}$

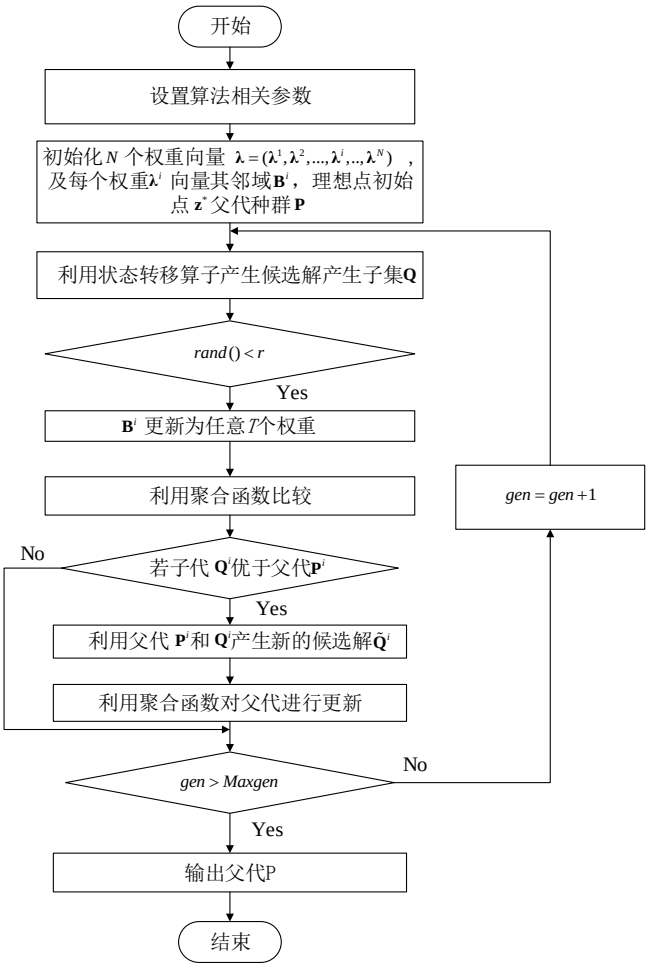


图 5-6 MOSTA/D 算法流程图

5.3 对比实验结果及分析

本小节利用 IGD 指标和 HV 指标，利用复杂前沿的 F1-F4 测试函数和 MOP1-MOP4 测试函数，通过和三种经典算法进行对比，对本章提出的算法 MOSTA/D 算法的性能进行评估。此外，为了验证本章提出的基于匹配度的修正切比雪夫函数的有效性，在算法其他部分与 MOSTA/D 保持一致的前提下，设计了一种采用切比雪夫分解方法的多目标状态转移算法进行比较。为了区别这两种算法，采用切比雪夫分解方法的多目标状态转移算法标记为 MOSTA/D-DCHE，实验结果和分析如图 5-7 至图 5-14 所示。

5.3.1 实验设置

本章实验的对比三种算法在实验平台 PlatEMO 上进行[79]。在本章算法实验对比时，采用 NSLS[36]，MOEA/D-DE[74]，NSGAIII[80]这三种算法进行比较；其算法参数为 PlatEMO 实验平台的提供的默认参数。

在本次实验中，MOSTA/D 算法参数的设置如下：最大迭代次数 *Maxgen* 设置为 500 次，目标函数最大计算次数为 10^5 次，种群大小 N 为 200 个，采样个数

T 为 20，α 初始值为 4，δ 初始值设置为 1，β 初始值设置为 10，$\delta_{\max}=1$，$\alpha_{\max}$ 与优化问题的维度相关，$\alpha_{\max}=0.75D$，其中 D 是多目标优化问题决策变量的维数，$\alpha_{\min}=1$，$\delta_{\min}=0.3$，$fc_T=1.2$，$fc_R=2$。

算法在不同测试函数上，独立运行 30 次之后，利用已知的标准前沿，对 IGD 指标和 HV 指标进行统计，来表征算法获得最优解集的分布性和收敛性。统计结果包括最大值，最小值，平均值及标准差。其中，IGD 指标可以同时对收敛性和分布性进行评估，其值越小，表示算法的收敛性和分布性越好；HV 指标更加侧重于算法的分布性；算法获得前沿解分布越广泛，HV 值越大，其算法的分布性越好。HV 指标的参考点取最优前沿中目标函数最大值的 1.2 倍。

5.3.2 实验结果

本小节利用 8 种测试函数，将本章提出的算法与第四章提出的算法和 MOSTA/D-DCHE，及 MOEA/D-DE，NSGAIII，NSLS 四种算法进行比较，实验结果及分析如下：

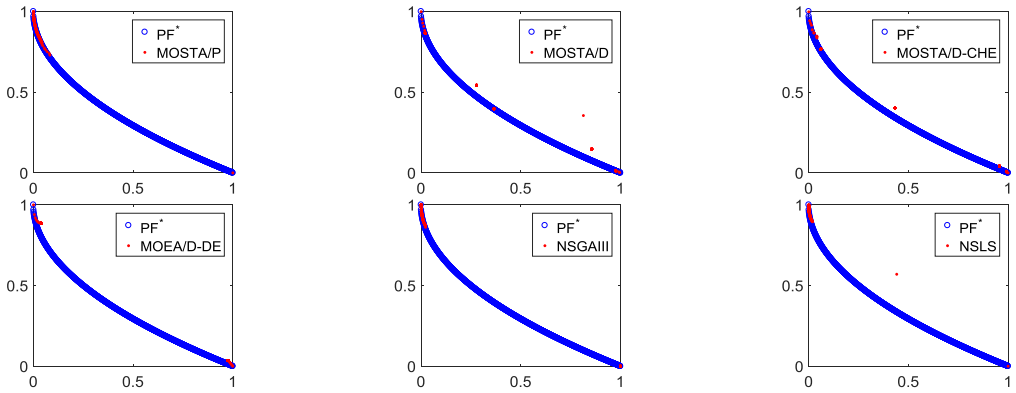


图 5-7 六种算法求解 MOP1 测试函数的前沿图

在求解 MOP1 问题时，6 种算法都未能成功找到全部的前沿；其中，第四章提出的算法 MOSTA/P 找到的 Pareto 前沿较多，MOEA/D-DE，NSGAIII，NSLS，MOSTA/D-CHE 四种算法找到的 Pareto 前沿解大多集中在在前沿两端，中部较少，或几乎没有。MOSTA/D 在中部有着较多的解并未收敛至 pareto 前沿解，但这些解的出现表明，MOSTA/D 还是有潜力求解 MOP1 测试问题。从 IGD 指标和 HV 指标上看，本章提出的算法的平均 IGD 指标最小，HV 指标指标最大，说明候选解的分布更加广泛，收敛性也较好。MOSTA/D 的性能指标的统计结果都优于 MOSTA/D-CHE 算法，表明本章提出的基于匹配度的修正切比雪夫分解方法是有效的。

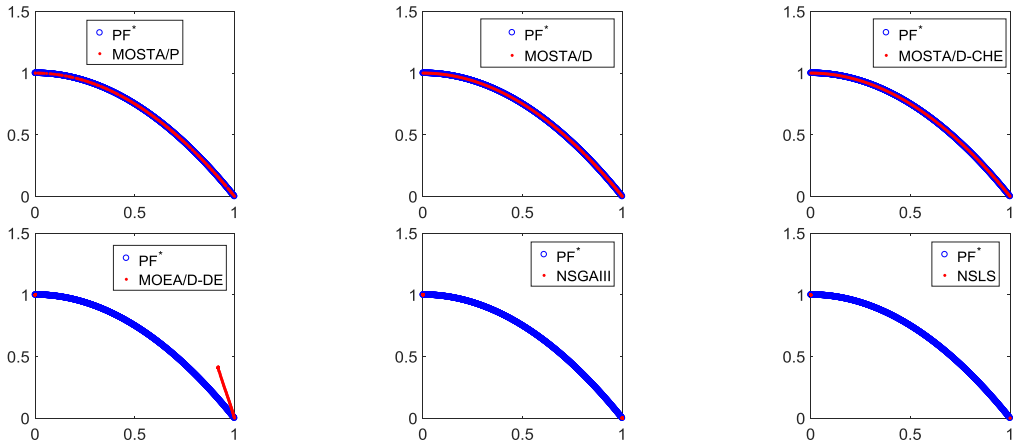


图 5-8 六种算法求解 MOP2 测试函数的前沿图

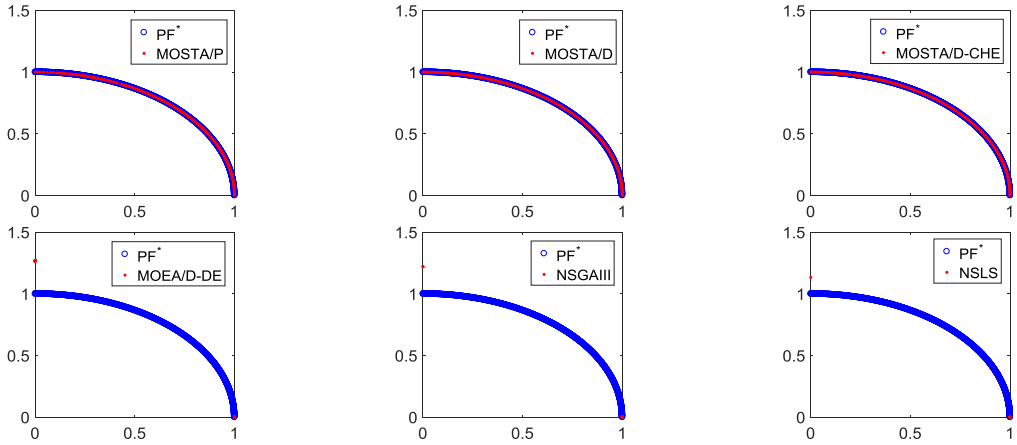


图 5-9 六种算法求解 MOP3 测试函数的前沿图

在求解 MOP2 测试问题的前沿图上看出，本论文提出的 MOSTA/P，MOSTA/D，及变种 MOSTA/D-CHE 算法都可以成功解决 MOP2 测试问题，而 MOEA/D-DE，NSGAII 及 NSLS 未能成功解决 MOP2 测试问题，其中，MOEA/D-DE 找到的最优解集分布较广，但是众多候选解未能收敛至 Pareto 前沿上。NSGAIII 和 NSLS 两种算法，仅仅找到了 Pareto 前沿的边界点。从 IGD 和 HV 性能指标上看，本章提出的算法 MOSTA/D 与 MOSTA/D-CHE 算法获得的平均 IGD 指标最小，HV 指标最大，处于第二位是本论文第三章提出的 MOSTA/P，在该测试问题上，本论文提出算法具有显著的优越性。

在求解 MOP3 问题时，与 MOP2 测试问题求解情况类似，仅仅本论文提出的 MOPSTA/P, MOSTA/D，及其变种 MOSTA/D-CHE 可以成功求解该问题。MOEA/D-DE，NSGAIII，NSLS 算法仅仅找到到了最优前沿两端的值,还有部分解没有收敛至 Pareto 最优前沿中；造成这种原因是算法最终找到的候选解集各不相同而他们的目标函数值是相同的，继而造成 Pareto 前沿只有两端解存在的情况。从 IGD 和 HV 指标上看，本章提出的算法 MOSTA/D 的 IGD 指标最小，

HV 指标最大，且指标比基于切比雪夫分解方法的 MOSTA/D-CHE 算法优，故本章提出的基于匹配度的修正切比雪夫聚合函数是有效的。

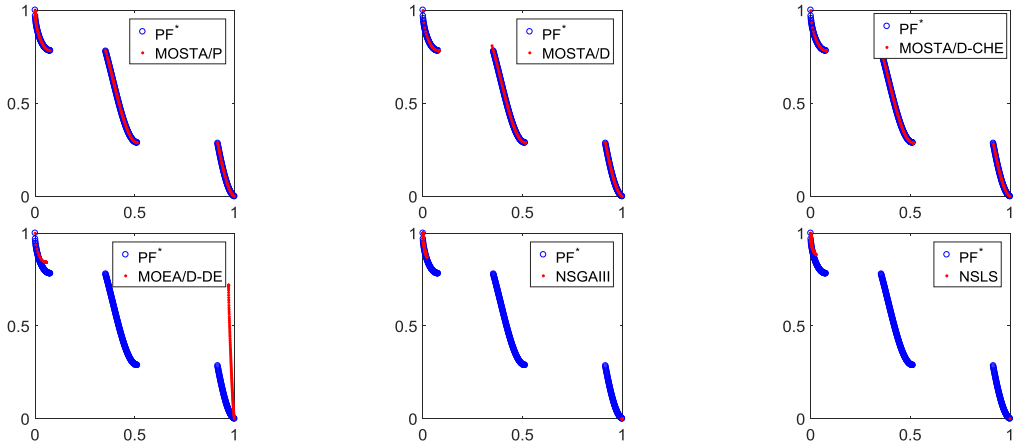


图 5-10 六种算法求解 MOP4 测试函数的前沿图

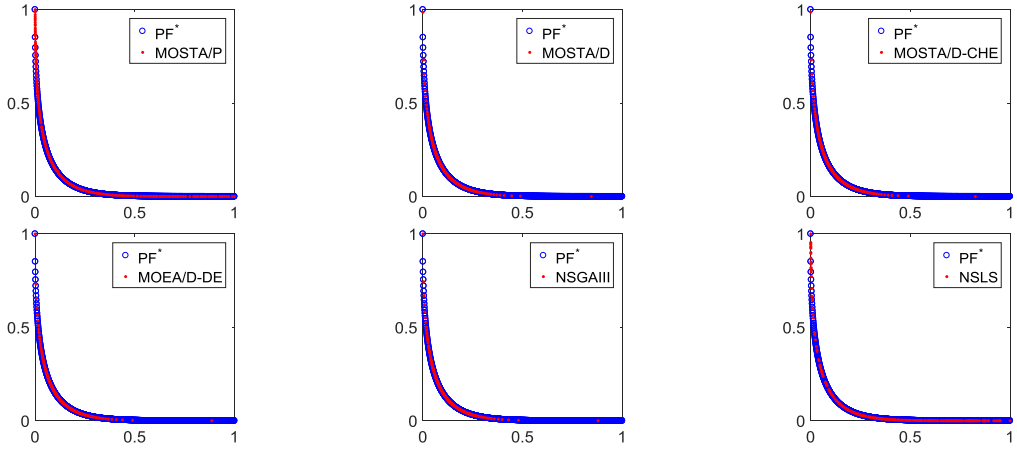


图 5-11 六种算法求解 F1 测试函数的前沿图

在求解 MOP4 测试问题上，本论文提出的算法及其变种成功解决了该测试问题，MOEA/D-DE 找的最优解大部分没有收敛至 Pareto 最优前沿中，Pareto 前沿第一个分支找到了一部分，NSGAIII 和 NSLS 算法也仅仅找到了一部分。从指标上看，本章提出的 MOSTA/D 算法 IGD 指标最小，HV 指标最大，然后是第四章提出的 MOSTA/P 算法，由于 MOSTA/D 的变种 MOSTA/D-CHE，表明本论文提出的算法及本章提出的聚合函数分解方法是十分有效的。

在求解 F1 测试问题时，MOSTA/D 算法并没有找到该测试问题的全部前沿，究其原因是由于该问题属于低尾问题。在尾部，Pareto 最优解的目标函数极小，基于权重向量的分解方法对于候选解进化引导能力减弱。而基于 Pareto 占优方法并不存在此问题。本论文提出的 MOSTA/P 算法和 NSLS 算法可以找到全部的前沿，从指标上看，MOSTA/P 的平均 IGD 指标最小。HV 指标最大，其次是 NSLS 算法，MOSTA/D 排在第三位。

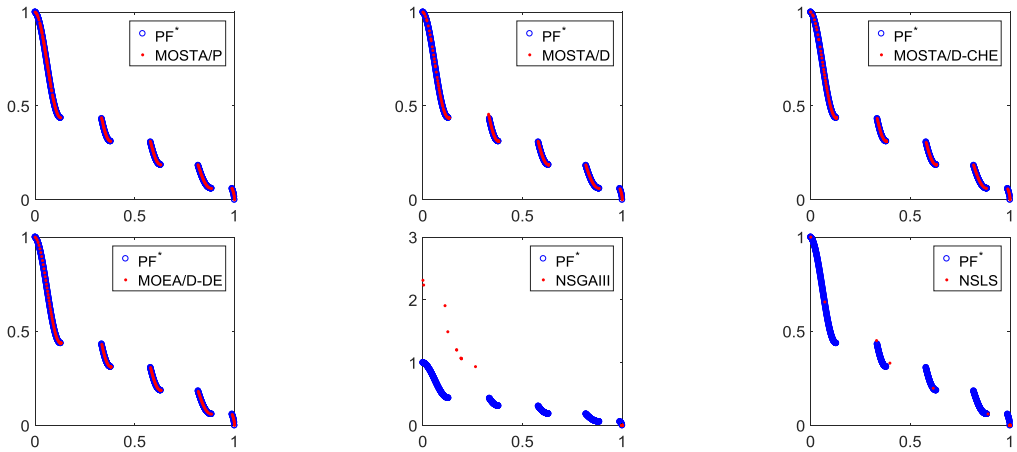


图 5-12 六种算法求解 F2 测试函数的前沿图

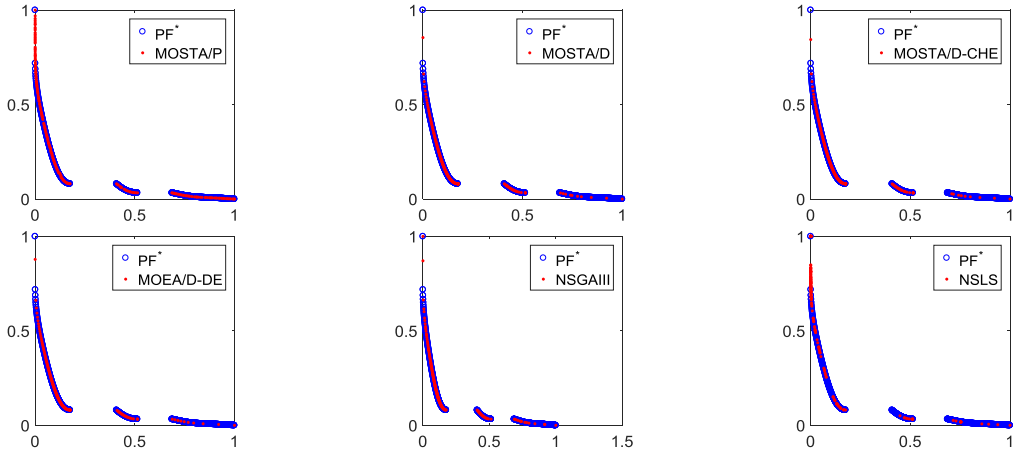


图 5-13 六种算法求解 F3 测试函数的前沿图

对于 F2 测试函数，本论文提出的两种算法及其变种，及 MOEA/D-DE 都能找到 Pareto 前沿，NSLS 算法获得了较少的 Pareto 最优解，在前沿图中分布过于稀疏，NSGAIII 算法并没有完全收敛至 Pareto 前沿内。从性能指标上来说，排在前列的依旧是本论文提出的两种算法 MOSTA/P 及 MOSTA/D 及其变种，表明本算法是有效的。

对于 F3 测试函数，本论文提出的两种算法及其变种，及其他算法都能找到 Pareto 前沿，从性能指标上来说，排在前列的依旧是本论文提出的算 MOSTA/P 表明本算法是有效的。

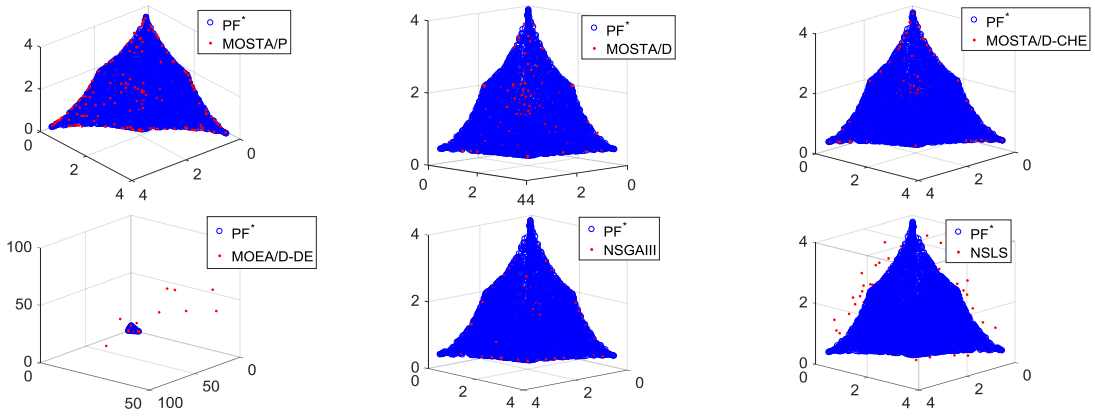


图 5-14 六种算法求解 F4 测试函数的前沿图

F4 测试函数的 Pareto 前沿是一个曲面，是较难求解的复杂前沿，从算法获得的前沿图上看，除 MOEA/D-DE 有较多的解，距离最优 Pareto 前沿较远，其他算法都找到了距离真实 Pareto 最优解较近的解；从指标上看，求解该问题最好的算法 NSGAIII，其 IGD 指标最小，HV 指标最大，本论文提出的算法的指标属于中等水平。

表 5-1 四种算法求解测试函数的 IGD 统计结果

测试函数	算法	最大值	平均值	最小值	标准差
MOP1	MOSTA/P	0.3062	0.2648	0.2196	**0.0227**
	MOSTA/D	**0.2147**	**0.1306**	**0.0665**	0.0312
	MOSTA/D-CHE	0.2830	0.1489	0.0918	0.0443
	MOEA/D-DE	0.3523	0.2600	0.1291	0.0728
	NSGAIII	0.3540	0.3446	0.3357	0.0043
	NSLS	0.3591	0.2599	0.1774	0.0639
MOP2	MOSTA/P	0.0027	0.0023	0.0021	0.001
	MOSTA/D	**0.0019**	**0.0019**	**0.0019**	6.5141e-6
	MOSTA/D-CHE	**0.0019**	**0.0019**	**0.0019**	**7.4585e-6**
	MOEA/D-DE	0.3546	0.2503	0.1861	0.0521
	NSGAIII	0.3546	0.3546	0.3546	5.5511e-17
	NSLS	0.3546	0.3546	0.3546	5.6460e-17
MOP3	MOSTA/P	0.0030	0.0024	0.0023	0.0002
	MOSTA/D	**0.0021**	**0.0021**	**0.0021**	**8.5434e-6**
	MOSTA/D-CHE	0.0021	0.0021	0.0021	1.0607e-5
	MOEA/D-DE	0.5278	0.4569	0.2367	0.0511
	NSGAIII	0.5327	0.4795	0.4117	0.0309
	NSLS	0.4805	0.3809	0.2186	0.0684
MOP4	MOSTA/P	0.1530	0.1529	0.1528	**2.7852e-5**
	MOSTA/D	**0.1497**	**0.1491**	**0.1488**	2.3020e-4
	MOSTA/ D-CHE	0.1559	0.1551	0.1547	2.4981e-4

	MOEA/D-DE	0.4035	0.2380	0.2021	0.0340
	NSGAIII	0.4217	0.4072	0.3802	0.0138
	NSLS	0.4197	0.4136	0.3924	0.0053
F1	MOSTA/P	**0.0032**	**0.0028**	**0.0025**	0.0002
	MOSTA/ D	0.0542	0.0456	0.0444	0.0016
	MOSTA/ D-CHE	0.0538	0.0454	0.0429	0.0017
	MOEA/D-DE	0.1327	0.0855	0.0473	0.0370
	NSGAIII	0.0507	0.0479	0.0292	0.0038
	NSLS	0.0034	0.0026	0.0022	**3.2828e-4**
F2	MOSTA/P	0.0777	**0.0776**	0.0075	0.0001
	MOSTA/D	**0.0768**	**0.0766**	**0.0765**	**7.4015e-5**
	MOSTA/D-CHE	0.0789	0.0787	0.0783	1.2436e-4
	MOEA/D-DE	0.6577	0.0983	0.0790	0.1056
	NSGAIII	0.5073	0.3632	0.2063	0.0865
	NSLS	0.2573	0.1452	0.1049	0.0371
F3	MOSTA/P	**0.0253**	**0.0251**	**0.0249**	0.0001
	MOSTA/D	0.0297	0.0291	0.0286	2.9971e-4
	MOSTA/D-CHE	0.0323	0.0306	0.0290	**8.8756e-4**
	MOEA/D-DE	0.0295	0.0292	0.0290	9.9140e-5
	NSGAIII	0.0322	0.0290	0.0279	9.7428e-4
	NSLS	0.0608	0.0373	0.0259	0.0091
F4	MOSTA/P	0.2402	0.1802	0.1547	0.0170
	MOSTA/D	0.1479	0.1314	0.1215	0.0061
	MOSTA/D-CHE	0.1776	0.1411	0.1261	0.0102
	MOEA/D-DE	0.1415	0.1372	0.1351	**0.0017**
	NSGAIII	**0.1132**	**0.1097**	**0.1043**	0.0018
	NSLS	0.2716	0.2197	0.1658	0.0297

表 5-2 四种算法求解测试函数的 HV 统计结果

测试函数	算法	最大值	平均值	最小值	标准差
MOP1	MOSTA/P	0.7845	0.7244	0.6503	**0.0391**
	MOSTA/D	**1.0116**	**0.9025**	**0.7412**	0.0514
	MOSTA/D-CHE	0.9704	0.8766	0.6441	0.0750
	MOEA/D-DE	0.9400	0.7022	0.5403	0.1330
	NSGAIII	0.5869	0.5657	0.5442	0.0098
	NSLS	0.7975	0.6766	0.5285	0.0931
MOP2	MOSTA/P	0.7708	0.7706	0.7699	1.6425e-4
	MOSTA/D	**0.7709**	**0.7709**	**0.7709**	1.2901e-5
	MOSTA/D -CHE	**0.7709**	**0.7709**	**0.7709**	**1.1239e-5**
	MOEA/D-DE	0.5681	0.4980	0.4400	0.0416
	NSGAIII	0.4400	0.4400	0.4400	2.7775e-16

	NSLS	0.4400	0.4400	0.4400	2.8230e-16
MOP3	MOSTA/P	**0.6525**	0.6523	0.6520	1.3214e-4
	MOSTA/D	0.6524	**0.6524**	**0.6523**	**1.7623e-5**
	MOSTA/D -CHE	0.6524	0.6523	0.6523	1.9380e-5
	MOEA/D-DE	0.3905	0.2494	0.2400	0.0315
	NSGAIII	0.2991	0.2420	0.2400	0.0108
	NSLS	0.3061	0.2510	0.2400	0.0251
MOP4	MOSTA/P	**0.9569**	**0.9569**	0.9567	4.8632e-5
	MOSTA/D	0.9564	0.9564	**0.9563**	**1.6514e-5**
	MOSTA/D-CHE	0.9564	0.9564	**0.9563**	2.4430e-5
	MOEA/D-DE	0.6526	0.6078	0.5740	0.0207
	NSGAIII	0.6082	0.5781	0.5575	0.0114
	NSLS	0.5717	0.5599	0.5491	0.0059
F1	MOSTA/P	**1.3914**	**1.3912**	**1.3909**	**1.1832e-4**
	MOSTA/D	1.3903	1.3903	1.3896	1.3142e-4
	MOSTA/D-CHE	1.3904	1.3903	1.3896	1.2803e-4
	MOEA/D-DE	1.3901	1.3897	1.3893	3.5046e-4
	NSGAIII	1.3908	1.3901	1.3899	1.8628e-4
	NSLS	1.3911	1.3906	1.3898	3.0957e-4
F2	MOSTA/P	**1.1203**	**1.1202**	**1.1201**	**4.7343e-5**
	MOSTA/D	1.1199	1.1197	1.1190	1.5156e-4
	MOSTA/D-CHE	1.1200	1.1198	1.1195	9.5667e-5
	MOEA/D-DE	1.1200	1.0906	0.2400	0.1607
	NSGAIII	0.8283	0.4556	0.2400	0.1647
	NSLS	1.0086	1.0226	0.9286	0.0470
F3	MOSTA/P	**1.3588**	**1.3587**	**1.3586**	**3.9926e-5**
	MOSTA/D	1.3580	1.3579	1.3568	2.1326e-4
	MOSTA/D-CHE	1.3580	1.3578	1.3570	1.6684e-4
	MOEA/D-DE	1.3578	1.3575	1.3573	1.3452e-4
	NSGAIII	1.3583	1.3578	1.3575	1.8546e-6
	NSLS	1.3579	1.3560	1.3496	0.0017
F4	MOSTA/P	86.1943	85.5546	83.7451	0.4832
	MOSTA/D	86.3345	85.9517	85.5171	0.2293
	MOSTA/D-CHE	86.6444	85.6214	84.0724	0.6086
	MOEA/D-DE	85.7035	85.3821	84.9053	0.1949
	NSGAIII	**87.0399**	**87.0119**	**86.9814**	**0.0160**
	NSLS	83.5883	82.4864	80.9858	0.7654

5.4 本章小结

本章首先以权重和的分解方法为例，对现有分解方法的不足进行了分析；现

有分解方法没有考虑权重向量和候选集的匹配关系，导致较好的候选解在选择过程中因为与权重向量不匹配而没有被保留。因此，本章提出一种候选解与权重向量匹配度的概念来度量两者之间的匹配程度。基于匹配度，本章提出一种基于匹配度的修正切比雪夫分解方法，且可证明该方法与切比雪夫方法是等价的；此外，本章基于分解框架，提出一种新的基于分解的多目标状态转移算法MOSTA/D。利用8种测试函数，与三种其他算法进行对比，其IGD指标和HV指标均处于算法前列，基于本章分解方法的算法和基于传统切比雪夫分解的算法进行对比发现，基于本章提出的分解方法的算法性能指标更优；本章提出的基于匹配度的修正切比雪夫分解函数可以有效地解决多目标优化问题。同时，本章在实验过程中还发现，对于目标函数个数较少的多目标优化问题，Pareto占优的比较策略不受目标函数大小的影响，可以获得优化问题的最优前沿。而基于分解方法在目标函数值较小时，权重向量对候选解进化的引导作用减弱。未来将Pareto占优与分解方法结合起来，可能是求解低尾问题一个重要研究方向。

6 总结与展望

6.1 研究工作总结

1) 分析了状态转移算子搜索性能对多目标优化算法收敛性的的影响

本文对状态转移算子的搜索性能对多目标优化算法的收敛性影响进行了研究，基于多目标算法的 GD 性能评价指标，提出了搜索算子对算法收敛性影响作用的评估算法，利用 4 种具有典型特征的标准测试函数，对现有的旋转变换算子，平移变换算子，伸缩变换算子及坐标变换算子进行了分析。分析结果表明，在单独使用这些算子进行搜索时，伸缩变换算子相比于其他算子，可以使算法较快的收敛至最优前沿附近，使用伸缩变换算子和坐标变换算子，算法可以一直保持向最优前沿进化的趋势。而使用平移变换算子，会让算法过早地停止向最优前沿进化；而组合使用这些状态变换算子产生候选解时，平移状态转移与其他算子组合使用可以相比于单独使用平移状态算子，算法朝向最优前沿进化的趋势增强，大大增加算法的收敛性。而伸缩状态算子与其他算子组合使用时，算法朝向最优前沿进化的趋势比单独使用伸缩状态算子有所减弱；旋转变换算子和坐标变换算子在与其他状态变换算子组合使用时，算法朝向最优前沿进化的趋势增强。

2) 提出了基于 Pareto 占优的多目标状态转移算法（MOSTA/P）

本文考虑了子代候选解在比较过程中的的计算资源消耗问题，定义了候选解的资源分配值这一概念，提出了一种基于计算资源动态分配的高效非支配排序策略。对于新产生的候选解，首先计算目标函数，根据其目标函数计算其资源分配值，选择资源分配值较大的候选解与父代候选解进行 Pareto 占优比较，确定其所在的 Pareto 前沿等级，减少了在确定前沿等级时的比较次数，节约了计算资源与时间。在多样性维护方面，采用拥挤距离的概念，将处于密集区域的候选解进行剔除。利用 IGD 指标和 HV 指标，与 3 种算法在 MOP 测试问题和 F 测试问题上，进行了对比，实验结果表明，本文提出的 MOSTA/P 算法可以成功解决多目标优化问题。

3) 提出了基于分解的多目标状态转移算法（MOSTA/D）

本文首先分析了现有分解方法的缺陷，现有分解方法没有考虑候选解与权重向量的匹配关系，在选择过程中，较好候选解可能因为与权重向量不匹配的而没有进入下一次迭代过程。本文提出权重向量和候选解匹配度的概念，基于匹配度，提出一种基于匹配度的修正切比雪夫分解方法，且可证明该方法与切比雪夫方法是等价的。此外，本章还提出一种新的状态变换算子，同化算子使得候选解可以在某一维度进行较为深度的搜索。基于分解框架，采用本章提出的基于匹配度的修正切比雪夫分解方法，设计了一种基于分解的多目标状态转移算法。利用 IGD

指标和 HV 指标，与 3 种算法在 MOP 测试问题和 F 测试问题上，进行了对比，实验结果表明，本文提出的 MOSTA/D 算法可以有效解决多目标优化问题。

6.2 后续工作展望

本文对状态转移算子的搜索对多目标优化算法的影响进行了分析，给出了建议的多目标状态转移算子使用方式，基于 Pareto 框架和分解框架，提出了两种多目标状态转移算法 MOSTA/P、MOSTA/D。然而，在多目标优化领域，状态转移算法还需在以下部分作出更为深入的研究：

1) 离散多目标状态转移算法的设计

实际生活中有许多优化问题如调度，特征提取，社区发现等问题可以建模成为离散多目标优化问题，本论文提出的状态转移算法主要用于求解决策变量是连续的多目标优化问题，而对于决策变量是离散的多目标优化算法研究不足，因此，状态转移算法在接下来多目标优化领域中，应该提出一种可以快速高效解决离散多目标优化问题的多目标离散状态转移算法。

2) 约束多目标状态转移算法的设计

如何求解多目标约束优化问题一直是多目标优化邻域的一大难点，本文仅仅提出了求解无约束问题的或简单约束的多目标优化算法，尚未涉及对复杂约束函数的处理，在之后的研究中，应结合单目标约束优化问题的求解思路，对多目标优化约束处理策略进行研究，提出一种多目标约束状态转移算法。

参考文献

[1] Deb.K, Multi-Objective Optimization Using Evolutionary Algorithms [M]. John Wiley & Sons, Inc., 2001.

[2] Nag K, Pal T, Pal N R. ASMiGA: An archive-based steady-state micro genetic algorithm[J]. IEEE Transactions on Cybernetics, 2015, 45(1): 40-52.

[3] Marler R T, Arora J S. Survey of multi-objective optimization methods for engineering[J]. Structural and Multidisciplinary Optimization, 2004, 26(6): 369-395.

[4] Chen G, Yi X, Zhang Z, et al. Applications of multi-objective dimension-based firefly algorithm to optimize the power losses, emission, and cost in power systems[J]. Applied Soft Computing, 2018, 68: 322-342.

[5] Yang D, Liu Z, Shu T, et al. An Improved Genetic Algorithm for Multiobjective Optimization of Helical Coil Electromagnetic Launchers[J]. IEEE Transactions on Plasma Science, 2018, 46(1): 127-133.

[6] Bayat M, Dehghani Z, Rahimpour M R. Dynamic multi-objective optimization of industrial radial-flow fixed-bed reactor of heavy paraffin dehydrogenation in LAB plant using NSGA-II method[J]. Journal of the Taiwan Institute of Chemical Engineers, 2014, 45(4): 1474-1484.

[7] Kong W, Chai T, Yang S, et al. A hybrid evolutionary multiobjective optimization strategy for the dynamic power supply problem in magnesia grain manufacturing[J]. Applied Soft Computing, 2013, 13(5): 2960-2969.

[8] Qiao J, Zhang W. Dynamic multi-objective optimization control for wastewater treatment process[J]. Neural Computing and Applications, 2018, 29(11): 1261-1271.

[9] 刘丁, 张新雨, 陈亚军. 基于多目标人工鱼群算法的硅单晶直径检测图像阈值分割方法[J]. 自动化学报, 2016, 42(3): 431-442.

[10] 苏兆品, 张国富, 蒋建国, 等. 基于非支配排序差异演化的应急资源多目标分配算法[J]. 自动化学报, 2017, 43(2): 195-214.

[11] 刘潭, 高宪文, 王丽娜. 补偿模型误差的采油过程多目标优化[J]. 控制理论与应用, 2015, 32(5): 615-622.

[12] Miettinen K. Nonlinear multiobjective optimization[M]. Springer Science & Business Media, 2012.

[13] Wang J, Yang W, Du P, et al. Research and application of a hybrid forecasting framework based on multi-objective optimization for electrical power system[J]. Energy, 2018, 148: 59-78.

[14] Babu B V, Jehan M M L. Differential evolution for multi-objective optimization[C]//The 2003 Congress on Evolutionary Computation, 2003. CEC'03. IEEE, 2003, 4: 2696-2703.

[15] 杨景明，穆晓伟，车海军等，多策略改进的多目标粒子群优化算法[J]，控制与决策，2017, (3):435-442.

[16] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II[J]. IEEE transactions on Evolutionary Computation, 2002, 6(2): 182-197.

[17] Zhang Q, Li H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition[J]. IEEE Transactions on Evolutionary Computation, 2007, 11(6): 712-731.

[18] Zitzler E, Künzli S. Indicator-based selection in multiobjective search[C]// International Conference on Parallel Problem Solving from Nature. Springer, Berlin, Heidelberg, 2004: 832-842.

[19] Zhou A, Qu B Y, Li H, et al. Multiobjective evolutionary algorithms: A survey of the state of the art[J]. Swarm and Evolutionary Computation, 2011, 1(1): 32-49.

[20] Trivedi A, Srinivasan D, Sanyal K, et al. A survey of multiobjective evolutionary algorithms based on decomposition[J]. IEEE Transactions on Evolutionary Computation, 2017, 21(3): 440-462.

[21] Coello C A C. Evolutionary multi-objective optimization: some current research trends and topics that remain to be explored[J]. Frontiers of Computer Science in China, 2009, 3(1): 18-30.

[22] Zhou X J, Yang C H, Gui W H, State transition algorithm [J]. Journal of Industrial and Management Optimization, 2017, 8(4):1039-1056.

[23] Vose M D. The simple genetic algorithm: foundations and theory[M]. MIT press, 1999.

[24] Eberhart R, Kennedy J. A new optimizer using particle swarm theory[C]//MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science. Ieee, 1995: 39-43.

[25] Storn R, Price K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces[J]. Journal of Global Optimization, 1997,

11(4): 341-359.

[26] Karaboga D, Basturk B. On the performance of artificial bee colony (ABC) algorithm[J]. Applied soft computing, 2008, 8(1): 687-697.

[27] Zhou X J, Gao D Y, Yang C H. A Comparative study of state transition algorithm with harmony search and artificial bee colony[C]//Proceedings of The Eighth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA), 2013. Springer, Berlin, Heidelberg, 2013: 651-659.

[28] Han J, Yang C, Zhou X, et al. A two-stage state transition algorithm for constrained engineering optimization problems[J]. International Journal of Control, Automation and Systems, 2018, 16(2): 522-534.

[29] Zhou X, Long J, Xu C, et al. An external archive based constrained state transition algorithm for optimal power dispatch [J]. Complexity, 2019, DOI:10.1155/2019/4727168.

[30] Zhou X J, Gao D Y, Yang C H, et al. Discrete state transition algorithm for unconstrained integer optimization problems[J]. Neurocomputing, 2016, 173: 864-874.

[31] 董天雪, 阳春华, 周晓君, 等. 一种求解企业员工指派问题的离散状态转移算法[J]. 控制理论与应用, 2016, 33(10):1378-1388.

[32] Wang Y, He H, Zhou X, et al. Optimization of both operating costs and energy efficiency in the alumina evaporation process by a multi-objective state transition algorithm[J]. The Canadian Journal of Chemical Engineering, 2016, 94(1): 53-65.

[33] Han J, Yang C, Zhou X, et al. Dynamic multi-objective optimization arising in iron precipitation of zinc hydrometallurgy[J]. Hydrometallurgy, 2017, 173: 134-148.

[34] Ishibuchi H, Murata T. A multi-objective genetic local search algorithm and its application to flowshop scheduling[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 1998, 28(3): 392-403.

[35] Murata T, Nozawa H, Tsujimura Y, et al. Effect of local search on the performance of cellular multiobjective genetic algorithms for designing fuzzy rule-based classification systems[C]//Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No. 02TH8600). IEEE, 2002, 1: 663-668.

[36] Chen B, Zeng W, Lin Y, et al. A new local search-based multiobjective optimization algorithm[J]. IEEE Transactions on Evolutionary Computation, 2015, 19(1): 50-73.

[37] Srinivas N, Deb K. Muiltiobjective optimization using nondominated sorting in genetic algorithms[J]. Evolutionary computation, 1994, 2(3): 221-248.

[38] Jensen M T. Reducing the run-time complexity of multiobjective EAs: The NSGA-II and other algorithms[J]. IEEE Transactions on Evolutionary Computation, 2003, 7(5): 503-515.

[39] Tang S, Cai Z, Zheng J. A fast method of constructing the non-dominated set: arena's principle[C]//2008 Fourth International Conference on Natural Computation. IEEE, 2008, 1: 391-395.

[40] McClymont K, Keedwell E. Deductive sort and climbing sort: New methods for non-dominated sorting[J]. Evolutionary Computation, 2012, 20(1): 1-26.

[41] Zheng J, Ling C X, Shi Z, et al. Some discussions about MOGAs: individual relations, non-dominated set, and application on automatic negotiation[C]// Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No. 04TH8753). IEEE, 2004, 1: 706-712.

[42] Zhang X, Tian Y, Cheng R, et al. An efficient approach to nondominated sorting for evolutionary multiobjective optimization[J]. IEEE Transactions on Evolutionary Computation, 2015, 19(2): 201-213.

[43] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach[J]. IEEE Transactions on Evolutionary Computation, 1999, 3(4): 257-271.

[44] 乔俊飞，李霏，杨翠丽，一种基于均匀分布策略的 NSGAII 算法[J]，自动化学报，2019，DOI 10.16383/j.aas.c180085.

[45] 张爱竹, 孙根云, 王振杰, 等. 一种基于数据场的多目标引力搜索算法[J]. 控制与决策, 2017, 32(1): 47-54.

[46] Murata T, Gen M. Cellular genetic algorithm for multi-objective optimization[C]//Proc. of the 4th Asian Fuzzy System Symposium. 2002: 538-542.

[47] Zhang Q, Li H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition[J]. IEEE Transactions on Evolutionary Computation, 2007, 11(6): 712-731.

[48] Marler R T, Arora J S. Survey of multi-objective optimization methods for engineering[J]. Structural and Multidisciplinary Optimization, 2004, 26(6): 369-395.

[49] Giagkiozis I, Purshouse R C, Fleming P J. Generalized decomposition[C]// International Conference on Evolutionary Multi-Criterion Optimization. Springer, Berlin, Heidelberg, 2013: 428-442.

[50] Das I, Dennis J E. Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems[J]. SIAM Journal on Optimization, 1998, 8(3): 631-657.

[51] Tan Y Y, Jiao Y C, Li H, et al. MOEA/D+ uniform design: A new version of MOEA/D for optimization problems with many objectives[J]. Computers & Operations Research, 2013, 40(6): 1648-1660.

[52] 李飞, 刘建昌, 石怀涛, 傅梓瑛, 基于分解和差分进化的多目标粒子群优化算法[J]，控制与决策，2017, 32(3):403-140.

[53] Qi Y, Ma X, Liu F, et al. MOEA/D with adaptive weight adjustment[J]. Evolutionary Computation, 2014, 22(2): 231-264.

[54] Sato H. Analysis of inverted PBI and comparison with other scalarizing functions in decomposition based MOEAs[J]. Journal of Heuristics, 2015, 21(6): 819-849.

[55] Wang Z, Zhang Q, Li H, et al. On the use of two reference points in decomposition based multiobjective evolutionary algorithms[J]. Swarm and Evolutionary Computation, 2017, 34: 89-102.

[56] Mohammadi A, Omidvar M N, Li X, et al. Sensitivity analysis of penalty-based boundary intersection on aggregation-based EMO algorithms[C]//2015 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2015: 2891-2898.

[57] Yang S, Jiang S, Jiang Y. Improving the multiobjective evolutionary algorithm based on decomposition with new penalty schemes[J]. Soft Computing, 2017, 21(16): 4677-4691.

[58] Ma X, Zhang Q, Tian G, et al. On Tchebycheff decomposition approaches for multiobjective evolutionary optimization[J]. IEEE Transactions on Evolutionary Computation, 2018, 22(2): 226-244.

[59] Cheng R, Jin Y, Olhofer M, et al. A reference vector guided evolutionary algorithm for many-objective optimization[J]. IEEE Transactions on Evolutionary Computation, 2016, 20(5): 773-791.

[60] Zhang Q, Liu W, Li H. The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances[C]//2009 IEEE congress on evolutionary computation. IEEE, 2009: 203-208.

[61] Liu H L, Gu F, Zhang Q. Decomposition of a multiobjective optimization problem

into a number of simple multiobjective subproblems[J]. IEEE Transactions on Evolutionary Computation, 2014, 18(3): 450-455.

[62] Kim J H, Han J H, Kim Y H, et al. Preference-based solution selection algorithm for evolutionary multiobjective optimization[J]. IEEE Transactions on Evolutionary Computation, 2012, 16(1): 20-34.

[63] E.Zitzler, S. Künzli, indicator-based selection in multiobjective search[M]. In International Conference on Parallel Problem Solving from Nature, 2004, 832-842. Springer, Berlin, Heidelberg.

[64] Emmerich M, Beume N, Naujoks B. An EMO algorithm using the hypervolume measure as selection criterion[C]//International Conference on Evolutionary Multi-Criterion Optimization. Springer, Berlin, Heidelberg, 2005: 62-76.

[65] Beume N, Naujoks B, Emmerich M. SMS-EMOA: Multiobjective selection based on dominated hypervolume[J]. European Journal of Operational Research, 2007, 181(3): 1653-1669.

[66] Zhou X, Yang C, Gui W. A statistical study on parameter selection of operators in continuous state transition algorithm[J]. IEEE Transactions on Cybernetics, 2018, doi: 10.1109/TCYB.2018.2850350.

[67] Coello C A C, Pulido G T, Lechuga M S. Handling multiple objectives with particle swarm optimization[J]. IEEE Transactions on Evolutionary Computation, 2004, 8(3): 256-279.

[68] 田红军, 汪磊,吴启迪,一种求解多目标优化问题的进化算法混合框架,控制与决策,2017,32(10):1729-1738.

[69] Zitzler E, Deb K, Thiele L. Comparison of multiobjective evolutionary algorithms: Empirical results[J]. Evolutionary Computation, 2000, 8(2): 173-195.

[70] Deb K, Thiele L, Laumanns M, et al. Scalable multi-objective optimization test problems[C]//Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No. 02TH8600). IEEE, 2002, 1: 825-830.

[71] Huband S, Hingston P, Barone L, et al. A review of multiobjective test problems and a scalable test problem toolkit[J]. IEEE Transactions on Evolutionary Computation, 2006, 10(5): 477-506.

[72] Bradstreet L, Barone L, While L, et al. Use of the WFG Toolkit and PISA for Comparison of MOEAs[C]//2007 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making. IEEE, 2007: 382-389.

[73] Zhang Q, Zhou A, Zhao S, et al. Multiobjective optimization test instances for the

CEC 2009 special session and competition[J]. University of Essex, Colchester, UK and Nanyang technological University, Singapore, special session on performance assessment of multi-objective optimization algorithms, technical report, 2008, 264.

[74] Li H, Zhang Q. Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II[J]. IEEE Transactions on Evolutionary Computation, 2009, 13(2): 284-302.

[75] Jiang S, Yang S. An improved multiobjective optimization evolutionary algorithm based on decomposition for complex Pareto fronts[J]. IEEE Transactions on Cybernetics, 2016, 46(2): 421-437.

[76] Yen G G, He Z. Performance metric ensemble for multiobjective evolutionary algorithms[J]. IEEE Transactions on Evolutionary Computation, 2014, 18(1): 131-144.

[77] Van Veldhuizen D A, Lamont G B. Multiobjective evolutionary algorithms: Analyzing the state-of-the-art[J]. Evolutionary computation, 2000, 8(2): 125-147.

[78] While L, Hingston P, Barone L, et al. A faster algorithm for calculating hypervolume[J]. IEEE Transactions on Evolutionary Computation, 2006, 10(1): 29-38.

[79] Tian Y, Cheng R, Zhang X, et al. PlatEMO: A MATLAB platform for evolutionary multi-objective optimization [educational forum][J]. IEEE Computational Intelligence Magazine, 2017, 12(4): 73-87.

[80] Deb k, Jain H , An evolutionary many-objective optimization algorithm usingreference-point based non-dominated sorting approach, part I: Solving problems with box constraints[J]. IEEE Transactions on Evolutionary Computation, 2014, 18(4): 577-601.

攻读学位期间主要的研究成果

发表论文情况

[1]. **Zhou J**, Zhou X, Yang C, Gui W, A multi-objective state transition algorithm for continuous optimization [C], Proceedings of the 36th Chinese Control Conference,China,Dalian, 2017:9859-9864. (EI 检索)

[2]. Zhou X, **Zhou J**, Yang C, Gui W, Set-point tracking and multi-objective optimization-Based PID control for the goethite process [J], 2018, IEEE ACCESS, 6, 36683-36698. (SCI 检索, IF:3.557)

申请发明专利

[1] 周晓君，**周佳佳**，徐冲冲，一种基于分解的多目标状态转移优化方法及系统[P]，2019年申请国家发明专利(专利号：201910313379.4)

参与科研项目情况

[1] 2019 年 1 月-至今，参与国家自然科学基金项目课题“有色冶金级联反应器不确定动态优化方法”（61873285）的研究工作；

[2] 中南大学创新驱动计划“流程工业过程数据建模与分布式优化方法研究”（2018CX012）的研究工作；

[3] 2016 年 9 月-2018 年 12 月，参与“基于分布式状态转移算法的有色冶金过程系统辨识方法研究”（61503416）；

[4] 2017 年 3 月-2019 年 3 月，主持中南大学研究生自主探索创新项目（2017zzts492）“基于多目标 PID 的针铁矿法沉铁过程优化控制方法研究”的研究工作。

获奖情况

[1] 2019 年 中南大学优秀毕业生

[2] 2018 年 国家奖学金 中南大学优秀研究生

[3] 2017 年-2019 年 研究生国家学业一等奖学金

致谢

时光荏苒，日月如梭，三年研究生生活匆匆划过。欣喜的是，这三年时间懂得了许多做人的道理，学到了让我一生受益的知识。在此，我谨向所有帮助我，引导我，激励我的老师、朋友、亲人们表示最诚挚的感谢。

特别感谢我的导师周晓君老师。周老师对待科研严肃认真，追求卓越，对我们悉心辅导；三年里取得的每一次进步都包含着老师的帮助和教导。在研究生这三年中，周老师积极资助我们参加各项学术交流，拓宽我们的学术眼界。在学术指导之外，还和我们分享人生感悟，促进我们综合能力的提高。他对我们的严格要求促使我们培养了耐心，细致，稳重的科研习惯，整个实验室在周老师的带领下，大家科研热情十分高涨，相互帮助，互相支持。

同时，衷心感谢控制工程研究所的桂卫华老师、阳春华老师、王雅琳老师、徐德刚老师、谢永芳老师、李勇刚老师，陈晓方老师、蒋朝辉老师、马山老师、吴同茂老师及其他老师的辛勤付出，你们渊博的学术知识和对于科研前沿的宏观把控，为我们的科学研究指引了方向。

感谢各位师兄师姐及同门师弟妹对我三年科研和生活的支持与帮助，感谢黄兆可师兄、韩洁师姐、张凤雪师姐对我的关心与帮助，感谢同门黄淼一直以来的相互扶持与相互促进，感谢民主楼 316-1 实验室的龙建朋师妹、王启安师弟、杨珂师弟、张润东师弟、徐冲冲师弟、王湘月师弟、张云祥师弟以及其他成员给予我的关心与温暖，是你们对科研的执着与追求，为实验室营造了良好的学习氛围。

感谢我的室友王瑞、吴书君、陈冠余、崔晓丽，是你们给予了我一个舒适、放松的生活环境，让我在研究生三年期间能够积极快乐地学习。

感谢我的家人和挚友，是你们的引导、鼓励和支持帮助我度过研究生生活的一切困难，帮助我逐渐成长，成熟，你们的无微不至的关心是我前进的动力！

最后，感谢中南大学，本科和研究生一起，在这里我度过了七年美好时光，收获了最宝贵的知识和最珍贵的友情，知行合一，经世致用的校训我将永远铭记。