混合拓扑结构的粒子群算法及其在测试数据生成中的应用研究

焦重阳[1] 周清雷[1] 张文宁[2,3]
(郑州大学信息工程学院 郑州 450001)[1] (中国人民解放军信息工程大学 郑州 450001)[2]
(中原工学院 郑州 450001)[3]

摘 要 粒子群算法(PSO)的拓扑结构是影响算法性能的关键因素,为了从根源上避免粒子群算法易陷入局部极值及早熟收敛等问题,提出一种混合拓扑结构的粒子群优化算法(MPSO)并将其应用于软件结构测试数据的自动生成中。通过不同邻域拓扑结构对算法性能影响的分析,采用一种全局寻优和局部寻优相结合的混合粒子群优化算法。通过观察粒子群的多样性反馈信息,对每一代种群粒子以进化时选择全局拓扑结构模型(GPSO)或局部拓扑结构模型(LPSO)的方法进行。实验结果表明,MPSO使得种群的多样性得到保证,避免了粒子群陷入局部极值,提高了算法的收敛速度。

关键词 粒子群算法,测试数据自动生成,拓扑结构,全局寻优,局部寻优,多样性
中图法分类号 TP301 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.12.045

MPSO and Its Application in Test Data Automatic Generation

JIAO Chong-yang[1] ZHOU Qing-lei[1] ZHANG Wen-ning[2,3]
(School of Information Engineering, Zhengzhou University, Zhengzhou 450001, China)[1]
(The PLA Information Engineering University, Zhengzhou 450001, China)[2] (Zhongyuan University of Technology, Zhengzhou 450001, China)[3]

Abstract To date, meta-heuristic search algorithms have been applied to automate test data generation. The topology of particle swarm optimization (PSO) is one of the key factors that affect algorithm performance. In order to overcome the phenomena of falling into local optimal and premature convergence of the standard particle swarm algorithm (PSO), a mixture neighborhood structure (MPSO) was proposed to generate software structure test data automatically. Based on the analysis of the different neighborhood topology structure effect on the performance of particle optimization, this paper presented a new particle swarm optimization with mix topological structure. MPSO is based on the combination of global optimization and local optimization. In each generation, by observing the feedback information of diversity of the population, the particle speed update method is selected by global topology model or local topology model. The experimental results show that MPSO increases the swarm diversity, avoids falling into local optimization, and improves the convergence speed of the proposed algorithm.

Keywords Particle swarm algorithm, Automatic test data generation, Topology structure, Global optimization, Local optimization, Diversity

1 引言

基于搜索的软件测试(Search Based Software Testing, SBST)[1]在测试数据生成中备受关注,之后粒子群优化算法(PSO)[2-3]以其全局搜索能力强、控制参数少、容易实现、计算量小、通用性强等优点成为元启发式搜索算法的后起之秀,在路径规划、神经网络、生物医学、工程设计、电力系统等领域已得到广泛应用。粒子群算法在搜索的中后期,种群缺乏多样性,且易陷入局部最优解,呈现出早熟性。针对这一严重缺陷,国内外研究者们提出了许多基于粒子群优化算法的改进算法,主要表现在粒子群算法参数的改进和粒子群邻域拓扑结构的调整上。参数的调整与改进主要体现在惯性权重、学习因子、时间因子等方面。

粒子群算法是一种基于群智能的进化计算技术。粒子群的信息共享方式直接影响着粒子群的进化,这种信息共享方式称为粒子的邻域拓扑结构。因此,为从根源上避免早熟收敛等问题,本文从粒子群的邻域拓扑结构着手。对此,国内外学者们对粒子群拓扑结构已有一些研究。Kennedy 和

到稿日期:2016-10-12 返修日期:2017-01-24 本文受国家自然科学基金项目(61250007),河南省科技厅基础与前沿技术研究项目(152300410055)资助。

焦重阳(1991-),女,硕士生,CCF会员,主要研究方向为软件工程、软件测试、智能计算,E-mail:Cyang8985@126.com;**周清雷**(1962-),男,博士,教授,博士生导师,主要研究方向为软件形式化方法、模型验证、信息安全、操作系统;**张文宁**(1982-),女,博士生,讲师,主要研究方向为软件工程、智能计算。

Mendes[4,5]对5种典型的邻域拓扑结构进行了全面的测试，结果显示不同拓扑结构的粒子群算法具有不同的优化性能。王雪飞等[6]提出了一种基于小世界网络模型的具有动态拓扑结构的新颖粒子群算法，有效地保持了优秀粒子与非优秀粒子所占比例的均衡性。Zhao 等[7]提出了动态多群粒子群算法，该算法将整个种群分成一些子群，用这些子群去优化各自的目标，然后按照一定的规则再重新组群。刘衍民等[8]提出了基于动态邻居和变异因子的粒子群算法，算法中粒子的邻居根据它的运行而动态变化。石松等[9]提出了层次环形拓扑结构的动态粒子群算法(HRPSO)，粒子组成的环被分配在规则树中，算法运行时，环在层次中动态移动。Mohais 等[10]将有向图的概念引入邻域拓扑结构模型中，提出了两种动态改变邻域结构的方法。温雯等[11]提出一种新颖的 PSO 改进算法(PSO-DT)来动态调整粒子群的拓扑结构，该算法首先计算粒子成为其他粒子邻域粒子的概率，然后在进化过程中根据这些概率对每个粒子选择邻域粒子。

针对静态拓扑结构不能自适应环境变化的缺点，以上动态粒子群算法的提出提高了 PSO 算法自适应环境变化的能力。但动态拓扑结构在建立过程中忽略了种群密度，增加了额外的时间耗费和算法的复杂度。针对不同拓扑结构对算法性能的影响结果分析，提出了一种全局寻优和局部寻优相结合的混合粒子群优化算法(MPSO)。通过分析粒子群的多样性指标，采用混合邻域拓扑结构进行进化寻优，提高了算法的收敛速度，避免了算法早熟现象的发生。

2 基本粒子群算法及其拓扑结构

2.1 基本粒子群算法(PSO)

1995 年，美国社会心理学家 James Kennedy[2]和电气工程师 Russell Eberhart[3]基于鸟群觅食过程的迁徙和聚集行为提出了粒子群优化算法(PSO)，该算法通过各个粒子的交流协作来搜索空间问题的最优解。粒子群优化算法的具体描述如下：

假设在一个 D 维的目标搜索空间中，t 代种群由 N 个粒子组成，其中第 i 个粒子的位置为 $X_i=(x_{i1}^t,x_{i2}^t,\cdots,x_{iD}^t)$，$i=1,2,\cdots,N$；第 i 个粒子的速度为 $V_i=(v_{i1}^t,v_{i2}^t,\cdots,v_{iD}^t)$，$i=1,2,\cdots,N$；第 i 个粒子迄今为止搜索到的最优位置为 P_{best}，称为个体极值，$P_{best}=(p_{i1}^t,p_{i2}^t,\cdots,p_{iD}^t)$，$i=1,2,\cdots,N$；整个粒子群迄今为止搜索到的最优位置为 g_{best}，称为全局极值，$g_{best}=(p_{g1}^t,p_{g2}^t,\cdots,p_{gD}^t)$。在找到这两个最优值时，粒子根据如下公式更新自己的速度和位置：

$$v_{id}^{t+1}=wv_{id}^t+c_1r_1(p_{id}^t-x_{id}^t)+c_2r_2(p_{gd}^t-x_{id}^t) \tag{1}$$

$$x_{id}^{t+1}=x_{id}^t+v_{id}^{t+1} \tag{2}$$

其中，w 为惯性权重，其目的是使微粒保持运动惯性。$c_1>0$，$c_2>0$ 为学习因子，$r_1,r_2\in[0,1]$为随机变量，$v_{id}\in[-v_{\max},v_{\max}]$，$x_{id}\in[-x_{\max},x_{\max}]$。

2.2 粒子群的邻域拓扑结构

粒子群的拓扑结构指整个种群中所有粒子之间相互连接的方式。而粒子群的邻域拓扑结构则是指单个粒子如何与其他粒子相连。粒子群的邻域拓扑结构是决定粒子群算法性能的关键因素，对于不同邻域拓扑结构的粒子群算法，其效果差别很大。目前粒子群拓扑结构主要包括：全互连型、环型、冯诺依曼型、星型、四类、金字塔等6种结构。

(1)全互连型结构：种群中所有粒子相互通信。这种结构中的信息的传递速度和收敛速度快，但容易陷入局部极值。

(2)环型结构：种群中所有粒子首尾互连形成一个环，每个粒子只与左右两个相邻的粒子进行信息交换，对于环中其他粒子则只能通过邻居节点间接进行信息交流。这种结构具有较强的寻优能力，陷入局部极值的概率小，但其收敛速度慢。

(3)冯诺依曼型结构：种群中所有粒子呈正方形排列，种群中的粒子与其上、下、左、右4个粒子相连，呈现一种立体正方形结构。这种结构在对每个区域进行充分搜索的同时加强了粒子间的信息交流，使得粒子更容易避开局部极值，并找到最优解，且收敛性也较快。

(4)星型结构：种群中所有粒子以其中一个粒子为中心，且其他粒子之间互不相连。在种群中，只有中心粒子与该邻域中所有粒子进行信息交流，而其他粒子不进行信息交流。这种结构使得信息传播速度大幅提高，收敛速度快，但易于陷入局部极值。

(5)四类结构：整个种群分为4个类，各个类内部互相通信，但类间两两相连进行通信，使信息在整个种群中得到分享。

(6)金字塔结构：也称四面体结构，粒子分布在四面体的4个顶点上，然后所有的四面体再相互连接，使得粒子之间的信息得到了全面的分享。

通过对以上6种拓扑结构的分析与研究可知，粒子群的拓扑结构能够控制种群信息的传播，直接影响算法的收敛性和寻优能力。其中，冯诺依曼型[12]拓扑结构的聚类系数为零，在收敛性上优于环形，在寻优能力上优于全互连型，是一种公认的较好的邻域拓扑结构。

3 混合拓扑结构(MPSO)模型

3.1 全局拓扑结构模型(GPSO)

在全局拓扑结构模型 GPSO[2-3]中，整个种群信息共享，每个粒子的邻居由群体中其他所有粒子构成，所有粒子朝着种群中的最优粒子进化。基本粒子群算法(PSO)采用的邻域拓扑为全局拓扑结构，其粒子速度与位置更新公式如下：

$$v_{id}^{t+1}=wv_{id}^t+c_1r_1(p_{id}^t-x_{id}^t)+c_2r_2(p_{gd}^t-x_{id}^t) \tag{3}$$

$$x_{id}^{t+1}=x_{id}^t+v_{id}^{t+1} \tag{4}$$

其中，p_{gd}^t为整个粒子群的全局最优位置。本文 GPSO 模型采用 2.2 节中的全互连型结构构造粒子群的邻域拓扑结构。

3.2 局部拓扑结构模型(LPSO)

1999 年，Kennedy[13]首次提出局部版本的粒子群优化算法，采用环形拓扑结构，如 2.2 节的环形结构所描述。在局部拓扑结构模型 LPSO 中，每个粒子的邻居由与它直接相连的粒子构成，粒子跟踪自身历史最优位置和邻域拓扑结构中所有邻居的最优位置进行进化，而不再跟踪种群最优位置。粒子速度和位置更新公式如下：

$$v_{id}^{t+1}=wv_{id}^t+c_1r_1(p_{id}^t-x_{id}^t)+c_2r_2(p_{id}^t-x_{id}^t) \tag{5}$$

$$x_{id}^{t+1}=x_{id}^t+v_{id}^{t+1} \tag{6}$$

其中，p_{id}^t为粒子拓扑结构中所有邻居的局部最优位置。本文

LPSO 模型构造的邻域拓扑采用 2.2 节中的冯诺依曼型结构。

3.3 混合模型(MPSO)

全局模型 GPSO 中,每个粒子向全局最优粒子的方向进化,更明显地收敛于当前最优解。每次对粒子位置进行更新时,综合了种群中所有粒子的特性,其收敛速度快,但易陷入局部极值。局部模型 LPSO 中,每个粒子仅仅与其邻居进行适应度函数值的比较,促使较差粒子向较好粒子进化,收敛速度变慢,但种群多样性得到了保证且不易陷入局部极值。Mendes 研究了不同固定模式的邻域拓扑结构对粒子群算法性能的影响,其研究结果表明:粒子个体间社会交互的平均连接度越高,种群中的信息传播速度就越快,但早熟收敛现象也较易发生。鉴于此,提出全局寻优和局部寻优相结合的混合粒子群优化算法,即 MPSO,通过观察粒子群多样性指标,每一代粒子在进化过程中选取 GPSO 或 LPSO 进行进化寻优。

定义 1(粒子多样性度量,Div)　采用 Riget[14] 提出的度量方法,公式如下:

$$Div=\frac{1}{|N|\times|L|}\sum_{i=1}^{N}\sqrt{\sum_{d=1}^{D}(x_{id}^{t}-\overline{x_{d}^{t}})^{2}} \tag{7}$$

其中,$|N|$ 为搜索空间中粒子的个数,$|L|$ 为搜索空间中最大对角线长度,D 为问题的维度,x_{id}^{t} 为第 t 次迭代第 i 个粒子位置的第 d 维分量,$\overline{x_{d}^{t}}$ 为第 t 次迭代所有粒子第 d 维分量的平均值。粒子多样性描述了种群中各个粒子相互之间分布的离散程度,Div 值越小则种群越集中,种群的多样性越低。种群的多样性丧失导致粒子群优化算法陷入局部极值。因此,通过观察种群多样性的反馈信息自动调节粒子群的拓扑结构来保持种群的多样性,避免算法陷入局部最优解。

4 MPSO 与测试数据自动生成

4.1 适应度函数构造

本文采用分支路径覆盖法来自动生成测试用例,在构造适应度函数时以分支为研究对象,给出程序在实参驱动下的轨迹与分支的偏离程度。借鉴 Korel[15] 和 Tracey 等[16] 的研究成果,针对几种典型的分支谓词设计分支距离的计算方式如表 1 所列(表 1 中 k 为大于 0 的常量)。

表 1　几种典型的分支谓词的距离函数构造

No.	Branch Predicate	Branch Distance Function f(bchi)
1	*boolean*	If true then 0 else k
2	$\neg a$	Negation is propagated over a
3	$a=b$	If abs$(a-b)=0$ then 0 else $(a-b)+k$
4	$a\neq b$	If abs$(a-b)\neq 0$ then 0 else k
5	$a<b$	If $a-b<0$ then 0 else $(a-b)+k$
6	$a\leqslant b$	If $a-b\leqslant 0$ then 0 else $(a-b)+k$
7	$a>b$	If $b-a<0$ then 0 else $(b-a)+k$
8	$a\geqslant b$	If $b-a\leqslant 0$ then 0 else $(b-a)+k$
9	a and b	$f(a)+f(b)$
10	a or b	$\min(f(a),f(b))$

分支路径覆盖由分支谓词和分支嵌套两部分组成。分支谓词的类型和分支嵌套的深度都会影响分支路径的全覆盖程度。一般而言,分支的嵌套深度越大,该分支的覆盖越难实现。为此,引入一个衡量分支嵌套的变量——分支嵌套权重 $wn(bchi)$,则程序的第 i 个分支 $bchi$ 嵌套权重计算如下:

$$wn(bchi)=0.417*e^{\frac{l_i-l_{\min}}{l_{\max}-l_{\min}}} \tag{8}$$

其中,l_i 为当前分支的深度,$l_{\max}$ 为最大分支嵌套深度,$l_{\min}$ 为最小分支嵌套深度。

分支谓词的类型对分支的可达性也有重要影响。通常,一个分支谓词(branch predicate)由多个条件通过"and"或"or"连接而成。引入一个衡量分支谓词的变量——分支谓词权重 $wp(bchi)$,则程序的第 i 个分支 $bchi$ 分支谓词权重计算如下:

$$wp(bchi)=\begin{cases}\sqrt{\sum_{j=1}^{u}w^{2}(c_j)}, & \text{如果连接词为 and}\\ \min\{w(c_j)\}, & \text{如果连接词为 or}\end{cases} \tag{9}$$

其中,第 i 个分支 $bchi$($1\leqslant i\leqslant s$,s 为分支的数目)包含 u 个条件,每个条件用 c_j($1\leqslant j\leqslant u$)表示,根据其条件关系的不同赋予不同的权重。从认知信息学的角度,针对不同类型的条件,给定的参考权重 $w(c_j)$($1\leqslant j\leqslant u$)如表 2 所列。

表 2　几种条件关系的参考权重

No.	Condition Type	Weight
1	$=$	0.9
2	$<,\leqslant,>,\geqslant$	0.6
3	Boolean	0.5
4	$\neq$	0.2

对于任意一条分支 $bchi$,其对应的惯性权重 w_i 可以表示为谓词权重和嵌套权重两者的综合形式:

$$w_i=0.5\times wn(bchi)+0.5\times wp(bchi) \tag{10}$$

对一个给定的程序 p,其对应的适应度函数构造如下:

$$fitness(p)=\sum_{i=1}^{s}w_i\times f(bchi) \tag{11}$$

其中,s 为被测程序 p 中的分支数目,w_i 为该分支对应的权重,$f(bchi)$ 为第 i 个分支的分支距离。

4.2 基于 MPSO 的测试数据自动生成模型

基于 MPSO 的测试数据自动生成模型分为 3 个部分,测试环境构造、MPSO 算法和测试运行,具体如图 1 所示。

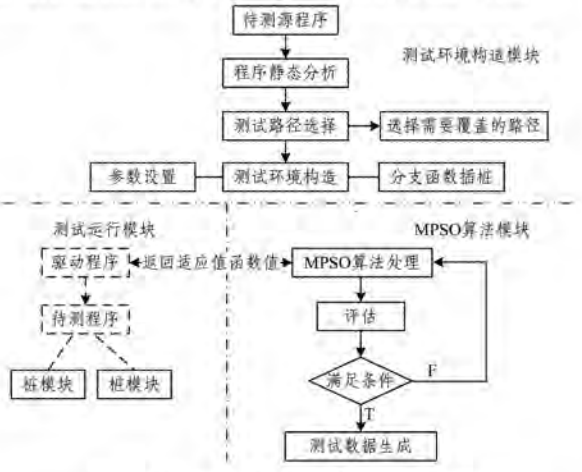

图 1　基于 MPSO 的测试数据自动生成模型

图 1 中,测试环境构造模块是整个模型的基础部分。在此部分,根据待测源程序进行静态分析,选择目标路径,并对待测源程序进行分支函数插桩,构造目标函数及对算法进行相关参数的设置。MPSO 算法模块是核心部分,根据测试环境构造模块中所提供的参数和搜索空间,初始化粒子群,并根

据粒子多样性公式和粒子位置、粒子速度进化公式更新粒子的位置和速度，通过适应度函数引导粒子向目标位置进化。在测试运行模块中，将粒子位置映射到实际问题参数的搜索域中，将其作为参数传递到待测程序中。

4.3 MPSO 算法的实现步骤

MPSO 算法的实现如算法 1 所示。

算法 1 MPSO 算法

输入：经分支函数插桩后的被测程序

输出：找到目标路径的测试数据，即空间最优位置

变量设置：种群规模 N、空间维数 D、最大进化代数 T、当前迭代次数 t、多样性阈值 Div_{thr}

step1 初始化种群规模为 N 的粒子群，设定学习因子 c1，c2 的值，惯性权重 w，多样性阈值 Div_{thr} 以及最大进化代数 T。

step2 每个粒子在定义域内随机初始化它们的位置 X_i 和速度 V_i，个体极值 P_{best} 为各个粒子的初始位置，全局极值 g_{best} 为适应值最好的粒子所对应的位置。

step3 根据目标函数 F(x)计算每个粒子的适应值 fitness(x_i)。

step4 对于每个粒子，将其适应值与个体极值 P_{best} 的适应值进行比较，若较好，则更新 P_{best}。

step5 对于每个粒子，将其适应值与全局极值 g_{best} 的适应值进行比较，若较好，则更新 g_{best}。

step6 利用式(7)计算粒子群多样性指标的大小，并与所设置的阈值作比较，若 $Div < Div_{thr}$，则根据式(5)和式(6)更新粒子的位置；否则，根据式(3)和式(4)更新粒子的位置。

step7 判断是否满足终止条件，若满足，则输出最优解；否则返回 step2。

5 实验

5.1 实验方案

本文将等边三角形判定问题、求最大最小值问题作为测试程序，分析比较 GPSO 算法、LPSO 算法和 MPSO 算法分别用于测试数据自动生成上的性能。其中，全局拓扑结构模型(GPSO)采用全互连型结构，局部拓扑结构模型(LPSO)采用冯诺依曼型结构来构造粒子群的邻域拓扑结构。本文主要使用两个重要指标进行算法的对比分析。

(1)平均进化代数：每个程序运行 100 次，若该程序在最大迭代次数 $T=1000$ 之内覆盖所选择的目标路径，则记录此时的进化代数；若没有找到目标路径，则记录为 T。平均进化代数即为所需进化代数的平均值。

(2)平均进化时间：被测程序生成目标路径所需进化时间的平均值。

5.2 参数设置

本文实验平台为：内存 2.0G，Windows7 系统，X4 处理器，MATLAB R2012b 辅助软件。对于三角形分类程序，3 条边的输入范围分别为：[0,100]，[0,200]和[0,500]。为避免参数不同对算法性能的影响，GPSO，LPSO，MPSO 算法采用统一的参数，设置为：空间维数 $D=3$，学习因子 $c_1=1.5$，$c_2=1.5$，惯性权重 $w=0.2$，最大进化代数 $T=1000$。MPSO 算法中多样性阈值 $Div_{thr}=0.1$。对于求最大最小值程序，3 种算法的参数统一设置为：种群规模 $N=100$，数据范围为：[0,100]，[0,200]，[0,500]。空间维数 $D=3$，学习因子 $c_1=1.5$，$c_2=1.5$，惯性权重 $w=0.5$，最大进化代数 $T=1000$。MPSO 算法中多样性阈值 $Div_{thr}=0.1$。一般而言，参数的选取会直接影响智能搜索算法的性能。为获取最优解，本文的参数设置是进行多次实验并对实验参数做出适当的调节而得到的。

5.3 实验结果

5.3.1 三角形分类程序

选择生成等边三角形作为测试路径，分别进行 100 次实验，记录 GPSO，LPSO，MPSO 3 种算法生成该路径测试数据所需的进化代数和所消耗的进化时间。实验结果如表 3－表 5 所列。

表 3 种群规模为 50 时 3 种算法的实验结果

种群规模	搜索空间	数据范围	平均进化代数			平均进化时间/s		
			GPSO	LPSO	MPSO	GPSO	LPSO	MPSO
50	3	[0,100]	62.3	56.5	54.3	0.152	0.281	0.276
	3	[0,200]	67.3	67.3	66.7	0.161	0.320	0.328
	3	[0,500]	75.1	74.0	73.1	0.181	0.339	0.361

表 4 种群规模为 100 时 3 种算法的实验结果

种群规模	搜索空间	数据范围	平均进化代数			平均进化时间/s		
			GPSO	LPSO	MPSO	GPSO	LPSO	MPSO
100	3	[0,100]	62.2	59.3	57.8	0.292	0.307	0.299
	3	[0,200]	69.8	66.0	64.9	0.321	0.310	0.329
	3	[0,500]	81.6	78.7	75.6	0.362	0.360	0.366

表 5 种群规模为 150 时 3 种算法的实验结果

种群规模	搜索空间	数据范围	平均进化代数			平均进化时间/s		
			GPSO	LPSO	MPSO	GPSO	LPSO	MPSO
150	3	[0,100]	61.7	60.0	56.2	0.439	0.295	0.284
	3	[0,200]	72.1	71.3	64.6	0.481	0.318	0.326
	3	[0,500]	76.1	71.2	67.3	0.488	0.328	0.337

由表 3－表 5 可以看出：1)在同一种群规模下，随着数据范围的不断扩大，3 种算法的平均进化代数和平均进化时间也不断增加。MPSO 的进化代数总是少于 LPSO，LPSO 的进化代数少于 GPSO。2)在同一数据范围下，随着种群数目的增加，3 种方法获得的最优解的平均进化代数的差距越来越大。MPSO 在种群规模较大时所需进化代数和进化时间明显少于 LPSO 和 GPSO，即 MPSO 在大规模种群数目下的寻优能力更强。3)在同一种群规模和同一数据范围内，3 种算法的平均进化代数和平均进化时间均不同。在表 3 和表 4 中，MPSO 的进化时间虽会多于 GPSO，但在表 5 中，MPSO 在进化时间上明显少于 GPSO，占据优势。

5.3.2 求最大最小值程序

选择生成最大值分支路径作为目标测试路径，选择种群规模为 100，分别进行 100 次实验，记录 GPSO，LPSO，MPSO 3 种算法生成该目标路径测试数据所需的进化代数和所消耗的进化时间，实验结果分别如图 2 和图 3 所示。

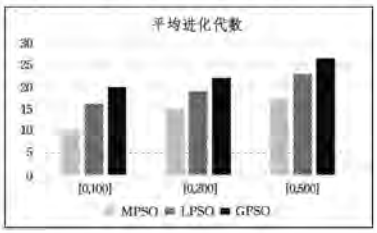

图 2 3 种算法所需进化代数的实验结果

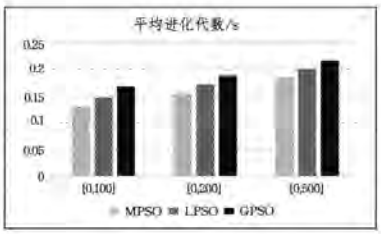

图 3　3 种算法所需进化时间的实验结果

由图 2 和图 3 可知，本文提出的 MPSO 算法在寻找目标测试数据时所需要的进化代数和所消耗的进化时间明显少于 LPSO 算法，而 LPSO 算法少于 GPSO 算法。显然，MPSO 策略使算法的全局搜索性能和局部搜索性能得到了平衡，在测试用例生成方面更具优势。

为进一步验证 MPSO 算法的有效性，以三角形分类程序为例分别进行 6 次统计实验，每次统计实验进行 100 次，记录 3 种算法覆盖测试路径所能达到的搜索成功率。实验结果如图 4 所示。

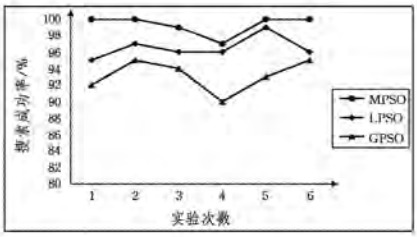

图 4　生成等边三角形路径的搜索成功率

从图 4 中可以看出，MPSO 算法在搜索成功率上优于 LPSO 算法，而 LPSO 算法优于 GPSO 算法。在不同的种群规模下，针对不同的输入范围做了大量的实验。对实验结果的统计分析可知，MPSO 算法在寻优能力上优于 LPSO 算法，LPSO 算法优于 GPSO 算法。

综上所述，MPSO 在进化速度、搜索效率和收敛精度上均具有明显优势，且性能更优。

6　有效性分析

本文提出的混合拓扑结构的粒子群优化算法应用于测试数据自动生成，主要是对基本粒子群算法的邻域拓扑结构进行分析改进。一方面，因为基本粒子群算法是一种随机进化搜索算法，所得实验结果具有一定的随机性，所以本文实验结果取 100 次重复实验的平均值以降低算法随机性带来的影响；另一方面，本文根据粒子群多样性度量指标来选择粒子群拓扑结构模型，没有动态改变种群密度，减小了算法的复杂度。

结束语　粒子群算法的拓扑结构决定了粒子之间的信息共享方式，是影响算法性能的关键因素。本文提出的混合拓扑结构粒子群优化算法结合了全局拓扑结构模型 GPSO 算法和局部拓扑结构模型 LPSO 算法的优点，使得 MPSO 算法的收敛速度快，收敛精度高；同时增加了种群的多样性，避免种群过早收敛，平衡了全局搜索能力与局部搜索能力，解决了粒子群易陷入局部极值的问题。软件结构性测试领域的一个关键难题是测试数据的生成，本文重点讨论了将粒子群算法应用于该难题，提出的 MPSO 算法不仅不需要构建复杂的动态拓扑结构，而且保持了种群的一定密度，提高了软件结构测试数据自动生成的效率。该策略保留了基本粒子群算法搜索能力强、简单易实现、收敛速度快的优点，有利于在实际优化领域中进行应用和推广。

参 考 文 献

[1] HARMAN M. Search-Based software engineering[C]// International Conference on Computational Science. Springer Berlin Heidelberg, 2006: 740-747.

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

[3] EBERHART R, KENNEDY J. A new optimizer using particle swarm theory[C]// Proceedings of the Sixth International Symposium on Micro Machine and Human Science. Nagoya, 1995: 39-43.

[4] KENNEDY J, MENDS R. Neighborhood topologies in fully informed and best-of-neighborhood particle swarms [J]. IEEE Transaction on Systems Man and Cybernetics Part C-Applications and Reviews, 2006, 36(4): 515-519.

[5] KENNEDY J, MENDS R. Population structure and particle swarm performance[C]// Proceeding of the 2002 Congress on Evolutionary computation. Honolulu, HI, USA, 2002: 1671-1676.

[6] WANG X F, WANG F, QIU Y H. Research on a novel particle swarm algorithm with dynamic topology[J]. Computer Science, 2007, 34(3): 205-207. (in Chinese)
王雪飞，王芳，邱玉辉. 一种具有动态拓扑结构的粒子群算法研究[J]. 计算机科学，2007，34(3)：205-207.

[7] ZHAO S Z, LIANG J J, SUGANTHAN P N, et al. Dynamic multi-swarm particle swarm optimizer with local search for large scale global optimization[C]// 2008 IEEE Swarm Intelligence Symposium. Hong Kong, 2008: 3845-3852.

[8] LIU Y M, ZHAO Q Z, SUI C L, et al. Particle swarm optimizer based on dynamic neighborhood topology and mutation operator [J]. Control and Decision, 2010, 25(7): 968-974. (in Chinese)
刘衍民，赵庆祯，隋常玲，等. 一种基于动态邻居和变异因子的粒子群算法[J]. 控制与决策，2010，25(7)：968-974.

[9] SHI S, CHEN Y. Dynamic particle swarm optimization algorithm with hierarchical ring topology[J]. Computer Engineering and Applications, 2013, 49(8): 1-5. (in Chinese)
石松，陈云. 层次环形拓扑结构的动态粒子群算法[J]. 计算机工程与应用，2013，49(8)：1-5.

[10] MOHAIS A S, MENDES R, WARD C, et al. Neighborhood restructuring in particle swarm optimization[C]// Australian Conference on Artificial Intelligence. 2005: 776-785.

[11] WEN W, HAO Z F. Improved Particle Swarm Optimizer Based on Dynamic Topology[J]. Computer Engineering and Applications, 2005, 41(34): 82-85. (in Chinese)
温雯，郝志峰. 一种基于动态拓扑结构的 PSO 改进算法[J]. 计算机工程与应用，2005，41(34)：82-85.

[12] CHEN Z Y, HE Z S, ZHANG C. Image multi-level thresholds based on PSO with Von Neumann neighborhood[J]. Application

Research of Computers, 2009, 26(5): 1977-2000. (in Chinese)
陈自郁,何中市,张程. 基于冯诺依曼邻居的粒子群多阈值分割算法[J]. 计算机应用研究,2009,26(5):1977-2000.

[13] KENNEDY J. Small World and mega-minds: effects of neighborhood topology on particle swarm performance[C]// IEEE Congress on Evolutionary Computation. Piscataway, NJ, 1999: 1931-1938.

[14] RIGET J, VESTERSTROEM J S. A diversity guided particle swarm optimizer- the ARPSO[R]. Denmark: University of Aaehus, 2002.

[15] KOREL B. Dynamic method for software test data generation [J]. Software Testing, Verification and Reliability, 1992, 2(4): 203-213.

[16] TRACEY N, CLARK J, MANDER K, et al. An automated framework for structural test-data generation[C]// Proceeding of the 13th International Conference on Automated Software Engineering. Honolulu, HI, USA, 1998: 285-288.

[17] MCMINN P. Search-based software test data generation: A survey[J]. Software Testing, Verification and Reliability, 2004, 14(2): 105-156.

[18] HARMAN M, MCMINN P. A theoretical and empirical study of search-based testing: local, global, and hybrid search[J]. IEEE Transactions on Software Engineering, 2010, 36(2): 226-247.

[19] MA S L, YE D Y, YANG L L. A new design criteria of particle swarm topology[J]. Computer Engineering, 2015, 41(1): 200-206. (in Chinese)
马胜蓝,叶东毅,杨玲玲. 一种新的粒子群拓扑结构设计准则[J]. 计算机工程,2015,41(1):200-206.

(上接第 226 页)

[2] WANG M, HUA X S. Active learning in multimedia annotation and retrieval: a survey[J]. ACM Transactions on Intelligent System and Technology, 2011, 2(2): 210-231.

[3] SETTLES B, CRAVEN M. An analysis of active learning strategies for sequence labeling tasks[C]// Proceedings of the Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2008: 1069-1078.

[4] LI Y, FANG B X, GUO L, et al. Supervised Intrusion Detection Based on Active Learning and TCM-KNN Algorithm [J]. Chinese Journal of Computers, 2007, 30(8): 1464-1473. (in Chinese)
李洋,方滨兴,郭莉,等. 基于主动学习和 TCM-KNN 方法的有指导入侵检测技术[J]. 计算机学报,2007,30(8):1464-1473.

[5] YU D, VARADARAJAN B, DENG L. Active learning and semi-supervised learning for speech recognition: A unified framework using the global entropy reduction maximization criterion[J]. Computer Speech & Language, 2010, 24(3): 433-444.

[6] NIU B, CHENG J, BAI X. A symmetric propagation based batch mode active learning for image retrieval[J]. Signal Processing, 2013, 93(6): 1639-1650.

[7] XU M X, SUN F M, LI H J. Online multi-label image classification with active learning[J]. Journal of Image and Graphics, 2015, 20(2): 237-244. (in Chinese)
徐美香,孙福明,李豪杰. 主动学习的多标签图像在线分类[J]. 中国图象图形学报,2015,20(2):237-244.

[8] MOHAMED T P, CARBONELL J G, GANAPATHIRAJU M K. Active learning for human protein-protein interaction prediction[J]. BMC bioinformatics, 2010, 11(Suppl 1): S57.

[9] ZHU J, HOVY E. Active Learning for Word Sense Disambiguation with Methods for Addressing the Class Imbalance Problem[C]// Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Prague, 2007: 783-790.

[10] HUANG G B, ZHU Q Y, SIEW C K. Extreme learning machine: theory and applications[J]. Neurocomputing, 2006, 70(1-3): 489-501.

[11] HUANG G B, ZHOU H, DING X, et al. Extreme learning machine for regression and multiclass classification [J]. IEEE Transactions on System, Man and Cybernetics, Part B: Cybernetics, 2012, 42(2): 513-529.

[12] HUANG G B, WANG D H, LAN Y. Extreme learning machine: a survey[J]. International Journal of Machine Learning and Cybernetics, 2011, 2(2): 107-122.

[13] ZONG W, HUANG G B, CHEN Y. Weighted extreme learning machine for imbalance learning [J]. Neurocomputing, 2013, 101(3): 229-242.

[14] MCCALLU A, NIGRAM K. Employing EM in pool-based active learning for text classification[C]// Proceedings of the International Conference on Machine Learning. Morgan Kaufmann, 1998: 350-358.

[15] ZHU X, ZHANG P, LIN X, et al. Active learning from stream data using optimal weight classifier ensemble[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 2010, 40(6): 1607-1621.

[16] YU H L, SUN C Y, YANG W K, et al. AL-ELM: One uncertainty-based active learning algorithm using extreme learning machine[J]. Neurocomputing, 2015, 166: 140-150.

[17] LIANG N Y, HUANG G B, SARATCHANDRAN P. A fast and accurate online sequential learning algorithm for feedforward networks[J]. IEEE Transactions on Neural Networks, 2006, 17(6): 1411-1423.

[18] ALCALA FDES J, FEMANDEZ A, LUENGO J, et al. KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework[J]. Journal of multiple-valued logic and soft computing, 2011, 17(2/3): 255-287.

[19] CHAWLA N V, BOWYER K W, HALL L O. SMOTE: Synthetic Minority Over-Sampling Technique[J]. Journal of Artificial Intelligence Research, 2002, 16(1): 321-357.

[20] GUYON I, CAWLEY G C, DROR G, et al. Results of the Active Learning Challenge[C]// JMLR: Workshop and Conference Proceedings. 2011: 19-45.