Journal of Computer Applications
计算机应用,2013,33(12):3380-3384
ISSN 1001-9081
CODEN JYIIDU
2013-12-01
http://www.joca.cn

文章编号:1001-9081(2013)12-3380-05
doi:10.11772/j.issn.1001-9081.2013.12.3380

# 基于多核机群的人工鱼群并行算法

李　双,李文敬*,孙环龙,林中明
(广西师范学院 计算机与信息工程学院,南宁 530023)
(*通信作者电子邮箱 liwj@gxtc.edu.cn)

**摘　要**:针对人工鱼群算法在复杂多峰函数优化问题上寻优精度低、后期搜索能力减弱且运行时间长等问题,提出一种基于多核机群的人工鱼群并行算法(PDN-AFS)。首先对人工鱼群算法的优势与不足进行分析,采用动态权衡因子策略并适时引入小生境机制,提出一种新的人工鱼群(DN-AFS)算法;然后根据多核机群的并行编程模型(MPI+OpenMP),对DN-AFS算法进行并行设计与分析,提出基于多核机群的人工鱼群并行算法;最后在多核机群环境下进行仿真实验。实验结果表明:该算法有效地提高了复杂多峰函数优化问题的收敛速度和寻优性能,并获得了较高的加速比。

**关键词**:人工鱼群算法;动态权衡因子;小生境;并行算法;MPI+OpenMP
**中图分类号**:TP301.6;TP18　　**文献标志码**:A

## Artificial fish swarm parallel algorithm based on multi-core cluster

LI Shuang, LI Wenjing*, SUN Huanlong, LIN Zhongming
(*College of Computer and Information Engineering, Guangxi Teachers Education University, Nanning Guangxi* 530023, *China*)

**Abstract:** Concerning the problems of low accuracy, limitations of stagnation and slow convergence speed in the later evolution process of Artificial Fish Swarm Algorithm (AFSA), a Parallel Dynamic weigh Niches Artificial Fish Swarm (PDN-AFS) algorithm based on multi-core cluster was proposed. Firstly, the advantages and disadvantages of AFSA were analyzed, and dynamic weighting factor strategy and niche mechanism were adopted, hence a new Dynamic weigh Niches Artificial Fish Swarm (DN-AFS) algorithm was put forward. Then parallel design and analysis of DN-AFS algorithm based on parallel programming model (MPI+OpenMP) were introduced. Finally, the simulation experiments on multi-core cluster environment were given. The experimental results show that PDN-AFS can effectively improve the convergence speed and optimization performance of the complex multimodal function optimization problem, and achieve high speed ratio.

**Key words:** Artificial Fish Swarm Algorithm (AFSA); dynamic weighting factor; niche; parallel algorithm; MPI+OpenMP

## 0　引言

函数优化问题是在工程、数学中普遍存在的一类优化问题,传统的优化方法对目标函数要求苛刻且对于复杂多峰函数的优化有效性低。人工鱼群算法(Artificial Fish Swarm Algorithm, AFSA)的提出,为多峰复杂函数的优化提供了一种有效手段。该算法在搜索空间的问题上有一定的自适应能力[1],并且算法对初值的设定要求不高,初值随机产生或设定成固定值都可以。但是AFSA的寻优精度不高,算法在运行后期搜索速度慢,盲目性较大。近年来许多学者对算法进行了不同方面的改进:文献[2]采用优胜劣汰策略筛选精英人工鱼群,以及对觅食行为进行改进,结合模式搜索法提高了算法搜索最优解的精度;文献[3]提出了一种改进的人工鱼群算法,通过引入变异算子策略和消亡操作对部分个体进行重新初始化或变异,具有较好的优化性能,但是算法的收敛时间还有待提高;文献[4]提出了一种新的小生境人工鱼群(Niche Artificial Fish Swarm, NAFS)算法,通过引入小生境排挤机制来维持种群的多样性,提高了算法的收敛精度,但是算法执行的随机性较强,使得算法后期搜索的稳定性较差;文献[5]提出一种基于图形处理器(Graphic Processing Unit, GPU)加速的并行人工鱼群算法,通过人工鱼的个体寻优进行并行化设计,提高了算法的运行速度,但是人工鱼算法的随机化程度很高,从而产生了影响负载均衡的多分支问题。目前的人工鱼群算法在获取问题的精确解以及运行时间上的能力相对较差,制约了人工鱼群算法的应用。为此,本文提出基于多核机群的人工鱼群并行算法,以解决大规模复杂多峰函数优化问题。

## 1　相关工作

### 1.1　人工鱼群算法基本思想

人工鱼个体状态表示为 $X=(x_1,x_2,\cdots,x_n)$,其中 $x_i(i=1,2,\cdots,n)$ 为预寻优变量;人工鱼个体之间的距离 $d_{ij}=\|x_i-x_j\|$;每次移动的步长用 $step$ 表示;$\delta$ 表示拥挤度因子;觅食行为中的最大重复尝试次数用 $Try\_number_{max}$ 表示。以求解多峰函数极大值问题为例描述鱼群行为如下:

1)觅食行为。

**收稿日期**:2013-07-19;**修回日期**:2013-08-21。　**基金项目**:国家自然科学基金资助项目(61163012);广西自然科学基金资助项目(2012GXNSFAA053218);广西高校科学技术研究项目(2013YB147)。

**作者简介**:李双(1988-),女,山东济南人,硕士研究生,主要研究方向:并行算法、人工智能;　李文敬(1964-),男(壮族),广西邕宁人,教授,主要研究方向:并行计算、Petri网;　孙环龙(1989-),男,江苏盐城人,硕士研究生,主要研究方向:神经网络;　林中明(1988-),男,广西贺州人,硕士研究生,主要研究方向:并行计算、云计算。

设人工鱼当前状态为 $x_i$,在其视野范围内($d_{ij} < visual$)随机选择一个状态 $x_j$,若食物浓度 $Y_j > Y_i$(若求解最小值,则相反),则朝 $x_i$ 方向移动一步;否则重新随机选择一个状态 $x_j$,若反复几次仍不满足前进条件,则在其视野范围内随机的移动一步。

2)聚群行为。

设人工鱼当前状态为 $x_i$,搜索其视野内($d_{ij} < visual$)的伙伴数目 $n_f$ 及中心位置 $x_c$,若 $Y_c/n_f > \delta Y_i$,说明伙伴中心有较多的食物且不太拥挤,则朝该位置移动一步;否则执行觅食行为。

3)追尾行为。

设人工鱼当前状态 $x_i$,搜索当前邻域($d_{ij} < visual$)内食物源浓度 $Y_j$ 最大的伙伴 $x_j$,若 $Y_j/n_f > \delta Y_i$,说明 $x_j$ 的状态有较高的食物浓度且其周围不太拥挤,则朝 $x_j$ 的方向前进一步;否则执行觅食行为。

4)行为选择。

根据所要解决问题的性质,人工鱼探索其当前所处的环境状况并进行评价,从而从 1)~3)中选择一个合适的行为执行。

5)公告板。

公告板记录人工鱼最优状态 $g_{best}$,各人工鱼每次寻优完成后就比较自身状态与公告板状态,若较优,则用自身状态取代公告板状态。

### 1.2　人工鱼群算法优势与不足

1)人工鱼视野与步长对算法收敛性的影响。

视野(*visual*)与步长(*step*)对算法的收敛性能有着较大的影响,较大的视野有利于人工鱼发现全局极值并收敛,较小的视野又可以使算法收敛到较好的结果。步长是 0 到 *step* 之间的随机值,这样虽然能够降低参数的敏感度,但是在算法运行前期可能产生的过小步长不利于算法早期的快速收敛,过大的步长又可能引起震荡现象的出现[6]。所以在视野与步长参数的设置上可以根据实际情况在开始时使用较大的视野与步长使算法快速收敛,然后逐渐减小视野与步长的值来提高算法的收敛精度。

2)种群多样性对算法收敛性的影响。

种群的多样性对 AFSA 寻优性能有明显影响,算法在运行前期,种群的多样性相对较强,收敛速度相对较快,但是随着算法的运行,大量鱼群聚集于局部最优点周围且不易跳出,造成种群多样性降低,从而导致求解数据精度变差。因此通过保持种群的多样性来提高算法的收敛性能是一种有效的手段。

### 1.3　小生境技术

小生境[7-9]原指在生物界中,相同种类的生物生活在一起,从而形成一个小的生活环境,不同种类的个体则分离开。科学研究中,小生境技术表现为:在一个固定规模的群体中形成许多子种群,每个子种群对应问题的子任务,每个子任务即是对应找出一个峰值。这样就避免大量数据聚集于局部最优点附近而造成种群多样性减少,因此可以考虑在算法中适时地引入基于共享机制的小生境技术来增加种群多样性。基于共享机制的选择策略如下:

1)计算个体之间的海明距离,生成小生境群体;

2)小生境内执行寻优运算;

3)利用共享机制更新个体的适应度值,对于适应度值低的个体施加罚函数来增加其淘汰率。

## 2　新的人工鱼群算法 DN-AFS

根据前面对人工鱼群算法收敛性能的分析,引入动态权衡因子策略,对人工鱼搜索过程中的步长和视野范围进行动态调整,在算法运行后期,引入小生境技术,设计了基于动态权衡与小生境技术的串行式人工鱼群算法 DN-AFS(Dynamic weigh Niches Artificial Fish Swarm),以提高算法的个体寻优速度以及增强整体的负载平衡能力。

### 2.1　引入动态权衡因子策略

对于多峰复杂函数的优化,在算法运行前期,采用较大的步长和视野,使人工鱼在较大的范围内粗略搜索,提高全局搜索能力和收敛速度;在算法运行后期,逐渐减小步长和视野范围,使算法定位在最优解附近进行局部精细搜索。步长 *step* 和视野 *visual* 按式(1)动态调整:

$$\begin{cases} step_{iter+1} = step_{iter} - step_{iter} \cdot \alpha \\ visual_{iter+1} = visual_{iter} - visual_{iter} \cdot \alpha \end{cases} ; \alpha = iter/I_{\max} \quad (1)$$

其中:$\alpha$ 为动态权衡因子,取值区间为(0,1),由于 *step* 是正数,这样的取值能保证在运行初期由于 $\alpha$ 很小,步长 *step* 和视野 *visual* 相应会较大,随着迭代次数变大 $\alpha$ 也逐渐变大,步长 *step* 和视野 *visual* 又逐渐减小,从而使视野随迭代次数的增加逐渐变小,减小了算法的强随机性,较好地平衡全局搜索能力和加强局部搜索能力;*iter* 为当前迭代次数;$I_{\max}$ 为最大迭代次数。

### 2.2　小生境技术引入的时机

根据种群多样性对人工鱼群算法收敛性能影响的分析可知,算法的早期收敛速度较快,因此直接引入小生境技术不合适。当后期鱼群出现聚集现象时,引入小生境机制增加种群的多样性,聚集程度用式(2)表示:

$$\frac{1}{(n-1)}\sum_{j=1}^{n} d_{ij} \leqslant \varepsilon \quad (2)$$

其中:$d_{ij}$ 是鱼群中心位置的鱼个体与其他鱼个体之间的距离;$n$ 为种群规模;$\varepsilon$ 为较小的常数阈值,它的取值根据具体的种群规模来确定,经多次验证,一般取[0, ln $n$] 较为合适。当中心位置的鱼个体与其他鱼个体之间的平均距离小于 $\varepsilon$ 时,说明大部分鱼群已经收敛到局部最优值或者全局最优值附近,此时引入小生境技术,生成新的人工鱼群体,利用前面提到的小生境技术的共享机制,使人工鱼群保留群体最优个体,直到满足终止条件。

### 2.3　DN-AFS 算法过程

DN-AFS 算法与 AFS 算法流程基本相同,改进部分分别在:1)计算人工鱼步长 *step* 和视野 *visual* 时引入了动态权衡因子策略减少搜索的强随机性;2)在鱼群搜索后期通过判断鱼群聚集程度引入小生境技术使鱼群跳出局部最优。具体步骤如下:

步骤 1　初始化人工鱼规模 $n$、视野 *visual*、步长 *step*、拥挤度因子 $\delta$、最大尝试重复次数 $Try\_number_{\max}$,最大迭代次数 $I_{\max}$ 等参数;

步骤 2　计算每条人工鱼的适应度值,并与公告板状态比较,若优于公告板状态,则将其值赋予公告板;

步骤 3　按式(1)计算人工鱼的步长 *step* 和视野 *visual*;

步骤 4　判断当前视野中适应度值最大的个体和中心位置个体的适应度值以及拥挤度情况,试探性执行追尾和聚群

行为,否则执行觅食行为;

步骤 5 按式(2)判断鱼群的聚集程度,若成立,引入小生境技术执行,否则执行下一步;

步骤 6 判断是否满足终止条件,若满足,输出最优值结果;否则转步骤 2 执行。

## 3 基于 MPI + OpenMP 的人工鱼群并行算法

### 3.1 人工鱼群并行算法的设计

在每一次迭代中,各个小鱼群在寻找最优解时是独立进行的,作为 MPI 进程分配到各个处理机上执行,具体到进程内,人工鱼个体的觅食、聚群、追尾行为也是独立的,并且主要集中在 for 循环中实现,用 OpenMP 生成线程并行执行,整个寻优过程中相互间的通信共享在于局部最优值与全局最优值之间,因此,提出基于动态权衡与小生境技术的人工鱼群并行算法(Parallel Dynamic weigh Niches AFS, PDN-AFS)。

### 3.2 PDN-AFS 算法具体实现

在集群机下,使用 MPI_INIT 初始化,将划分好的小鱼群作为一个进程分配到不同的处理机上利用 MPI 实现并行,进程间通过消息传递交换信息,使用 OpenMP 编译指导语句在 MPI 进程内创建线程组来完成小鱼群内人工鱼的寻优部分(主要是觅食、聚群、追尾行为),其中 MPI 进程中通过 omp-set_num-threads( )来确定 OpenMP 并行区域的线程数量;在结束 OpenMP 多线程并行区域之后,MPI 可以进行节点间的通信,实现局部最优解与全局最优解之间的比较与替换,当所有的任务完成以后,使用 MPI-FINALIZE 结束 MPI 进程。并行算法主要步骤如下:

步骤 1 初始化各项参数 $(n, visual, step, \delta, Try\_number_{max})$ 及迭代次数 $iter = 0$。

步骤 2 随机划分成处理机个数的小鱼群作为进程,并分配到不同的处理机上。

步骤 3 进程内计算每条人工鱼的适应度值,并与公告板状态比较,若较优,则将其值赋予公告板。

步骤 4 按式(1)计算人工鱼的步长 *step* 和视野 *visual*。

步骤 5 人工鱼当前状态为 $x_i$,在自己视野范围内随机游动寻找食物源。

步骤 6 进程内部,人工鱼的主要循环部分被随机分配到不同的线程上执行:

1)觅食行为。

(a)设人工鱼当前状态为 $x_i$,随机生成一个新状态 $x_j = \text{random}(n(x_i, visual))$。

(b)如果 $f(x_i) < f(x_j)$,则人工鱼状态修改为:$x_{inext} = x_i + \text{random}(step) * (x_j - x_i)/\| x_j - x_i \|$;判断是否达到最大尝试次数 $Try\_number_{max}$,若没达到返回(a)执行,否则执行下一步。

(c)随机游动前进一步:$x_i = x_i + \text{random}(step)$。

2)聚群行为。

(a)探索其视野范围,并生成其同伴的集合 $k = \{x_j \mid x_j - x_i \leqslant visual\}$。

(b)如果 $k \neq \varnothing$,取其中心位置:$x_c = \sum x_j / n_f (n_f \geqslant 1)$。

(c)如果 $n_f/n < \delta (0 < \delta < 1)$ 且 $f(x_i) < f(x_c)$,则人工鱼向中心位置 $x_c$ 前进一步生成新位置 $x_{inext}$:$x_{inext} = x_i + \text{random}(step) * (x_c - x_i)/\| x_c - x_i \|$,否则返回步骤 5 执行。

3)追尾行为。

(a)搜寻小鱼群内状态最优的人工鱼 $x_{max}$,如果没有返回步骤 5 执行,否则执行下一步;

(b)如果 $f(x_i) < f(x_{max})$,且 $x_{max}$ 满足 $n_f/n < \delta (0 < \delta < 1)$,则向 $x_{max}$ 位置前进一步:$x_i = x_i + \text{random}(step)(x_{max} - x_i)/\| x_{max} - x_i \|$。

步骤 7 判断是否达到最大迭代次数,若满足,退出 OpenMP 多线程并行区,MPI 进行节点间通信,对比得出各进程的最优解 $x_{best}$,与公告牌 $g_{best}$ 比较,如果 $x_{best} > g_{best}$,则赋值给公告牌;否则返回步骤 4。

步骤 8 结束 MPI 进程,输出最优解。

MPI + OpenMP 混合模型下 PDN-AFS 算法框架如图 1 所示。

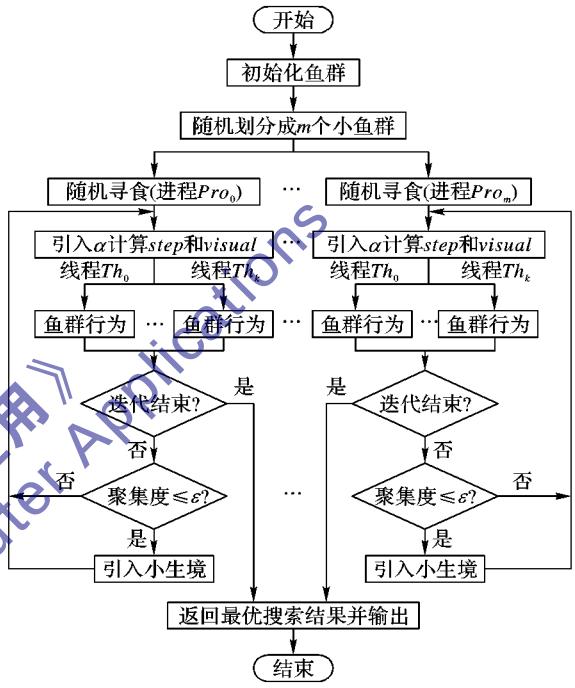


图 1 PDN-AFS 算法框架

## 4 仿真实验及分析

选取文献[3]提出的改进人工鱼群算法(下文称改进 AFS)和文献[4]提出的小生境人工鱼群算法(NAFS)与本文提出的基于动态权衡小生境串行人工鱼群算法(DN-AFS)和动态权衡小生境人工鱼群并行算法(PDN-AFS)进行性能对比分析。仿真实验在 4 台 CPU 为 AMDAthlon 64 位酷睿双核 PC,内存 4 GB;通信协议为 TCP/IP;操作系统为 Windows XP;编程语言为 C ++;IDE 为 Visual Studio2005(OpenMP);MPI 为 mpich2-1.2-win-ia32 的集群环境下进行。

### 4.1 测试函数及相关参数

表 1 为实验选取的 5 个典型基准函数及其取值范围和最优解。其中:$f_1$ 是一个变峰、多极值点函数,该函数峰值较密且函数值之间相差较大,算法中,取 $visual = 4$,$step = 1$;函数 $f_2$ 的局部极大值点有无数个,此函数最大值的周围有一圈脊(局部极大值点),值为 0.990 28,一般算法极易停滞在该处,取 $step = 0.5$,$visual = 6$;$f_3$ 的最佳点在端点(2.048, -2.048)取得,取 $visual = 6$,$step = 0.5$;$f_4$ 是 Benchmark 函数中的 Sphere 函数,该函数是连续的单峰函数,最小值为 0,取 $visual = 1$,$step = 0.05$;$f_5$ 是 Benchmark 函数中的 Griewank 函数,该函数是复杂的非线性多峰值函数,最小值为 0,取 $visual = 10$,$step = 0.8$。这 5 个函数可以有效测试算法的全局寻优能力以及对最优点位置的逼近能力。仿真实验中,本文算法

采用的其他运算参数为:$n = 30, Try\_number_{max} = 10, \delta = 0.6$,　　最大迭代次数设置为 1 000,函数维数为 50。

**表 1　测试函数基本表**

| 函数 | 公式 | 变量范围 | 最优解 |
|---|---|---|---|
| $f_1$ | $x_1\sin(4\pi x_1) - x_2\sin(4\pi x_2 + \pi + 1)$ | [-1,2] | 3.3099 |
| $f_2$ | $0.5 - (\sin^2 x_1^2 + x_2^2)0.5/([1 + 0.001(x_1^2 + x_2^2)]^2)$ | [-4,4] | 1 |
| $f_3$ | $100(x_1^2 - x_2)^2 + (1 - x_1)^2$ | [-2.048,2.048] | 3905.9262 |
| $f_4$ | $\sum_{i=1}^{n} x_i^2$ | [-100,100] | 0 |
| $f_5$ | $\frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$ | [-600,600] | 0 |

### 4.2　实验结果及分析

实验结果采用每个函数独立运行 30 次的平均值。

1)固定迭代次数为 1 000,算法的运算结果、寻优精度对比如表 2 所示。对于复杂函数 $f_1$、$f_3$,DN-AFS 算法虽然表现出很好的寻优能力,但是它的收敛时间不太理想,这是因为改进后的算法在运行时存在小生境的重新生成以及人工鱼适应度值的更新等问题,PDN-AFS 算法的寻优精度与文献[3]的改进 AFS 算法相当;对于函数 $f_2$,一般算法在 0.990 28 附近极易陷入该局部极值点,而无论改进后的串行还是并行算法都具有较好的突破局部最优的能力,且 PDN-AFS 的寻优精度明显优于对比的三种算法;对于单峰函数 $f_4$,DN-AFS 算法具有较好的全局搜索能力,而 PDN-AFS 算法并没有体现出它的优势;对于非线性多峰函数 $f_5$,PDN-AFS 算法能快速搜索到最优解附近,搜索精度明显高于 NAFS 算法。

**表 2　测试函数实验结果对照表**

| 函数 | 算法 | 平均值 | 最优值 | 最小值 |
|---|---|---|---|---|
| $f_1$ | 改进 AFS | 3.2990 | 3.3094 | 3.2733 |
| | NAFS | 3.3098 | 3.3099 | 3.3096 |
| | DN-AFS | 3.3099 | 3.3099 | 3.3096 |
| | PDN-AFS | 3.3098 | 3.3099 | 3.3099 |
| $f_2$ | 改进 AFS | 0.9967 | 0.9999 | 0.9901 |
| | NAFS | 0.9998 | 1.0000 | 0.9994 |
| | DN-AFS | 0.9999 | 1.0000 | 0.9996 |
| | PDN-AFS | 0.9999 | 1.0000 | 0.9998 |
| $f_3$ | 改进 AFS | 2997.0673 | 3905.9262 | 3874.9260 |
| | NAFS | 3905.9262 | 3905.9262 | 3905.9262 |
| | DN-AFS | 3905.9260 | 3905.9262 | 3905.9260 |
| | PDN-AFS | 3905.9262 | 3905.9262 | 3905.9262 |
| $f_4$ | 改进 AFS | 4.55E-36 | 4.93E-54 | 5.93E-54 |
| | NAFS | 4.10E-158 | 5.73E-169 | 5.73E-169 |
| | DN-AFS | 2.49E-160 | 4.64E-137 | 4.09E-147 |
| | PDN-AFS | 2.50E-160 | 9.46E-142 | 4.66E-170 |
| $f_5$ | 改进 AFS | 6.86E-02 | 2.88E-02 | 2.88E-02 |
| | NAFS | 6.09E-06 | 4.42E-06 | 4.42E-06 |
| | DN-AFS | 7.91E-10 | 2.98E-04 | 7.43E-17 |
| | PDN-AFS | 5.49E-14 | 4.25E-07 | 5.42E-19 |

2)函数 $f_1 \sim f_5$ 在集群机下独立运行 30 次的函数值优化曲线分别如图 2~6 所示,其中图 2~4 与文献[3]中改进 AFS 算法对比,图 5、6 与文献[4]的 NAFS 算法对比。对于函数 $f_1, f_2, f_3$,DN-AFS 算法在平均迭代 600 次后能找到函数最优值且趋于稳定,PDN-AFS 算法在平均迭代次数为 400 时找到函数最优值,收敛速度略优于文献[3]中改进的 AFS 算法,而 AFS 算法在达到迭代次数 1 000 时仍没有准确搜索到全局最优点。对于函数 $f_4$,DN-AFS 算法收敛的速度明显快于改进的 AFS 算法,与 NAFS 算法的寻优速度不相上下。由图 6 可知在处理多峰函数时,DN-AFS 算法和 PDN-AFS 算法都能够进行比较全面的解空间的搜索,并且都优于 NAFS 算法。PDN-AFS 算法在后期收敛的效率明显提高,这是因为并行程序运行时存在线程间的启动、切换、销毁等开销,随着寻优难度的增加,PDN-AFS 算法的并行优势就开始体现出来。

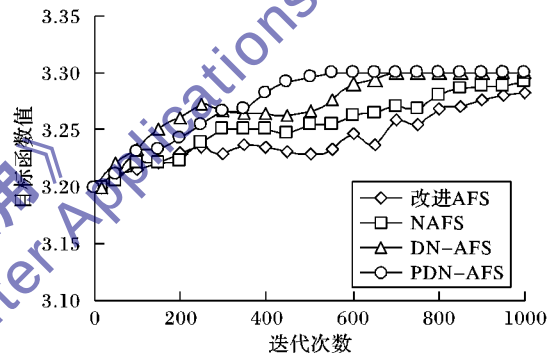


图 2　函数 $f_1$ 目标函数值进化曲线

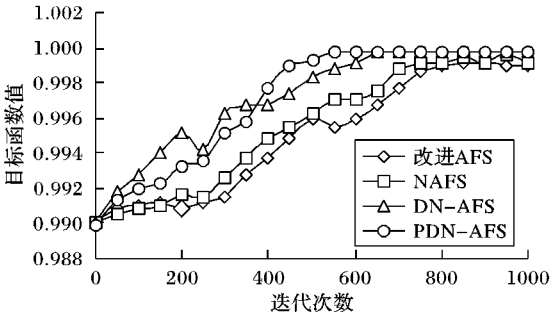


图 3　函数 $f_2$ 目标函数值进化曲线

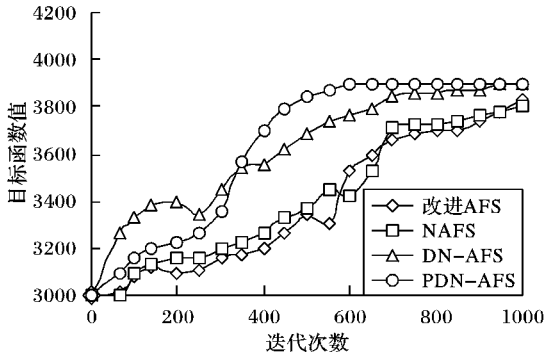


图 4　函数 $f_3$ 目标函数值进化曲线

3)表 3 是 DN-AFS 算法与 PDN-AFS 算法在迭代 1 000 次之后的加速比,可以看出采用多核机群的并行算法有效提高了计算性能,不过并没有达到理论上的线性加速比,这是因为机群中进程的启动、进程间数据传递、进程内的线程启动等都会造成通信延时,影响机群总体运行效率。但是总体来说

PDN-AFS 算法还是获得了较高的加速比和收敛效率。

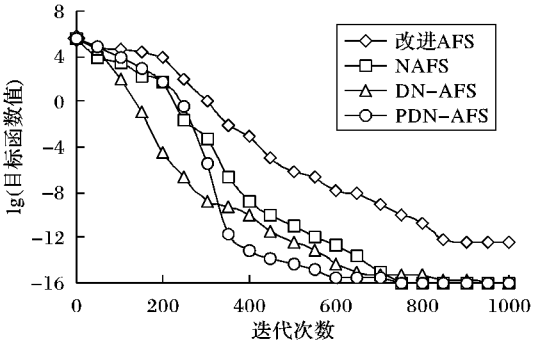


图 5 函数 $f_4$ 目标函数值进化曲线

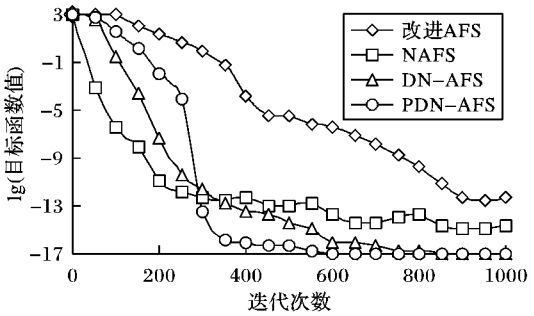


图 6 函数 $f_5$ 目标函数值进化曲线

**表 3 算法运行时间及加速比**

| 函数名称 | 运行时间/s | | 加速比 |
|---|---|---|---|
| | DN-AFS | PDN-AFS | |
| $f_1$ | 0.001 25 | 0.001 07 | 1.168 |
| $f_2$ | 2.264 10 | 1.388 20 | 1.630 |
| $f_3$ | 2.382 40 | 1.008 20 | 2.363 |
| $f_4$ | 0.003 07 | 0.001 64 | 1.872 |
| $f_5$ | 0.002 63 | 0.001 28 | 2.055 |

## 5 结语

本文通过在基本人工鱼群算法原有结构上引入动态权衡因子策略和小生境技术，提高了算法寻优精度；利用 MPI + OpenMP 混合编程模式，实现了在多处理器计算机集群体系结构特下的人工鱼群并行算法；并通过仿真实验证明改进后的算法的有效性和较强的鲁棒性，为高维多峰函数以及大规模复杂工程的优化问题提供了一种有效手段。下一步工作是将智能群算法在 OpenMP、MPI、TBB 等并行编程模型下进行对比分析，并尝试将 TBB 与 MPI 结合，为多核处理器集群提供有效的并行层次化结构的同时也提高算法的可扩展性。

**参考文献：**

[1] 李晓磊，邵之江，钱积新. 一种基于动物自治体的寻优模式：鱼群算法[J]. 系统工程理论与实践，2002，22(11)：32－38.

[2] 邓涛，姚宏，杜军. 多峰函数优化的改进人工鱼群混合算法[J]. 计算机应用，2012，32(10)：2904－2906.

[3] 陈广洲，汪家权，李传军等. 一种改进的人工鱼群算法及其应用[J]. 系统工程，2009，27(12)：105－110

[4] 王培崇，雷凤君，钱旭. 改进人工鱼群算法及其收敛性分析[J]. 科学技术与工程，2013，13(3)：616－620

[5] 胡一凡. 基于 GPU 加速的并行人工鱼群算法及其应用[D]. 杭州：浙江大学，2011.

[6] 李晓磊. 一种新型的智能优化算法——人工鱼群算法[D]. 杭州：浙江大学，2003.

[7] 王培崇，钱旭，雷凤君. 新的混合小生境鱼群聚类算法[J]. 计算机应用，2012，32(8)：2189－2192.

[8] 华杰，崔杜武. 基于个体优化的自适应小生境遗传算法[J]. 计算机工程，2010，36(1)：194－196.

[9] LING Q，WU G，YANG Z Y，*et al*. Crowding clustering genetic algorithm for multimodal function optimization [J]. Applied Soft Computing，2008，8(1)：88－95.

（上接第 3374 页）

## 3 结语

如何评价 PSO 在寻优过程中的粒子分布性，是衡量仿生类优化算法的一个重要性能指标。本文提出一种粒子在搜索过程的多样性分布计算方法，将这种方法引入到粒子的勘探和开发两种模式，并根据多样性设计了一种自适应惯性权值的调整策略。

本文下一步的工作将进一步对 APSO 在寻优过程中的种群拓扑结构进行研究，其初步设想通过多样性的评价值来选择不同的粒子邻域结构，从而提高 APSO 在高维复杂优问题中的寻优速度，提高算法的优化性能。

**参考文献：**

[1] KENNEDY J，EBERHART R C. Particle swarm optimization [C]// Proceedings of the 1995 IEEE International Conference on Neural Networks. Piscataway：IEEE，1995：1942－1948.

[2] RAMAZAN C. A fuzzy controller design for nuclear research reactors using the particle swarm optimization algorithm [J]. Nuclear Engineering and Design，2011，241 (5)：1899－1908.

[3] CHEN W N，ZHANG J，CHUNG H S H，*et al*. A novel set-based particle swarm optimization method for discrete optimization problems [J]. IEEE Transactions on Evolutionary Computation，2010，14 (2)：278－300.

[4] 陈仕涛，陈国龙，郭文忠，等. 基于粒子群优化和邻域约简的入侵检测日志数据特征选择[J]. 计算机研究与发展，2010，47(7)：1261－1267.

[5] 陈自郁，何中市，张程. 一种邻居动态调整的粒子群优化算法[J]. 模式识别与人工智能，2010，23(4)：586－592.

[6] 汤可宗，柳炳祥，杨静宇，等. 双中心粒子群优化算法[J]. 计算机研究与发展，2012，49(5)：1086－1094.

[7] HU X H，EBERHART R C. Adaptive particle swarm optimization：detection and response to dynamic systems [C]// CEC '02：Proceedings of the 2002 Congress on Evolutionary Computation. Piscataway：IEEE，2002，2：1666－1670.

[8] BLACKWELL T M，BENTLEY P J. Dynamic search with charged swarms [C]// GECCO '02：Proceedings of the Genetic and Evolutionary Computation Conference. San Francisco：Morgan Kaufmann，2002：19－26.

[9] 窦全胜，周春光，徐中宇，等. 动态优化环境下的群核进化粒子群优化方法[J]. 计算机研究与发展，2006，43(1)：89－95.

[10] 俞欢军，张丽平，陈德钊，等. 基于反馈策略的自适应粒子群优化算法[J]. 浙江大学学报：工学版，2005，39(9)：1286－1291.

[11] 黄泽霞，俞攸红，黄德才，等. 惯性权自适应调整的量子粒子群优化算法[J]. 上海交通大学学报，2012，46(2)：228－232.

[12] 姜长元，赵曙光，郭力争，等. 改进型耗散粒子群优化算法[J]. 东华大学学报：自然科学版，2012，38(3)：312－317.