

加入捕猎算子和死亡算子的改进遗传算法研究

刘抒桥¹ 白俊卿¹

LIU Shuqiao BAI Junqing

摘要

针对经典遗传算法 (genetic algorithms, GA) 易陷入局部最优解和收敛速度慢的问题, 受生命演化过程的启发, 加入了捕猎和死亡两种算子对其进行改进 (predation and death genetic algorithms, PDGA)。首先, 在遗传算法的基础上, 划分出两个种群——捕食者和被捕食者, 通过捕食建立新的筛选机制, 增加收敛速度; 其次, 引入寿命机制来强制种群中个体的死亡, 增加跳出局部最优解的概率; 最后, 采用一种过度繁殖的方式来补充种群规模。为了验证改进算法的有效性, 选取 6 个基准函数进行仿真实验。实验结果表明, 与 GA 算法相比, PDGA 算法在寻优性能上有明显提升。

关键词

遗传算法; 捕猎算子; 死亡算子; 测试函数

doi: 10.3969/j.issn.1672-9528.2024.09.005

0 引言

生命演化是一种适应过程, 还是一种改良过程, 或者两者都不是, 但是模仿这种“生命游戏”的一些规则已经产生了强大的最优搜索算法^[1]。遗传算法 (genetic algorithms, GA) 是由美国的 Holland 教授^[2]于 1975 年在他的专著《自然界和人工系统的适应性》中首先提出的。遗传算法就是模仿了生物的遗传、进化原理, 并引用了随机统计理论而形成的, 它模拟了自然选择和自然遗传过程中发生的繁殖、杂交和突变现象。

随着研究的进行, 遗传算法慢慢显露出了问题, 如早熟收敛、收敛速度。协同演化是针对遗传算法的不足而兴起的一个研究热点, 最早由 Ehrlich 和 Raven 提出^[3]。协同演化理论认为某些物种的演化与另一些物种的演化相互关联、相互受益, 不同物种不同个体之间既有相互受益又有相互制约。

协同演化算法 (co-evolution algorithm, CEA) 通过构造两个或多个种群, 建立它们之间的竞争或合作关系, 多个种群通过相互作用来提高各自性能适应复杂系统的动态演化环境, 以达到种群优化的目的^[4]。该算法最早开始于 1990 年 Hillis^[5]的研究, 他将一种寄生虫和寄主协同演化的机制应用到优化搜索的改进中。1994 年, Paredis 引入生命周期适应度评价 (life-time fitness evaluation, LTFE) 的方法^[6]使 CEA 的健壮性更好。正是由于 CEA 考虑到进化的环境和个体之间的复杂联系对个体进化的影响, 对生命演化的模仿更加完善, CEA 表现结果更加令人满意。

本文所提出的捕食算子的核心思想与 CEA 一脉相承, 但实现方式有所不同, 并且本文在此基础上加入了死亡算子, 意在模拟生命过程中的寿命对个体的影响以及最后的死亡, 来探究此“消极”现象会对物种的演化带来何种影响, 以及采用了一种新的选择方式。以上这些改进策略都是为了更加接近生命演化的一种“模仿游戏”, 作者相信生命本身就蕴藏着最完美的问题解决方案。

1 遗传算法基础

1.1 编码策略

编码是把一个问题的可行解从其解空间转换到遗传算法所能处理的搜索空间的转换方法^[7]。在遗传算法编码方式的问题上, Holland 提出的二进制编码是遗传算法中最常用的一种编码方法。除此之外, 常见的还有格雷编码、实数编码、非数值编码等^[8]。

1.2 遗传算子

遗传算法的操作算子包括选择、交叉和变异三种基本形式, 构成了遗传算法强大搜索能力的核心, 是模拟自然选择和遗传过程中发生的繁殖、杂交和突变现象的主要载体^[7]。

1.3 选择算子

选择操作体现“适者生存”的原理, 通过适应度选择优质个体而抛弃劣质个体, 其主要作用是避免基因缺失, 提高全局收敛性和计算效率。

1.4 交叉算子

交叉能使个体之间的遗传物质进行交换, 从而产生更好的个体。

1. 西安石油大学计算机学院 陕西西安 710065

1.5 变异算子

变异能恢复个体失去的或未开发的遗传物质，以防止个体在形成最优解过程中过早收敛。

2 PDGA 算法

2.1 算法介绍

PDGA 算法是在 GA 算法的基础上加入了捕食算子和死亡算子。因为捕食具有随机性，每次捕猎结束存活的个体数量不确定，所以选择算子采用了一种过度繁殖的方式来补充因捕食和死亡而消失的种群个体数。本研究关注的重点是两种新算子以及新的选择方式，故交叉和变异都采用最简单的方式。

2.1.1 初始化

初始化时，创建两个种群狼群和羊群（捕食者和被捕食者）。狼群和羊群用同一个测试函数作为适应度函数，染色体选用浮点数编码。两个种群在各自所在种群中参与选择、交叉、变异，通过捕猎算子产生交互。初始化种群个数时，设置狼群数量小于羊群数量，可以保证每次捕食都有一定数量的羊群参与繁殖。

2.1.2 捕猎算子

模拟自然界中捕食者捕获猎物的过程，具体操作是：捕猎时遍历整个狼群，对每只狼随机选择一只羊，比较两者的适应度值，以求最小值为例，若狼的适应度小于羊的适应度，则视为此次该狼捕食羊成功，该狼在捕猎中存活，该羊死亡。值得注意的是，对每只狼赋予了一定的捕猎次数 chance，而该次数与死亡算子相关，即用死亡算子来控制狼的捕猎次数，当 chance 为 0 时，也代表着该狼的死亡。若在捕猎次数内，该狼都没有成功捕食到羊，则该狼在此次捕猎中死亡，该羊存活。若求最大值，则情况相反。

2.1.3 死亡算子

本文给狼群和羊群加入 lifestage 属性，模拟自然界中的寿命，寿命随着时间增减，个体能力也会有所改变，当寿命达到一定值时，个体死亡。lifestage 初始化为 1，每迭代一轮，lifestage 就增加 1。并且采用了一种较为温和的死亡方式，不是简单设定 lifestage 达到某一特定值个体直接死亡，而是在过程中采用了一种过渡的方式。将狼群的捕猎次数 chance 和 lifestage 反相关，随着 lifestage 的增加，chance 减小直到 0，失去捕食能力。lifestage 与 chance 满足如下关系式：

$$\text{lifestage} + \text{chance} = N \quad (1)$$

式中：N 为设定的一个大于 0 的自然数

将羊群的适应度值 fitness 与 lifestage 反相关，随着 lifestage 的增加，羊会更容易被捕食。适应度值与 lifestage 满足如下关系式：

$$\text{fitness} = \frac{\text{fitness}}{1 - 0.1 \times (\text{lifestage} - 1)} \quad (2)$$

需要注意的是，fitness 的缩放只是表现在捕猎过程中，并不是真正改变羊群的 fitness。

在捕猎结束时，遍历所有狼群和羊群，将 lifestage 达到设定值 N 的个体淘汰，存活的个体更新 lifestage，并且更具关系式（1）更新狼的捕猎次数，参与之后的选择。

2.1.4 选择算子

本文引入了一种新的选择方式，旨在模拟自然界“赢者通吃”的场景，同时也尽可能保证遗传物质的多样性，让适应度较低的个体也能参与进繁衍之中，具体做法如下。

将种群中 n 个个体按适应度值从低到高排序（若求最大值，则为从高到低），从最佳个体（第一个个体）依次往后交配，直到第 n 个。然后从第 2 个个体依次往后交配，直到第 n-1 个，在第 $\lfloor \frac{n}{2} \rfloor$ 个按此法遍历完后停止。选择示意图如图 1 所示。

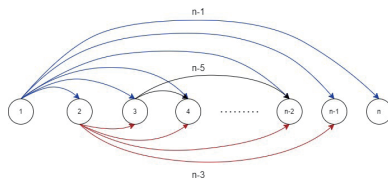


图 1 选择示意图

根据该流程，可以计算出新的种群数量：

$$\begin{cases} \text{sum} = \frac{n^2 - 1}{4} + n, & n \text{ 为奇数时} \\ \text{sum} = \frac{n^2}{4} + n, & n \text{ 为偶数时} \end{cases} \quad (3)$$

该选择方式的复杂度为 $O(n^2)$ ，而且保证在子代种群中的遗传物质按适应度值的优劣所占比例逐次减小。

2.1.5 交叉算子

由选择算子选择出的两个个体采用简单的线性插值公式：

$$\overrightarrow{X_C} = \overrightarrow{X_A} \cdot \theta + \overrightarrow{X_B} \cdot (1 - \theta) \quad (4)$$

式中： θ 为取值在 0 ~ 1 的一组随机数，作为插值权重； $\overrightarrow{X_A}$ 和 $\overrightarrow{X_B}$ 分别代表两个父代的染色体； $\overrightarrow{X_C}$ 为产生的新个体的染色体。该公式产生的新染色体体现出基因融合的思想。

2.1.6 变异算子

对于所有个体的染色体，以一定概率采用加入权重的均值归一化公式进行变异：

$$X_C = X_C + \frac{(X_C - \overline{X_C})}{\text{MAX} - \text{MIN}} \cdot \theta \quad (5)$$

式中： X_C 为变异染色体； $\overline{X_C}$ 为种群所有个体染色体的平均值；MAX 和 MIN 为染色体搜索范围的上下界； θ 为取值在 0 ~ 1 的一组随机数，作为权重。该公式产生的新染色体可以在较小的范围内改变值。

2.2 算法介绍

PDGA 算法的流程图如图 2 所示。

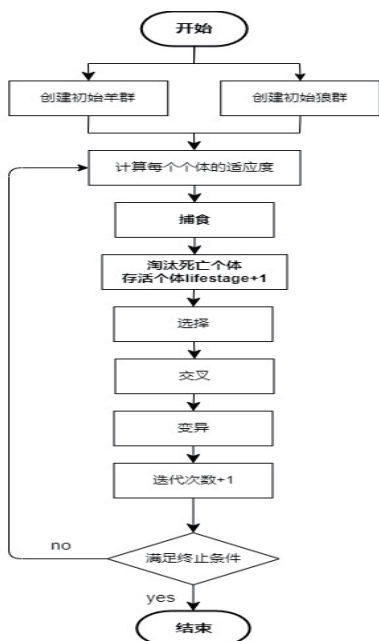


图 2 PDGA 算法流程图

3 实验与结果分析

3.1 实验环境与基准函数

所有仿真实验都在相同环境下进行：实验设备为一台安装 Windows 10 专业版，64 位操作系统的电脑，RAM 为 16 GB，处理器为 Intel(R) Core(TM) i3-10100F 3.60 GHz，测试软件为 MATLAB R2016a。

为了评估 PDGA 算法的收敛能力和跳出局部最优的能力，本文选取了 CEC2017^[9] 以及文献 [10-11] 中标准测试集中部分标准测试函数集中几个典型的优化函数。这组基准函数包括单峰函数和多峰函数：其中 $f_1 \sim f_2$ 为多维单峰函数， $f_3 \sim f_4$ 为多维多峰函数， $f_5 \sim f_6$ 为 2 维多峰函数，具体如表 1 所示。

表 1 基准函数

函数	函数名称	维度	搜索空间	$f_{\min}$
f_1	Sphere Function	d	$[-100,100]$	0
f_2	Zakharov Function	d	$[-5,10]$	0
f_3	Ackley's Function	d	$[-32,32]$	0
f_4	Rastrigin's Function	d	$[-5.12,5.12]$	0
f_5	Branin Function	2	$[-5,5]$	0.398
f_6	Holder Table	2	$[-5,10]$	-19.208

将改进的遗传算法 PDGA、经典遗传算法 GA 以及作为对照只加入捕食算子的遗传算法 PGA 放在一起进行实验对比。其中，这 3 个算法均采用相同的交叉、变异算子。

为了减少实验中随机因素造成的偏差，所有对比算法

在每个测试函数上均独立运行 30 次，用其寻优结果的平均值（AVG）和标准差（STD）来评估算法寻优性能和稳定性。所有优化算法均采用相同的公共参数进行测试。需要注意的是，狼群数量设定为 $0.8n$ ，种群数根据维度变化，GA 算法的种群数为其他算法种群数的 2 倍。各算法参数设置如表 2 所示。

表 2 各算法的参数设置

算法	参数设置
PDGA	$n=200/300/500$, $d=2/10/30$, $T=300$, cross=0.8, mutation=0.05, chance=3
PGA	$n=200/300/500$, $d=2/10/30$, $T=300$, cross=0.8, mutation=0.05
GA	$n=400/600/1000$, $d=2/10/30$, $T=300$, cross=0.8, mutation=0.05

3.2 实验结果及分析

表 3 ~ 5 为各算法对 6 个测试函数的优化结果。由实验结果可以看出，PDGA 算法和 PGA 算法的最优值以及收敛速度均比 GA 算法的表现优异，同时除了 $d=30$ 的 f_3 函数，PDGA 算法也比 GA 算法的 STD 更低，说明改进后的算法更加稳定。低维时，PDGA 算法能稳定收敛到理论最优值；在高维时，结果虽然比 GA 算法优异不少，但很难收敛到理论最优值。并且，与作为对照组的 PGA 算法相比，PDGA 算法实验结果差异不大（最优 AVG 适应度值已加粗表示），还有部分函数的实验结果不及 PGA 算法。

表 3 不同优化算法的测试结果 ($d=2$)

函数	PDGA		PGA		GA	
	AVG	STD	AVG	STD	AVG	STD
f_1	4.44e-19	1.87e-18	5.23e-13	2.51e-12	0.096	0.108
f_2	3.09e-19	1.60e-18	8.84e-12	4.77e-11	0.165	0.140
f_3	1.27e-07	5.36e-07	3.60e-05	1.30e-04	1.744	0.860
f_4	1.11e-04	5.80e-04	0.033	0.182	0.717	0.497
f_5	0.398	6.11e-11	0.398	1.82e-09	0.581	0.237
f_6	-19.194	0.071	-19.208	0.004	-19.180	0.096

表 4 不同优化算法的测试结果 ($d=10$)

函数	PDGA		PGA		GA	
	AVG	STD	AVG	STD	AVG	STD
f_1	30.294	20.659	33.689	27.557	6.77e+03	1.70e+03
f_2	0.309	0.267	0.284	0.248	90.503	24.586
f_3	2.997	0.735	3.520	0.809	11.040	0.950
f_4	10.180	2.707	11.022	3.255	70.005	8.081

表 5 不同优化算法的测试结果 ($d=30$)

函数	PDGA		PGA		GA	
	AVG	STD	AVG	STD	AVG	STD
f_1	1.82e+03	570.991	2.08e+03	507.032	4.87e+04	3513.929
f_2	35.570	9.504	37.041	11.251	983.446	134.195
f_3	8.865	0.807	9.565	0.689	15.522	0.553
f_4	94.206	15.918	92.476	15.495	343.472	19.185

为了更直观地观察对比各个算法的收敛速度、收敛精度以及算法跳出局部最优的能力,图3(a)~(d)给出了三种算法在部分函数10维时的平均收敛曲线图,其中横轴表示迭代次数,纵轴表示适应度。

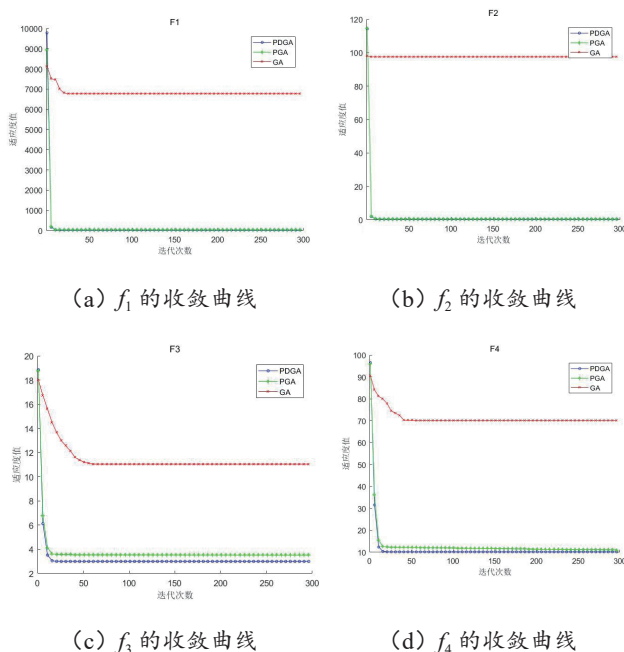


图3 各算法在测试函数上的收敛曲线

根据实验结果,可以得出结论:相较于GA算法,其改进算法PDGA的收敛速度更快,收敛精度更高,跳出局部最优的能力更强,鲁棒性更好。因此,加入两种新算子的改进方法是有效的。

4 存在问题与可能原因

(1) PDGA算法在高维度的实验结果表现不令人满意,可能是因为捕猎算子的设定只关注了这种方式,没有使用严谨的生物模型和数学公式来给予支撑。比如捕猎次数的设定以及chance和lifestage的关系定义,fitness缩放时的公式选取;选择算子和变异算子若采用其他更加成熟的算法,实验结果的精准度会大大提升;在选择操作时,由于捕猎的随机性,当剩余的父代个体过多,接近最大种群规模时,产生的新个体数量有限,会导致很多染色体被淘汰,“较优”个体的染色体会占比过大,使种群基因库的多样性遭到破坏。

(2) 加入死亡算子的PDGA算法与没有加入的PGA在实验结果上相差不大。这可能是因为自然界中的环境是时时变化的,且评价标准是多种因素叠加之后呈现的结果,若评价函数为动态适应度函数,结果可能会有改善;生命演化的评价标准或许不是达到最优,而是与环境达到平衡。

以上问题,可以在之后的研究中重点关注。

5 结语

本文提出了一种加入捕猎和死亡算子以及新的选择机制

来改进的遗传算法(PDGA),旨在解决遗传算法易发生早熟收敛、陷入局部最优的问题。捕猎算子让捕食者与被捕食者加入了“军备竞赛”,这种筛选压力加快了收敛速度。死亡算子的寿命机制通过影响捕猎过程来控制物种的死亡,一定程度上解决了“较优”个体支配种群基因库的问题,增加了算法跳出局部最优的能力。另外,采用了一种过度繁殖的选择方式来补充种群数量。选取了6个基准函数进行测试,实验结果表明,相较于遗传算法,PDGA算法在寻优性能上有明显提升。

参考文献:

- [1] RUDOLPH G. Convergence properties of evolutionary algorithms[M]. Hamburg: Verlag Dr. Kovač, 1997.
- [2] HOLLAND J H. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence[M]. 2nd ed. Cambridge: MIT Press, 1992.
- [3] EHRlich P R, RAVEN P H. Butterflies and plants: a study in coevolution[J]. Evolution, 1964, 18(4): 586-608.
- [4] 董红斌, 黄厚宽, 印桂生, 等. 协同演化算法研究进展[J]. 计算机研究与发展, 2008(3): 454-463.
- [5] HILLIS D. Co-evolving parasites improves simulated evolution as an optimization procedure[J]. Physica D, 1990, 42(1): 228-234.
- [6] PAREDIS J. Coevolutionary computation[J]. Artificial life, 1995, 2(4): 355-375.
- [7] 马永杰, 云文霞. 遗传算法研究进展[J]. 计算机应用研究, 2012, 29(4): 1201-1206+1210.
- [8] 葛继科, 邱玉辉, 吴春明, 等. 遗传算法研究综述[J]. 计算机应用研究, 2008(10): 2911-2916.
- [9] AWAD N H, ALI M Z, SUGANTHAN P N. Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective real-parameter numerical optimization [R]. Singapore: Nanyang Technological University, 2016.
- [10] DIGALAKIS J G, MARGARITIS K G. On benchmarking functions for genetic algorithms[J]. International journal of computer mathematics, 2001, 77(4): 481-506.
- [11] YAO X, LIU Y, LIN G. Evolutionary programming made faster[J]. IEEE transactions on evolutionary computation, 1999, 3(2): 82-102.

【作者简介】

刘抒桥(1996—),男,陕西汉中,人,硕士研究生,研究方向:智能算法、路径规划。

白俊卿(1983—),女,河南商丘,人,博士,副教授,硕士生导师,研究方向:计算机视觉、人工智能、数字电路设计及FPGA应用。

(收稿日期:2024-06-05)