基于自适应粒子群算法求解单目标任务问题研究

摘要

与标准粒子群优化（Canonical Particle Swarm Optimization Algorithm, PSO）算法相比，自适应粒子群优化（Adaptive Particle Swarm Optimization Algorithm, APSO）算法具有更高的搜索效率，更快的收敛速度，更优的全局搜索能力。APSO主要包括两个步骤：通过评估种群分布和粒子个体适应值($fitness$)，确定当前种群处于某种进化状态，包括勘探、开发、收敛和迁移；实现惯性权重、加速度系数的自动控制，实时更新进化参数。当进化状态被归类为收敛状态时，APSO通过精英学习策略得到更精确的解，相反，迁移状态通过增加全局最优粒子（$Gbest_i$）的加速度系数，帮助种群跳出局部最优困境。APSO已分别在CEC2017数据集的10D、30D、50D和100D维度上进行实验测试，并与标准PSO比对。实验结果表明APSO 在收敛速度、全局最优性、解的准确性和算法可靠性方面均显著优于PSO。

关键词：自适应粒子群优化算法；粒子群优化算法;演化计算

1 引言

粒子群优化算法（Particle Swarm Optimization, PSO）以其实现简便、易于理解和高效等特性，在解决现实世界中的多种复杂优化问题方面已获得广泛应用 [4] - [3]。这些应用涵盖了工程设计优化、机器学习参数调优、路径规划、生物信息学等多个领域，充分展示了PSO在处理连续和离散优化问题上的广泛适用性和卓越性能。

然而，尽管PSO在许多应用中表现出色，它仍然存在一些固有的局限性。与其他进化计算方法类似，PSO基于种群的迭代优化，其核心思想是通过模拟鸟群觅食行为来寻找搜索空间中的最优解 [8]。这种基于种群的特性使得PSO在搜索过程中能够并行探索多个解空间区域，从而提高了全局搜索能力。然而，在处理某些复杂的优化问题，尤其是多模态问题时，标准PSO（Canonical Particle Swarm Optimization Algorithm, PSO）易陷入局部最优困境，且难以摆脱局部最优解的束缚，导致算法最终收敛于次优解。此外，PSO在某些情况下的收敛速度较慢，尤其是在高维度和复杂约束条件下，进一步限制了其应用范围和效率。这些弱点在一定程度上制约了PSO算法在更广泛、更复杂的实际问题中的推广和应用 [7]。

近年来，提升PSO的收敛速度以及增强其摆脱局部最优困境的能力，成为该领域内最重要且具有吸引力的两个主要目标。为此，研究人员提出了许多改进PSO，这些改进算法的创新点主要集中在算法结构的设计优化、搜索策略调整以及动态自适应参数控制等方面 [5]、[9]、 [8]、 [1]。例如，一些研究通过引入自适应惯性权重和加速度系数，动态调整粒子的搜

索步伐，从而在不同的搜索阶段实现平衡探索与开发的效果；另一些研究则通过结合局部搜索算子或全局优化策略，增强算法的全局搜索能力，避免过早收敛于局部最优解。

然而，尽管已有众多研究致力于提升PSO的性能，但迄今为止，如何同时实现算法加速收敛和有效规避局部最优解的问题，仍然是一个难以克服的挑战。例如，[1]中提出的综合学习粒子群优化（Comprehensive Learning PSO, CLPSO）算法主要侧重于通过增强全局搜索能力来避免局部最优，但在某些情况下其收敛速度相对较慢，未能充分解决优化效率的问题。为克服这些局限性，复现论文提出了一种自适应粒子群优化（Adaptive Particle Swarm Optimization, APSO）算法，该算法通过开发系统参数适应方案和精英学习策略（Elite Learning Strategy, ELS），在保证搜索效率的同时，有效提升了算法的全局搜索能力。

2 相关工作

2.1 标准PSO

在标准PSO以及大多数改进PSO算法中，适应值（$fitness$）被视作单个粒子个体进化结果的评判标准。将适应值优异的粒子选做学习对象是目前惯用的粒子群种群优化策略。在简单的单模态问题中，基于适应值的学习对象选择策略可以提供更快的收敛速度，但在复杂的多模态问题中，它可能会导致粒子受困于局部最优解。标准粒子群算法速度更新公式和位置更新公式分别见公式（1）和公式（2）：

$$v_{i,j} = \omega v_{i,j} + c_1 r_1 (pb_{i,j} - v_{i,j}) + c_2 r_2 (gb_{i,j} - x_{i,j}) \tag{1}$$

$$x_{i,j} = x_{i,j} + v_{i,j} \tag{2}$$

其中 w 代表先前的速度惯性权重; c_1 和 c_2 是两个加速度系数; $r_{1,j}$和$r_{2,j}$是两个随机数并均匀分布在区间 [0, 1]上.

基于以上讨论，PSO 可概括为算法 1。

Algorithm 1 标准PSO算法(Canonical Particle Swarm Optimization, PSO).

Input: :粒子个数N,问题维度P,求解方程F,最大迭代次数gen;

1: 初始化X,V;初始化$Pbest$，$Gbest$;
2: **while** $fitness$未达到最优 or 未达到最大迭代次数gen **do**
3: 　更新粒子速度 V(Eq. (1));
4: 　计算粒子位置 X(Eq. (2));
5: 　个体最优[i] = $Pbest$[i];
6: 　全局最优 = $Gbest$;
7: **end while**

Output: : 全局最优$Gbest$;

2.2 最新改进PSO算法

鉴于其有效性和简单的概念，PSO 已成为一种流行的优化算法，并被广泛应用于解决实际问题。因此，对该算法的理论研究和性能改进变得非常重要。Clerc 和 Kennedy [1]、Trelea [10]、Yasuda 等人 [12]、Kadirkamanathan 等人 [6]以及 van den Bergh 和 Engelbrecht [11]等人发表了对算法的收敛性分析和稳定性的研究。同时，关于PSO性能改进的研究也有很多文献，包括参数研究、与辅助运算的结合以及相关的拓扑结构 [4]、 [7]、 [3]。

此外，有作者 [?]还提出将模糊自适应参数ω用于调节速度惯性的影响，而 [2]则尝试将ω设为由 $0.5 + random(0,1)/2$ 得到的随机数值，用于动态系统优化。由于这种随机ω的期望值为0.75，因此它与Clerc的约束因子(为分析收敛行为引入) [1]具有相似的设计思路。

3 方法复现

3.1 划分PSO种群拓扑结构

在自适应粒子群算法(Adaptive Particle Swarm Optimization Algorithm, APSO)的初始优化阶段，作者首先分析了PSO的迭代过程中种群的分布情况，并将其大致划分为三种结构(四种情况)，如图 1, 图 2, 图 3所示，分别指勘探阶段、开发阶段(收敛阶段)、迁移阶段。

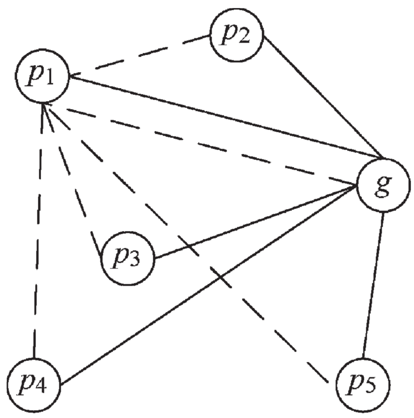

图 1. 勘探拓扑结构

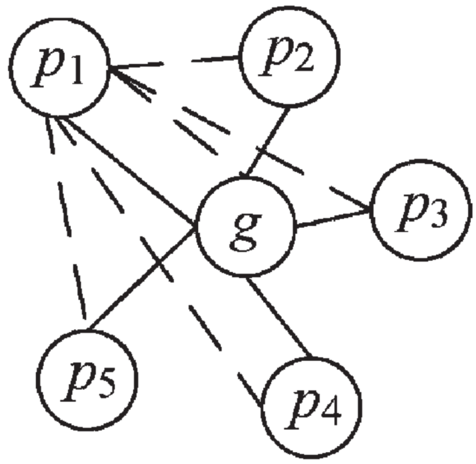

图 2. 开发拓扑结构

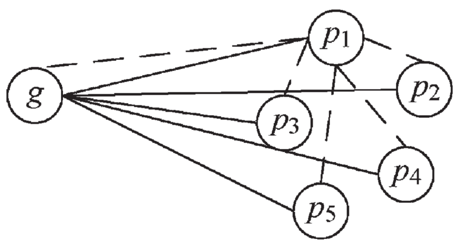

图 3. 迁移拓扑结构

进化迭代过程中，种群分布结构的划分依据是公式（3）和公式（4）：

$$d_i = \frac{1}{N-1}\sum_{\substack{j=1\\ j\neq i}}^{N}\sqrt{\sum_{k=1}^{D}(x_{ki}-x_{kj})^2} \tag{3}$$

其中，N 和 D 分别为群体规模和粒子个体维数。

$$f = \frac{dg - d_{\min}}{d_{\max} - d_{\min}} \in [0,1] \tag{4}$$

进化系数f数值作为模糊控制系统的重要输入参数，其不同数值控制着APSO对当前拓扑结构的划分判断，详细请参考论文中相关部分的具体描述。

3.2　模糊控制自适应参数

在划分了迭代过程中的种群分布结构后，原论文作者进一步将模糊控制理论应用于进化参数的自适应调整，即实现对w、c_1和c_2的模糊输出控制。如图 4所示，S1、S2、S3和S4分别指收敛阶段、开发阶段、勘探阶段、迁移阶段。

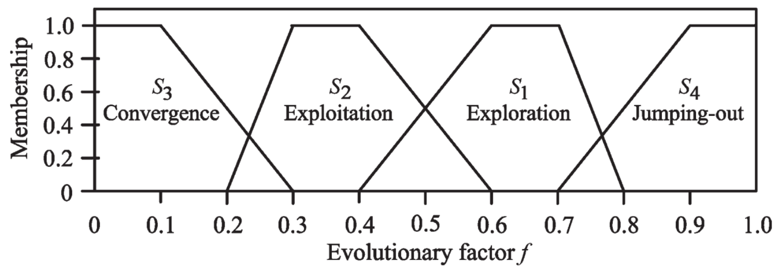

图 4. 四种种群进化结构的模糊隶属度函数

依据进化系数f，模糊控制系统将得到与f相对应的隶属度计算结果，并依据划分得到的模糊区间以及公式（5）和公式（6），计算或对自适应参数w、c_1和c_2施加约束。

$$\omega(f) = \frac{1}{1 + 1.5e^{-2.6f}} \in [0.4, 0.9] \quad \forall f \in [0, 1] \tag{5}$$

$$c_i = \frac{c_i}{c_1 + c_2} \cdot 4.0, \quad i = 1, 2 \tag{6}$$

最终，将计算出的自适应参数运用于标准PSO中的种群速度、位置迭代过程中。至此，APSO的完整框架设计与创新点介绍完毕。

4 复现细节

4.1 与已有开源代码对比

本次复现工作没有参考任何相关源代码，本次工作完全基于标准PSO框架与论文改进策略设计并编写，在此明确申明。本次复现工作基本实现了APSO中绝大多数创新策略，读者可以下载该代码，并进一步用于自己的科研工作。

4.2 对比实验设置

本研究将APSO在CEC2017基准函数（10维、30维、50维、100维）上进行了实验测试，并将标准PSO算法作为对比算法，分别记录了运行30次的最优解平均值和最优解方差的结果。

5 实验结果分析

	PSO(10D)	APSO(10D)	PSO(30D)	APSO(30D)	PSO(50D)	APSO(50D)
F_1 $\frac{Mean}{S.D.}$	3.130E+03(2) 2.260E+03	**2.506E+03(1) 2.021E+03**	1.034E+08(2) 3.250E+08	**2.537E+03(1) 2.456E+03**	3.610E+07(2) 2.410E+07	**2.932E+05(1) 4.927E+05**
F_2 $\frac{Mean}{S.D.}$	9.287E-14(2) 5.254E-13	**0.000E+00(1) 0.000E+00**	4.544E+16(2) 7.070E+16	**5.072E+15(1) 1.359E+16**	1.430E+49(2) 2.820E+49	**2.800E+45(1) 6.831E+45**
F_3 $\frac{Mean}{S.D.}$	7.330E-11(2) 2.200E-10	**0.000E+00(1) 0.000E+00**	**1.058E+03(1) 5.380E+02**	6.322E+03(2) 3.066E+03	7.830E+04(2) 1.020E+04	**7.345E+04(1) 1.075E+04**
F_4 $\frac{Mean}{S.D.}$	4.840E+00(2) 1.540E+00	**3.076E+00(1) 1.774E+00**	9.998E+01(2) 2.394E+01	**6.696E+01(1) 3.348E+01**	2.070E+02(2) 5.360E+01	**1.969E+02(1) 4.007E+01**
F_5 $\frac{Mean}{S.D.}$	**1.090E+01(1) 3.600E+00**	3.981E+00(2) 2.350E+00	7.165E+01(2) 1.830E+01	**5.569E+01(1) 1.350E+01**	**1.450E+02(1)1.990E+01**	1.523E+02(2) 2.710E+01
F_6 $\frac{Mean}{S.D.}$	5.680E-14(2) 5.990E-14	**0.000E+00(1) 0.000E+00**	2.365E-01(2) 1.800E-01	**2.009E-08(1) 3.094E-08**	3.980E+00(2) 1.920E+00	**7.112E-01(1) 1.112E+00**
F_7 $\frac{Mean}{S.D.}$	1.590E+01(2) 7.020E+00	**1.405E+01(1) 1.719E+00**	1.176E+02(2) 2.196E+01	**9.930E+01(1) 2.092E+01**	3.230E+02(2) 4.160E+01	**2.670E+02(1) 5.324E+01**
F_8 $\frac{Mean}{S.D.}$	1.100E+01(2) 4.930E+00	**5.472E+00(1) 1.774E+00**	7.356E+01(2) 1.460E+01	**6.526E+01(1) 1.893E+01**	1.850E+02(2) 3.040E+01	**1.399E+02(1) 2.865E+01**
F_9 $\frac{Mean}{S.D.}$	3.400E-12(2) 1.070E-11	**0.000E+00(1) 0.000E+00**	2.795E+02(2) 3.050E+02	**7.540E+01(1) 7.454E+01**	**3.310E+03(1)8.960E+02**	3.314E+03(2) 2.267E+03
F_{10} $\frac{Mean}{S.D.}$	2.970E+02(2) 2.300E+02	**1.050E+02(1) 6.350E+01**	3.050E+03(2) 1.070E+03	**2.245E+03(1) 6.481E+02**	8.030E+03(2) 9.950E+02	**5.201E+03(1) 2.345E+03**
F_{11} $\frac{Mean}{S.D.}$	4.890E+00(2) 2.490E+00	**3.080E+00(1) 1.420E+00**	9.600E+01(2) 3.100E+01	**5.226E+01(1) 2.682E+01**	2.290E+02(2) 9.190E+01	**1.320E+02(1) 6.687E+01**
F_{12} $\frac{Mean}{S.D.}$	1.790E+04(2) 1.900E+04	**1.360E+04(1) 1.020E+04**	2.395E+06(2) 4.525E+06	**2.101E+05(1) 9.701E+04**	8.250E+06(2) 3.350E+06	**3.979E+06(1) 2.230E+06**
F_{13} $\frac{Mean}{S.D.}$	8.910E+03(2) 9.330E+03	**4.782E+03(1) 4.351E+03**	1.316E+04(2) 8.370E+03	**6.824E+03(1) 5.319E+03**	7.100E+03(2) 9.800E+03	**2.649E+03(1) 2.295E+03**
F_{14} $\frac{Mean}{S.D.}$	1.860E+01(2) 1.720E+01	**3.338E+00(1) 2.301E+00**	1.688E+04(2) 1.700E+04	**9.068E+03(1) 6.744E+03**	2.340E+05(2) 3.200E+04	**7.393E+04(1) 5.846E+04**
F_{15} $\frac{Mean}{S.D.}$	1.920E+01(2) 1.820E+01	**2.281E+00(1) 1.749E+00**	1.077E+04(2) 1.270E+04	**6.621E+03(1) 7.268E+03**	9.890E+03(2) 8.520E+03	**6.670E+03(1) 7.445E+03**

图 5. F_1到F_{15}实验结果

在10维、30维问题上，大多数函数优化问题的求解结果都比较接近最优值，说明两算法在低维度优化时都表现较好。具体来说，在10维问题上，标准PSO表现最优，这反映了基于$Gbest$的引导有助于种群在低维问题上快速求解精准的最优解;在30维问题上，虽然标准PSO在某些问题上取得了最优结果，但APSO在最优结果数量上占更大比例，尤其是在混合函数与复杂函数等复杂多模态函数问题上。

	PSO(10D)	APSO(10D)	PSO(30D)	APSO(30D)	PSO(50D)	APSO(50D)
F_{16} $\frac{Mean}{S.D.}$	1.860E+01(2) 1.670E+01	**2.671E-01(1) 2.188E-01**	**5.210E+02(1) 2.020E+02**	7.377E+02(2) 2.286E+02	**1.540E+03(1) 3.820E+02**	1.601E+03(2) 6.324E+02
F_{17} $\frac{Mean}{S.D.}$	1.140E+01(2) 1.210E+01	**7.762E+00(1) 8.541E+00**	2.156E+02(2) 1.260E+02	**1.793E+02(1) 1.399E+02**	9.960E+02(2) 3.090E+02	**8.202E+02(1) 2.410E+02**
F_{18} $\frac{Mean}{S.D.}$	5.470E+03(2) 1.040E+04	**2.018E+03(1) 3.624E+03**	2.510E+05(2) 2.280E+05	**3.874E+05(1) 3.290E+05**	**1.120E+06(1) 1.800E+06**	2.280E+06(2) 1.756E+06
F_{19} $\frac{Mean}{S.D.}$	5.140E+01(2) 7.560E+01	**1.372E+00(1) 8.555E-01**	1.527E+04(2) 1.670E+04	**9.195E+03(1) 1.060E+04**	2.660E+04(2) 1.830E+04	**2.144E+04(1) 1.075E+04**
F_{20} $\frac{Mean}{S.D.}$	4.770E+00(2) 8.470E+00	**6.854E-01(1) 6.081E-01**	**1.870E+02(1) 1.230E+02**	2.316E+02(2) 1.549E+02	7.020E+02(2) 2.670E+02	**6.547E+02(1) 2.807E+02**
F_{21} $\frac{Mean}{S.D.}$	**1.160E+02(1) 5.760E+01**	1.335E+02(2) 5.357E+01	2.740E+02(2) 1.590E+01	**2.600E+02(1) 1.283E+01**	3.950E+02(2) 2.250E+01	**3.491E+02(1) 2.356E+01**
F_{22} $\frac{Mean}{S.D.}$	1.030E+02(2) 3.320E+00	1.022E+02(1) 6.391E-01	1.010E+03(2) 1.490E+03	**3.922E+02(1) 9.226E+02**	1.020E+04(2) 2.200E+03	**7.433E+03(1) 3.013E+03**
F_{23} $\frac{Mean}{S.D.}$	3.180E+02(2) 7.540E+00	3.106E+02(1) 3.742E+00	4.306E+02(2) 1.700E+01	**4.214E+02(1) 2.798E+01**	6.360E+02(2) 3.380E+01	**5.941E+02(1) 2.960E+01**
F_{24} $\frac{Mean}{S.D.}$	**2.250E+02(1) 1.320E+02**	2.991E+02(2) 1.058E+02	5.166E+02(2) 2.640E+01	**5.113E+02(1) 3.230E+01**	7.230E+02(2) 4.350E+01	**6.724E+02(1) 4.505E+01**
F_{25} $\frac{Mean}{S.D.}$	4.370E+02(2) 1.950E+01	**4.227E+02(1) 2.425E+01**	3.875E+02(2) 1.730E+00	**3.873E+02(1) 2.054E+00**	6.280E+02(2) 6.870E+01	**5.965E+02(1) 3.953E+01**
F_{26} $\frac{Mean}{S.D.}$	3.320E+02(2) 6.640E+01	**2.765E+02(1) 9.871E+01**	1.920E+03(2) 2.190E+02	**1.857E+03(1) 3.130E+02**	3.500E+03(2) 4.770E+02	**2.919E+03(1) 4.484E+02**
F_{27} $\frac{Mean}{S.D.}$	4.020E+02(2) 2.760E+01	**3.947E+02(1) 3.563E+00**	5.135E+02(1) 5.480E+00	5.226E+02(2) 1.120E+01	6.560E+02(2) 8.600E+01	**6.466E+02(1) 9.819E+01**
F_{28} $\frac{Mean}{S.D.}$	4.470E+02(2) 1.130E+02	**3.000E+02(1) 0.000E+00**	4.479E+02(2) 2.470E+01	**4.119E+02(1) 1.924E+01**	5.760E+02(2) 4.170E+01	**5.749E+02(1) 3.011E+01**
F_{29} $\frac{Mean}{S.D.}$	**2.530E+02(1) 2.240E+01**	2.481E+02(2) 8.632E+00	6.455E+02(2) 1.600E+02	**6.289E+02(1) 1.933E+02**	1.050E+03(2) 2.090E+02	**8.874E+02(1) 2.056E+02**
F_{30} $\frac{Mean}{S.D.}$	8.590E+04(2) 2.570E+05	**3.325E+03(1) 2.681E+03**	1.020E+04(2) 1.000E+04	**6.482E+03(1) 2.775E+03**	**1.170E+06(1) 4.590E+05**	8.455E+06(2) 1.650E+05

图 6. F_{16}到F_{30}实验结果

在50维的高维问题上，两算法的最终优化结果相较于低维问题显著提高，这反映了高维优化问题的搜索难度随着维度的上升有着明显的提升。随着维度的增加，在50维问题上，标准PSO的求解结果偏高，仅在少部分简单函数上表现出色，整体来看，APSO在大多数问题上更具有效性。对比中低维函数优化问题的实验结果，APSO的优势在复杂低维问题以及绝大多数高维问题上更为明显，这说明通过模糊逻辑控制的自适应粒子个体的全局搜索能力更优，可以有效应对复杂高维问题。

6 总结与展望

自适应粒子群优化（APSO）算法提升了标准PSO的算法性能，具有更高的搜索效率，更快的收敛速度，更优的全局搜索能力。APSO已在CEC2017数据集的10维、30维、50维度上问题上完成实验测试，并与标准PSO比对。实验结果证实了APSO在CEC2017数据集中的复杂高维问题上具有有效性。在本次算法复现工作的基础之上，还可以通过简化算法内部的函数表达式来提高APSO的可解释性。在尝试降低当前算法计算量的同时，可以进一步研究模糊逻辑控制理论对应的种群拓扑结构的种类划分，提升自适应参数计算系统的求解性能。

参考文献

[1] M. Clerc and J. Kennedy. The particle swarm-explosion, stability and convergence in a multidimensional complex space. *IEEE Trans. Evol. Comput.*, 6(1):58–73, Feb. 2002.

[2] R. C. Eberhart and Y. Shi. Tracking and optimizing dynamic systems with particle swarms. In *Proc. IEEE Congr. Evol. Comput.*, pages 94–97, Seoul, Korea, 2001.

[3] R. C. Eberhart and Y. Shi. Guest editorial. *IEEE Trans. Evol. Comput.*, 8(3):201–203, Jun. 2004.

[4] R. C. Eberhart and Y. H. Shi. Particle swarm optimization: Developments, applications and resources. In *Proc. IEEE Congr. Evol. Comput.*, pages 81–86, Seoul, Korea, 2001.

[5] S.-Y. Ho, H.-S. Lin, W.-H. Liauh, and S.-J. Ho. Opso: Orthogonal particle swarm optimization and its application to task assignment problems. *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, 38(2):288–298, Mar. 2008.

[6] V. Kadirkamanathan, K. Selvarajah, and P. J. Fleming. Stability analysis of the particle dynamics in particle swarm optimizer. *IEEE Trans. Evol. Comput.*, 10(3):245–255, Jun. 2006.

[7] X. D. Li and A. P. Engelbrecht. Particle swarm optimization: An introduction and its recent developments. In *Proc. Genetic Evol. Comput. Conf.*, pages 3391–3414, 2007.

[8] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Trans. Evol. Comput.*, 10(3):281–295, Jun. 2006.

[9] B. Liu, L. Wang, and Y. H. Jin. An effective pso-based memetic algorithm for flow shop scheduling. *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, 37(1):18–27, Feb. 2007.

[10] I. C. Trelea. The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Inf. Process. Lett.*, 85(6):317–325, Mar. 2003.

[11] F. van den Bergh and A. P. Engelbrecht. A study of particle optimization particle trajectories. *Inf. Sci.*, 176(8):937–971, Apr. 2006.

[12] K. Yasuda, A. Ide, and N. Iwasaki. Stability analysis of particle swarm optimization. In *Proc. 5th Metaheuristics Int. Conf.*, pages 341–346, 2003.