第 38 卷 第 4 期
Vol.38 No.4

计 算 机 工 程
Computer Engineering

2012 年 2 月
February 2012

·人工智能及识别技术· 文章编号：1000—3428(2012)04—0182—03 文献标识码：A 中图分类号：TP18

并行协同差异进化算法研究

王 磊[a]，林鸿飞[a]，滕弘飞[a,b]

(大连理工大学 a. 计算机科学与技术学院；b. 机械工程学院，辽宁 大连 116023)

摘 要：在协同差异进化(CCDE)算法和整体同步并行(BSP)计算模型的基础上，提出一种并行协同差异进化算法。采用 Archive 协同机制取代 CCDE 原有的协同机制，有助于得到算法最优解，并使用 BSP 模型实现 CCDE 的并行计算。利用标准测试函数进行仿真实验，结果表明，该算法具有较高的计算效率和计算质量。

关键词：并行计算；协同差异进化；大规模优化问题；同步并行

Research on Parallel Cooperative Coevolutionary Differential Evolution Algorithm

WANG Lei[a], LIN Hong-fei[a], TENG Hong-fei[a,b]

(a. School of Computer Science and Technology; 2. School of Mechanical Engineering, Dalian University of Technology, Dalian 116023, China)

【Abstract】This paper presents a Parallel Cooperative Coevolutionary Differential Evolution(PCCDE) algorithm based on Cooperative Coevolutionary Differential Evolution(CCDE) and Bulk Synchronous Parallel(BSP) computing model. In this algorithm, Archive collaboration mechanism replaces the original mechanism in CCDE, the PCCDE uses the BSP model to implement the parallel computation. Simulation experimental results based on a set of widely used benchmark function show that the algorithm has superior calculation efficiency and quality.

【Key words】parallel computation; Cooperative Coevolutionary Differential Evolution(CCDE); large-scale optimization problem; synchronous parallel

DOI: 10.3969/j.issn.1000-3428.2012.04.059

1 概述

目前，求解大规模优化问题是比较困难的，产生这种困难的主要原因是优化问题的复杂性和搜索空间一般会随着变量维数、约束条件以及目标函数的增多而增加，同时优化问题的特征也有可能随着变量维数的增加而改变。近年来，出现了基于合作式协同演化算法(Cooperative Coevolutionary Algorithm, CCEA)框架[1]的协同差异进化(Cooperative Coevolutionary Differential Evolution, CCDE)算法[2]，用于求解大规模优化问题。然而，CCDE 算法大多是在顺序执行的算法，对于一个大规模优化问题，这些算法计算耗时很长。为了提高计算效率和计算质量，本文在 CCDE 基础上，提出一种并行协同差异进化算法(Parallel Cooperative Coevolutionary Differential Evolution, PCCDE)，基于整体同步并行(Bulk Synchronous Parallel, BSP)计算模型[3]实现 CCDE 并行计算，并用文献[4]的 Archive 机制改进了 CCDE 算法的现有协同机制，使其能更好地适用于并行计算环境，最后在 MPI 并行环境中实现了本文算法。

2 协同差异进化算法

Potter M A 于 2000 年提出一个通用的协同进化(CCEA)模型[1]，它是一种求解复杂优化问题的有效方法[5]。2005 年，史彦军等人为求解诸如航天器舱布局这样带性能约束的高维优化问题，根据协同进化算法框架，以差异进化为算法基础，构造了协同差异进化方法[2]。其数值仿真实验表明，对于高维函数优化问题，CCDE 相对于 DE、CCGA 和 GA 有更好的计算性能，该 CCDE 算法验证算例为 100 维(本文将其验证到 1 000 维)。2007 年，Yang Zhenyu 等提出新的 DECC 算法，用来求解 1 000 维的函数优化问题取得了良好的效果[6]。可见 CCDE 对于高维优化问题十分有效，同时由于 CCDE 采用了协同进化框架，易于并行处理，但文献[2,6]的算法都是顺序执行的算法，因此设计一种并行的合作式协同差异进化算法用以求解理大规模优化问题是可行的。此外，CCDE 需解决以下关键问题：一是如何合理地进行问题分解；二是合作个体选择及其协同机制。

2.1 大规模优化问题的分解

在文献[1]提出的合作式协同进化框架中，目标函数的解空间按变量划分为多个子种群，若有 n 个变量，则有 $x=(x_1,x_2,\cdots,x_l,\cdots x_p)\in R^n$，其中，$x_l\in R^{m_l}$；$\sum_{l=1}^{p} m_l = n$，$p$ 是子问题的个数。在并行处理中，为了使得各个处理机的负载尽可能的均衡，每个子问题中所包含的变量数目尽可能的相等，即 $m_l = n/p$。

2.2 CCDE 子种群间的协同机制

在协同进化框架中，子种群中个体的适应度评价时，要为其选择合作个体从而组成完整解。一个子种群的个体可以有多个合作个体，如何选取合作个体，直接影响子种群中个体适应度的计算。目前有 4 种常用的合作个体选择机制：完全选择，随机选择，最优选择，混合选择的方式。CCDE 算法采用了最优选择的方法。在最优选择协同机制中，一个子

基金项目：国家自然科学基金资助项目(50575031, 50975033)

作者简介：王 磊(1987－)，男，硕士研究生，主研方向：智能计算，布局优化；林鸿飞、滕弘飞，教授、博士生导师

收稿日期：2011-07-08 **E-mail：**dgjx309@163.com

种群的合作个体是其他子种群中最优个体。这种协同机制能够使得进化快速稳解定的聚集到一个方向，但也容易陷入局部最优。

2.3　本文的改进合作个体选择方法

本文的算法使用了 Archive 机制[4]，替换了 CCDE 现有的协同机制。文献[4]中的算法 iCCDE 为了减少评价函数次数，同时能够利用过去有用的合作个体信息，在算法中维持一个由每个子种群较好的合作个体组成的 Archive。本文算法在保持 Archive 机制的上述特点的基础上，改变了更新和选择 Archive 个体这 2 个操作。在有 p 个子种群的系统中，本文的 Archive 中有 p 个个体，每个个体 r 被划分成 p 块，即 $r=(r_1,r_2,\cdots,r_l,\cdots,r_p)$。初始时，Archive 中的个体 r 随机产生。在以后的每一代进化中，Archive 都要根据子种群的不断进化独立更新。r 的更新如下：在每个子种群的每一代进化中，用第 l 个子种群的第 i 个个体去替换对应的 r 中的第 l 块，l 子种群对应的 Archive 产生新的个体 r^*，如果 r^* 的适应度函数值优于 r 的适应度函数值，那么就用 r^* 替代 r 作为 Archive 的新成员，否则不替换。如此循环直到遍历完子种群的所有个体($l\in\{1,2,\cdots,p\}$)。

每个子种群，在每一代进化中通过如上的更新产生一个对应的 Archive 新个体，p 个子种群产生 p 个 Archive 新个体。在同步阶段将这 p 个新个体按适应度排序，从中选择适应度最好的 r_{best}。然后再产生一个由这 p 个子种群对应的 Archive 个体组成新的完整解个体 r_{new}，新合作个体 r_{new} 中的第 l 块来自于第 l 个子种群对应的 Archive 个体的第 l 块($l\in\{1,2,\cdots,p\}$)。最后计算 r_{best} 和 r_{new} 的适应度，选择一个最优的个体作为下一代进化每个子种群对应的 Archive 个体。本文将 CCDE 算法最优选择协同机制用 Archive 机制取代，避免了最优选择协同机制过于简单，过于贪婪而引起的收敛到局部最优的问题。Archive 协同机制可以将一些系统级的较优解保存下来，在下一代评价子种群个体适应度值时就可以利用这些信息，从而使进化快速的聚向一个方向，趋向系统最优解。

3　并行 CCDE 算法

在设计并行算法时不能局限于某种具体的并行计算机，要求能够在不同的体系结构下方便地移植和有效运行，因此必须借助于抽象的计算模型。为此学者们提出了各种并行计算模型，几种典型的计算模型包括 PRAM 模型、BSP 模型、LogP 模型以及 C3 模型等。许多学者认为 BSP 模型较其他几种在计算能力、简单性及可移植性方面稍强一些，针对求解大规模优化问题本文选择了 BSP 模型[3]。

本文算法中将问题为分解为 p 个子问题空间，对应 p 个子种群，每一个子种群对应一个进程，这些进程被分配到多个处理机。这些处理机使用主从式模型(Master-Slaves)组织起来(如图 1 所示)，主节点起协调和全局同步作用，支持从节点进化，每一个从节点进行一个 DE 进化操作。每个子种群的进化分成 $t(t=1,2,\cdots)$个超步，每个超步对应一代进化，t 个超步也就是 t 代进化，t 为进化的总代数。每一代进化中分为本地计算、全局通信和同步 3 个过程。在本地计算时，每个子种群使用 DE 算法独自进化，此时各个种群的计算与其他子种群无关。在计算个体适应度时，使用本地维持的 Archive 中的合作个体进行计算，同时更新各自的 Archive，这样可以避免频繁的通信开销，从而减少计算时间。通过全局通信来传递各个子种群维持的合作个体和最优解，每个节点将其发送给主节点，从节点此时就阻塞进入路障同步，直到所有从节点都完成本次进化。最后在同步阶段主节点等到所有子种群本地计算结束，主节点收到所有从节点发来的数据，比较从节点进化信息，得出当前的原优化问题的最优解以及下一次进化各个子种群所需的合作个体，将这些信息发回给从节点，从节点在收到并存储主节点的数据后，再开始继续进行下一代进化过程。

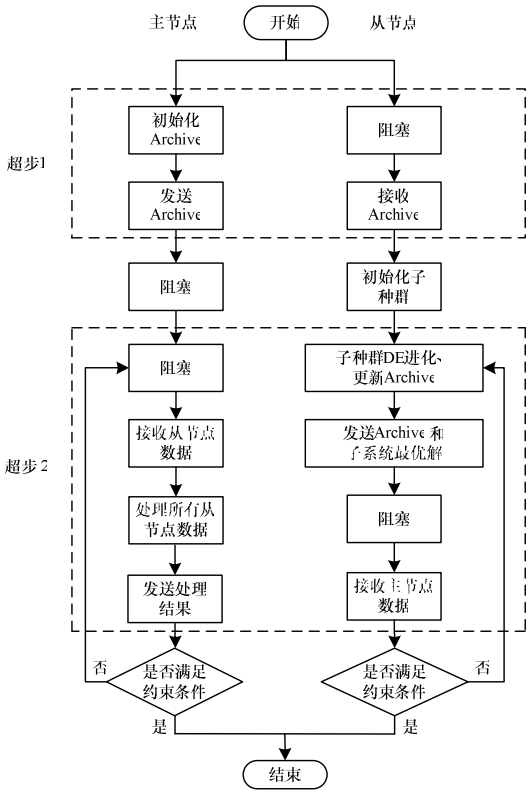

图 1　PCCDE 算法流程

本文算法具体过程如下：

主节点：

Step1 分解原问题的解空间为 p 个子种群，将这 p 个子种群分配到 p 个处理机上，并初始化 Archive 中的个体，发送给每个从节点；

Step2 对于每一代进化；

Step3 等待接收来自于从节点的最优解和 Archive；

Step4 从所有从节点中的最优解中选择最优解，并保存；根据 2.3 节所述的方法选择下一代进化所需的 Archive；

Step5 把 Step4 中得到的 Archive 发回给各个从节点；

Step6 如果满足结束条件或到达最大进化代数则结束；否则，转到 Step3 继续执行。

从节点：

Step1 等待接收主节点的数据；

Step2 初始化子种群；

Step3 对于每一代进化；

Step4 每一个个体执行 DE 进化操作，并更新 Archive；

Step5 发送最优解和 Archive 到主节点；

Step6 等待主节点处理数据，并接收主节点发送的数据；

Step7 如果满足结束条件或到达最大进化代数则结束，否则，转到 Step4 继续执行。

由于算法是并行的，因此 PCCDE 算法的时间复杂度可

以看做子种群的时间复杂度加上子种群之间的通信时间复杂度。在每一代进化中，差异进化算法的变异和交叉时间复杂度为 $O(NP\times D\times C_r)$，适应度函数的计算时间复杂度为 $O(D\times SN)$ (测试函数不同其适应度函数的复杂度也不同，这里假设适应度只和变量维数线性相关)，其中，NP 为子种群规模，D 子种群的变量维数，SN 为子种群数目，C_r 为交叉率。$O(t)$ 为每次通信时间复杂度，因此，算法的时间复杂度为：$O(D\times SN\times NP\times G+NP\times D\times C_r\times G+G\times t)$，$G$ 为迭代次数。

4 数值实验

4.1 实验函数和参数设置

为了测试本文给出的算法的计算性能，本文采用了来自于 2010 年会议“Special Issue in Soft Computing Journal on Large Scale Optimization”提供的 19 个测试函数，详见文献[7]。本文实验软件环境是 Windows XP professional 操作系统下 MPICH2-1.2.1 并行计算环境，硬件环境是单机配置 AMD Athlon 64×2 QL-65 处理器，2 GB 内存，100 Mb/s 以太网网络。算法参数设置如下：优化问题的变量维数 D 分别取，D=500 和 D=1 000；每个从节点子种群的种群个体规模 NP=40，子种群个数 $PopSize$=10；差异进化算法缩放因子 F 取值范围[0.3,1]，交叉因子 CR 取值范围[1/D,1]；本文算法选择的差异进化变异策略是 *DE/rand/*1/*bin*；评价函数最大计算次数(*Max_FEs*)，其取值由变量维数决定 *Max_FEs*=5 000×D。此外每个测试函数都独立随机的执行 25 次，试验结果取 25 次计算的平均误差值。对于一个优化解 x，其误差定义如下：$f(x)-f(x^*)$，x^* 为测试函数的最优解。

4.2 数值实验结果

在数值仿真实验中，将本文算法与以下其他算法做了比较：(1)比较 PCCDE 算法、差异进化算法(DE)、改进的遗传算法——跨世代异物种重组大变异算法(CHC)以及协同差异进化算法(CCDE)的计算精度误差值，表 1 给出了部分测试函数比较结果；(2)比较 PCCDE 和一种顺序执行的算法 CCDE，在相同的评价函数计算次数(*Max_FEs*)之下的计算耗时。

表 1 测试函数计算统计结果比较

测试函数	DE 平均误差	CHC 平均误差	CCDE 平均误差	PCCDE	
				平均误差	标准差
F_1	1.08E−15	1.36E−11	1.48E−16	3.68E−22	4.10E−22
F_3	9.69E+02	8.75E+02	9.76E+02	9.59E+02	5.68E+00
F_5	5.21E−16	7.02E−03	2.44E−15	1.79E−15	2.05E−17
F_8	2.75E+05	3.85E+11	1.19E+05	1.99E+04	1.01E+03
F_{10}	1.79E−27	3.77E+01	6.39E−18	2.87E−29	2.74E−30

图 2 显示了 CCDE 算法和 PCCDE 算法在 1 000 维的情况下的平均耗时，图 3 给出了 PCCDE 和 CCDE 的收敛曲线(限于篇幅，选择部分测试函数结果)。

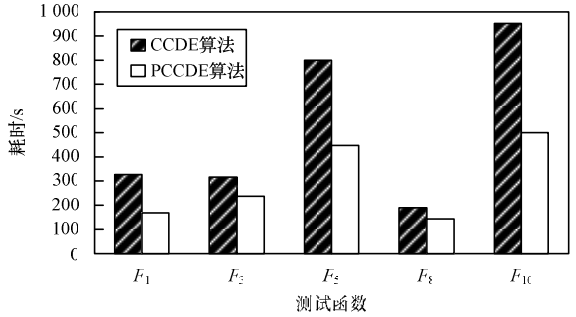

图 2 D=1 000 维时 CCDE 和 PCCDE 计算耗时比较

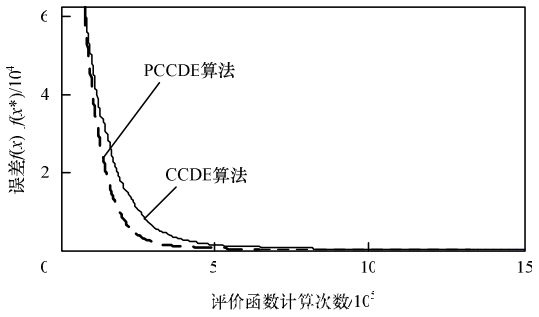

图 3 函数 F_{10} 收敛曲线

4.3 结果分析与讨论

表 1 给出了 DE、CHC、CCDE 和 PCCDE 的计算结果对比，由实验中可知，本文 PCCDE 算法的求解结果要优于其他 3 种算法，能找到大多数测试函数的最优解。在测试函数 F_1，F_8，F_{10} 上，PCCDE 算法求解结果是所有算法里最好的，在其余的几个测试函数上，PCCDE 算法求解结果和其他算法相当。由表 1 还可知 CCDE 算法对于 1 000 维的高维函数优化问题求得的结果并不是最好的，原因是对于高维弱藕合函数优化问题，CCDE 通过问题分解有效地降低了问题的求解难度，提高了计算效率，但是对于强耦合的不可分函数优化问题，CCDE 算法的计算效率并不是很高，有时甚至比不上 DE 算法，PCCDE 算法却能够在一定程度上克服这个问题。原因在于一般的合作个体产生方法类似于数值优化中依次轮换优化各个变量而固定其他变量值的方法。这类方法只对各变量相对独立的低耦合可分函数优化有效，而对各变量相互依赖的高耦合不可分函数，可能各子种群相对于其他子种群都达到了最优，即系统解的组成部分相对于其他部分达到最优，然而系统解却没有到达最优[8]。本文中的 Archive 机制可以避免最优选择合作机制过于简单和贪婪而造成的上述问题，在一定的程度上达到了系统最优。在计算计算耗时方面，由图 2 可知，PCCDE 算法的耗时明显比 CCDE 算法少，原因在于各个子种群的进化时并行执行的，但是由于各个子种群之间有一定的通信时间，并行加速比不能达到理想的状态。由此可见本文算法在解决复杂大规模优化问题上能够有效节省求解时间。所以，PCCDE 算法在这些测试函数上表现出了很好的性能，总体上要比 CCDE 算法和 DE 算法好。

5 结束语

本文提出了一种并行的协同进化差异进化算法，用于求解高维复杂函数优化问题。本文的算法基于协同差异进化(CCDE)算法，在其基础上做了以下改进：(1)改进了合作式协同进化算法的协同机制，引入了 Archive 机制；(2)基于整体同步并行计算模型(BSP)实现了 CCDE 算法并行计算，使其能够在并行计算环境下运行，对于高维优化问题的求解可以减少计算耗时。数值实验表明，本文给出的 PCCDE 算法相对于 DE、CHC 和 CCDE 算法具有更好的计算性能。

参考文献

[1] Potter M A, De J K A. Cooperative Coevolution: An Architecture for Evolving Coadapted Subcomponents[J]. Evolutionary Computation, 2000, 8(1): 1-29.

[2] Shi Yanjun, Teng Hongfei, Li Ziqiang. Cooperative Co-evolutionary Differential Evolution for Function Optimization[C]//Proc. of Advances in Natural Computation. Changsha, China: [s. n.], 2005: 1080-1088

(下转第 190 页)