软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2011,22(8):1738−1748 [doi: 10.3724/SP.J.1001.2011.03830]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种基于动态小生境的自组织学习算法*

周传华[1,3], 谢安世[2,3+]

[1](中国科学技术大学 计算机科学与技术学院,安徽 合肥 230026)

[2](中国科学院 科技政策与管理科学研究所,北京 100190)

[3](安徽工业大学 管理科学与工程学院,安徽 马鞍山 243002)

Dynamic Niche-Based Self-Organizing Learning Algorithm

ZHOU Chuan-Hua[1,3], XIE An-Shi[2,3+]

[1](School of Computer Science and Technology, University of Science and Technology of China, Hefei 230026, China)

[2](Institute of Policy and Management, The Chinese Academy of Sciences, Beijing 100190, China)

[3](School of Management Science and Engineering, Anhui University of Technology, Maanshan 243002, China)

+ Corresponding author: E-mail: shermanxas@163.com

Zhou CH, Xie AS. Dynamic niche-based self-organizing learning algorithm. *Journal of Software*, 2011,22(8): 1738−1748. http://www.jos.org.cn/1000-9825/3830.htm

Abstract: A dynamic niche-based self-organizing learning algorithm (DNSLA) was proposed in this paper. The dynamic learning mechanism based on 0-1 coding method was carried out, and the individuals involved in this algorithm were able to adapt to the dynamic environments through active learning, which was different from the passive adaptive search strategy in traditional evolutionary algorithms. As a result of self-organizing learning and friendly interaction with the environments, DNSLA was more robust to adapt to the dynamic problems, and it was able to accurately detect the slight changes of the environments and track the extreme points in the solution domain. A series of dynamic simulation tests for comparative experiments showed that, even in the turbulent environments, DNSLA was still able to perform friendly interactive learning with the dynamic environments. DNSLA showed a strong robustness in the comparative experiments, whose dynamic search capabilities were far superior to other search methods.

Key words: dynamic niche; self-organizing learning; evolutionary computing; dynamic environment

摘 要: 提出了一种基于动态小生境的自组织学习算法(dynamic niche-based self-organizing learning algorithm,简称 DNSLA),实现了基于 0-1 编码的动态学习机制.种群中的个体由被动适应转为主动学习,即通过系统的自组织学习而实现与环境的友好交互,因而具有更强健的动态环境适应能力,能够及时、准确地侦测到环境的变化并跟踪极值点在搜索空间内的运动轨迹,具有良好的可移植性和很强的泛化能力.一系列动态测试问题的对比仿真实验结果表明,该算法即使在剧烈动荡的环境中也能很好地与环境进行稳定而友好的交互学习,表现出了很强的鲁棒性,其动态搜索能力和极值点跟踪能力远优于同类搜索方法.

关键词: 动态小生境;自组织学习;进化计算;动态环境

* 基金项目: 安徽省教育厅重大项目基金(ZD200904)
收稿时间: 2009-05-17; 修改时间: 2009-11-14; 定稿时间: 2009-12-02; jos 在线出版时间: 2010-06-30

中图法分类号: TP181 文献标识码: A

现实中的许多优化问题往往是动态变化的,其最优解会随着时间和环境的改变而改变,也即问题的最优解不是固定不变的.如 Job Shop 调度问题中,新的 Job 不断到达,需要立即加入到当前已有调度中,而机器可能会发生故障或降低加工速度,原材料会发生改变以及各种约束条件也可能发生变化等等.虽然传统的优化方法也可以在每一次变化之后的环境条件基础上从头开始求解,但这实质上是开始了又一个新的优化问题,这将会导致无法重用过去的有用信息而重复计算浪费时间等问题;而且实际情况是环境变化的信息往往无法及时获得,因此传统的针对静态问题的优化方法对这类动态问题无能为力.这就需要一种功能强大的能够处理这些现实世界中普遍存在的不确定性的启发性方法,以便随时侦测出所处环境的变化并作出迅速而准确的反应.

传统进化算法的目标是使种群逐渐收敛,最终获得一个满意解.因此,进化算法在动态环境中所面临的主要挑战就是如何在进化过程中保持种群的多样性,从而保持对环境变化的适应能力.许多学者在这方面做了一些有益的工作,如,为了平衡算法的开发性和探索性,Cobb 在传统遗传算法(GA)中采用了一种过度变异的策略[1].当探测到环境的变化后立刻猛烈增大变异率,使得趋于收敛的种群发散.在后来的研究中,过度变异被进一步检验[2],结果发现,环境变化的频率对变异率的增大幅度有一定的影响,频率越大,过度变异越强烈.Vavak 等人[3]提出一种称为可变局部搜索的变异算子,最初只是采用较小的变异(如仅仅允许二进制编码最后几位发生变异),当种群的性能在一定时期内没有得到改善时,逐步增大局部搜索的范围.文献[4]提出一种称为随机移民的方法,即在每一代,种群中的部分个体都会被随机产生的个体所替代.Cobb 等人[5]比较了 3 种不同变异方案在动态环境中的应用:固定变异率、过度变异和随机移民的方法,结果发现,采用过度变异的 GA 在缓慢变化的环境中(低强度)表现最好;如果环境变化较大,则随机移民的方法表现会更好.Goldberg 等人提出了基于二倍体和基因显性机制的 GA[6],在动态背包问题的应用中,该方法比标准 GA 表现出更好的性能.Hadad 等人提出了多倍体的方法[7],将一个统治变量作为个体的一部分,在动态背包问题中,该方法能够获得与二倍体方法一样好的性能;而且他们发现,多倍体的方法更适用于变化频率较高的环境.Ryan 使用了一种附加多倍体方法[8],附加的基因用于决定个体表现型的某个特性.在某些动态问题中,这种方法取得了较好的结果.自组织侦察群方法[9]的基本思想是,当一个峰被发现之后,种群就会分离出一个子种群来监视这个峰;而种群中的其余个体展开,继续搜索新的峰.Oppacher 等人提出一种 Shifting Balance GA[10],整个种群分成核心种群和许多小的殖民地种群,核心种群的任务是探索极值点,小殖民地种群负责在适值曲线上几个孤立的区域进行搜索,即能够保证探索性.当适值曲线变化频繁而变化幅度较小时,这种算法明显优于常规 GA.文献[11]还讨论了一种更为有效的计算个体与核心种群之间距离的方法.Ursem 提出一种多种族 GA[12],在该算法中,子种群的分组利用一种“峰谷探测过程”来确定.文献[13]提出的原对偶遗传算法(PDGA)在进行交叉变异等遗传运算之前,首先对种群中的部分染色体进行对偶运算,取得了较好的效果.

上述改良方法大体上如文献[14]所述,主要是以修改某些进化操作算子、引入某种记忆策略和采用多种群策略等措施来保持种群的多样性,以便种群能够更容易适应环境中的变化.上述措施虽然有一定的作用,但都是具体操作层面上的改良,而算法总体的进化策略仍然是被动适应的.本文提出的基于动态小生境的自组织学习算法(dynamic niche-based self-organizing learning algorithm,简称 DNSLA)一反传统的进化策略,使种群中的个体由传统的被动适应转化为主动学习,通过种群的自组织学习与环境进行友好交互,可随时侦测出所处环境的变化并作出迅速而准确的反应,因而具有更强健的动态环境适应能力,能够及时而准确地跟踪极值点在搜索空间内的运动轨迹,具有良好的可移植性和较强的泛化能力.

1 动态自组织学习算法(DNSLA)

DNSLA 的搜索模式为:设立 $np+1$ 个公告板(np 是生态系统内小生境种群的数目),分别记录各个小生境种群和整个生态系统的最优个体,分别代表所求解问题的局部最优解和全局最优解,并随着迭代过程不断更新.种群内个体的搜索过程是一个学习过程,即根据学习对象来调整自己的搜索方法和搜索步长.随着学习的进行,当

某个小生境种群发现了较好的全局解时,其他小生境种群将指派部分个体进入该小生境种群以协助进行密集搜索;但如果某个小生境种群在进化过程中一直都没有发现较好的全局解,则其中所有个体将会逐渐被调配往其他小生境种群,最终导致该小生境种群消亡.同理,如果某个体能够发现较好的全局解,将会从其他小生境种群中吸引更多的个体到其邻域内而形成新的小生境种群.这种自组织学习过程体现了强者更强、弱者更弱的马太效应,符合自然界物竞天择、适者生存的自然法则.

各个种群的自组织学习是这样进行的:每个种群中一部分个体进行全局学习,即向整个生态系统内具有最佳适应度的个体学习;另一部分个体进行邻域学习,即向该个体所在种群内具有最佳适应度的个体学习;其余部分进行基于对偶映射的自发学习.每个个体的学习欲望和学习强度各不相同,其大小是根据该个体自身的适应度与其学习对象的适应度(fitness)及海明距离(Hamming distance,简称 HD)的大小来决定的.下面我们对其分别加以详细描述.

1.1 自组织学习设计

本文讨论的学习机制是基于 0-1 编码方案的,因此,个体对搜索方向和搜索步长的调整主要体现在对自身基因坐标的调整上,学习欲望主要体现在学习率上,学习强度主要体现在该个体所调整的基因位个数上.

1.1.1 基于个体适应度值的全局学习

设 X_E^{best} 是整个生态系统内具有最佳适应度的个体,其对应的基因表达式为 G_E^{best};X_K^i 是种群 P_K 所属的第 i 个个体,其对应的基因表达式为 G_K^i,则 X_K^i 的全局学习率为

$$\begin{cases}\max f(x): Grate_K^i = Grate' + f_K^i / \tilde{f}_K - 1\\ \min f(x): Grate_K^i = Grate' + \tilde{f}_K / f_K^i - 1\end{cases} \tag{1}$$

其中,$Grate'$表示全局学习率的初始值,f_K^i 表示个体 X_K^i 的适应度,$\tilde{f}_K$ 表示种群 P_K 的平均适应度.X_K^i 进行全局学习是指 G_K^i 中与 G_E^{best} 相异的基因以 $Grate_K^i$ 的概率替换成 G_E^{best} 中相应的基因,即 X_K^i 主动缩小与 X_E^{best} 的海明距离.从公式(1)中可以看出,某个体的适应度越符合优化目的(根据优化目的,其适应度大于或小于平均适应度),则其全局学习欲望越强烈,全局学习率就越大.通过这种机制,生态系统内那些有前途的个体主动聚集到全局最佳个体所在的搜索邻域内,能够起到协助搜索的作用.

1.1.2 基于海明距离的邻域学习

设 X_E^{best} 是小生境种群 P_K 内具有最佳适应度的个体,其对应的基因表达式为 G_K^{best};该种群内第 i 个个体 X_K^i 所对应的基因表达式为 G_K^i,则 X_K^i 的邻域学习率为

$$Brate_K^i = Brate' - HD_{k,h} / Length + 1 \tag{2}$$

其中,$Brate'$表示邻域学习率的初始值,$HD_{k,h}$ 为该个体与 X_E^{best} 的海明距离,$Length$ 为个体基因表达式的编码长度.X_K^i 进行邻域学习是指 G_K^i 中与 G_K^{best} 相异的基因以 $Brate_K^i$ 的概率替换成 G_K^{best} 中相应的基因,即 X_K^i 主动缩小与 X_E^{best} 的海明距离.由公式(2)可知,当小生境种群中某个体与该种群的最佳个体的海明距离较小时,其邻域学习欲望会自动增加,迅速迁移到该种群中最佳个体的搜索邻域内,以协助该个体进行搜索.

这里的全局学习和邻域学习都是缩小个体之间的海明距离,看似相同且可能会让所有个体出现相同的基因型,但实事上两者却有很大的差异,并且这也是本算法的核心思想之一:拉近与最优个体海明距离的行为,有利于种群进行集中搜索,同时也是保持种群多样性的最佳手段,因为每个个体的学习对象是动态变化的.后文将对此进行详述(见第 2 节算法分析部分).

1.1.3 基于对偶映射的自学习设计

个体进行基于对偶映射的自学习,指该个体的基因表达式中每个基因位都执行对偶映射,即 $0\leftrightarrow 1$.设 X_K^i 是隶属小生境种群 P_K 的个体,则 X_K^i 的自学习率为

$$\begin{cases}\max f(x): Srate_K^i = Srate' \times \tilde{f}_K / f_K^i\\ \min f(x): Srate_K^i = Srate' \times f_K^i / \tilde{f}_K\end{cases} \tag{3}$$

其中,*Srate'*表示自学习率的初始值, f_K^i 表示个体 X_K^i 的适应度, $\tilde{f}_K$ 表示种群 P_K 的平均适应度.从公式(3)可以看出,当优化目的是最大化时,如果个体的适应度小于其所在种群的平均适应度,则其学习欲望会快速增强,学习率将快速提高到一个较大的数值.于是有较多机会得到其对偶个体,以提高其自身的适应度;但如果该个体的适应度已经比该种群的平均适应度大,则其学习欲望会快速消退,学习率也快速降低到一个较小的数值,这样可以保护优良基因免遭破坏.同理,当优化目的是最小化时,其自学习率也会相应地自动调整,以符合优化目的.

1.2 伪代码

设 $E=\{P_1,P_2,\ldots,P_{np}\}$是包含 *np* 个小生境种群的生态系统,$N_i$ 表示种群 P_i 中个体的数目, P_i^j 表示 P_i 中第 *j* 个个体, P_i^{best} 表示 P_i 中具有最佳适应度的个体, $\tilde{f}_i$ 表示 P_i 在当前代的平均适应度, $\tilde{f}_E$ 表示生态系统当前代的平均适应度,P_{best} 表示当前生态系统中具有最佳适应度的个体,max*gen* 表示最大的迭代次数,则 DNSLA 的伪代码可表示如下:

1. 初始化小生境种群个数、各小生境种群中个体数目、学习率等参数
2. for *gen*=1:max*gen*, do
   a) for *i*=1:*np*, do
      i. 评估 P_i 中每个个体的适应度
      ii. 计算 P_i 的平均适应度 $\tilde{f}_i$
      iii. 记录并更新 P_i 中具有最佳适应度的个体 P_i^{best} ,即局部最优解
   b) 记录并更新整个生态系统中具有最佳适应度的个体 P_{best},即全局最优解
   c) $\tilde{f}_E=\left(\sum \tilde{f}_i\right)\Big/np$
   d) for *i*=1:*np*, do
      i. $\begin{cases}\max f(x): N_i=N_i+\tilde{f}_i/\tilde{f}_E-1\\ \min f(x): N_i=N_i-\tilde{f}_i/\tilde{f}_E+1\end{cases}$
      ii. if N_i==0
          删除种群 P_i(消亡)
      iii. endif
   e) for *i*=1:*np*, do
      i. 种群 p_i 中个体 P_i^j 进行基于适应度值的全局学习
      ii. 如果个体 P_i^j 的适应度没有得到改善,则进行基于海明距离的邻域学习
      iii. 如果个体 P_i^j 的适应度仍然没有得到改善,则进行基于对偶映射的自学习
   f) if 与上一代相比 $\tilde{f}_E$ 没有得到改善或 P_{best} 没有发生改变 do
      生态系统内各小生境种群之间相互交换具有最佳适应度的个体
3. 输出全局最优解

2 算法分析

与传统的智能优化算法相比,DNSLA 有一些独特之处,下面重点分析其两个突出特点.

2.1 动态因素

在 DNSLA 中有 4 个因素是动态变化的.

第 1 个动态变化的因素是各小生境种群的规模,每个种群的规模是基于该种群的平均适应度和整个生态系统的平均适应度的对比而动态地改变的.在种群 P_i 中,个体的数目是依照下述法则动态变化的:

$$\begin{cases}\max f(x): N_i = N_i + \tilde{f}_i / \tilde{f}_E - 1 \\ \min f(x): N_i = N_i - \tilde{f}_i / \tilde{f}_E + 1\end{cases} \tag{4}$$

这里, N_i^t 表示种群 P_i 在第 t 代时包含的个体数目,即种群规模.以优化目的是实现最大化为例,当某种群的平均适应度比整个生态系统平均适应度高时,其规模会进一步扩大;反之则会进一步萎缩.因此可以设想,将会有越来越多的个体自发地集中在搜索空间中更具前景的区域,这体现了算法的开发性(exploitation).

第 2 个动态变化的因素是迁移率.种群个体的迁移,本算法特指生态系统内各小生境种群相互交换具有最佳适应度的个体.通过迁移,有利于发现由各个种群中个体的学习行为而产生的新的基模.当整个生态系统从上一代进化到下一代,其平均适应度或最佳个体的适应度没有变化时,迁移将会发生;因此,当种群处于一种稳态时,通过迁移有助于探寻和发现符合优化目标的新的基模,也有利于保持生态系统内的种群多样性.

第 3 个动态变化的因素是学习欲望.由上文所述可知,个体的全局学习率是基于该个体的适应度和该个体所属种群的平均适应度的,当优化目标是实现最大化时,如果个体的适应度比种群的平均适应度大,则提高全局学习率以协助全局最佳个体进行密集搜索;个体的邻域学习率是基于该个体与其所属种群中最佳个体的海明距离而动态调整的,当海明距离较小时,增强邻域学习率,尽可能地减小与最佳个体的海明距离以协助种群内最佳个体进行密集搜索.个体的自学习率也是根据该个体的适应度和该个体所属种群的平均适应度的来动态调整的,如果适应度不符合优化目的,则其学习率快速上升到一个较高的水平,通过对偶映射得到其对偶个体;反之,则自学习率快速下降到一个较低的水平,从而有利于保护优良基因进化到下一代.

第 4 个动态变化的因素是学习行为.本算法设计了 3 种学习策略,包括基于个体适应度值的全局学习、基于海明距离的邻域学习和基于对偶映射的自学习.但每一个体不一定都要同时执行这些学习策略,只有当其执行某一学习策略而适应度没有得到相应改善时才会执行另一学习策略.这样不仅避免了进化过程中可能产生的退化现象,也大大节省了计算资源,提高了进化速度,同等条件下比遗传算法(SGA 和 PDGA,见后文实验部分)的计算时间减少了一半以上.

以上 4 个因素的动态变化是环境变量、约束条件等发生变化引发的.生态系统内外部环境的改变引起了个体和种群适应度的变化,即环境的变化反映在个体和种群的适应度的改变上.而上述 4 个基于个体和种群适应度的因素也动态地、自适应地调整,以便追踪最优个体在解空间内的运动轨迹.这表明 DNSLA 的确具有良好的与环境交互的能力.

2.2 种群多样性

与传统的静态导向的优化算法相反,DNSLA 不追求种群的最终收敛,而是要在进化过程中自始至终保持种群的多样性.环境变量、约束条件等的改变会引起种群外部环境的变化,有时甚至引起搜索空间的变形(如维数发生改变等),从而问题的最优解不再固定不变.因此,针对这类问题的算法,其首要任务是随时侦测到环境的变化并作出快速反应,进而追踪到问题最优解在搜索空间内的漂移轨迹.基于这样的需求,对进化算法来讲,如何保持种群在进化过程中的多样性成为主要问题.

基于 0-1 编码的进化算法大多是以个体之间的海明距离来界定收敛的,即生态系统内所有个体之间的海明距离以某种概率变为 0,即种群中所有个体都具有相同基因型时达到收敛.因此,本算法保持种群多样性就是要保持生态系统中有足够多的相互之间海明距离不为 0 的个体.本算法中,个体的学习行为本质上是该个体根据当前最佳个体的基因表达式来调整自己的基因表达式,个体在进行全局学习后有可能缩小与当前全局最佳个体的海明距离,在进行邻域学习后有可能缩小与其所属种群中当前最佳个体的海明距离.但是,随着新的全局最佳个体的出现和各小生境种群之间最佳个体的相互交换,所有个体的学习对象在不断地发生变化,而且个体在自学习后会得到其对偶个体.因此,不可能出现所有个体呈现相同基因型的情形,这就保证了整个生态系统和各小生境种群内的多样性,从而保证了算法在搜索过程中的探索性(exploration).关于算法保持种群多样性能力的理论和实证分析,详细内容参见 http://www.sciencenet.cn/u/shermanxas.

3 实验与分析

为了验证DNSLA的动态搜索能力,分别与标准遗传算法(SGA)和原对偶遗传算法(PDGA)[13]在3个动态测试问题中进行仿真对比实验.这3个问题分别是动态位匹配问题、时变背包问题和移动峰函数优化问题.

以上3种算法都是在MATLAB7.6环境下编程实现的,涉及到的控制参数众多且不尽相同,但均采用二进制编码,个体总数目均设为100.其中:DNSLA在初始阶段有10个小生境种群,每个种群包含10个个体;全局学习率和局部学习率的初始值都设为0.5,自学习率的初始值设为0.8;SGA采用锦标赛选择,反异或交叉和单点变异,其交叉率和变异率的初始值分别设为0.95和0.2.为便于对比,PDGA采用了与SGA相同的参数设置.

这里,DNSLA的全局学习率和局部学习率的初始值都设为0.5,自学习率的初始值设为0.8.其依据是,由于本算法特殊的学习策略,即个体的学习是通过将自己的基因位值用学习对象相应的基因位值以一定的概率进行替换来实现的,因此,当这3个学习率的初始值取值适中时,算法在运行过程中能够保持种群最好的多样性,从而具有最佳的搜索性能.为了体现DNSLA的优越性,在以下几个实验中,特意将自学习率的初始值设为一个较大的值0.8,这实际上降低了DNSLA保持种群多样性的能力,从而在总体上降低了DNSLA的搜索性能.

SGA和PDGA的交叉率和变异率分别设为0.95和0.2,是因为这两种算法都是主要依靠交叉操作来产生新的可行解,将交叉率设为较大的值时,有利于发挥其搜索性能.传统上,变异率一般被设置得很小,但实验中发现,当变异率过小时,变异操作对产生新的可行解的贡献几乎为0;而当变异率取值大于0.4时,则容易导致退化现象,或出现所谓随机漫游现象.因此,在以下几个实验中将变异率的初始值取0.2较为合适.

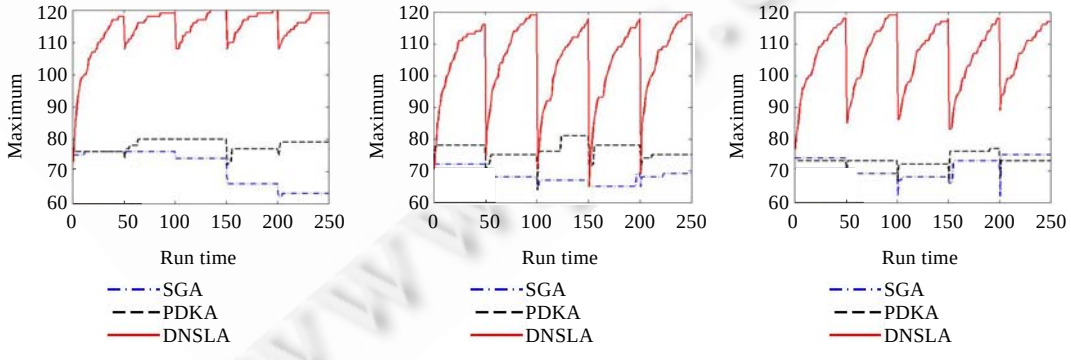
3.1 动态位匹配问题(dynamic bit matching,简称DBM)

3.1.1 问题定义

位匹配问题即给定一个字符串(本文采用二进制串,称为样本),个体与字符串相匹配的位数为该个体的适应度,该问题是要求出最大的位匹配数(显然,其最大值就是样本长度).而动态位匹配问题是指这个给定的字符串会动态地不断改变,并且种群内个体的基因表达式也会按照某种方式发生变化,即构成动态环境.本实验的动态环境是这样构建的:定义一个与样本同等长度的二进制字符串(本文称之为模板),所有个体与模板进行位异或运算(即相同则为0,相异则为1).可见:如果模板中某一位为0,则个体基因表达式中对应的基因位值保持不变;而如果模板中的某一位为1,则个体基因表达式中对应的基因位值将取反.因此,模板中其值为1的位在模板中所占的比率可以用来表征环境变化的强度.

3.1.2 实验结果与分析

本实验中样本串长设为120,因此最大的位匹配数是120.3种算法都迭代250次,每隔50次随机产生一个新的模板,且所有个体都与新的模板进行异或运算,即环境总共经历5次振荡.用*CR*表示模板中值为1的位所占的比率,即环境变化的强度,图1给出了3种算法在*CR*分别等于0.1,0.5和0.9时的搜索性能比较结果.



(a) *CR*=0.1 (b) *CR*=0.5 (c) *CR*=0.9

Fig.1 Test result of DBM

图1 对DBM问题的测试结果

从图 1 可以看出,NDSLA 在每个环境变化的周期内都是积极主动地进行搜索,基本上没有出现停滞现象,并且在每个变化周期内都搜索到了全局极值点或附近;相比之下,SGA 和 PDGA 在每个环境变化周期内基本上处于停滞状态,其搜索性能令人失望.尤其是 SGA,在不同的环境变化强度下却有着同样偏低的搜索能力.大多数情况下,SGA 的性能是随着环境的变化而每况愈下的.PDGA 的搜索性能略优于 SGA,在每一次因环境变化,其适应度下降后基本上都能重新搜索到较好的解,这是由于其原对偶映射策略发挥了作用.可见,NDSLA 的动态环境适应能力以及在动态环境下的搜索能力远远超过 SGA 和 PDGA.

需要说明的是,以上 3 种算法在环境发生变化之后,即每个新周期开始时,并没有重新初始化以产生新的个体和种群,也即算法并没有重启(下面两个实验中也没有重启).这不同于当前大多数研究动态算法的做法,但这也正是本文研究的核心目的之一,即算法在运行过程中如何及时、准确地侦测到环境的变化而跟踪极值点在搜索空间内的运动轨迹.如果算法重启,固然能够提供一个新的发散的种群,有利于搜索到最优解,但这也正如本文在前言部分所指出的那样,环境改变之后算法重启,相当于在新的环境条件基础上从头开始求解.这实际上是开始了一个新的优化问题.一方面,这将会导致无法重用过去的有用信息,即无法重用之前种群积累的关于外部环境的知识,会因此浪费大量计算时间;另一方面,与 NDSLA 相比,SGA 和 PDGA 的搜索性能显得较差,与本文设置的周期长度有关.每种算法都只迭代 250 次,经历 5 个周期,每个周期长度也只有 50 次.因此,对于一个相对较大的搜索空间来讲,SGA 和 PDGA 的搜索能力无法发挥出来;但 NDSLA 却能在规定的周期内搜索到全局极大值或附近,且从图 1 中可以看出,其适应度在每个周期内一直都处于上升态势,如果再多迭代几次就一定能搜索到全局最优解.这表明主动学习的搜索策略确实能够更好地与环境进行友好交互,从而快速发现更好的可行解,比纯粹的随机搜索或单纯的被动适应性搜索更具优势.

3.2 时变背包问题(time-varying knapsack problem,简称TVKP)

3.2.1 问题定义

0-1 背包问题可以描述为:现有背包一只和物件若干,其中背包装载容量为定值,每一物件均有重量和价值两个属性,如何在不超出背包装载容量的前提下,使装入背包中物件的总价值最大.本文是这样定义 0-1 时变背包问题的,即在这若干物件中,某些物件可用与否是不定的,某些物件的价值大小是随着时间而改变的,还有背包的装载容量是动态变化的,以及这 3 种情况的交叉情形.本文主要研究第 3 种情况,即背包的装载容量发生变化的情形.本实验主要研究有 100 个物件的动态 0-1 背包问题,背包装载容量的变化分两种情况:一是在两个数值之间变化;二是在两个以上数值之间变化.当装载容量在多个数值之间变化时,背包问题就相当于一个振荡系统,背包装载容量变化的大小体现了环境振荡的强度.本实验中物件的重量和价值见表 1.

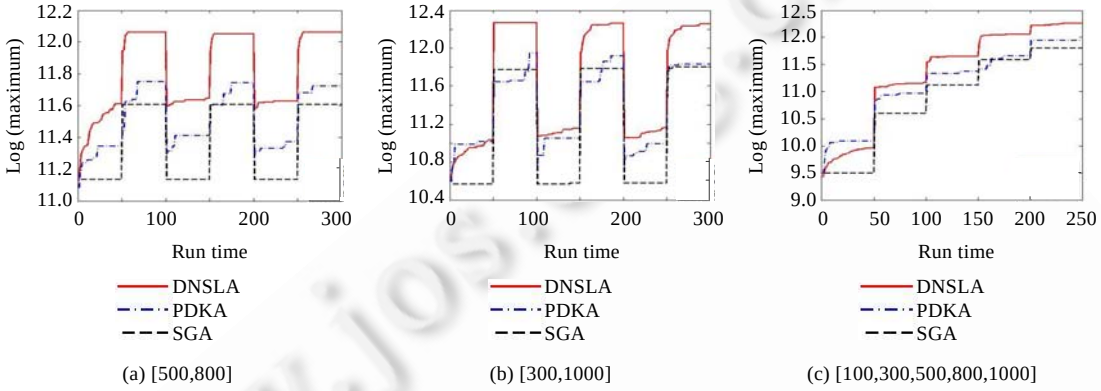
Table 1 Date used for TVKP

表 1 时变背包问题的相关数据

ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Weight	82	91	13	92	64	10	28	55	96	97	16	98	96	49	81	15	43
Cost	41	199	78	133	42	151	66	164	173	188	113	21	58	229	39	207	135
ID	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
Weight	92	80	96	66	4	85	94	68	76	75	40	66	18	71	4	28	5
Cost	250	20	111	27	241	2	194	205	218	22	100	65	201	108	228	46	66
ID	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
Weight	10	83	70	32	96	4	44	39	77	80	19	49	45	65	71	76	28
Cost	37	35	218	145	138	37	214	156	88	129	101	19	60	31	46	60	105
ID	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68
Weight	68	66	17	12	50	96	35	59	23	76	26	51	70	90	96	55	14
Cost	13	226	237	123	123	85	226	93	28	196	98	61	101	25	33	236	240
ID	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85
Weight	15	26	85	26	82	25	93	35	20	26	62	48	36	84	59	55	92
Cost	144	15	59	89	206	4	11	43	163	183	162	113	137	75	187	48	172
ID	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	–	–
Weight	29	76	76	39	57	8	6	54	78	94	13	57	47	2	34	–	–
Cost	46	93	157	196	21	233	194	122	109	112	77	128	128	205	199	–	–

3.2.2 实验结果与分析

3 种算法都是在运行时每隔 50 代,背包装载容量发生一次变化,下面是 3 种算法在不同振荡强度下搜索性能的对比情况.图 2 中的适应度曲线是原适应度值经过取对数之后绘制的,可以看出,当背包装载容量在两个不同的数值之间振荡时,NDSLA 的搜索性能总是大大优于 SGA 和 PDGA.尽管 PKGA 在第 50 代前搜索到的解比 NDSLA 要好,如图 2(b)所示,但 NDSLA 在这一周期结束时,其搜索的最终解仍然优于 PKGA.从图 2 中可以看出,NDSLA 在各个振荡周期内都是处于积极主动的搜索过程中,其搜索结果是连续的,且都在 50 代内搜索到了全局最优解.SGA 在环境发生振荡之后,在每个搜索周期内完全处于停滞状态,根本无法进行有效搜索.可以看出,PKGA 在环境发生振荡之后进行了一定程度上的有效搜索,使搜索到的最优解有所改善.从图 2 中可以看出,NDSLA 的最大特色是其在每个环境变化的同期内都是积极主动地进行搜索的,并且能够很快找到新环境条件下的全局最优解.



(a) [500,800] (b) [300,1000] (c) [100,300,500,800,1000]

Fig.2 Test result of TVKP

图 2 对 TVKP 问题的测试结果

3.3 移动峰函数优化问题(dynamic function optimization,简称DFO)

3.3.1 问题定义

本实验的动态函数(见表 2)是这样构造的:对于 $F1$,使其变量范围发生变化.这样,其全局最优解会随之发生漂移,其定义域变化的大小程度就表征了环境振荡的强度.现在分别测试其变量在±50 和±100 之间以及±100 和±500 变化的情况,然后测试其变量在±5,±50,±100,±500,±1000 这 5 个数量级中逐级变化的情况;对于后面 3 个函数,变量范围不变,但采用动态模板的方式来使进化过程中个体的基因表达式发生变化.

Table 2 Testbed functions

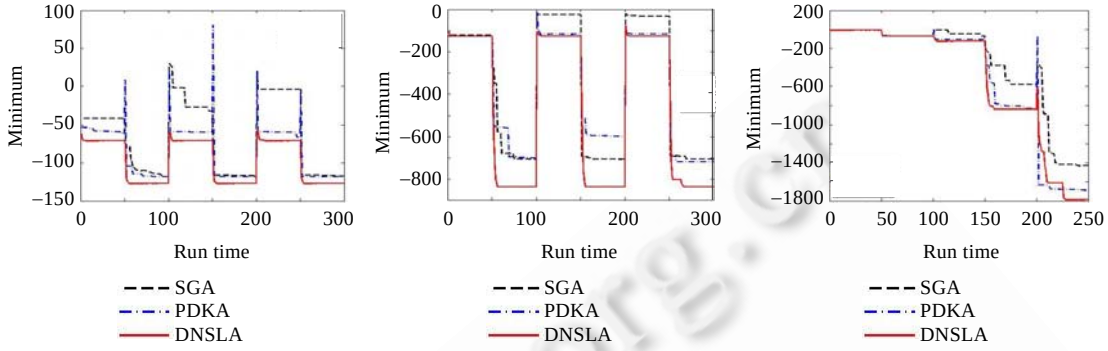
表 2 测试函数

	Function	x_i
$F1$	$\min f(x)=\sum_{i=1}^{2}\left[-x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)\right]$	x_i to be determined
$F2$	$\min f(x)=\sum_{i=1}^{2}[100(x_{i+1}-x_i^2)^2+(1-x_i)^2]$	$x_i\in[-2.048,2.048]$
$F3$	$\min f(x)=\sum_{i=1}^{2}[x_i^2-10\cos(2\pi x_i)+10]$	$x_i\in[-5.12,5.12]$
$F4$	$\min f(x)=\sum_{i=1}^{2}\frac{x_i^2}{4000}-\prod_{i=1}^{2}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	$x_i\in[-600,600]$

3.3.2 实验结果与分析

从图 3 中可以看出,SGA 和 PDGA 的搜索性能很不稳定,当变量的取值范围发生变化之后,往往无法搜索到

新的全局最优解,即无法跟踪最优解的运动轨迹.相比之下,NDSLA 的搜索能力要强得多,当侦测到环境变化之后,能够迅速作出搜索方向和搜索步长方面的调整,从而快速搜索到较好的候选解.从图中 NDSLA 的适应度曲线在变量取值范围发生变化后有一些小的弯曲现象中可以看出这一点,这表明 NDSLA 的搜索结果是持续改进并且能够快速定位的.难能可贵的是,在每一个新周期开始不久,NDSLA 就能迅速搜索到新环境中的全局最优值,且无论环境变化的强度大小如何,NDSLA 找到的最优解总是一致的.

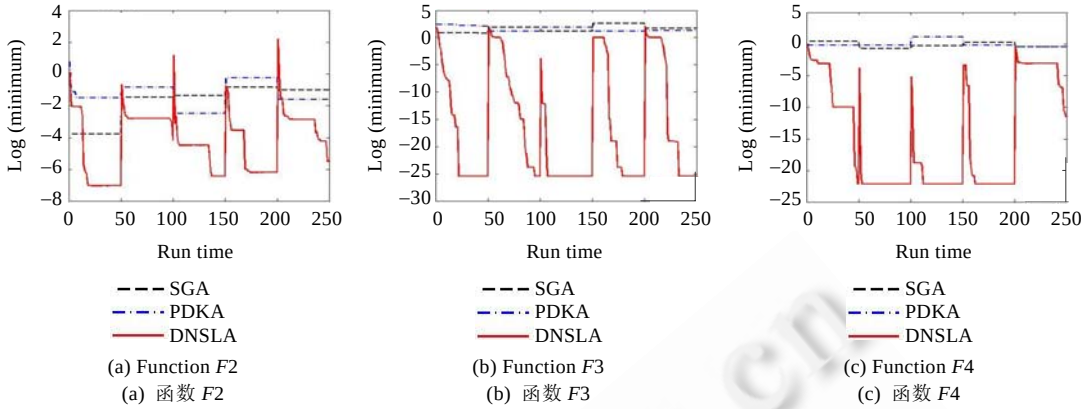


(a) [−50,+50], [−100,100] (b) [−100,100], [−500,500] (c) ±5, ±50, ±100, ±500, ±1000

Fig.3 Test result of $F1$

图 3 对函数 $F1$ 的测试结果

图 4 显示的是 SGA,PDGA 和 NDSLA 分别对 Rosenbrock 函数、Rastrigin 函数和 Griewank 函数在环境振荡强度为 0.9 时(即值为 1 的位在动态模板中所占的比率为 0.9)搜索性能的比较情况.因 3 种算法搜索到的适应度原始数据的绝对值差别不大,但对于较小的定义域和优化目的来讲,即使细微的差距也是相当重要的,因此,图 4 中的适应度曲线是原始数据经过取对数之后绘制的.这样,就将 3 种算法性能的优劣程度完全展示出来了.Rosenbrock 函数虽然在其定义域内只有 1 个全局最优值,但其全局最优点处于一个平滑而狭长的抛物线形峡谷之内,在缺乏启发性信息的情况很难被搜索到.可以看出,NDSLA 搜索到的结果大大优于 SGA 和 PDGA.需要说明的是,在图 4 的这 3 个子图中,SGA 和 PDGA 的数据是经过几次运行后挑选出来的较好的一次运行结果,而 NDSLA 的数据是算法第 1 次运行的结果.从 Rastrigin 函数和 Griewank 函数的优化结果中可以看出,SGA 和 PDGA 在环境变化强度极为剧烈的情况下,其适应度曲线并没有太大的起伏,而 NDSLA 的适应度曲线却发生了剧烈的起伏变化.这表明 NDSLA 对环境的变化极为敏感,并且能够与环境进行友好而透明的交互,从而能够迅速地搜索到新环境中的较好的解,并持续地改善.针对表 2 中 4 个待优化函数都是二维的情况,个体的二进制基因表达式长度被设为 50,即用 25 个比特位来代表一个变量.可以看出,同样长度的基因表达式,在采用双精度浮点数时(MATLAB7.6 默认的数据类型),SGA 和 PDGA 的搜索精度最多只到 0.1;而 NDSLA 的搜索精度却达到了小数点后 12 位,并且非常稳定(从图中可以看出,其在每个周期内搜索到的最优解是一致的).事实上,NDSLA 每次都能搜索到全局最优解,这个误差完全是由二进制数与双精度浮点数之间的转换所引起的.



(a) Function *F*2 (b) Function *F*3 (c) Function *F*4

(a) 函数 *F*2 (b) 函数 *F*3 (c) 函数 *F*4

Fig.4 Test result of *F*2, *F*3 and *F*4

图 4 对函数 *F*2,*F*3 和 *F*4 的测试结果

4 结论和展望

针对传统的智能进化算法由于其被动适应的搜索策略而无法应对动态问题求解的事实,本文提出的 DNSLA 实现了基于 0-1 编码的动态学习机制,种群中的个体由被动适应转为主动学习,实现了与环境的友好交互,从而具有更强健的动态环境适应能力,能够及时准确地侦测到周遭环境的变化并跟踪极值点在搜索空间内的运动轨迹.此外,本文还构造了一系列动态测试问题,在此基础上,经过 3 个动态测试问题的对比仿真实验表明,该算法在剧烈动荡的环境中的确能够很好地与环境进行稳定而友好的交互学习,表现出了很强的鲁棒性.

DNSLA 侧重于进化思想层面上的改造,即用主动学习的进化策略代替了传统的被动适应的进化思想.但这种主动学习的策略是建立在二进制编码基础上的,且实验部分的几个测试问题相对于实际应用中的问题还是比较简单的,其涉及的动态环境变量毕竟不多.因此,如何将自组织学习的思想用于其他编码情形以及当问题涉及到较多的动态环境变量时怎样与环境交互并组织学习,是我们下一步研究的课题.

致谢 在此,我们向本文的审稿专家表示衷心的感谢.为使本文能够顺利刊出,编辑部老师付出了辛勤劳动,这里一并表示诚挚的谢意!

References:

[1] Cobb HG. An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environment. Technical Report, 6760, Washington: Naval Research Laboratory, 1990.

[2] Morrison RW, de Jong KA. Triggered hypermutation revisited. In: Zalzala A, *et al*., eds. Proc. of the Congress on Evolutionary Computation. Piscataway: IEEE Service Center, 2000. 1025−1032. [doi: 10.1109/CEC.2000.870759]

[3] Vavak F, Fogarty TC, Jukes KA. A genetic algorithm with variable range of local search for tracking changing environments. In: Voigt HM, Rechenberg I, Schwefel HP, eds. Proc. of the Parallel Problem Solving From Nature IV. Berlin: Springer-Verlag, 1996. 376−385. [doi: 10.1007/3-540-61723-X_1002]

[4] Grefenstette JJ. Genetic algorithms for changing environments. In: Maenner R, Manderick B, eds. Proc. of the Parallel Problem Solving from Nature II. Elsevier Science Publishers, 1992. 137−144.

[5] Cobb HG, Grefenstette JJ. Genetic algorithms for tracking changing environments. In: Forrest S, ed. Proc. of the 5th Int'l Conf. on Genetic Algorithms. San Francisco: Morgan Kaufmann Publishers, 1993. 523−530.

[6] Goldberg DE, Smith RE. Nonstationary function optimization using genetic algorithms with dominance and diploidy. In: Grefenstette JJ, ed. Proc. of the Int'l Conf. on Genetic Algorithms. Hillsdale: Lawrence Erlbaum Associates, 1987. 59−68.

[7] Hadad BS, Eick CF. Supporting polyploidy in genetic algorithms using dominance vectors. In: Angeline PJ, *et al*., eds. Proc. of the 6th Int'l Conf. on Evolutionary Programming. San Francisco: Morgan Kaufmann Publishers, 1997. 223−234. [doi: 10.1007/BFb0014814]

[8] Ryan C. Diploidy without dominance. In: Angeline PJ, *et al*., eds. Proc. of the 3rd Nordic Workshop on Genetic Algorithms. 1997. 63−70.

[9] Branke J, Kaubler T, Schmidt C, Schmeck H. A multi-population approach to dynamic optimization problems. In: Proc. of the Adaptive Computing in Design and Manufacturing 2000. Berlin: Springer-Verlag, 2000. 299−308.

[10] Oppacher F, Wineberg M. The shifting balance genetic algorithm: Improving the GA in a dynamic environment. In: Banzhaf W, *et al*., eds. Proc. of the Genetic and Evolutionary Computation Conf. San Francisco: Morgan Kaufmann Publishers, 1999. 504−510.

[11] Wineberg M, Oppacher F. Enhancing the GA's ability to cope with dynamic environments. In: Whitley D, Goldberg D, Cantu-Paz E, Spector L, Parm I, eds. Proc. of the Genetic and Evolutionary Computation Conf. San Francisco: Morgan Kaufmann Publishers, 2000. 3−10.

[12] Ursem RK. Mutinational GA optimization technique in dynamic environments. In: Whitley D, Goldberg D, Cantu-Paz E, Spector L, Parm I, eds. Proc. of the Genetic Evolutionary Computation Conf. San Francisco: Morgan Kaufmann Publishers, 2000. 19−26.

[13] Yang S. The primal-dual genetic algorithm. In: Abraham A, Koppen M, Franke K, eds. Proc. of the 3rd Int'l Conf. on Hybrid Intelligent System. Amsterdam: IOS Press, 2003.

[14] Wang HF, Wang DW, Yang SX. Evolutionary algorithms in dynamic environments. Control and Decision, 2007,22(2):127−131 (in Chinese with English abstract).

附中文参考文献:

[14] 王洪峰,汪定伟,杨圣祥.动态环境中的进化算法.控制与决策,2007,22(2):127−131.

周传华(1968－),男,安徽含山人,博士,教授,主要研究领域为机器学习,Web 智能,信息安全.

谢安世(1983－),男,硕士,CCF 学生会员,主要研究领域为现代智能优化计算技术.