第 42 卷 第 2 期　　自 动 化 学 报　　Vol. 42, No. 2
2016 年 2 月　　ACTA AUTOMATICA SINICA　　February, 2016

基于划分的多尺度量子谐振子算法多峰优化

陆志君 [1]　　安俊秀 [1]　　王 鹏 [1]

摘 要 针对多峰优化问题, 本文结合多尺度量子谐振子算法的全局优化特性提出了基于划分的多尺度量子谐振子算法. 对定义域进行合理均匀划分, 根据划分区域长度构建初始基态高斯曲线, 随着标准差衰减高斯曲线逐渐收敛, 从而在各个区域内快速搜索到极值点. 对于实际函数的维度和极值数不同, 本文提出固定分辨率策略和多级分辨率策略来解决实际问题, 通过寻优精确性、全极值点寻优和全局多峰优化三个角度进行实验, 对比蚁群算法、差分进化算法等主流群智能算法, 可以表明该算法参数设置简单, 具有很好的寻优准确性、快速收敛性和记忆性.

关键词 量子谐振子, 多峰优化, 全极值, 群智能算法

引用格式 陆志君, 安俊秀, 王鹏. 基于划分的多尺度量子谐振子算法多峰优化. 自动化学报, 2016, **42**(2): 235−245

DOI 10.16383/j.aas.2016.c150429

Partition-based MQHOA for Multimodal Optimization

LU Zhi-Jun[1]　　AN Jun-Xiu[1]　　WANG Peng[1]

Abstract To solve the problem of multimodal optimization, a partition-based multi-scale quantum harmonic oscillator algorithm (MQHOA) is proposed depending on MQHOA′s global optimization characteristic. It divides reasonably a domain into uniform areas, and then Gauss curves with ground state can be constructed according to the lengths of these uniform areas. With the attenuation of standard deviation, the Gauss curves will converge gradually, thus, extreme points can be found quickly. In addition, two strategies comprising fixed wavelength resolution and muti-level resolution are used for practical problems. Experiments are carried out from three aspects including optimization′s accuracy, all extremal points optimization and global multimodal optimization. Compared with the ant colony algorithm, differential evolution algorithms and other mainstream swarm intelligence algorithms, the algorithm has, in addition to its simpleness on setting parameters, superior optimization accuracy, fast convergence and memory property.

Key words Quantum harmonic oscillator, multimodal optimization, all the extremum, swarm intelligence algorithms

Citation Lu Zhi-Jun, An Jun-Xiu, Wang Peng. Partition-based MQHOA for multimodal optimization. *Acta Automatica Sinica*, 2016, **42**(2): 235−245

在控制与决策等领域中, 往往会遇到许多优化问题, 有时我们不仅需要找到全局最优解, 还需要找到局部最优解进行辅助决策, 这类优化问题称为多峰搜索 (Multi-peak searching) 或多峰优化 (Multimodal optimization) 问题. 随着问题的复杂性不断提升, 传统的数学规划理论很难解决.

大规模可并行的群智能算法的出现为解决多峰优化问题带来了新的思路. 其中粒子群算法 (Particle swarm optimization, PSO) 是一种基于群体协作的随机搜索算法, 但由于其对局部最优解搜索能力不好并且需要较多的先验知识, 许多学者对其进行了改进[1−4]. 同样, 差分进化算法 (Differential evolution, DE) 作为进化算法的一个重要分支, 收敛速度比 PSO 快, 但由于贪婪选择策略, 在求解具有大量局部优点的问题时, 种群极易陷入局部最优. 有学者使用临近衰减策略[5] 和双目标[6] 等策略来解决这个问题. 这些算法能够在局部找到接近最优的极值点, 但无法给出所有的极值点. 也有学者尝试使用蚁群算法[7] 和在极值点附近循环扰动的 Nelder-Mead 单纯形法[8] 来找出所有极值, 但采用的测试函数极值点均较少, 而且搜索时间较长, 不大适合用于搜索有极多极值的函数.

量子算法以其完备的理论和强大的并行能力[9], 被很多学者用于群智能算法中. 文献 [10] 在 2010 年提出了模拟量子谐振子的概念, 文献 [11] 中给出了算法实现, 即多尺度量子谐振子算法 (Multi-scale quantum harmonic oscillator algorithm, MQHOA). 该算法数学原理简单物理含义明确且无需先验知识. 完备的采样能力和精确的搜索聚焦能力使 MQHOA 能较为准确地在多维空间实现对最优中把 MQHOA 与模拟退火算法 (Simula-

收稿日期 2015-07-02　录用日期 2015-11-02
Manuscript received July 2, 2015; accepted November 2, 2015
国家自然科学基金 (60702075), 国家社会科学基金 (12XSH019) 资助
Supported by National Natural Science Foundation of China (60702075), National Social Science Foundation of China (12XSH019)
1. 成都信息工程大学并行实验室 成都 610225
1. Parallel Laboratory, Chengdu University of Information and Technology, Chengdu 610225

ted annealing, SA)、量子解的搜索. 文献 [12] 将退火算法 (Quantum annealing, QA)、量子粒子群算法 (Quantum-behaved particle swarm optimization, QPSO) 进行对比, 对 15 种不同的标准测试函数求最小值, 实验结果进一步说明了其具有更好的计算速度和计算精度.

MQHOA 集中处理单峰优化问题, 主要根据量子隧道效应[12] 保证全局采样点数目足够大以避免陷入局部较优解位置. 算法收敛的依据是这些采样点都收敛到一个最优解上, 但对于多峰函数, 采样点将分散在各个峰上而得不到聚集, 导致算法无法收敛. 为扩展 MQHOA 对多峰优化问题的应用, 本文提出了基于划分的多尺度量子谐振子算法 (Partition-based MQHOA, P-MQHOA), 它把求解域均匀划分成 N 个局部域, 把 MQHOA 的收敛特性应用在局部域内, 每个局部域中心内放置一个采样点进行谐振子迭代操作, 那么这些采样中心点的平衡位置即为所求的解. 实验使用可调节参数的双势阱量子系统函数来验证算法对于等值极值点和不等极值点寻优准确性. 对于全部极值点的搜寻效果, 本算法对比了蚁群算法对同一个函数的全极值搜寻, 实验结果超出了蚁群算法. 此外, 通过对 14 个基准函数的全局多峰优化, 对比遗传算法、粒子群算法和差分进化算法效果都要好. 当极值点极多的情况下也可以采用多级分辨率策略求解, 求解精度高, 查全率高, 速度快.

1 多峰优化问题描述

工程上的多峰优化是把这个工程问题映射到一个多维函数上, 要得到多个最优方案就是在这个多维函数上求多个极大值 (或者是求这个多维函数取相反数的多个极小值).

为了全文描述方便, 本文给出以下四个定义:

定义 1. 对于某 D 维 N 极值函数 $(N \geq 2)$, 第 i 个极值点和第 j 个极值点相邻, 他们坐标之间第 k 维差的绝对值记作 $\lambda_{i,j,k}$. 取 $\lambda_{i,j} = \min_{1\leq k\leq D} \lambda_{i,j,k}, \lambda_{\min} = \min_{1\leq i<j\leq N} \lambda_{i,j}$, 把 $\lambda_{\min}$ 称为该函数实际波长分辨率.

定义 2. 工程上需要识别某 D 维多极值函数上的极值点满足如下要求: $\lambda_{i,j} \geq \lambda_0 \geq \lambda_{\min}$. 把 λ_0 称为需求波长分辨率, 这些极值点称之为满足需求波长分辨率 λ_0 的极值点.

定义 3. 把算法求得的极值点坐标与对应的实际极值点坐标之间允许的各维最大差的绝对值称作该算法求解的位置精度, 记为 A_l.

定义 4. 某 D 维函数定义域上界为 $\boldsymbol{u}$, 下界为 $\boldsymbol{s}$, 那么称 $\boldsymbol{l} = \boldsymbol{u} - \boldsymbol{s} = (l_1, l_2, \cdots, l_D)$ 为该函数的长度域, 其中 l_k 为该函数在第 k 维上定义域的长度.

多峰优化问题的数学描述即寻找 D 维函数 $f(x_1, x_2, \cdots, x_D)$ 在定义域上某点 $X_i(x_{i,1}, x_{i,2}, \cdots, x_{i,D})$ 处取得极值. $X_i^- = (x_{i,1} - dx_1, x_{i,2} - dx_2, \cdots, x_{i,D} - dx_D), X_i^+ = (x_{i,1} + dx_1, x_{i,2} + dx_2, \cdots, x_{i,D} + dx_D)$, f_i^- 为函数在点 X_i^- 求得的值, f_i^+ 为函数在点 X_i^+ 求得的值, f_i 为函数在点 X_i 取得的函数值, X_i^* 为实际极值点, 那么判断该函数在 X_i 处取得极小值的依据:

$$\begin{cases} f_i^- > f_i, f_i^+ > f_i \\ X_i \in \{X \mid |X - X_i^*| \leq A_l\} \end{cases} \tag{1}$$

判断该函数在 X_0 处取得极大值的依据:

$$\begin{cases} f_i^- < f_i, f_i^+ < f_i \\ X_i \in \{X \mid |X - X_i^*| \leq A_l\} \end{cases} \tag{2}$$

2 基于划分的 MQHOA 原理

量子谐振子是物理上描述微粒动态振动的物理量, 可以近似描述分子的运动规律. MQHOA 算法把量子谐振子群作为智能粒子群来解决优化问题, 振子的平衡位置即势能最低位置, 就是所求函数的最优解. 量子谐振子从高能态到低能态是一个逐渐收敛的过程[13], 直到收敛到基能态: $\psi_0(x) = \frac{\sqrt{\alpha}}{\sqrt[4]{\pi}} \mathrm{e}^{-\alpha^2 \frac{(x-x_i)^2}{2}}$, 定义 MQHOA 算法在尺度 σ_s 下的波函数为 k 个以 x_i 为中心的高斯概率密度函数的迭加:

$$\psi_{QHO}(x) = \sum_{i=1}^{k} \frac{1}{\sqrt{2\pi}\sigma_s} \mathrm{e}^{-\frac{(x-x_i)^2}{2\sigma_s^2}} \tag{3}$$

$\psi_{QHO}(x)$ 近似代表了以 k 个最优解为中心的目标函数可行域上的概率分布, 通过引入多尺度优化函数二进信息高斯采样模型, 二进小波尺度函数采用高斯函数按照不同尺度和不同精度要求进行聚焦搜索[9], 可以保证信息的不遗漏采样, 从而获得全局最优解, 详细算法描述参见文献 [11].

P-MQHOA 扩展了 MQHOA 对多峰优化问题的处理能力, 并且具有对所有最优解的记忆能力和更快的收敛速度. 其主要思想是把 MQHOA 的全局优化特性局限在了一个较小划分区域内, 在这个区域内只设置一个高斯采样中心点, 并且让这个采样中心点在反复迭代过程中不断收敛到精度范围内. P-MQHOA 对整个函数定义域划分的 N 个区域与 N 个采样中心点一一对应, 最终找到 N 个解 (包括重复解). 由于这 N 个采样中心点之间的采样计算是独立进行的, 并不会由于某几个采样中心点收敛不了而影响其余采样中心点, 因此其收敛速度要比 MQHOA 要快得多, 并且算法通过对记忆的 N 个解进行整理可以得到全局最优解.

算法对函数定义域进行均匀划分, 划分区域内如何选择要生成的高斯采样点的标准差直接影响算法的性能和执行结果. 各维度划分成的各均匀区段中心位置处随机生成 M 个标准差为 σ_s 的高斯分布点, 在同一个区段内, σ_s 与这些分布点有如下关系: σ_s 越大分布点越接近均匀分布, σ_s 越小分布点越集中在中心点, 通过 σ_s 的衰减可以使得这些分布点最终聚集在搜索精度范围内. 算法初始的 σ_s 大小很重要, 分布点太均匀的话很大概率会搜索到临近的其他区段, 分布点太集中的话很大概率会忽略区域边界附近的极值点. 假如某个区段内初次生成的高斯分布点超出边界的概率为 p, M 次伯努利实验后可得到超出边界的分布点数目平均值为 Mp. 算法一般选择 σ_s 为该区段长度的四分之一, 通过查找正态分布表可知, $p=0.0456$, 即每个分布点超出区段的概率为 0.0456. 这样设置不仅可以保证对本区段内的极值搜索, 又不失高斯分布的中心点集中特性.

对定义域下界为 $\boldsymbol{s}$ 长度域为 $\boldsymbol{l}$ 的 D 维函数在第 j 维上进行均匀划分, 划分成 k 个区段后高斯概率密度叠加可得第 j 维上采样点的概率分布:

$$\psi_{jQHO}(x)=\sum_{i=1}^{k}\frac{1}{\sqrt{2\pi}\sigma_s}\mathrm{e}^{-\frac{\left(x-\frac{l_i(i-\frac{1}{2})}{k}-s_i\right)^2}{2\sigma_s^2}} \tag{4}$$

其对应的密度图如图 1 所示, 虚线为原来的高斯概率分布, 实线为叠加后的概率分布.

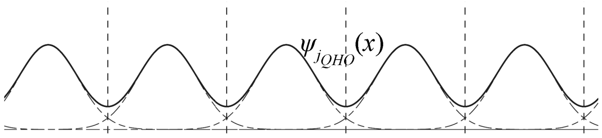


图 1　k 个高斯分布函数均匀叠加概率密度图

Fig. 1　Probability density map of the uniform superposition with k Gauss distribution functions

叠加后某一区段的高斯分布由于临近区段分布点的 p 概率跳入而弥补了本区段 p 概率跳出流失的分布点, 这样就保证了 M 个采样点在各区段数目相等, 并且有 Mp 个点作为区段之间的交流媒介保证了各区段的联系. 这种联系使得在某区段内找不到极值点的情况下可以跳出该区域协助临近区段寻找极值点. 把各个维度上的均匀划分区段组成一个个均匀划分区域, 在各个划分区域内寻找极值是相互独立的操作, 故算法可以多线程并行执行.

为了保证在需求波长分辨率 λ_0 下识别该分辨率下的所有极值点, 本算法需要在可行域上划分 K 个区域并设置 K 个采样中心点, K 的大小决定算法能准确寻找极值的数目. 以寻找一维函数极小值为例, 定义搜索区域长度为函数定义域上均匀划分的区段长度. 如图 2 所示, $\psi(x)$ 为初始采样点在区域内分布的概率密度函数. 当搜索区域长度超出该一维函数的波长分辨率 λ_0 (两个波谷之间的水平距离) 时, 算法在当前搜索区域内至少能找到一个极值点, 如图中函数 $f_1(x)$ 所示; 当搜索区域长度等于 λ_0 时, 极值点或者在该区域被找到或者在邻近区域被找到, 如图中函数 $f_2(x)$ 所示; 当搜索区域长度小于 λ_0 时, 当前搜索区域无法找到极值点, 只能在邻近区域找到, 如图中函数 $f_3(x)$ 所示. 因此, 本算法需要通过设置合理的采样中心点数使得搜索区域长度大于等于 λ_0.

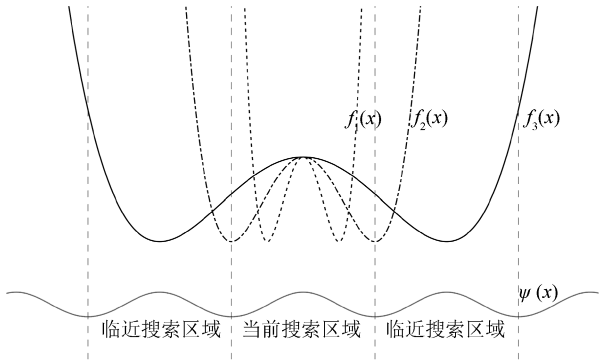


图 2　不同函数的极小值搜索

Fig. 2　Search minimum values of different functions

K 的大小也影响算法的执行速度. 以某 D 维函数为例, 其可行域 $\boldsymbol{l}=(l_1,l_2,\cdots,l_D)$, 如果需要找到所有极值, 那么搜索区域长度正好为该函数的实际波长分辨率 $\lambda_{\min}$, 以此算出合理的采样中心点数. 在第 i 维上置采样中心点个数 k_i 为 $\lceil\frac{l_i}{\lambda_{\min}}\rceil$, 即在该维度上划分出 k_i 个区段, 要找到 D 维函数所有极值点则需初始划分区域数 $K=\prod_{i=1}^{D}k_i$. 当 $\lambda_{\min}$ 很小而 D 很大时, 算法需要花费极多 CPU 时间在所有区域内进行计算. 类似多分辨率的小波分析[14], 本文提出多级分辨率策略, 其基本思想是先在大波长分辨率下进行区域划分, 在找到极值点的区域内再进行小波长分辨率的划分, 而在不存在极值点的区域就不再进行划分计算, 如此重复可以使搜索区域长度逼近 $\lambda_{\min}$, 从而找到满足需求分辨率极值点. 这种策略避免了原来固定分辨率下对无极值区域的计算浪费, 而把更多的 CPU 计算集中在极值较多的区域. 实际应用中需求波长分辨率 λ_0 一般比 $\lambda_{\min}$ 大得多, K 取 $\prod_{i=1}^{D}\left\lceil\frac{\lambda_0}{l_i}\right\rceil$. 当 K 不是很大时, 可以直接采用固定分辨率策略, 反之采用多级分辨率策略.

3　P-MQHOA 算法过程描述

3.1　P-MQHOA 算法的三个阶段

P-MQHOA 主要有三个阶段. 初始划分阶段: 该阶段主要是对 D 维度函数定义域进行区域均匀

划分, 然后在区域中心放置一个采样中心点; 量子谐振子收敛阶段: 该阶段采用高斯函数来进行采样最终定位中心点; 归并阶段: 该阶段把位置精度范围内的中心点归为一类, 并从该类中选出最好的中心点.

3.1.1 初始划分阶段

输入需求波长分辨率 λ_0 下所求 D 维度函数定义域上界为 $\boldsymbol{u}$, 下界为 $\boldsymbol{s}$, 由定义 4 可求得长度域 $\boldsymbol{l}$. 对于第 j 维划分区段数记作 k_j, 若算法采取固定分辨率策略, 初始第 j 维上划分区段数 k_j 为 $\lceil \frac{l_i}{\lambda_0} \rceil$; 若算法采取 N 级分辨率策略, 初始第 j 维上划分区段数为 $\lceil \sqrt[N]{\frac{l_j}{\lambda_0}} \rceil$. 总采样点数 $K = \prod\limits_{j=1}^{D} k_i$, 初始化采样中心点, 某个中心点在第 i 个区段第 j 维的坐标值可以记作 $c_{i,j} = \frac{l_j(i-1/2)}{k_i} i + s_j, i \in [1, k_j], j \in [1, D]$, i 与 j 均为整数.

3.1.2 量子谐振子收敛阶段

基于划分的 MQHOA 的采样中心点之间互不影响, 在每个划分区域内的收敛过程都是独立的, 因此各中心点可以并行执行. 如图 3 所示为寻找某一维函数 $f(x)$ 极大值的收敛过程. 初始过程的采样点概率分布为 6 个高斯分布函数的叠加, 见式 (4). 其区域长度正好为这个叠加函数的周期长度. 按照前面说明, 该区域的高斯函数提供 $M(1-p)$ 个采样点, 由于在可行域外产生的采样点很少可以忽略不计, 那么很容易证明该区域内约有 Mp 个采样点由其他区域的高斯采样提供, 而越靠近的区域提供越多. 在图 3 第一步中左起第二个区域内约有 $Mp/2$ 个采样点落在第三个区域, 而第三个区域内的采样点效果显然比原区域好, 因此下一轮采样将在第三个区域进行. 在找到当前区域最大值采样点后, 以这个点为中心, 搜索区域长度衰减, 重新在新的搜索区域寻找最大值, 如图 3 中第二第三步, 以此类推, 直到所有中心点要么收敛在极值点附近, 要么找不到极值点. 图 3 最后一步中所有中心点都收敛在极值点附近.

3.1.3 归并阶段

图 3 最后收敛阶段有 6 个中心点分别收敛于 4 个波峰, 其中第 2、3、4 个中心点聚集在一个极值点附近, 需要对这些点进行归并操作. 若任两个点各维坐标最大差的绝对值小于位置精度 A_l, 按照定义 3, 把这两个点归为一类, 最终从此类中取出对应函数值最大 (最小) 的点作为最终的极值点.

3.2 P-MQHOA 算法流程

综上所述, 整个算法流程可以简要描述如下:

步骤 1. 初始化:

步骤 1.1. 输入 $\lambda_0, M, A_l, \boldsymbol{u}, \boldsymbol{s}$.

步骤 1.2. 根据公式初始化长度域 $\boldsymbol{l}$, 标准差 $\sigma_s = \frac{\lambda_0}{4}$, 在各维度上均匀划分产生 K 个中心点, 其中第 i 个点 $center_i = (c_{i,1}, c_{i,2}, \cdots, c_{i,D})$.

步骤 2. 收敛过程:

步骤 2.1. 在各维上以 c_i 为中心点, σ_s 为标准差进行 M 次高斯采样, 当采样点超出可行域边界时以边界点为准.

步骤 2.2. 从 M 个采样点中选择对应函数值最大 (最小) 的一个点作为新的中心点, σ_s 衰减为原来一半. 如果满足 $\sigma_s \geq A_l$, 重复执行步骤 2.1, 否则继续执行步骤 3.1.

步骤 3. 归并:

步骤 3.1. 对最后生成的 K 个解根据式 (1) 或式 (2) 作极值判定, 去除不满足式子的点.

步骤 3.2. 计算各相邻点之间各维度差值的最小值, 若小于 A_l 则合并这些点并取其中最大 (最小) 的作为最终解.

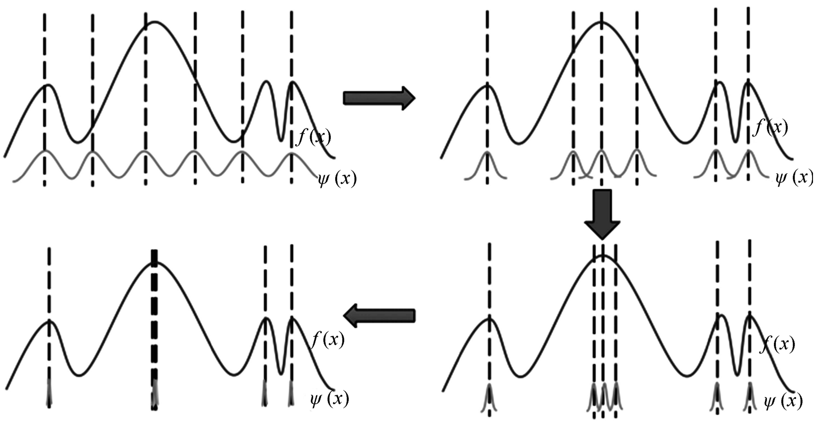


图 3 P-MQHOA 收敛过程示意图

Fig. 3 Schematic diagram of P-MQHOA convergence process

3.3　P-MQHOA 算法两种策略

针对维度较低、需求波长分辨率不高的函数进行多峰寻优, 采用固定分辨率策略即可解决. 但维度较高或者需求波长分辨率很小情况下, 应当采用多级分辨率策略.

对于 N 级分辨率, 算法经历初始划分阶段和量子谐振子收敛阶段后, 找到输出中心点不为空所对应的区域, 对这些区域执行下一级的初始划分阶段的操作并对划分的子区域执行量子谐振子收敛阶段的操作, 递归搜寻到第 N 级后对所有输出的中心点执行归并阶段的操作, 可输出最终的极值点.

以图 4 为例, 对 $f(x,y)$ 在需求分辨率为 0.1 下求极值, 长度域 $l_x=l_y=5$, 采用三级分辨率策略初始划分区域数为 $\lceil\sqrt[3]{\frac{5}{0.1}}\times\sqrt[3]{\frac{5}{0.1}}\rceil=16$, 如子图 4 (a) 所示; 经过收敛和归并过程后确认极值点在 6 号与 11 号区域, 重新在这两个区域进行第二级划分, 如子图 4 (b) 所示; 重复以上过程第三级划分后实际分辨精度可达到 $\frac{5}{4^3}=0.078$, 满足需求分辨率终止迭代, 如子图 4 (c) 所示. 若每个中心点的采样次数为 M, 可以计算总的采样次数为 $16(1+2+3)M=96M$, 而采取固定分辨率需要次数为 $50\times50M=2\,500M$, 可见其省去很多计算时间, 但多级数情况下也容易丢失部分极值点. 因为 M 在粗分辨率下相对于细分辨率就要小, 就好比把黄豆随机撒在球场上和撒在球门里的区别. 如子图 4 (b) 所示若 11 号区域未找到极值点那么在子图 4 (c) 中就会丢失很多极值点. 因此 M 的选择和级数的选择应该根据实际情况而定.

此外, 多级分辨率策略下不仅在同级区域内横向可并行执行, 不同级区域也可纵向并行执行, 计算过程不会相互干涉, 在保持原 QHOA 算法低耗时、快收敛、高精度的基础上该算法有很好的多峰寻优能力.

4　算法实例分析

本节通过寻优精确性、全极值点寻优和全局多峰优化三个角度来测试该算法的综合性能, 实验使用的计算机采用 Mac 系统, 主频 29 GHz, 在 JDK7 环境下的浮点计算精度为小数点后 16 位, 实验寻找的极值点默认为极小值点, 设置位置精度 A_l 为 0.000001.

4.1　寻优准确性

准确性是衡量一个算法好坏的重要标准, 为了测试本算法在需求波长分辨率下寻找极值点的准确度, 实验使用双势阱量子系统函数来验证. 双势阱量子系统是研究微观物理学的一个重要物理模型, 其函数表达式为

$$V_{sys}=V_0\frac{(x^2-a^2)^2}{a^4}+\delta x \tag{5}$$

该函数有两个极小值, 其中 V_0 表示函数的值域长度, $2|a|$ 表示两个极小值点之间的水平距离, $2|\delta||a|$ 表示两个极小值点之间的垂直距离. 置 $V_0=10$, a 分别取 $1,0.5,0.1,0.05,\cdots,0.0000001$, 对应函数实际分辨率 $\lambda_{\min}$ 为 $2,1,0.5,0.1,\cdots,0.0000005$. 设置函数定义域为 $[-1.5,1.5]$, 初始采样中心点数 $K=30$, 固定分辨率下可识别波长分辨率为 $\lceil\frac{1.5-(-1.5)}{30}\rceil=0.1$, 实验设置需求波长分辨率 $\lambda_0=0.1$, 测试算法在实际波长分辨率 $\lambda_{\min}$ 取不同值时识别极值的准确度. 此外, 为了测试极值点间的垂直距离对寻找极值点准确度的影响, 实验分别取 $\delta=1$ 与 0, 前者对应不等值极值双势阱函数, 后者对应等值极值双势阱函数. 经过 1 000 次试验, 统计寻找极值点的次数, 不等值极值点函数统计结果如图 5 所示, 等值极值点函数统计结果如图 6 所示. 其中横坐标为 a, 三条曲线分别代表找到第一个极值点的次数、找到第二个极值点的次数和全部找到极值点次数.

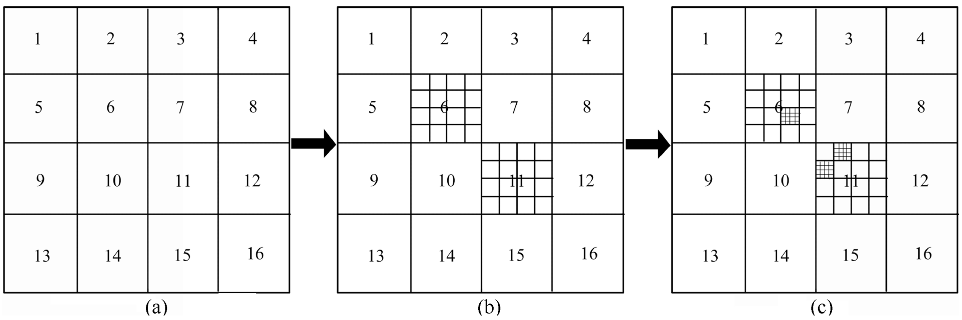


图 4　3 级分辨率策略下对 $f(x,y)$ 求极值

Fig. 4　3-level resolution strategy for the extreme value of $f(x,y)$

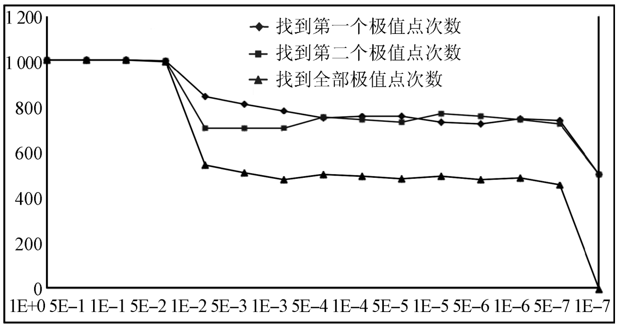


图 5 不等值极值点函数搜索准确性随 a 变化图

Fig. 5 Searching accuracy of unequal-value-extremum function varing with a

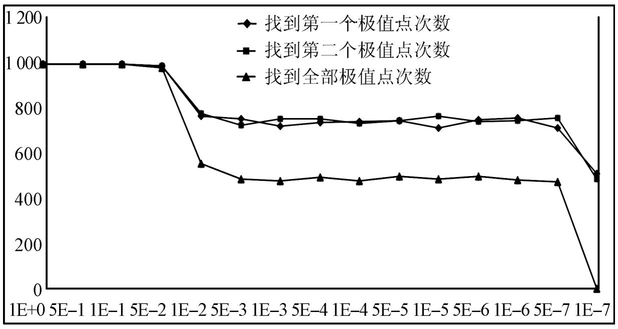


图 6 等值极值点函数搜索准确性随 a 变化图

Fig. 6 Searching accuracy of equal-value-extremum function varing with a

从两张图可以发现本算法可以找出需求分辨率下几乎所有极值. 令 $\lambda_0 = 2|a|$, 解出 $a_0 = 0.05$; 令 $A_l = 2|a|$, 解出 $a_1 = 0.0000005$. a_0 与 a_1 正好是两张图曲线趋势变化的分界点. 当 $a < a_0$ 时均能找到两个极值点; 当 $a = a_0$ 时极大概率找到两个极值点; 当 $a_0 < a \leq a_1$ 时, 找到两个极值点的概率趋于稳定; 当 $a > a_1$ 时找到两个极值点的概率逐渐减少. 实验结果与前面图 2 说明相符合, 证明了需求波长分辨率下寻优准确性.

此外可以发现:

1) 不管对于等极值函数和不等极值函数, 该算法在定义域内均能至少找到一个极值点, 并且随着 $\lambda_{\min}$ 减小, 三条曲线变化趋势几乎一致.

2) 图 4 中 a 取 0.05 与 0.0005 之间时算法更容易找到极值更小的第一个极值点, 也就是说对于不等值极值点, 该算法容易找到极值更明显的点, 这个特性使得该算法适合用于全局优化.

4.2 全极值点寻优

4.2.1 全极值点寻优性能

函数的全部极值点往往可以反映函数整体变化趋势, 大部分算法对全极值点的搜索消耗了很多时间, 而且精度不高, 例如文献 [7] 中的 Ant 算法与文献 [8] 中 RePAMO 算法全极值点的搜寻成为多峰优化问题上的一个难题, 这里采用两种策略的 P-MQHOA 作全极值寻优.

实验使用简单一维震荡函数 $y = x\sin\frac{1}{x}, x \in [-0.5, 0.5]$, 如图 7. 该函数在原点附近有无穷极值, 越靠近原点极值点分布越紧密, 直至波长分辨率趋近于 0. 故不可能找出理论上的所有极值点, 实验指定需求波长分辨率, 然后找出该波长分辨率下的极值点数.

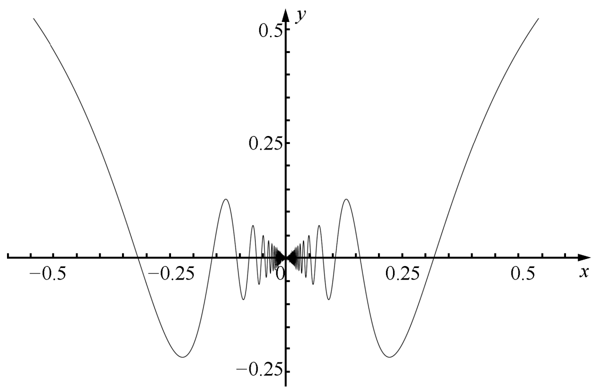


图 7 一维震荡函数曲线

Fig. 7 One-dimensional shock function curve

指定需求分辨率下的理论极值点数目可以这样求. 对 y 求导, $y' = -\frac{1}{x}\cos\frac{1}{x} + \sin\frac{1}{x}$, 令 $y' = 0$, 有 $\tan\frac{1}{x} = \frac{1}{x}$, 当 $\frac{1}{x} \in [-0.5, 0.5]$ 时可以得到近似解, 其中正近似解 $x_+ = \frac{2}{(2k+1)\pi}, k \in N^*$. 相邻两个波谷水平距离 $\lambda = \frac{2}{(2k+1)\pi} - \frac{2}{(2k+5)\pi}$, 在 $\lambda_0 < \lambda$ 条件下可解得

$$k < \sqrt{1 + \frac{2}{\lambda_0 \pi}} - 1.5, k \in N^* \tag{6}$$

这是正近似解数, 由于函数对称性, 负近似解数等于正近似解数, 所以满足式 (6) 的最大正整数的两倍记作 K_0, 即为满足需求分辨率的极值点个数.

为验证该算法对多维函数的多极值寻优性能分析, 实验使用 Griewank 函数, 函数表达形式: $f(x_i) = \sum\limits_{i=1}^{D} \frac{x_i^2}{4000} - \prod\limits_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$ 这里选择维度为 2, 如图 8 所示, x_i 在 [−100, 100] 内, 该函数有固定等分辨率的极值点数记为 K_0.

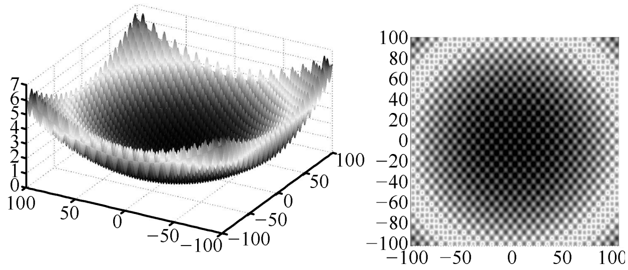


图 8　二维 Griewank 函数

Fig. 8　Two-dimensional Griewank function

实验设置不同需求波长分辨率 λ_0, 进行 50 次独立实验. 对比固定分辨率和二级分辨率下算法的性能, 前者找到的极值点个数用 K_1 表示, 后者用 K_2 表示, 平均耗时 (Time consuming) 分别为 TC_1、TC_2, 单位为秒, 采样中心点数 (Center-point numbers) 分别用 CN_1、CN_2 表示. 震荡函数结果见表 1.

表 1　一维震荡函数两种策略寻优性能比较

Table 1　Comparison of two strategies for one dimensional shock function

K_0	λ_0	K_1	CN_1	TC_1	K_2	CN_2	TC_2
2	0.1	3.9	10	0.0030	4.0	4	0.0038
12	0.01	13.9	100	0.0180	11.9	10	0.0084
46	0.001	47.7	1 000	0.1499	41.6	31	0.0348
156	0.0001	157.0	10 000	1.312	144.2	100	0.167
500	0.00001	493.2	100 000	10.374	461.0	316	0.767
1 592	0.000001	1 515.0	1 000 000	95.614	1 419.6	1 000	3.067

K_0 用来表示满足需求波长分辨率 λ_0 的极值点数, 实际上算法可以在原点附近找到两个比需求波长分辨率还小的极值点, 如 $\lambda_0 = 0.1$ 时算法还可以找到分别在 $(-0.1, 0)$ 与 $(0, 0.1)$ 域内的两个极值点, 但这两个极值点距离最近极值点的坐标距离远小于 λ_0, 因此找到满足 λ_0 的极值点数实际为 $K_1 - 2$ 和 $K_2 - 2$. 比较 K_0 与 $K_1 - 2$ 和 $K_2 - 2$ 的关系可知: 随着 λ_0 减小, 算法可以找到更多的极值点, 而且相比多级分辨率策略固定分辨率策略下找到满足 λ_0 的极值点数更接近 K_0, 但多级分辨率策略下算法花费时间较短. 如取 λ_0 为 0.000001 时固定分辨率与二级分辨率时间消耗相差约三十倍, 其原因是单个采样中心点迭代次数恒定的前提下两种策略算法所用的采样中心点数相差很大, 采用固定分辨率策略共需要 $CN_1 = \lceil(\frac{L}{\lambda_0})^D\rceil = \lceil(\frac{(0.5-(-0.5))}{0.000001})^1\rceil = 1\,000\,000$ 个采样中心点, 而采用二级分辨率策略初始只需 $CN_2 = \lceil(\frac{L}{\lambda_0})^{D/N}\rceil = \lceil(\frac{(0.5-(-0.5))}{0.000001})^{1/2}\rceil = 1\,000$ 个采样中心点, 初始平均划分 1 000 个区域, 这 1 000 个区域中存在很多无极值的区域, 在第二级划分时不会对这些无极值区域划分, 这样总的采样中心点相对来说就变很少了. 同样把函数换成二维 Griewank 函数进行 50 次独立实验, 对结果取平均值, 见表 2 所示.

表 2　二维 Griewank 函数两种策略寻优性能比较

Table 2　Comparison of two strategies for two dimensional Griewank function

K_0	λ_0	K_1	CN_1	TC_1	K_2	CN_2	TC_2
1 369	20	75.7	10	0.051	40.3	4	0.036
1 369	10	307.8	400	0.183	231.2	10	0.291
1 369	5	1 049.2	1 600	0.726	617.5	31	1.472
1 369	2	1 333.4	10 000	4.078	670.2	100	1.657
1 369	1	1 340.5	40 000	15.312	783.0	316	9.64
1 369	0.5	1 353.10	160 000	57.272	1 924.1	1 000	21.04

随着 λ_0 减小算法找到的极值点变多, 到 $\lambda_0 < 5$ 后两种策略找到的极值点数均开始变得稳定. 这是由于 Griewank 函数极值点坐标均匀分布在各维度上, 各个相邻极值点在各维度上的坐标差均相等, 等于实际波长分辨率, 当需求分辨率小于实际波长分辨率后算法找到的极值点数就趋于稳定了. 此外, 采用固定分辨率策略划分的 MQHOA 几乎能够找到需求波长分辨率下所有极值点, 而二级分辨率策略下查全率较低. 如上表所示, 二维 Griewank 函数在 $\lambda_0 = 2$ 时采用固定分辨率策略几乎能够找到所有极值点, 而采用二级分辨率策略下只能找到实际数目的一半, 这是因为在粗分辨率下忽略的极值点所在区域不会参与到第二级的划分. 但整体来说多级分辨率策略可以用较短的时间完成算法搜索, 尤其当需求波长分辨率很小的情况下这种时间差别更明显.

与其他智能算法比较, 本算法在时间和精度上都有很大优势. 如文献 [5] 中利用蚁群算法对函数 $y = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2)$, $x_1, x_2 \in [-1, 1]$ 求所有极大值点. 与利用 P-MQHOA 算法得出的结果比较如表 3 所示. 对 36 个极值点按照各维坐标进行基数排序, No. 为找到极值的序号, $((x_1, x_2, f(x_1, x_2))$ 分别为坐标点和对应的值.

以上实验数据是在固定分辨率策略下设置中心点数为 200, 可以算出可识别波长分辨率为 0.142, 小于函数实际的波长分辨率, 因此可找到所有极值点. 另外, 相对于蚁群算法求出所有极值点花费的 3 203.297 s, P-MQHOA 只需 0.123 s, 并且其精度相对于蚁群算法的 0.01 精确到 0.0000005, 无论是时间还是精度上 P-MQHOA 都占很大优势. 实际上, 这里每个中心点只生成 20 个高斯采样点也能找出所有极值点, 时间花费可以更少, 因此 P-MQHOA 性能上还有进一步提高的空间.

综上讨论, P-MQHOA 适合函数的全极值点寻优, 并且可以根据工程具体需求采用不同的策略: 当

表 3 P-MQHOA 与蚁群算法全极值搜索比较

Table 3 Comparison of total-extremum-searching of P-MQHOA and ant colony algorithm

NO.	P-MQHOA	ANT
	$(x_1, x_2, f(x_1, x_2))$	
1	(−0.878093608, −0.878093599, 3.53255484)	(−0.8781, −0.8781, 3.5326)
2	(−0.878093550, −0.526852857, 3.042136418)	−0.8737, −0.5310, 3.0362
3	(−0.878093617, −0.17561706, 2.796928371)	(−0.8725, −0.1810, 2.7873)
4	(−0.878093568, 0.175617049, 2.796928371)	(−0.8725, −0.5310, 3.0362)
5	(−0.878093628, 0.526852825, 3.042136418)	(−0.8700, 0.5175, 3.0176)
6	(−0.878093611, 0.878093569, 3.53255484)	(−0.8675, 0.8675, 3.4966)
7	(−0.52685281, −0.878093573, 3.042136418)	(−0.5310, −0.8737, 3.0362)
8	(−0.526852791, −0.526852812, 2.551717996)	(−0.5268, −0.5268, 2.5517)
9	(−0.526852794, −0.175617069, 2.306509949)	(−0.5352, −0.1773, 2.3056)
10	(−0.526852803, 0.17561705, 2.306509949)	(−0.5215, 0.1700, 2.2969)
11	(−0.526852866, 0.526852858, 2.551717996)	(−0.5200, 0.5200, 2.5367)
12	(−0.526852827, 0.878093628, 3.042136418)	(−0.5175, 0.8700, 3.0176)
13	(−0.175617027, −0.878093592, 2.796928371)	(−0.1810, −0.8725, 2.7873)
14	(−0.175617075, −0.526852805, 2.306509949)	(−0.1773, −0.5252, 2.3056)
15	(−0.175617028, −0.175617046, 2.061301903)	(−0.1756, −0.1756, 2.0613)
16	(−0.175617006, 0.175617028, 2.061301903)	(−0.1756, 0.1756, 2.0559)
17	(−0.175617067, 0.526852847, 2.306509949)	(−0.1700, 0.5215, 2.2969)
18	(−0.175617056, 0.878093620, 2.796928371)	(−0.1675, 0.8700, 2.7759)
19	(0.175617041, −0.878093592, 2.796928371)	(0.1675, −0.8700, 2.7759)
20	(0.175617068, −0.526852815, 2.306509949)	(0.1700, −0.5215, 2.2969)
21	(0.175617061, −0.175617041, 2.061301903)	(0.1715, −0.1715, 2.0559)
22	(0.175617051, 0.175617057, 2.061301903)	(0.1756, 0.1756, 2.0613)
23	(0.175617054, 0.526852783, 2.306509949)	(0.1773, 0.5252, 2.3056)
24	(0.175617039, 0.878093590, 2.796928371)	(0.1810, 0.8725, 2.7873)
25	(0.526852762, −0.878093542, 3.042136418)	(0.5175, −0.8700, 3.0176)
26	(0.526852876, −0.526852885, 2.551717996)	(0.5200, −0.5200, 2.5367)
27	(0.526852785, −0.175617064, 2.306509949)	(0.5215, −0.1700, 2.2969)
28	(0.526852823, 0.175617064, 2.306509949)	(0.5252, 0.1773, 2.3056)
29	(0.52685278, 0.52685282, 2.551717996)	(0.5268, 0.5268, 2.5517)
30	(0.526852781, 0.878093598, 3.042136418)	(0.5310, 0.8737, 3.0362)
31	(0.878093552, −0.878093598, 3.53255484)	(0.8675, −0.8675, 3.4966)
32	(0.87809359, −0.526852814, 3.042136418)	(0.8700, −0.5175, 3.0176)
33	(0.878093528, −0.175617038, 2.796928371)	(0.8700, −0.1675, 2.7759)
34	(0.878093604, 0.175617041, 2.796928371)	(0.8725, 0.1810, 2.7873)
35	(0.878093617, 0.526852807, 3.042136418)	(0.8737, 0.5310, 3.0362)
36	(0.878093597, 0.878093581, 3.53255484)	(0.8781, 0.8781, 3.5326)

需要查找所有极值点时可以采用固定分辨率策略, 当对时间要求较高时可以采用多级分辨率策略. 实际上实验还发现这两种策略所找到的极值都包含全局最优解, 因此在不考虑求所有极值的情况下, 对于全局优化问题可以考虑用多分辨率策略.

4.3　全局多峰优化

在控制与决策领域中, 例如无人机的紧密编队飞行控制应用、航空发动机的燃烧主动控制应用、导弹控制参数的自动调节应用等[15], 系统的代价函数普遍地存在多个全局极值, 如何能够使系统的代价函数稳定地工作在全局极值点, 最大地发挥系统的性能, 是工业控制界一个亟待解决的问题. 大部分多峰优化智能算法致力于解决这个问题, 而基于划分的 MQHOA 是适合全极值点寻优的算法, 可以从众多极值点中筛选出最佳极值点.

本节实验选用 14 个函数[6] 作为测试集合, 其中 $f_1 \sim f_7$ 和 f_{12} 为一维函数, f_{14} 为三维函数, 其余都是二维函数. 比较 P-MQHOA 与文献 [6] 中其他智能算法的的区别. 这些算法包括: 基于双目标差分进化的多峰优化算法 (Multimodal optimization using a biobjective differential evolution algorithm, MOBi-DE)、拥挤度差分进化算法 (Crowding DE, CDE)、基于形态的差分进化算法 (Speciation-based DE, SDE)、欧氏适应度距离半径的粒子群算法 (Fitness-Euclidean distance ratio PSO, FER-PSO)、基于物种形成原理的粒子群算法 (Speciation-based PSO, SPSO)、协方差矩阵适应进化算法 (Covariance matrix adaptation evolution strategy, CMA-ES)、固定小生境半径的协方差矩阵适应进化算法 (CMA-ES with fixed niche radius, CMA)、双目标多群体遗传算法 (Biobjective multipopulation genetic algorithm, BMPGA)、基于小生境的带有精英策略的非支配排序的遗传算法 (Niching-based NSGA-II) 和环型拓扑粒子群优化算法 (Particle swarm optimization using a ring topology, rpso). 其中 r2pso 中每个粒子只与环形拓扑中右边的粒子交互, r3pso 与环形拓扑中的左右两边的粒子均交互. 如表 4 所示, 所有算法都经过 50 次独立重复实验, 统计找到的全局峰数, 对结果取平均值, 然后对每个算法进行排名记录到括号中去, 最终统计总排名. 其中 ε 用来控制极值点的精度, No. 表示对应函数应有全局峰数. P-MQHOA 算法对一维函数采用固定分辨率策略, 二维函数采用二级分辨率策略, 三维函数采用三级分辨率策略.

参与比较的算法主要是对差分进化算法 (DE)、遗传算法 (GA) 和粒子群优化算法 (PSO) 的各种改进算法, 属于进化算法的分枝, 也都是基于群体智能的随机并行优化算法, 通过模仿生物群体内个体间的合作与竞争产生的启发式群体智能来指导优化搜

表 4　二维 Griewank 函数极值分布拟合结果

Table 4　Fitting results of extremum values′ distribution of Griewank function

Func.	No.	ε	P-MQHOA	MOBi-DE	CDE	SDE	S-CMA	CMA	SPSO	PER-PSO	r2pso	r3pso	Niching-based NSGA-II	BNPGA
f_1	1	0.05	1(4)	1(4)	1(4)	1(4)	1(4)	1(4)	0.44(12)	0.82(10)	0.76(11)	0.84(9)	1(4)	1(4)
f_2	1	0.05	1(5)	1(5)	1(5)	1(5)	1(5)	1(5)	0.40(12)	1(5)	0.88(11)	0.96(10)	1(5)	1(5)
f_3	2	1E−6	2(2.5)	2(2.5)	2(2.5)	1.96(5)	1.92(7)	1.95(6)	1.38(9)	0.8(10)	0.48(12)	0.6(11)	2(2.5)	1.5(8)
f_4	5	1E−6	5(1.5)	5(1.5)	3.84(10)	4.70(6)	0.04(12)	0.6(11)	4.88(3)	4.84(4)	4.68(7)	4.74(5)	4.37(9)	4.64(8)
f_5	1	1E−6	1(6)	1(6)	0.72(12)	1(6)	1(6)	1(6)	1(6)	1(6)	1(6)	1(6)	1(6)	1(6)
f_6	5	1E−6	5(1.5)	5(1.5)	3.96(9)	4.6(7)	0(12)	0.64(11)	4.92(4)	4.96(3)	4.88(5)	4.72(6)	4.52(8)	3.56(10)
f_7	1	1E−6	1(5.5)	1(5.5)	0.8(11)	1(5.5)	1(5.5)	1(5.5)	1(5.5)	1(5.5)	1(5.5)	1(5.5)	0.6(12)	1(5.5)
f_8	4	5E−4	4(1.5)	4(1.5)	0.32(12)	3.72(4.5)	3.72(4.5)	3.43(7)	0.84(11)	3.68(6)	2.92(9)	2.76(10)	3.74(3)	3.34(8)
f_9	2	1E−6	2(3)	2(3)	0.04(12)	2(3)	1.6(7)	2(3)	0.08(11)	1.96(6)	1.44(9)	1.56(8)	2(3)	1.24(10)
f_{10}	1	1E−6	1(2)	1(2)	0.52(9)	0.32(10)	0.06(12)	0.18(11)	0.56(8)	1(2)	0.88(6)	0.76(7)	0.94(5)	0.96(4)
f_{11}	18	5E−2	17.5(1)	17.4(2)	11.39(12)	12.78(9)	12.04(10)	12(11)	14.36(7)	15.61(6)	15.95(5)	16.45(4)	16.94(3)	13.72(8)
f_{12}	6	0.05	6(1.5)	6(1.5)	5.56(6.5)	4.88(12)	5.56(6.5)	5.81(3)	5.6(5)	5.28(10)	5.52(8)	5.16(11)	5.36(9)	5.71(4)
f_{13}	36	1E−3	35.6(1)	35.4(2)	33.8(3)	23.8(7)	24.6(6)	23.6(8)	25.72(5)	21.8(12)	22.4(9)	22.2(10)	22.05(11)	31.76(4)
f_{14}	218	1E−3	163.56(2)	175.88(1)	152(3)	50.6(8)	0.6(12)	32.5(11)	70.12(6)	68.6(7)	40.6(10)	45.4(9)	92.76(5)	121.54(4)
Total ranks			38	39	111	92	109.5	102.5	104.5	92.5	113.5	111.5	85.5	88.5

表 5 P-MQHOA 算法与 MOBi-DE 算法时间消耗 (s)

Table 5 Time consumption of P-MQHOA algorithm and MOBi-DE algorithm (s)

Func.	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}
P-MQHOA	0.01	0.01	0.01	0.01	0.01	0.01	0.02	0.1	0.1	0.1	1.76	0.28	4.32	12.14
MOBi-DE	1.56	1.46	2.82	2.64	1.54	2.04	1.68	18.34	15.28	41.24	46.28	36.49	51.74	72.36

索. 这些算法使用小生境来提高种群多样性避免早熟, 可以同时使用多个子种群来定位和跟踪多个最优解. 小生境的半径和小生境的数量影响整体的运算速度和精度, 算法性能对该参数很敏感, 需要调试多次, 其中大部分改进算法需要较多控制参数才能达到比较满意的结果, 而 P-MQHOA 只需要设置与需求波长分辨率相对应的采样中心点数即可. 根据表 4 统计的排名数据绘制热力图, 如图 9 所示, 其中同一行颜色越淡表示找到的全局最优点越多, 反之越少. 比如对于 f_1 函数, 由颜色深浅可见 SPSO 算法的寻优效果最差, 其他 PSO 算法表现也不佳, 而 P-MQHOA 与其他遗传算法均能找到对应的全局最优点.

从上图可看出, P-MQHOA 与 MOBi-DE 对 14 个函数测试下所找到的最优解数目差不多, 而其他算法相比之下较为逊色. 单独比较两种算法对这 14 个函数测试所花费的平均时间, 如表 5 所示, 时间单位为秒.

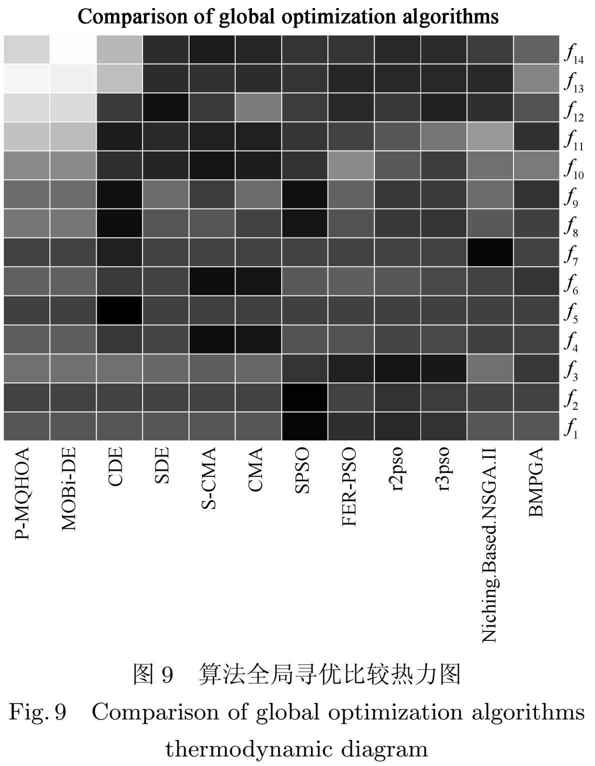


图 9 算法全局寻优比较热力图

Fig. 9 Comparison of global optimization algorithms thermodynamic diagram

可见采取固定分辨率策略的 P-MQHOA 的一维函数全局寻优时间消耗比 MOBi-DE 算法低两个数量级, 采取多级分辨率策略的 P-MQHOA 的多维函数全局寻优时间消耗比 MOBi-DE 算法也低很多. 这是因为 P-MQHOA 局部收敛速度要比 MOBi-DE 等大部分智能算法要快的多. 例如 P-MQHOA 对 f_{14} 全局寻优过程中, 能够在 13 秒内找到约 30 000 个极值点, 然后筛选出全局极值点.

通过以上实验对比进行分析, 归纳如下:

GA 采用二进制编码, 需要进行交叉变异选择遗传操作, 比较繁杂, 并且以前的知识随着种群的改变被改变, 没有记忆性, 有较多敏感的控制参数, 收敛速度慢; PSO 一般采用实数编码, 粒子通过内部速度变化进行更新, 没有 GA 复杂的遗传操作, 此外通过当前搜索得到最优点进行信息共享, 较优解被所有粒子保存, 具有记忆能力, 收敛速度相对 GA 较快但很容易陷入局部最优解, 算法不稳定; DE 采用实数编码、基于差分的简单变异操作和一对一的竞争生存策略, 降低了遗传操作的复杂性, 其独特的记忆能力使其可以动态跟踪当前的搜索情况以调整其搜索策略, 控制参数较少, 收敛速度相对于 GA 与 PSO 都要快. P-MQHOA 无需编码, 拥有固定分辨率与多级分辨率两种策略, 每个区域的搜索相互独立, 每个采样中心点具有对所在区域极值的记忆能力, 操作简单, 控制参数较少, 收敛速度最快.

5 结论

为解决多峰优化问题, 本文改进了 MQHOA, 提出 P-MQHOA 算法, 把 MQHOA 的全局优化特性分散在多个均匀划分区域, 可以快速在各个区域找到极值点, 从而解决多峰优化问题. 当实际问题除了最优解还需要备择解时, 该算法为全极值点优化中提供次优解提供了条件. 此外针对具体问题本文提出了固定分辨率和多级分辨率两种策略, 前者有很好的全极值搜索能力, 后者有较强的实时性, 两种策略都能用于解决全局多峰优化问题. 后面的研究工作将集中在该算法在多级分辨率策略下的全极值寻优性能研究, 提高其处理高维问题的能力, 扩展该算法在工业上的应用.

References

1 Chang Y W, Yu G F. Multi-Sub-Swarm PSO algorithm for multimodal function optimization. In: Proceedings of the 2014 International Conference on Computer Science and Information Technology. India: Springer, 2014. 687−695

2 Qu B Y, Suganthan P N, Das S. A distance-based locally informed particle swarm model for multimodal optimization. *IEEE Transactions on Evolutionary Computation*, 2013, **17**(3): 387−402

3 Shih C J, Teng T L, Chen S K. A niche-related particle swarm meta-heuristic algorithm for multimodal optimization. In: Proceedings of the 2nd International Conference on Intelligent Technologies and Engineering Systems. Switzerland, Germany: Springer, 2014. 313−321

4 Agrawal S, Sanjay S. FRPSO: fletcher-reeves based particle swarm optimization for multimodal function optimization. *Soft Computing*, 2014, **18**(11): 2227−2243

5 Qu B Y, Suganthan P N, Liang J J. Differential evolution with neighborhood mutation for multimodal optimization. *IEEE Transactions on Evolutionary Computation*, 2012, **16**(5): 601−614

6 Basak A, Das S, Tan K C. Multimodal optimization using a biobjective differential evolution algorithm enhanced with mean distance-based selection. *IEEE Transactions on Evolutionary Computation*, 2013, **17**(5): 666−685

7 Pang C Y, Li X, Liu H, Wang Y F, Hu B Q. Applying ant colony optimization to search all extreme points of function. In: Proceedings of the 5th IEEE Conference on Industrial Electronics and Applications. Taichung, China: IEEE, 2010. 1517−1521

8 Dasgupta B, Divya K, Mehta V K, Deb K. RePAMO: recursive perturbation approach for multimodal optimization. *Engineering Optimization*, 2013, **45**(9): 1073−1090

9 Vidya R C, Phaneendra H D, Shivakumar M S. Quantum algorithms and hard problems. In: Proceedings of 5th IEEE International Conference on Cognitive Informatics. Washington, DC, USA: IEEE, 2006. 783−787

10 Wang Peng. *The Key Technology and Application Examples of Cloud Computing*. Beijing: People′s Posts and Telecommunications Press, 2010. 168−194
(王鹏. 云计算的关键技术与应用实例. 北京: 人民邮电出版社, 2010. 168−194)

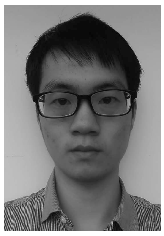
11 Wang Peng, Huang Yan, Ren Chao, Guo You-Ming. Multi-Scale quantum harmonic oscillator for high-dimensional function global optimization algorithm. *Acta Electronica Sinica*, 2013, **41**(12): 2468−2473
(王鹏, 黄焱, 任超, 郭又铭. 多尺度量子谐振子高维函数全局优化算法. 电子学报, 2013, **41**(12): 2468−2473)

12 Wang Peng, Huang Yan. Physical model of multi-scale quantum harmonic oscillator optimization algorithm. *Journal of Frontiers of Computer Science and Technology*, 2015, **9**(10): 1271−1280
(王鹏, 黄焱. 多尺度量子谐振子优化算法物理模型. 计算机科学与探索, 2015, **9**(10): 1271−1280)

13 Xiao Li-Bin, Wang Peng, Chen Lei, Guo You-Ming. Quantum harmonic oscillator optimization algorithm. *Journal of Computer Applications*, 2013, **32**(S2): 1−4, 44
(肖黎彬, 王鹏, 陈磊, 郭又铭. 量子谐振子优化算法. 计算机应用, 2013, **32**(S2): 1−4, 44)

14 Mallat S G. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1989, **11**(7): 674−693

15 Zuo Bin, Hu Yun-An, Shi Jian-Hong. Research and development of extremum seeking algorithm. *Journal of Naval Aeronautical Engineering Institute*, 2006, **21**(6): 611−617
(左斌, 胡云安, 施建洪. 极值搜索算法的研究与进展. 海军航空工程学院学报, 2006, **21**(6): 611−617)



陆志君 成都信息工程大学软件工程学院硕士研究生. 2013 年获得南京信息工程大学软件工程学院学士学位. 主要研究方向为分布式计算, 智能算法.
E-mail: bingjungg@126.com
(**LU Zhi-Jun** Master student at the College of Software, Chengdu University of Information Technology. He received his bachelor degree from the College of Software, Nanjing University of Information Technology in 2013. His research interest covers distributed computing and intelligent algorithm.)

安俊秀 成都信息工程大学软件工程学院教授. 2004 年获得西安交通大学工学硕士学位. 主要研究方向为社会计算, 智能搜索, 分布式计算.
E-mail: anjunxiu@cuit.edu.cn
(**AN Jun-Xiu** Professor at the College of Software, Chengdu University of Information Technology. She received her master degree from Xi′an Jiaotong University in 2004. Her research interest covers social computing, intelligent searching, and distributed computing.)

王 鹏 成都信息工程大学软件工程学院教授. 2004 年获得中国科学院成都计算机应用研究所博士学位. 主要研究方向为分布式计算, 智能算法. 本文通信作者. E-mail: wp002005@163.com
(**WANG Peng** Professor at the College of Software, Chengdu University of Information Technology. He received his Ph. D. degree from Chengdu Institute of Computer Application, Chinese Academy of Sciences in 2004. His research interest covers distributed computing and intelligent algorithm. Corresponding author of this paper.)