计算机科学
COMPUTER SCIENCE

基于两层知识迁移的多代理多任务优化方法

马慧, 冯翔, 虞慧群

引用本文

马慧, 冯翔, 虞慧群. 基于两层知识迁移的多代理多任务优化方法[J]. 计算机科学, 2023, 50(10): 203-213.

MA Hui, FENG Xiang, YU Huiqun. Multi-surrogate Multi-task Optimization Approach Based on Two-layer Knowledge Transfer [J]. Computer Science, 2023, 50(10): 203-213.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.220900242

基于两层知识迁移的多代理多任务优化方法

马　慧　　冯　翔　　虞慧群
华东理工大学计算机科学与工程系　上海 200237
上海智慧能源工程技术研究中心　上海 200237
(531199628@qq.com)

摘　要　进化多任务优化是计算智能领域一个新兴的研究方向,它致力于研究通过进化算法如何同时、有效地求解多个优化问题,从而提高单独求解每个任务的性能。基于此,提出了一种基于两层知识迁移的多代理多任务优化算法(AMS-MTO),其通过在代理间和代理内同时进行知识迁移来达到跨域优化的目的。具体来讲,代理内的知识迁移是通过差分进化实现决策变量信息的跨维迁移,从而避免算法陷入局部最优;代理间的学习采用了隐式知识迁移和显式知识迁移两种策略。隐式知识迁移利用种群的选择性交叉来产生后代,促进遗传信息的交流;显式知识迁移是对精英个体的迁移,可以弥补隐式迁移随机性很强的缺点。为了评估两层知识迁移的多代理多任务优化方法的有效性,在8个高达100维的基准问题上进行了实证研究,同时给出了收敛证明,并将其与现有的算法进行了对比。实验结果表明,在求解单目标优化的昂贵问题时,AMS-MTO算法效率更高,性能更好,收敛速度更快。

关键词:进化多任务优化;多代理;知识迁移;精英个体;隐式迁移
中图法分类号　TP391

Multi-surrogate Multi-task Optimization Approach Based on Two-layer Knowledge Transfer

MA Hui,FENG Xiang and YU Huiqun
Department of Computer Science and Engineering,East China University of Science and Technology,Shanghai 200237,China
Shanghai Engineering Research Center of Smart Energy,Shanghai 200237,China

Abstract　Evolutionary multi-task optimization is a new research direction in the field of computational intelligence. It focuses on how to handle multiple optimization tasks effectively and simultaneously through evolutionary algorithm,so as to enhance the performance of solving each task individually. Based on this,a multi-surrogate and multi-task optimization approach based on two-layer knowledge transfer(AMS-MTO) is proposed,which achieves the purpose of cross-domain optimization by transferring knowledge between surrogates and within surrogates at the same time. Specifically,the knowledge transfer within the surrogates realizes the cross-dimensional transfer of decision variable information through differential evolutionary,so as to avoid the algorithm falling into local optimum. The learning between surrogates adopts two strategies:implicit knowledge transfer and explicit knowledge transfer. The former uses the selective crossover of populations to generate offspring and promote the exchange of genetic information. The latter is mainly the transfer of elite individuals,which can make up for the strong randomness of implicit transfer. For the sake of evaluate the effectiveness of the AMS-MTO algorithm,we carry out an empirical study on 8 benchmark problems up to 100 dimension. At the same time,we give the convergence proof and compare it with the existing algorithms. Experiment resultsshow that when solving expensive problems of single objective optimization,the AMS-MTO algorithm has higher efficiency,better performance and faster convergence speed.

Keywords　Evolutionary multi-task optimization,Multi-surrogate,Knowledge transfer,Elite individuals,Implicit transfer

到稿日期:2022-09-26　返修日期:2023-03-09
基金项目:国家自然科学基金面上项目(62276097);国家自然科学基金重点项目(62136003);国家重点研发计划(2020YFB1711700);上海市经信委"信息化发展专项资金"(XX-XXFZ-02-20-2463);上海市科技创新行动计划(21002411000)
This work was supported by the National Natural Science Foundation of China(62276097),Key Program of National Natural Science Foundation of China(62136003),National Key Research and Development Program of China(2020YFB1711700),Special Fund for Information Development of Shanghai Economic and Information Commission(XX-XXFZ-02-20-2463) and Scientific Research Program of Shanghai Science and Technology Commission(21002411000).
通信作者:冯翔(xfeng@ecust.edu.cn)

1 引言

众所周知，世界上的许多任务存在相关性，因此也具有一定的相互借鉴和学习的地方，同时，人类可以高效地并发完成多项任务，由此引发了研究者们对多任务计算智能方法的研究。多任务优化(Multi-task Optimization，MTO)是优化领域一个新兴的研究方向，它致力于在求解多个优化问题的过程中应用进化算法相互进行知识迁移，从而提升每个任务自身的优化性能。因此，在求解多个任务的性能和效率上，MTO由于结合了任务间的共性和互补性而显著优于原始的单任务优化方法。

进化算法(Evolutionary Algorithms，EA)是一种基于种群迭代的智能优化算法，它模拟达尔文的自然选择和遗传变异的生存过程。EA的设计一般从某个初始种群开始，通过模拟生物界的遗传和变异生成子代群体，并将其与父代群体合并，只保留表现好的个体作为下一迭代的初始种群。然后不断重复上述过程，直到到达预先设定的停止条件。进化算法不依赖于优化函数的表达式，且不易陷入局部最优，因此被学者广泛地应用于实际场景问题下的优化和求解。但是它也有一定的局限性，即解决昂贵优化问题时通常会进行大量耗时的适应度评估。为此，学者们提出了许多代理辅助进化算法，即同时求解计算便宜问题和计算昂贵问题，利用计算成本廉价的代理模型来代替部分耗时的适应度评估，以降低计算昂贵问题的计算代价[1]。在代理模型中，最著名和最常用的模型有多项式回归[2]、人工神经网络[3]、支持向量机(Support Vector Machine，SVM)、高斯过程(Gaussian Process，GP)[4]和径向基函数网络(Radial Basis FunctionNetwork，RBFN)[5]等。

近二十年来，越来越多的学者们投身于代理模型的研究[6-7]，提出了众多代理辅助进化算法。DE-S算法[8]在差分进化算法、二次代理近似辅助进化优化的基础上，引入了一种新的填充抽样策略和评分标准，以提高算法的搜索性能。MGP-SLPSO算法[9]在高斯过程辅助社会学习粒子群优化算法(Social Learning Particle Swarm Optimization，SL-PSO)的基础上提出了一种多目标填充准则，把最小化近似适应度和最大化近似不确定性视为两个独立的目标，同时使用非支配排序进行模型管理，从而显著提高了算法在性能和不确定性两者间的平衡能力。SCCMA算法[10]将协同进化与代理辅助模因算法相结合，并引入随机问题分解的思想和多目标排序策略，从而提高代理辅助进化的效率。EvoLs算法[11]通过构建支持可信域局部搜索策略的代理模型，实现代理辅助模因算法的自配置，从而提高了解决计算昂贵问题的速率。GORS-SSLPSO算法[12]是在SL-PSO的基础上提出的一种基于代的最优重启策略，即在全局代理模型中每隔几代重新启动一次SL-PSO算法，重启前选择最佳近似适应度的个体进行精确评估。SAMSO算法[13]是一种代理辅助的多群体优化，提出了一种动态种群规模调整策略，以提高高维昂贵问题的搜索效率。

单个代理虽然已经在某种程度上提高了进化算法的搜索速度，但依然具有一定的局限性。单代理模型通常包括单个的全局代理模型和单个局部代理模型。全局代理模型由于代理质量太差而无法加快进化算法的收敛速度，同时可能会陷入局部最优，从而无法快速帮助进化算法找到全局最优解；而局部代理模型得到的局部最优解也很难帮助进化算法进行全局性的收敛。因此，多代理进化优化的概念被提出来，即应用多个代理协同工作，从而快速找到全局最优解。一种很典型的设计是：构建一个粗粒度的全局模型来开发存在局部最优的区域和一个精细的局部模型，从而探索原始适应度范围的局部细节。Zhou等[14]提出了一种分层代理辅助进化优化算法，先利用GP全局代理模型来筛选出可能存在最优值的局部区域，再利用精准的局部代理模型进行局部最优搜索。Lim等[15]将多个代理模型应用到广义进化框架的进化搜索中，使用集成的局部代理模型和平滑的全局代理模型同时进行局部搜索，从而实现对搜索空间的探索和开发。Sun等[16]提出了一种两层代理辅助粒子群算法，采用全局代理模型平滑目标函数的局部最优，同时采用多个局部代理模型精确地逼近局部最优。Yu等[17]提出了一种代理辅助的分层粒子群优化框架，在该算法中粒子群优化(Particle Swarm Optimization，PSO)和SL-PSO协同工作，两个优化器均使用了RBF代理模型。其中，PSO用于探索搜索空间，学习适应度函数的全局特征；而SL-PSO用于开发搜索空间，即准确地学习适应度函数的局部细节，从而找到局部最优值。

受上述多代理辅助优化技术研究的启发，本文提出了一种多代理多任务进化优化算法。该算法构造了两个RBF代理模型：一个是在所有个体上进行训练的全局代理模型，负责探索整个搜索空间；另一个是只在适应度最佳的个体上进行训练的局部代理模型，负责帮助局部区域的开发。两个代理模型可以看作两个任务，它们协同工作，以加速进化算法的收敛速度。然而，在现有算法中，局部代理模型在执行局部搜索的时候对寻找全局最优解的贡可能献不大，从而导致搜索效率低下；另外，代理模型间知识的迁移具有随机性，可能会产生负迁移。针对这两个问题，本文引入一种两层知识迁移策略，主要贡献如下：

(1)提出了代理间与代理内两层知识迁移的算法。其中，代理间的知识迁移利用模型间的共性和相似性来加快每个代理间的收敛速度，实现的是跨任务间的迁移；代理内的知识迁移通过差分进化将决策变量从一个维度迁移到另一个维度，实现的是跨维迁移，能够帮助算法逃离局部极小点，同时加快收敛速度。

(2)在代理间的知识迁移学习中，采用了隐式知识迁移和显式知识迁移两种策略。其中，隐式知识迁移是利用种群的选择性交配来产生后代，促进遗传信息的交流；显式知识迁移是对精英个体的迁移，可以弥补隐式迁移随机性很强的缺点。

(3)在代理内的学习策略中，对每一代的每一个任务的最优解应用拟牛顿法以获得高质量的改进解，从而提高迁移的有效性。

本文第2节简要介绍了相关背景知识；第3节详细介绍了所提出的基于两层知识迁移的多代理多任务优化(AMS-MTO)的模型框架；第4节详细介绍了AMS-MTO的算法实现及计算复杂度分析；第5节给出了算法参数设置过程，以及与多个最近所提算法的对比实验，并对结果进行了分析阐述；最后总结全文并展望未来。

2　相关工作

2.1　进化多任务优化

进化多任务优化（Evolutionary Multi-task Optimization，EMTO）是一系列 MTO 技术中的代表，它将进化算法作为优化器，同时借助知识迁移技术来实现对多任务的并行处理。迁移技术是机器学习中新兴的研究方向，它利用任务或数据的相似性，迁移旧模型学到的知识，以辅助新模型的学习。应用最广泛的多因子进化算法（Multifactor Evolutionary Algorithm，MFEA）可以看作 EMTO 的先驱。自 MFEA 提出之后，学者们对 EMTO 的研究不断深入。Bali 等[18]提出了 LDA-MFEA 算法，该算法引入了线性域自适应策略来缓解任务之间的知识负迁移问题。该策略将简单任务的统一搜索空间通过线性变换映射到复杂任务的搜索空间，从而提升任务间的高阶相关性，减少负迁移的产生。Wen 和 Ting[19]提出了 MFEARR 算法，该算法通过检测到资源分离时停止任务间的知识迁移来重新分配不同类型子代的适应度进行评估，从而实现分离检测和资源重分配机制。Liaw 和 Ting[20]提出的 EBS-CMAES 算法，是一种基于共生生物群落的进化多任务框架。该框架可以自适应地控制级联后代之间的信息交换，从而根据所有任务生成后代的候选解进行评估，以此提高多任务知识迁移的效率。Li 等[21]提出了 MPEF 算法，其是一个通用的多种群进化框架。该框架嵌入了进化算法，其中每个种群都利用自己的随机交配概率（Random Mating Probability，RMP）去处理自己唯一负责的优化任务，并在不同种群间设计不同的知识迁移策略来实现知识共享。Tan 等[22]提出的 TEMO-MPS 算法，是一个自适应知识重用的多问题代理辅助多目标优化框架。该框架具有跨问题获取知识和自适应迁移学习的能力，有效地提高了任务的全局优化速率。

2.2　多因子进化算法

2.2.1　定义

多任务优化以同时求解多个不同优化任务为目标，它借助了任务间的共性和相似性，通过迁移任务间有用的知识来提升各自的优化性能。

MFEA 构造了一个多任务的环境，然后利用选择性交配和垂直文化传播两种策略进化出一个单一的个体种群来解决 MTO 问题，其中每个任务都作为一个独特的文化因子影响着群体的进化。

考虑一个 MTO 问题，假设其同时处理 k 个不同的优化任务，且每一个任务都是连续单目标、无约束的最小化问题。换言之，MTO 的最终目的是找到一组解，使得解的每个分量都对应一个任务的最小值。若将 T_j 定义为第 j 个任务，其对应的目标函数为 $f_j:X_j\rightarrow R$（R 表示实数域），则 MTO 问题可表述如下：

$$x_j^*=\arg\min_{x\in X_j} f_j(x),j=1,2,\cdots,k$$

其中，X_j 表示 D_j 维的解空间，$\{x_1^*,x_2^*,\cdots,x_K^*\}$ 即为多任务优化的一组最优解。

这里需要注意的是，如果每个任务对应的目标函数是多目标的，那么 MTO 处理的是多目标多任务优化问题。如果对其中某一任务的解空间增加了约束条件，那么需要将这些约束条件和目标函数结合起来一起进行优化。

为了在多任务环境中比较个体的优劣，MFEA 给每个个体 p_i 分配适应度，定义了以下属性[23]。

定义 1（因子代价，Factorial Cost）　对于个体 p_i，其因子代价定义为 $\alpha_{ij}=\gamma\delta_{ij}+F_{ij}$，表示 p_i 在任务 T_j 上的适应度值。其中，F_{ij} 和 δ_{ij} 分别是个体 p_i 在 T_j 上的目标函数值和违反约束的总次数；系数 γ 是一个很大的惩罚乘数。

定义 2（因子排名，Factorial Rank）　对于个体 p_i，其在任务 T_j 上的因子排名 r_j^i 是所有种群个体按照因子代价 α_{ij} 在 T_j 上升序排序后，p_i 的索引值。

需要注意的是，当个体 p_1 和 p_2 在任务 T_j 上具有相同的因子代价时，它们的排名先后顺序将随机决定。

定义 3（标量适应度，Scalar Fitness）　对于个体 p_i，其标量适应度 φ_i 是根据其在 k 个任务上所得到的因子排名中最好的排名所计算出来的，$\varphi_i=1/\min\{r_k^i\},k=1,2,\cdots,K$。该属性描述了个体在每个任务上的表现。

定义 4（技能因子，Skill Factor）　对于个体 p_i，其技能因子 τ_i 表示在所有任务中，p_i 表现最好的那个任务。$\tau_i=\arg\min_k\{r_k^i\},k=1,2,\cdots,K$。

基于以上定义，在进化过程中可以快捷地比较出个体的优劣。例如，若个体 p_1 和 p_2 的标量适应度大小情况为 $\varphi_1>\varphi_2$，则表明个体 p_1 优于 p_2。

MEFA 算法主要包含 3 个模块：统一编解码、选择性交配和垂直文化传播。

2.2.2　统一编解码方案

对于 MTO 问题，不同任务的维数不同，搜索空间不同，解空间也有所不同。若在原始空间中直接进行任务间的交配，那么遗传物质在任务间的隐性转移将会受到影响或部分丢失。编码-解码方案可以将不同任务的搜索空间映射到一个统一的搜索空间中，同时管理着多个维数不同的任务并且能够保存不同任务对应的完整遗传物质，从而有效地加快算法的收敛速度。

在 MFEA 算法中，统一搜索空间 Y 被定义为 $D_Y=\max\{D_1,D_2,\cdots,D_K\}$，$D_k$ 表示第 k 任务的维数。在种群初始化阶段，所有个体都用 D_Y 维的向量进行编码，其中每一个决策变量的值都在 0 和 1 之间进行标准化。每一个个体的染色体均为 D_Y 维的向量，因此都包含了所有任务对应的完整遗传物质。

在评估个体前，需要对其进行解码，即将其解码到该个体所负责任务的决策空间中。假设个体 y 的技能因子为 k，则其负责处理第 k 个任务。若第 k 个任务的上边界是 U_k，下边界是 L_k，那么个体 y 将在原始决策空间中被解码成 x，$x=y(1:D_k)\times(U_k-L_k)+L_k$。为了形象地描述编解码的过程，图 1 给出了一个简单的示例。

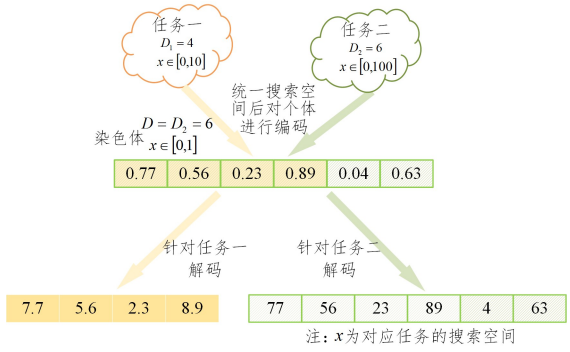


图 1　编码-解码图示

Fig. 1　Encoding and decoding diagram

2.2.3 选择性交配和垂直文化传播

选择性交配是 MFEA 的繁衍策略，同一任务下的个体相当于具有相同的文化背景，选择性交配的原则是具有相同文化背景的个体可以自由交配。若文化背景不相同，那么个体间的交配由 RMP 控制。当父代个体之间无法交配时，那么子代则由父代执行变异操作生成。假设，两个父代个体分别为 p_i 和 p_j，技能因子分别为 τ_i 和 τ_j，生成子代为 c_i 和 c_j，那么选择性交配的操作可以表示为：

$$\begin{cases} p_i \otimes p_j \rightarrow c_i \cup c_j, & \text{if}(\tau_i = \tau_j) \text{ or } (rand \leqslant rmp) \\ p_i \xrightarrow{\text{mutation}} c_i \cup p_j \xrightarrow{\text{mutation}} c_j, & \text{otherwise} \end{cases}$$

其中，$\otimes$ 为交配操作，$\cup$ 为取并集，$rand$ 表示 $[0,1]$ 的随机数，$\xrightarrow{\text{mutation}}$ 表示个体的变异操作。

垂直文化传播的原理是通过允许子代继承父代的技能因子，实现子代个体在特定任务上的评估，从而大幅减少 MTO 问题的总评估次数。假设子代为 c_i，若其由父代个体 p_i 和 p_j 交配而来，则其技能因子同等概率地继承自 p_i 或 p_j；若由父代个体 p_i 变异而来，则其技能因子只能继承于 p_i。具体可以表示如下：

$$\begin{cases} \begin{cases} c_i \infty p_i, & \text{if}(rand < 0.5) \\ c_i \infty p_j, & \text{otherwise} \end{cases}, & \text{if } (p_i \otimes p_j \rightarrow c_i) \\ c \infty p_i, & \text{otherwise} \end{cases}$$

其中，$c_i \infty p_i$ 表示子代 c_i 继承父代 p_i 的技能因子。

2.3 径向基函数网络

RBFN 是最常用的代理模型之一，并且在函数近似、时间序列预测和控制方面取得了成功的应用[24]。RBFN 的原理比较简单，它可以看作人工神经网络的变体，即激活函数采用的是径向基函数，且可以依据给定的数据点进行插值和外推[24]。因此，本文中的代理辅助模型都采用了 RBFN。

RBFN 是一种只有 3 层的前向人工神经网络，其中第一层是输入层，第二层是隐含层，第三层是网络输出层。输入层由一个 D 维向量组成，可表示为 $\boldsymbol{x}=(x_1, x_2, \cdots, x_D)$；隐含层是使用径向基函数作为激活函数的神经元；输出层是对输入模式的作用做出响应，简单来说就是输入向量的标量函数。输入层到隐含层属于非线性变换，而隐含层到输出层是简单的线性连接。给定 D 维输入空间中的一组训练数据点 S，$x_i=(x_{i1}, x_{i2}, \cdots, x_{iD})$，基函数以输入点与中心点的距离作为函数自变量，则输出 $F(x)$ 为基函数的线性组合：

$$F(x)=\sum_{i=1}^{H}\lambda_i \varphi(\| x-c_i \|)$$

其中，基函数 $\phi(\| \cdot \|)$ 为非线性函数，H 为隐含层神经元的个数，c_i 表示训练数据点 S 的中心，λ_i 为线性输出层中第 i 个节点的权重。目前很多典型的径向基函数，如高斯函数、拟多二次函数、三次函数、反演 S 型函数等，已被提出。由于已有实验证明在给定预算很少时，三次函数的表现较好[25-26]，因此本文中的激活函数采用三次函数，其表示如下：

$$\varphi(\| x-c_i \|)=(\| x-c_i \|)^3$$

3 AMS-MTO 框架

在多代理多任务优化中，两个代理模型作为两个相关的任务来加速算法的收敛过程。其中一个作为全局模型，用于探索搜索空间中全局最优解所在的区域；另一个作为局部模型来辅助局部区域最优解的开发。两个代理模型协同进化，从而快速逼近原始目标函数。

多代理多任务优化使用广义的 MFEA 算法[27]来实现种群的进化。虽然 MFEA 在解决相似任务间知识迁移的策略上具有显著的有效性而被广泛应用，但是在本文中其不能保证两个不同的代理任务具有相似的最优解，即使它们在相同的初始数据集上进行训练。当两个具有不同技能因子的父代个体进行交叉时，只有其各自所负责的任务存在一定相似性时才能促进任务间的迁移，否则会导致知识的负迁移。针对该缺点，本文对 MS-MTO 算法[28]的迁移策略进行了改进，提出了一种代理间与代理内的两层知识迁移学习算法 AMS-MTO。

3.1 AMS-MTO 模型框架

本节主要介绍 AMS-MTO 算法的框架和主要流程。首先，两层知识迁移分为代理间的知识迁移和代理内部知识迁移，其中，代理间的迁移利用模型的共性和相似性来加快每个任务的收敛速度，实现的是跨任务间的迁移；而代理内部的知识迁移是将决策变量从一个维度迁移到另一个维度，实现的是跨维迁移，从而在防止算法陷入局部最优的同时提高任务的搜索速率。

AMS-MTO 算法的主要流程如图 2 所示。首先，在决策空间中采用拉丁超立方抽样技术生成 N_l 个解，并用实际目标函数对其进行评估后保存在档案 Arc 中。然后，构建 RBF 全局代理模型和局部代理模型，其中全局代理模型使用档案 Arc 中的所有数据进行训练，而局部代理模型只选取 Arc 中前 N_l 个最优适应度的数据进行训练。这里的 N_l 作为局部代理模型训练所需的样本数，要小于或等于 Arc 中的总数据量。最后，采用改进的广义 MFEA 算法同时优化两个代理模型。在 AMS-MTO 框架中，每隔 5 代进行一次任务内的跨维迁移学习，其余均执行任务间的知识迁移。以上两个过程的详细流程将在后文介绍。

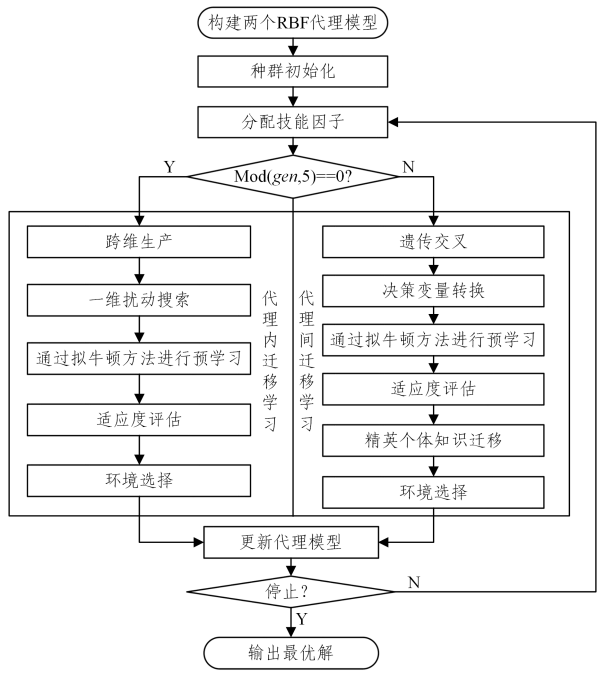


图 2 AMS-MTO 算法的流程图

Fig. 2 Flowchart of AMS-MTO algorithm

3.2 MFEA 算法框架

本文中多任务优化方法使用了多因子进化算法，这是一种

常见的进化多任务优化方法，目标是借助问题的相似性来同时求解多个不同的优化问题，从而加快每个问题的收敛速度。

在 MFEA 算法中，首先在统一搜索空间 Y 中随机初始化一个种群，并根据适应度评估为每一个个体随机分配一个技能因子。个体在其技能因子所代表任务上的目标值为适应度评估值，而在其他任务上的目标值设为无穷大。图 3 描述了 MFEA 框架的流程。从图中可以看出，该算法主要由两个模块来完成知识迁移，分别是选择性交配和垂直文化传播，它们协同运作，从而将一个任务的知识迁移到其他不同的任务组。需要注意的是，在选择性交配中，两个随机选择的父代必须满足一定的条件才能进行交配，上一节中已进行详细阐述。在垂直文化传播中，后代随机地继承父代的技能因子，并进入到相应任务组，只在该任务上进行评估。通过这种方法，可以实现解在不同任务间的不断传递。在每一次迭代结束后，会根据标量适应度对父代和后代个体进行排序，选择表现良好的个体作为下一轮迭代的父代。

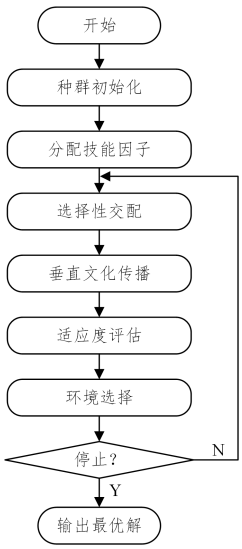


图 3　MFEA 算法的框架

Fig. 3　Framework of MFEA algorithm

3.3　代理间的知识迁移

多代理多任务框架中将两个代理模型作为两个任务同时进行优化。在种群进化过程中，个体的技能因子可能会在迭代过程中随着适应度值的变化而变化。当其发生改变时，表明个体从原来的技能因子所代表的任务组中迁移到了一个新的任务组内，该过程充分地体现了知识迁移的思想。本节应用了两种代理间知识迁移的策略，一种是通过选择交叉来实现隐式知识迁移，另一种是通过交换任务间的精英个体来实现显式知识迁移。

隐式知识迁移通过染色体交叉来实现隐式知识迁移，首先从当前种群中随机选择两个父代个体，使用选择性交配来产生子种群，并进行垂直文化传播，使得每一个后代个体都从父代个体中随机地选择一个并模仿其技能因子。因此在该操作中，具有不同技能因子的父代交配可以实现遗传信息的传递，同时技能因子的随机继承制可以促进任务间的知识迁移。而在变异操作中，子代个体直接模仿父代个体的技能因子。特别地，两个随机选择的亲本个体基于平衡因子 RMP 进行交叉或变异。

显式知识迁移通过精英个体在任务间的迁移来实现显式知识迁移，具体来说就是在种群进化过程中不断地改变适应度最好个体的技能因子。在每一代中，对每个任务的精英个体都进行存档。考虑到不同任务之间具有共性和相似性，为每个精英个体分配一个新的技能因子，使其进入其他任务组。因此，当多个优化任务具有很强的共性和相似性时，那么一个任务的最优解在其他任务上的适应性也会很好。

然而，选择性交配和垂直文化传播具有很强的随机性，从而限制了种群的搜索效率和收敛速度；而显式的精英个体迁移可以有效弥补隐式迁移的不足。

3.4　代理内的知识迁移

除代理间迁移学习外，该算法还在代理内进行迁移学习。代理内迁移学习是将知识从一个维度迁移到同一代理任务内的其他维度。跨维一维搜索算法与模拟二进制变异算法有很好的互补性，可以避免算法陷入局部最优。

下面将通过图 4 详细地阐述跨维知识迁移的方法。假定优化问题为 $f(x)=x_1+2x_2+x_3$，在当前种群中随机选择个体 p_1，p_2 和 p_3 是与 p_1 具有相同技能因子的两个个体。首先，从 3 个个体中选取出第一维的决策变量，分别是 3，4，6，对这 3 个基因进行差分进化得到新基因 2，2，8。然后，对个体 p_1 启动跨维搜索，即利用新的基因依次替换父代个体 p_1 的基因。因为 2 小于 3，所以替换后的个体适应度值比 p_1 的小，因此保留该替换操作；同理，基因 2 替换掉基因 6。最后，由于 8 大于 7，若执行替换步骤，则会导致适应度值增大，因此最后一维的决策变量不进行替换。从图中可以看出，个体 p_1 的适应度值从 22 减小到 13，由此证明通过代理内的跨维搜索可以有效地进行知识迁移。

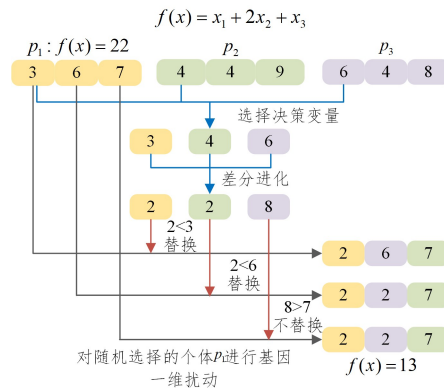


图 4　跨维知识迁移图解

Fig. 4　Cross-dimensional knowledge transfer diagram

4　AMS-MTO 算法

本节主要介绍所提出算法伪代码的具体实现并分析计算复杂度。

4.1　AMS-MTO 算法的具体实现

算法 1 给出了 AMS-MTO 算法的整体框架。首先，使用统一的编码方案初始化种群中的所有个体，并为每个个体分配一个技能因子，表示其被指派到该技能因子所对应的任务组。算法中每经过 5 代，就会进行一次代理内部的跨维知识

迁移作为局部搜索，通过跨维产生后代，然后对选定的某一个体进行一维决策变量的扰动搜索，最后对形成的新个体进行适应度的评估，同时利用选择操作保留效果好的个体，否则，将执行代理间的跨任务知识迁移学习。该过程通过遗传交叉、变异、决策变量迁移[27]、拟牛顿预学习、适应度评估、精英个体知识迁移和环境选择等操作来实现。父代种群通过二进制交叉和多项式变异来产生后代种群，为了避免两个代理模型最优解的位置不同，引入了决策变量迁移策略，并采用拟牛顿法来进行个体预学习，从而加快搜索；然后进行个体的适应度评估；在对个体进行适应度优劣排序后，应用精英个体迁移策略，并从当前种群中选取 N 个精英个体作为下一代父代种群。最终，两个代理模型的最优解将使用昂贵的实际目标函数进行评估，并保存在档案 Arc 中，用于两个代理模型参数的更新。

算法 1 AMS-MTO 算法

输入：给定目标函数和决策空间的 K 个优化任务$\{T_k|k=1,\cdots,K\}$

输出：K 个最优解$\{x_k^*|k=1,\cdots,K\}$

1. 在决策空间中，对每一个任务使用拉丁超立方体抽样方法初始化档案 Arc；
2. while 计算预算未耗尽 do；
3. 构建/更新两个 RBF 代理模型：全局代理和局部代理模型；
4. 设置交叉和变异操作的参数线性更新；
5. 初始化 N 个个体以获得初始种群P_0；
6. 在全局和局部模型中分别评估每个个体；
7. for $p_i \leftarrow 1$ to P_0 do；
8. 分配技能因子τ_i；
9. 重置两个代理模型的因子代价；
10. end for；
11. $t=0$；
12. while 停止准则未满足 do
13. 为每个任务提取相应的数据集；
14. if mod(gen,5)==0 do；
15. 代理内迁移学习(算法 3)；
16. else do；
17. 代理间迁移学习(算法 2)；
18. 记录两个代理模型的最优解；
19. 合并P_t和C_t作为临时种群R_t；
20. 更新R_t中每一个个体的标量适应度 φ；
21. 选择种群R_t中的 N 个精英个体作为下一代的父代种群P_{t+1}；
22. $t=t+1$；
23. end if；
24. end while；
25. 用实际目标函数评估两个最优解的适应度值；
26. 将两个适应度值保存在 Arc 中；
27. end while；
28. return $\{x_k^*|k=1,\cdots,K\}$.

算法 2 给出了代理间知识迁移学习的伪代码。该部分应用了基于选择交叉的隐式知识迁移和基于任务间精英个体交换的显式知识迁移的两种迁移策略。具体来讲，首先执行决策变量迁移策略，然后从当前种群P_t中随机选择两个父代个体，使用选择性交配来产生子种群C_t，再对C_t进行决策变量迁移，从而解决两个代理模型最优解位置不同的问题。对后代C_t中的每一个个体c_i，根据垂直文化传播分配一个技能因子s_i，并随机交换兄弟个体间的决策变量，然后通过拟牛顿方法进行个体预学习，最后对所有的个体在其所负责的任务上进行评估。

算法 2 代理间迁移学习算法

输入：当前种群P_t

输出：具有技能因子的后代种群C_t

1. 通过选择性交配产生后代种群C_t；
2. 对后代C_t执行决策变量转换；
3. for $c_i \leftarrow 1$ to C_t do：
4. 通过垂直文化传播分配技能因子s_i；
5. 随机交换兄弟个体间的决策变量；
6. 通过拟牛顿算法进行个体预学习；
7. 在任务s_i上评估后代个体c_i；
8. end for；
9. return C_t；

算法 3 给出了代理内知识迁移学习的伪代码。首先，在统一编码空间中，从当前种群P_t中随机生成个体p_i，同时随机筛选出和个体p_i有共同技能因子的个体，对它们的一维决策变量进行差分演化；然后，依次替换个体p_i相应的决策变量，从而得到后代种群。换言之，上述过程是利用个体p_i每一维的决策变量任务组内的其他个体信息来进行突变。对每一维突变后得到的后代，首先进行一轮个体预学习，然后在p_i所支配的任务上进行评估和环境选择。具体地，当突变后代的适应度值比父代的适应度更小时，则保留替换操作；否则，再还原回父代原始的决策变量。

算法 3 代理内迁移学习算法

输入：当前种群P_t；统一编码空间中决策变量的数目 D

1. 从种群P_t中随机地选择个体p_i；
2. $Diff_{p_i}(1,D) \rightarrow Off_{p_i}(1,D)$；/* 对每一维的决策变量进行差分进化得到后代种群 */
3. for $j \leftarrow 1$ to D do：
4. $d_j=(p_i(1),\cdots,p_i(j-1),Off_{p_i}(j),p_i(j+1),\cdots,p_i(D))$；/* 对$p_i$进行一维突变 */
5. 利用拟牛顿算法进行个体预学习；
6. 在p_i所支配的任务上评估d_j；
7. if $Fitness(d_j)<Fitness(p_i)$ do：
8. $p_i(j)=Off_{p_i}(j)$；
9. end if；
10. end for；

4.2 AMS-MTO 算法计算复杂度分析

对于一组多任务，AMS-MTO 算法的计算复杂度主要来源于：实际代价函数的适应度评估、档案 Arc 中个体适应度值的排序、两个 RBF 代理模型的训练、代理间和代理内通过知识迁移搜索最优解的过程，以及诸如新种群的产生、选择性交配、变异等计算复杂度可以忽略的其他操作的过程。因此，AMS-MTO 的总计算时间复杂度(T)可以写成如下表达式：

$$T=T_e+\sum_{i=1}^{n}(T_s^i+T_{\mathrm{RBF}}^i+T_{\mathrm{AMSMTO}}^i+T_t^i)$$

其中，T_e是使用实际代价函数进行适应度评估的时间复杂度，n 是算法的运行次数，T_s是根据适应度值对档案 Arc 中的

个体进行排序的时间复杂度。假设 Arc 中保存了 N 个历史表现最优的个体，即 N 条数据，那么 $T_s=O(N^2)$。T_{RBF} 是训练两个代理模型的时间复杂度，其中全局代理模型使用档案 Arc 中的所有数据进行训练，时间复杂度为 $O(N^3)$；局部代理模型只选取 Arc 中前 N_l 个最优适应度的数据进行训练，时间复杂度为 $O(N_l^3)$，因此 $T_{RBF}=O(N^3+N_l^3)$。T_{AMSMTO} 是 AMS-MTO 多任务进化算法的时间复杂度，若算法中的种群大小为 N_p，任务的维数为 D，算法的最大迭代次数为 M，那么 $T_{AMSMTO}=O\left(N_p\times D+I\times\left(\frac{4}{5}N_p^2+\frac{1}{5}D^2\right)\right)$。$T_t$ 是其他可忽略的操作的计算时间复杂度。

5　实验与结果分析

本节为了验证 AMS-MTO 算法的有效性和性能，分别在 10，20，30，100 维的 8 个基准测试问题上将其与一些最近针对昂贵问题优化所提出的算法进行了对比实验，包括 MS-EO[28]，TLSAPSO[16]，SHPSO[17] 和 MS-MTO[28]。

5.1　数据集描述

本文使用具有不同景观特征的 8 个基准问题来验证 AMS-MTO 算法的有效性。表 1 列出了 8 个基准问题的特征和全局最优解以及每个问题的维数。

表 1　实验所使用的 8 个基准函数的特征

Table 1　Characteristics of eight benchmark functions used in experiment

	Benchmark problem	Dimension	Characteristics	Global optimum
F1	Ellipsoid	10，20，30，100	Unimodal	0
F2	Rosenbrock	10，20，30，100	Multimodal with narrow valley	0
*F*3	Ackley	10，20，30，100	Multimodal	0
*F*4	Griewank	10，20，30，100	Multimodal	0
*F*5	Rastrigin	10，20，30，100	Multimodal	0
*F*6	Shifted Rotated Rastrigin	10，20，30，100	Very complicated multimodal	-330
*F*7	Rotated Hybrid Composition Function 1	10，20，30，100	Very complicated multimodal	120
F8	Rotated Hybrid composition function with a Narrow Basin for the Global Optimum	10，20，30，100	Very complicated multimodal	10

5.2　参数设置

在多任务优化中，假设有 K 个不同的优化任务，那么种群相应地被分为 K 个不重叠的任务组。当多任务优化开始搜索两个代理任务的最优解时，首先生成一个新的初始种群，种群数量 N_p 设置为 100。对于昂贵问题，为了搜索比当前的最优解更好的解，将在初始种群中包含一些由实际函数评估的存档 Arc 中的历史解。在本文中，设置初始群体中 $0.25N_p$ 个个体将从存档 Arc 中复制。此外，将 AMS-MTO 的迭代次数设置为 10，以便在单峰和多峰问题上获得更稳定的性能。在进化计算过程中，将交叉概率设置为 1，变异概率设置为 10，随着迭代的进行，交叉概率线性增加，从而获得更好的性能。在本文中，对于维数不大于 30 的问题和维数等于 100 的问题，分别设置了不同的停止标准，即将最大代价适应度评估数分别设置为 $11\times D$ 和 1 000。以上的参数确定过程均来源于文献[28]。

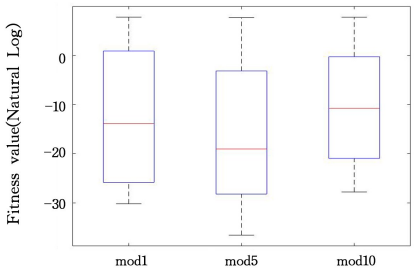


图 5　不同 mod 值在 30 维 F1 问题上 AMS-MTO 的性能

Fig. 5　Performances of AMS-MTO with different values of mod on 30-dimensional F1

另外，AMS-MTO 算法设置每经过 5 代进行一次代理内的迁移学习。图 5 分别给出了每隔 1 代、每隔 5 代和每隔 10 代进行一次代理内迁移最终所获 F1 问题结果（运行 10 次的平均值）的箱线图。从图中可以看出，参数值设置为 5 时，多任务进化优化算法可以获得相对而言最好的性能。

实验中的所有算法均在 MATLAB © 2016b 中实现，并在 AMD Ryzen 7 5800H with Radeon Graphics @3.20 GHz 笔记本电脑上运行。

5.3　与最近所提算法的比较

为了对 AMS-MTO 算法的性能做出更好的评估，下面将其与最近提出的 MS-EO，TLSAPSO 和 SHPSO 算法在 8 个基准问题上进行实验对比。由于文献[17]中并没有提供 10 维和 20 维的 $F1-F8$ 基准问题上的优化信息，因此本文只在 30 维和 100 维的问题上与 AMS-MTO 进行了结果的比较。

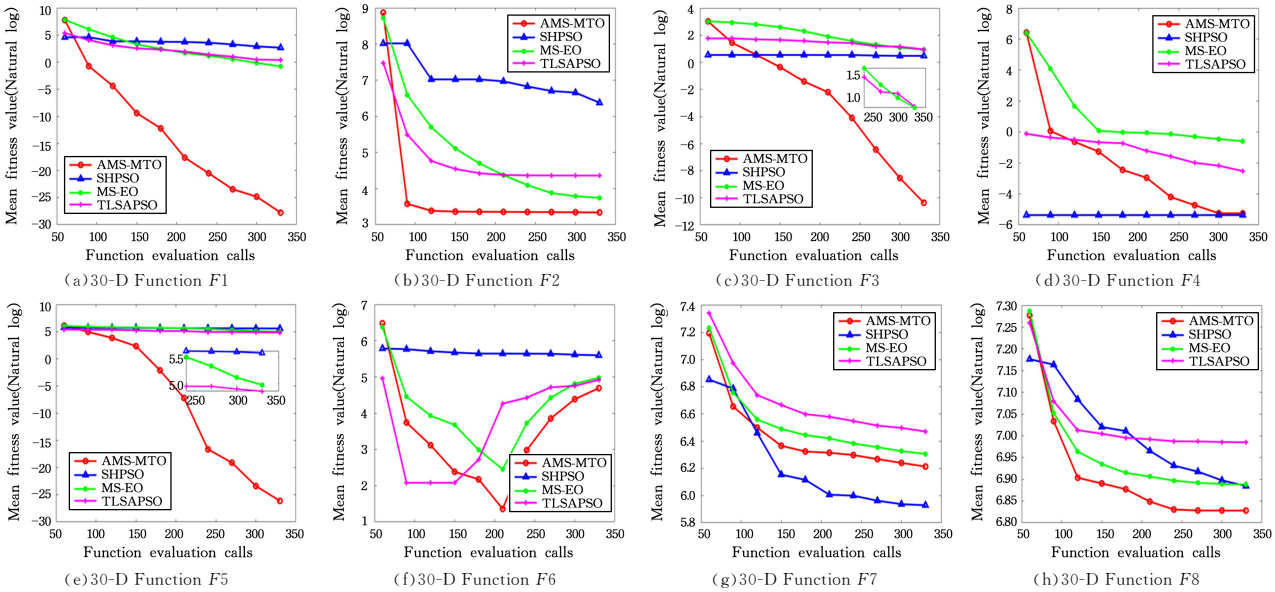
表 2 列出了在 10 次迭代优化中，AMS-MTO，MS-EO，TLSAPSO 和 SHPSO 算法分别在 10 维、20 维、30 维和 100 维的 $F1-F8$ 这 8 个基准问题上的平均最优适应度的静态数据。从表中可以看出，AMS-MTO 算法在 10 维、20 维、30 维、100 维的 $F1-F6$ 问题和 $F8$ 问题上均获得了更好的结果；在 10 维、30 维和 100 维的 $F7$ 问题上，AMS-MTO 获得了具有可比性的结果。此结果表明，AMS-MTO 算法在解决低维和高维问题上都具有一定的竞争力。

图 6 和图 7 分别给出了在低维（30D）和高维（100D）两种问题上的 4 种算法各自的优化收敛曲线。从图中可以看出，AMS-MTO 算法相比于 MS-EO，TLSAPSO 和 SHPSO 算法有更好的收敛速度。图中还可看出，在 30 维和 100 维的复合基准函数 $F7$ 上，SHPSO 的收敛性也明显优于 AMS-MTO，MS-EO 和 TLSAPSO；另外，AMS-MTO 算法虽然在 100 维的 $F8$ 问题上可能陷入局部最优点，但其性能仍远优于其他两种算法。

表 2 4 种算法的优化结果

Table 2 Optimization results of four algorithms

Problems		MS-EO	TLSAPSO	SHPSO	AMS-MTO
F1	10D	1.06×10^{-1}	1.09×10^{-2}		$\mathbf{6.58\times10^{-8}}$
	20D	1.56×10^{-1}	1.27×10^{-2}		$\mathbf{3.59\times10^{-12}}$
	30D	4.57×10^{-1}	6.14×10^{-1}	4.06×10^{2}	$\mathbf{8.62\times10^{-13}}$
	100D	2.01×10^{1}	1.36×10^{3}	3.75×10^{3}	$\mathbf{6.47\times10^{-12}}$
F2	10D	1.13×10^{1}	9.69		**8.76**
	20D	2.61×10^{1}	3.95×10^{1}		$\mathbf{1.84\times10^{1}}$
	30D	4.23×10^{1}	7.82×10^{1}	3.04×10^{4}	$\mathbf{2.82\times10^{1}}$
	100D	1.75×10^{2}	2.36×10^{4}	9.20×10^{4}	$\mathbf{9.68\times10^{1}}$
F3	10D	3.47	6.90		$\mathbf{3.87\times10^{-2}}$
	20D	2.83	2.65		$\mathbf{1.10\times10^{-3}}$
	30D	2.53	2.17	1.61	$\mathbf{3.15\times10^{-5}}$
	100D	4.70	3.40	1.06×10^{-1}	$\mathbf{1.96\times10^{-5}}$
F4	10D	8.11×10^{-1}	3.94×10^{-1}		$\mathbf{1.87\times10^{-1}}$
	20D	6.42×10^{-1}	3.57×10^{-1}		$\mathbf{5.80\times10^{-2}}$
	30D	5.53×10^{-1}	5.65×10^{-2}	1.62×10^{-1}	$\mathbf{5.20\times10^{-3}}$
	100D	6.13×10^{-1}	9.65×10^{-2}	1.12	$\mathbf{2.00\times10^{-5}}$
F5	10D	6.34×10^{1}	5.86×10^{4}		$\mathbf{4.40\times10^{-2}}$
	20D	1.34×10^{2}	1.44×10^{2}		$\mathbf{3.14\times10^{-10}}$
	30D	1.44×10^{2}	1.20×10^{2}	3.12×10^{2}	$\mathbf{4.50\times10^{-12}}$
	100D	5.67×10^{2}	8.34×10^{2}	1.07×10^{3}	**0**
F6	10D	-2.58×10^{2}	-2.83×10^{2}		$\mathbf{-2.64\times10^{2}}$
	20D	-1.69×10^{2}	-1.85×10^{2}		$\mathbf{-1.81\times10^{2}}$
	30D	-1.45×10^{2}	-1.39×10^{2}	3.14×10^{2}	$\mathbf{-1.50\times10^{2}}$
	100D	4.07×10^{2}	1.76×10^{3}	1.21×10^{3}	$\mathbf{3.46\times10^{2}}$
F7	10D	$\mathbf{4.75\times10^{2}}$	5.76×10^{2}		4.88×10^{2}
	20D	4.42×10^{2}	5.28×10^{2}		$\mathbf{4.15\times10^{2}}$
	30D	5.48×10^{2}	6.14×10^{2}	$\mathbf{3.76\times10^{2}}$	4.99×10^{2}
	100D	4.71×10^{2}	5.32×10^{2}	$\mathbf{3.57\times10^{2}}$	4.47×10^{2}
F8	10D	1.17×10^{3}	1.23×10^{3}		$\mathbf{1.01\times10^{3}}$
	20D	1.17×10^{3}	1.07×10^{3}		$\mathbf{9.10\times10^{2}}$
	30D	9.80×10^{2}	1.45×10^{3}	1.16×10^{3}	$\mathbf{9.23\times10^{2}}$
	100D	1.21×10^{3}	1.09×10^{3}	1.41×10^{3}	$\mathbf{9.10\times10^{2}}$

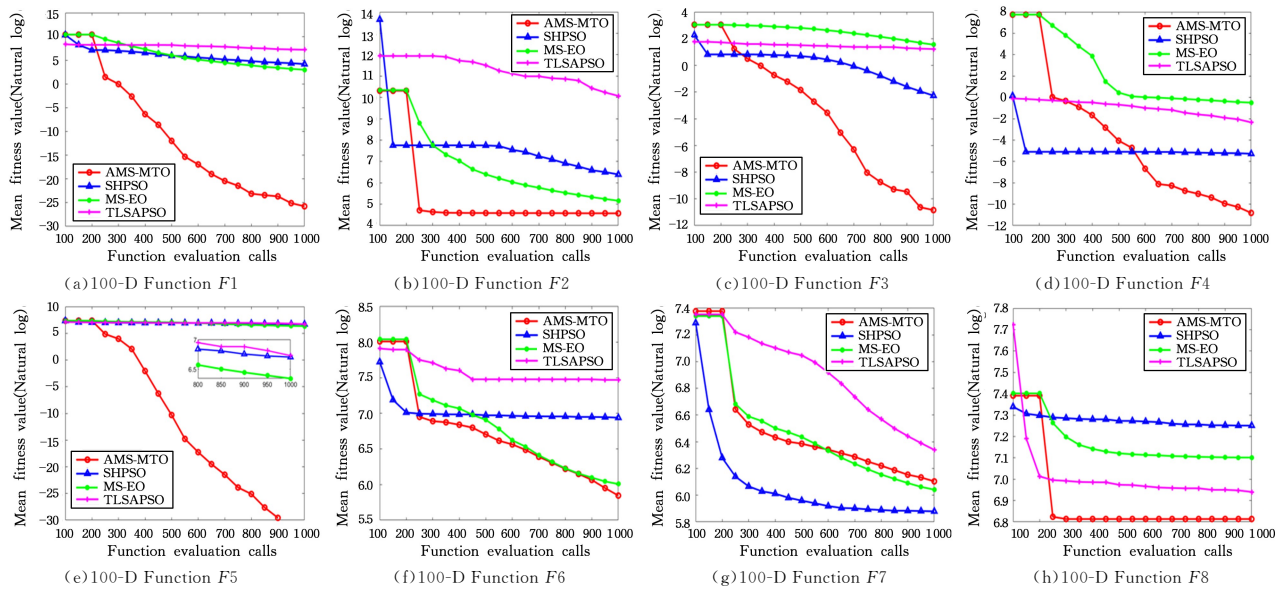


(a)30-D Function *F*1 (b)30-D Function *F*2 (c)30-D Function *F*3 (d)30-D Function *F*4

(e)30-D Function *F*5 (f)30-D Function *F*6 (g)30-D Function *F*7 (h)30-D Function *F*8

图 6 MS-EO,TLSAPSO,SHPSO 和 AMS-MTO 算法在 30 维 *F*1-*F*8 问题上的收敛结果图

Fig. 6 Convergence result graphs of MS-EO, TLSAPSO, SHPSO and AMS-MTO algorithms on 30-dimensional *F*1-*F*8 problems



(a)100-D Function *F*1 (b)100-D Function *F*2 (c)100-D Function *F*3 (d)100-D Function *F*4

(e)100-D Function *F*5 (f)100-D Function *F*6 (g)100-D Function *F*7 (h)100-D Function *F*8

图 7 MS-EO,TLSAPSO,SHPSO 和 AMS-MTO 算法在 100 维 *F*1－*F*8 问题上的收敛结果图

Fig. 7 Convergence result graphs of MS-EO, TLSAPSO, SHPSO and AMS-MTO algorithms on 100-dimensional *F*1－*F*8 problems

5.4　与多代理多任务优化 MS-MTO 的比较

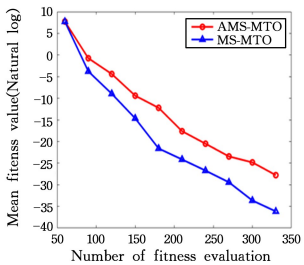
为了证明改进后的 AMS-MTO 优化算法的有效性，分别在 10，20，30 和 100 维的 8 个基准问题上将其与 MS-MTO 算法进行了比较。在实验中，AMS-MTO 和 MS-MTO 的种群和交配参数设置为相同的值。此外，两种算法的迭代次数均设置为 10，以确保搜索最优值时的总适应度评估次数相同。

表 3 列出了 10 次迭代期间 AMS-MTO 和 MS-MTO 的最优目标函数值。从表中可以看出，在 32 个测试问题中，AMS-MTO 可以在其中 25 个问题上获得更优化的结果，其比 MS-MTO 更有效。图 8 和图 9 分别显示了 30 维 $F1-F8$ 问题上 AMS-MTO 和 MS-MTO 的收敛曲线和 100 维 $F1-F8$ 问题上的单次运行时间曲线。从图中可以看出，AMS-MTO 在 $F2$，$F3$，$F5$，$F6$ 和 $F7$ 上具有更好的收敛性能，在 $F1$ 和 $F4$ 上具有可比性。在 MS-MTO 算法中，每一代都必须基于遗传算法在代理之间进行知识迁移；而 AMS-MTO 设置为每 5 代在代理之间执行简单的知识迁移。因此，AMS-MTO 在每一代的运行速度上都有了显著提高。通过以上分析，我们可以得出结论：AMS-MTO 算法的收敛速度比 MS-MTO 方法快得多；同时，这也有力地证明了两层知识迁移策略的有效性。

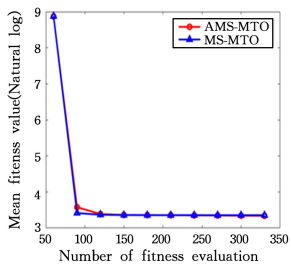
表 3　MS-MTO 和 AMS-MTO 的优化结果

Table 3　Optimization results of MS-MTO and AMS-MTO

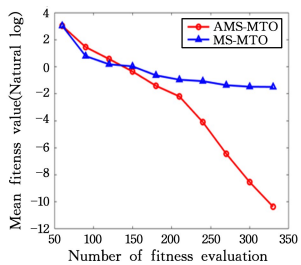
Problems		MS-EO	AMS-MTO	Problems		MS-EO	AMS-MTO
F1	10D	$\mathbf{4.37\times10^{-15}}$	1.34×10^{-12}	$F5$	10D	9.38×10^{-9}	$\mathbf{2.62\times10^{-9}}$
	20D	$\mathbf{4.38\times10^{-17}}$	2.30×10^{-13}		20D	8.64×10^{-12}	$\mathbf{3.81\times10^{-12}}$
	30D	$\mathbf{7.04\times10^{-18}}$	2.48×10^{-14}		30D	2.88×10^{-13}	$\mathbf{2.13\times10^{-14}}$
	100D	4.90×10^{-15}	$\mathbf{3.27\times10^{-15}}$		100D	0	0
$F2$	10D	8.7887	**8.6382**	$F6$	10D	-2.74×10^{2}	$\mathbf{-2.81\times10^{2}}$
	20D	1.87×10^{1}	$\mathbf{1.82\times10^{1}}$		20D	-2.07×10^{2}	$\mathbf{-2.25\times10^{2}}$
	30D	2.85×10^{1}	$\mathbf{2.78\times10^{1}}$		30D	-2.31×10^{1}	$\mathbf{-2.28\times10^{2}}$
	100D	9.65×10^{1}	$\mathbf{9.65\times10^{1}}$		100D	3.24×10^{2}	$\mathbf{1.98\times10^{2}}$
$F3$	10D	1.55×10^{-2}	$\mathbf{5.60\times10^{-4}}$	$F7$	10D	$\mathbf{3.53\times10^{2}}$	3.64×10^{2}
	20D	3.10×10^{-2}	$\mathbf{1.60\times10^{-5}}$		20D	4.42×10^{2}	$\mathbf{3.82\times10^{2}}$
	30D	2.04×10^{2}	$\mathbf{1.35\times10^{-5}}$		30D	4.03×10^{2}	$\mathbf{3.93\times10^{2}}$
	100D	$\mathbf{1.10\times10^{-6}}$	6.29×10^{-6}		100D	4.02×10^{2}	$\mathbf{3.57\times10^{2}}$
$F4$	10D	6.80×10^{-3}	$\mathbf{2.80\times10^{-3}}$	$F8$	10D	9.10×10^{2}	$\mathbf{9.10\times10^{2}}$
	20D	1.37×10^{-5}	$\mathbf{1.43\times10^{-6}}$		20D	9.10×10^{2}	$\mathbf{9.10\times10^{2}}$
	30D	$\mathbf{1.20\times10^{-9}}$	1.14×10^{-8}		30D	9.10×10^{2}	$\mathbf{9.10\times10^{2}}$
	100D	$\mathbf{7.70\times10^{-11}}$	3.20×10^{-8}		100D	9.10×10^{2}	$\mathbf{9.10\times10^{2}}$



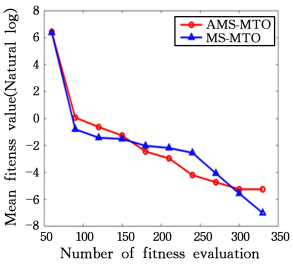
(a)30-D Function $F1$



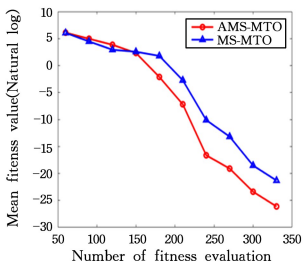
(b)30-D Function $F2$



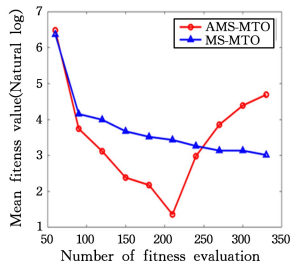
(c)30-D Function $F3$



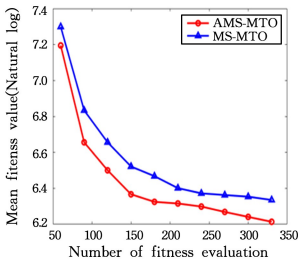
(d)30-D Functrion $F4$



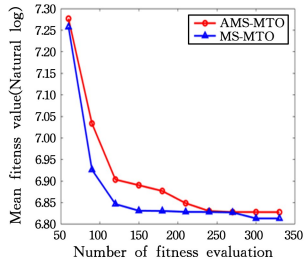
(e)30-D Function $F5$



(f)30-D Function $F6$



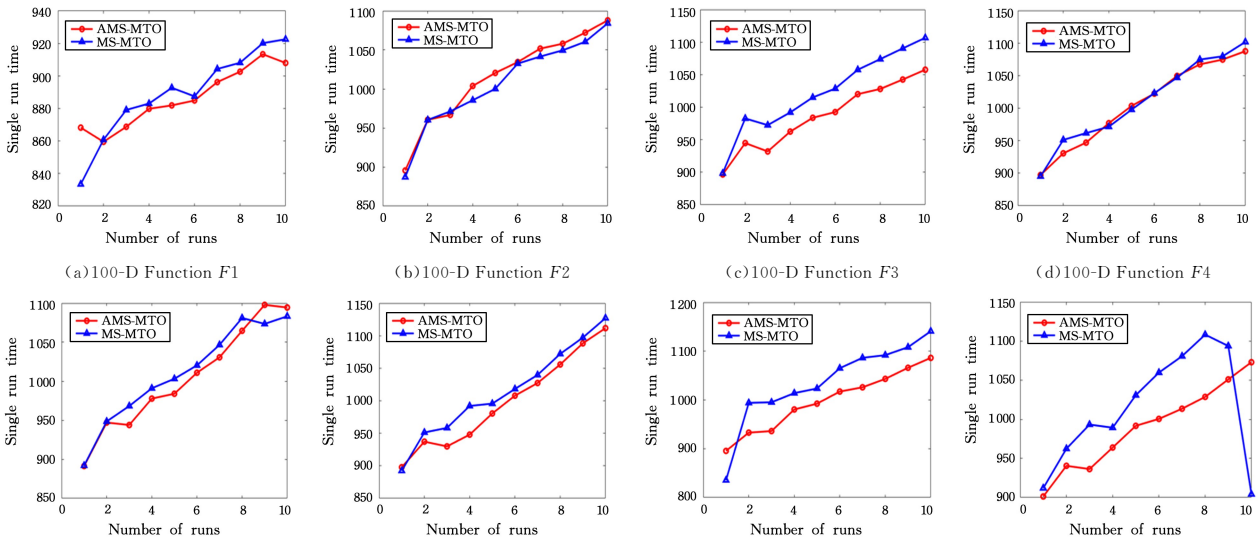
(g)30-D Function $F7$



(h)30-D Function $F8$

图 8　MS-MTO 和 AMS-MTO 算法在 30 维 $F1-F8$ 问题上的收敛结果图

Fig. 8　Convergence result graphs of MS-MTO and AMS-MTO algorithms on 30-dimensional $F1-F8$ problems



(a)100-D Function $F1$ (b)100-D Function $F2$ (c)100-D Function $F3$ (d)100-D Function $F4$

(e)100-D Function $F5$ (f)100-D Function $F6$ (g)100-D Function $F7$ (h)100-D Function $F8$

图 9 MS-MTO 和 AMS-MTO 算法在 100 维 $F1-F8$ 问题上的单次运行时间结果比较

Fig. 9 Comparison graphs of single run time results of MS-MTO and AMS-MTO algorithms on 100-dimensional $F1-F8$ problems

图 10 给出了 AMS-MTO 和 MS-MTO 算法在 100 维的 8 个基准函数上，10 次迭代的平均运行时间直方图。从图中可以直观地看出 AMS-MTO 只有在 $F2$ 函数上的寻优速度比 MS-MTO 慢，在其余问题上均比 MS-MTO 的速度快。综合两个算法的寻优能力和寻优速度来看，AMS-MTO 有着突出的优化性能，并且可以高效快速地求解多任务优化问题。

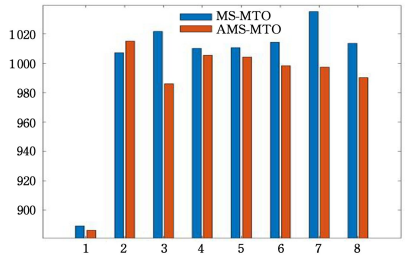


图 10 MS-MTO 和 AMS-MTO 算法运行时间直方图

Fig. 10 Run time histogram of MS-MTO and AMS-MTO

结束语 本文提出了一种基于两层知识迁移的多代理辅助多任务优化算法 AMS-MTO，构造了一个全局代理和局部代理模型，并将它们作为两个相关的优化任务，然后通过在代理间和代理内同时进行知识迁移来达到跨任务优化的目的。其中，代理内通过差分进化实现了决策变量信息的跨维迁移，从而有助于帮助算法逃离局部最优点；代理间使用了基于遗传交叉的隐式知识迁移和基于精英个体迁移的显式知识迁移两种策略，这两种策略协同工作，有利于解决知识迁移的强随机性问题，从而加快任务间信息交流的速度。

最后，通过实验验证了 AMS-MTO 算法的有效性，并在 CEC2005 的 8 个基准测试问题上与最近所提出的其他算法进行了对比。结果表明，AMS-MTO 算法在低维和高维上的收敛性能显著优于 MS-EO 和 SHPSO 算法。由于 AMS-MTO 每隔 5 代执行一次代理内的知识迁移，因此在相同的适应度评估次数上，其相比于 MS-MTO，AMS-MTO 不仅获得了明显更好或者相当的收敛性能，而且在运行速度上也显著快于后者。

尽管 AMS-MTO 在本文的研究中展示了很好的性能，但在解决高维度计算量大的昂贵问题时，该技术仍然具有很大的挑战性。在未来的研究中，会进一步探索和优化代理间知识迁移的有效性和代理模型的多样性。此外，我们也会致力于研究一些新的框架和技术，使其借助简单问题优化过程中的知识来加速昂贵问题的收敛速度。

参 考 文 献

[1] DING J L, YANG C, JIN Y C, et al. Generalized multitasking for evolutionary optimization of expensive problems [J]. IEEE Transactions on Evolutionary Computation, 2019, 23(1): 44-58.

[2] THEIL H. A rank-invariant method of linear and polynomial regression analysis[J]. Advanced Studies in Theoretical and Applied Econometrics, 1992, 23: 345-381.

[3] KOURAKOS G, MANTOGLOU A. Pumping optimization of coastal aquifers based on evolutionary algorithms and surrogate modular neural network models [J]. Advances in Water Resources, 2009, 32(4): 507-521.

[4] BUCHE D, SCHRAUDOLPH N, KOUMOUTSAKOS P. Accelerating evolutionary algorithms with gaussian process fitness function models [J]. IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews, 2005, 35(2): 183-194.

[5] GONZALEZ J, ROJAS I, ORTEGA J, et al. Multiobjective evolutionary optimization of the size, shape, and position parameters of radial basis function networks for function approximation[J]. IEEE Transactions on Neural Networks, 2003, 14(6): 1478-1495.

[6] JIN Y C. Surrogate-assisted evolutionary computation: Recent advances and future challenges [J]. Swarm and Evolutionary Computation, 2011, 1(2): 61-70.

[7] HAFTKA R T, VILLANUEVA D, CHAUDHURI A. Parallel surrogate-assisted global optimization with expensive functions — a survey [J]. Structural and Multidisciplinary Optimization, 2016, 54: 3-13.

[8] VINCENZI L，GAMBARELLI P. A proper infill sampling strategy for improving the speed performance of a surrogate-assisted evolutionary algorithm[J]. Computers and Structures，2017，178：58-70.

[9] TIAN J，TAN Y，ZENG J C，et al. Multiobjective infill criterion driven gaussian process-assisted particle swarm optimization of high-dimensional expensive problems[J]. IEEE Transactions on Evolutionary Computation，2019，23(3)：459-472.

[10] GOH C K，LIM D，MA L，et al. A surrogate-assisted memetic co-evolutionary algorithm for expensive constrained optimization problems[C]// 2011 IEEE Congress of Evolutionary Computation(CEC). 2011：744-749.

[11] LE M N，ONG Y S，MENZEL S，et al. Evolution by adapting surrogates[J]. Evolutionary Computation，2013，21(2)：313-340.

[12] YU H B，TAN Y，SUN C L，et al. A generation-based optimal restart strategy for surrogate-assisted social learning particle swarm optimization[J]. Knowledge-Based Systems，2019，163：14-25.

[13] LI F，CAI X W，GAO L，et al. A surrogate-assisted multiswarm optimization algorithm for high-dimensional computationally expensive problems[J]. IEEE Transactions on Cybernetics，2021，51(3)：1390-1402.

[14] ZHOU Z Z，ONG Y S，NAIR P. Hierarchical surrogate-assisted evolutionary optimization framework[C]// Proceedings of the 2004 Congress on Evolutionary Computation. 2004：1586-1593.

[15] LIM D，JIN Y C，ONG Y S，et al. Generalizing surrogate-assisted evolutionary computation[J]. IEEE Transactions on Evolutionary Computation，2010，14(3)：329-355.

[16] SUN C L，JIN Y C，ZENG J C，et al. A two-layer surrogate-assisted particle swarm optimization algorithm[J]. Soft Computing，2015，19：1461-1475.

[17] YU H B，TAN Y，ZENG J C，et al. Surrogate-assisted hierarchical particle swarm optimization[J]. Information Sciences，2018，454-455：59-72.

[18] BALI K K，GUPTA A，FENG L，et al. Linearized domain adaptation in evolutionary multitasking[C]// 2017 IEEE Congress on Evolutionary Computation(CEC). 2017：1295-1302.

[19] WEN Y W，TING C K. Parting ways and reallocating resources in evolutionary multitasking[C]// 2017 IEEE Congress on Evolutionary Computation(CEC). 2017：2404-2411.

[20] LIAW R T，TING C K. Evolutionary many-tasking based on biocoenosis through symbiosis：A framework and benchmark problems[C]// 2017 IEEE Congress on Evolutionary Computation(CEC). 2017：2266-2273.

[21] LI G H，ZHANG Q F，GAO W F. Multipopulation evolution framework for multifactorial optimization[C]// Genetic and Evolutionary Computation Conference. Association for Computing Machinery，2018：215-216.

[22] MIN A T W，ONG Y S，GUPTA A，et al. Multiproblem surrogates：Transfer evolutionary multiobjective optimization of computationally expensive problems[J]. IEEE Transactions on Evolutionary Computation，2019，23(1)：15-28.

[23] GUPTA A，ONG Y S. Genetic transfer or population diversification? deciphering the secret ingredients of evolutionary multitask optimization[C]// 2016 IEEE Symposium Series on Computational Intelligence(SSCI). 2016：1-7.

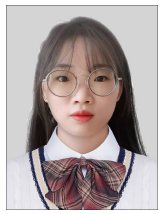
[24] KATTAN A，GALVAN E. Evolving radial basis function networks via GP for estimating fitness values using surrogate models[C]// 2012 IEEE Congress on Evolutionary Computation. 2012：1-7.

[25] WILD S M，SHOEMAKER C A. Global convergence of radial basis function trust region derivative-free algorithms[J]. SIAM Journal of Optimization，2011，21(3)：761-781.

[26] WILD S M，SHOEMAKER C A. Global convergence of radial basis function trust region algorithms for derivative-free optimization[J]. SIAM Rev，2013，55(2)：349-371.

[27] DING J L，YANG C，JIN Y C，et al. Generalized multitasking for evolutionary optimization of expensive problems[J]. IEEE Transactions on Evolutionary Computation，2019，23(1)：44-58.

[28] LIAO P，SUN C L，ZHANG G C，et al. Multi-surrogate multitasking optimization of expensive problems[J]. Knowledge-Based Systems，2021，551：23-38.



MA Hui，born in 1997，postgraduate. Her main research interests include artificial intelligence and evolutionary multi-task optimization.

FENG Xiang，born in 1977，Ph.D，professor，is a member of China Computer Federation. Her main research interests include artificial intelligence，swarm intelligence and evolutionary computing，big data intelligence.

（责任编辑：柯颖）