带淘汰制的自适应联盟竞赛算法*

卢协平,刘秉瀚
(福州大学 数学与计算机科学学院,福建 福州 350108)

摘 要:提出一种带淘汰制的自适应参数调整方法。实验结果表明,改进后的算法在计算效果上有一定程度的提高,同时解决了参数设置问题。

关键词:联盟竞赛算法;鲁棒性;全局搜索;自适应参数调整;淘汰制
中图分类号:TP301 **文献标识码**:A **文章编号**:1674-7720(2013)15-0082-04

Adaptive league championship algorithm with knockout system

Lu Xieping, Liu Binghan
(College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China)

Abstract: This paper proposes an adaptive parameter adjustment method with knockout system. Experimental results show that the improved algorithm has much better global search capability and solves the problem of parameter setting.

Key words: league championship algorithm; robustness; global search; adaptive parameter adjustment; knockout system

联盟竞赛算法 LCA[1](League Championship Algorithm)是一种基于迭代的群体智能优化算法。

参考文献[1]中的实验结果表明,联盟竞赛算法相比粒子群优化算法,计算效果和效率都有一定的优势,因此该算法具有较好的应用前景。目前国内外对该算法做的研究主要集中在算法的扩展上,参考文献[2]和参考文献[3]都是将算法扩展为能够解决约束优化问题的算法,参考文献[4]将算法扩展为解决组合优化问题的算法。

通过初步分析和实验证明,算法主要存在两个缺点:(1)需手动设置多个参数,且在算法的运行中参数是静态不变的,因此对每个优化问题,需尝试不同的参数组合分别运行算法以找到一个较好的结果;(2)算法鲁棒性较低,在优化不同函数时,全局搜索能力差距较大,大大降低算法的实用性。目前还没有文献解决这些问题。

针对以上两个缺点,本文提出一种带淘汰制的自适应联盟竞赛算法 ALCAKS(Adaptive League Championship Algorithm with Knockout System)。

1 联盟竞赛算法

1.1 联盟竞赛算法基本流程

(1)初始化团队数量、阵型、比赛轮数 S 等,令当前迭代次数 $t=0$;

(2)产生一个赛季的比赛赛程;

(3)将最优适应值的团队阵型保存到 $\hat{X}^t$;

(4)根据第 t 轮的比赛赛程进行比赛,并按照一定规则判断比赛输赢;

(5)$t=t+1$,若 $t>S$,执行步骤(9);

(6)更新团队阵型:每个团队产生一个新阵型,若新阵型适应值比当前阵型适应值更优,则替换当前阵型;

(7)判断是否要产生新赛程,若是则根据某种规则产生新赛程并替换旧赛程,否则继续使用旧赛程;

(8)转步骤(3);

(9)输出全局最优解 $\hat{X}^s$。

1.2 联盟竞赛算法主要步骤

设 L 为参赛团队数量,S 为最大比赛轮数,t 为迭代轮数,n 为团队的成员数(维数)。$X_i^t=(x_{i1}^t,x_{i2}^t,\cdots,x_{in}^t)$ 为团队 i 在第 t 轮比赛的团队阵型,$f(X_i^t)$ 为团队 i 在第 t 轮比赛的适应值。

1.2.1 产生比赛赛程

常见的单循环赛制方法是逆时针轮转法。设有 8 个团队,团队编号从 1 开始。一个赛季有 7 轮比赛,赛程如图 1 所示。

*基金项目:福建省科技计划重点项目(2011Y0040);福建省自然科学基金项目(2012J01263)

$$(1)\begin{bmatrix}1:8\\2:7\\3:6\\4:5\end{bmatrix}(2)\begin{bmatrix}1:7\\8:6\\2:5\\3:4\end{bmatrix}(3)\begin{bmatrix}1:6\\7:5\\8:4\\2:3\end{bmatrix}\cdots(7)\begin{bmatrix}1:2\\3:8\\4:7\\5:6\end{bmatrix}$$

图 1 逆时针轮转法赛程的一个例子

LCA 允许在每个赛季后生成新赛程或使用旧赛程。

1.2.2 判断比赛输赢

假设团队 X_i^t 和 X_j^t 在第 t 轮比赛中是对手，适应值分别是 $f(X_i^t)$ 和 $f(X_j^t)$。令 p_i^t 是团队 i 赢得这场比赛的概率，则 p_i^t 满足式(1)：

$$p_i^t=\frac{f(X_j^t)-f(\hat{X}^t)}{f(X_j^t)+f(X_i^t)-2f(\hat{X}^t)} \tag{1}$$

若 $p_i^t\geqslant r_0$，则团队 i 赢、j 输；否则团队 j 赢、i 输。r_0 为 $[0,1]$ 区间随机数。

1.2.3 更新团队阵型

LCA 使用 SWTO 分析法对团队阵型进行更新，该方法包括内部优势、弱势和外部机会、威胁。设第 t 轮比赛中，团队 i 和 j 是对手，团队 l 和 k 是对手；第 $t+1$ 轮比赛，团队 i 和 l 是对手。则根据团队 i 和 l 在第 t 轮比赛的输赢，分为四种情况对 $X_i^t=(x_{i1}^t,x_{i2}^t,\cdots,x_{in}^t)$ 更新：

i 和 l 赢：$x_{id}^{t+1}=x_{id}^t+y_{id}^t(c_1r_1(x_{id}^t-x_{kd}^t)+c_1r_2(x_{id}^t-x_{jd}^t))$ (2)

i 赢 l 输：$x_{id}^{t+1}=x_{id}^t+y_{id}^t(c_2r_1(x_{kd}^t-x_{id}^t)+c_1r_2(x_{id}^t-x_{jd}^t))$ (3)

i 输 l 赢：$x_{id}^{t+1}=x_{id}^t+y_{id}^t(c_1r_1(x_{id}^t-x_{kd}^t)+c_2r_2(x_{jd}^t-x_{id}^t))$ (4)

i 和 l 输：$x_{id}^{t+1}=x_{id}^t+y_{id}^t(c_2r_1(x_{kd}^t-x_{id}^t)+c_2r_2(x_{jd}^t-x_{id}^t))$ (5)

其中 $d=1,\cdots,n$ 表示维数，c_1 和 c_2 是权重系数，r_1 和 r_2 是随机数，c_1、c_2、r_1 和 $r_2\in[0,1]$。y_{id}^t 是二进制数，为 1 则表示允许第 d 维值更新，否则不更新。$Y_i^t=(y_{i1}^t,y_{i2}^t,\cdots,y_{in}^t)$ 为 n 维二进制向量，q_i^t 表示团队 i 在第轮比赛后团队队员更新幅度，公式如下：

$$q_i^t=\sum_{j=1}^{n}y_{ij}^t \tag{6}$$

LCA 中使用截断几何分布[5]来定义 q_i^t 的值，公式如下：

$$q_i^t=\left\lceil\frac{\ln(1-(1-(1-p_c)^n)r_3)}{\ln(1-p_c)}\right\rceil:q_i^t\in\{1,2,\cdots,n\} \tag{7}$$

其中，r_3 是 $[0,1]$ 区间的随机数，$p_c\in(0,1)$ 是输入参数。

综上所述，算法要求手动设置 p_c、c_1 和 c_2 三个参数，其在运行中是静态不变的。对每个优化问题，需设置不同的参数组合分别运行算法以找到一个较好的结果。此外，本文通过实验（实验结果见 2.2 节）发现算法鲁棒性较低。针对这些问题，本文提出一种带淘汰制的自适应参数调整方法。

2 带淘汰制的自适应联盟竞赛算法

为下文提出自适应参数调整方法和引入淘汰制提供基础，本文先给出一种检测收敛过慢和陷入局部极值的方法。

2.1 检测收敛速度

为检测收敛速度，在算法中增加变量数组 Gbest，Gbest[i]保存算法迭代到第 i 轮时的最好适应值。对于求解函数最小值时，任意 $i(0\leqslant i\leqslant S-2)$ 都满足 Gbest[$i+1$]$\leqslant$Gbest[i]。当算法在第 i 轮（$i\geqslant w$）出现|Gbest[i]−Gbest[$i-w$]|小于某阈值时，认为算法收敛过慢或陷入局部极值，其中 w 是窗口大小。为区分收敛过慢与陷入局部极值，再增加一个变量数组 Tbest，Tbest[i]保存算法在第 i 轮中产生的新团队阵型中的最好适应值。如图 2 和图 3 所示，给出算法在优化 Rosenbrock 函数时出现的收敛过慢及陷入局部极值这两种结果，每种结果随着比赛轮数 i 的增加，Gbest[i]和 Tbest[i]的变化曲线。

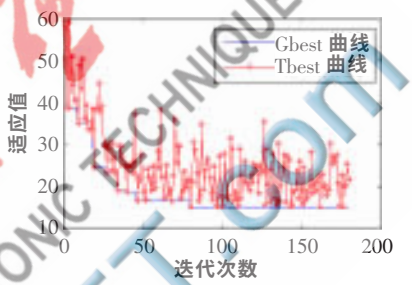

图 2 收敛过慢时 Gbest 和 Tbest 曲线

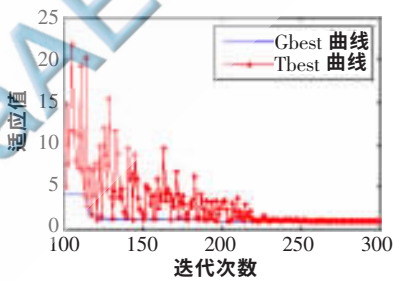

图 3 陷入局部极值时 Gbest 和 Tbest 曲线

在正常收敛速度下 Gbest 曲线稳步下降，Tbest 曲线波动较大；从图 2 看出，当算法收敛过慢时，如在第 80 轮~180 轮迭代期间，Gbest 曲线无变化，但 Tbest 曲线波动较大；从图 3 看出，当算法陷入局部极值时，Gbest 曲线和 Tbest 曲线变化幅度都较小，两条曲线最终重叠一起。

本文给出收敛速度的检测方法，如式(8)所示：

$$\begin{cases}|\text{Gbest}[t]-\text{Gbest}[t-w]|>\lambda_s, & \text{收敛正常}\\ |\text{Gbest}[t]-\text{Gbest}[t-w]|\leqslant\lambda_s\ \&\ tc\leqslant w/2, & \text{收敛过慢}\\ |\text{Gbest}[t]-\text{Gbest}[t-w]|\leqslant\lambda_s\ \&\ tc>w/2, & \text{局部极值}\end{cases} \tag{8}$$

$$tc=\sum_{k=t-w}^{t-1}\text{fun}(\text{Tbest}[k+1],\text{Tbest}[k]) \tag{9}$$

$$\text{fun}(x,y)=\begin{cases}1,\text{if}|x-y|\leqslant\lambda_s\\0,\text{其他}\end{cases} \tag{10}$$

其中，w 是窗口大小，λ_s 为波动阈值。

2.2 自适应参数调整

由式(2)~式(7)可知，c_1 和 c_2 可调整团队阵型成员

的更新幅度，所以 c_1 和 c_2 值越大，更新的幅度越大，导致更新速度也会越快。式中变量 y_{id}^t 的值受 q_i^t 影响，根据式(7)，p_c 值越小，q_i^t 值越大，导致每个团队阵型可能被更新的成员越多。

现通过实验证明，设团队数量 $L=10$、比赛轮数 $S=9\ 000$、选择 5 个测试函数(见第 3 节)维数均设为 5，当搜索到的最优值小于 1E-06 时，认为收敛到全局最优算法停止，或算法迭代次数大于 S 时停止。采用 12 种不同参数组合对测试函数试验 100 次，实验结果如表 1 所示，列出每种参数组合下 100 次试验中收敛到全局最优的百分比。其中 $c_1=c_2=C$。

表 1 收敛到全局最优值的比例

p_c	C	Sphere/%	Rosenbrock/%	Rostrigin/%	Griewank/%	Ackley/%
0.01	1	100	89	79	90	55
0.01	0.5	100	88	70	56	98
0.01	0.1	100	91	0	83	62
0.1	1	100	93	79	88	47
0.1	0.5	100	93	82	60	99
0.1	0.1	100	95	0	77	61
0.5	1	100	96	97	98	75
0.5	0.5	100	90	92	97	99
0.5	0.1	100	93	0	61	63
0.9	1	100	0	99	100	90
0.9	0.5	100	0	96	100	98
0.9	0.1	100	0	0	58	58

根据上述的实验结果和分析得出：

(1)固定 p_c 值，c_1 和 c_2 值越大，收敛速度越快。当 c_1 和 c_2 值过小时，会导致收敛过慢而最终无法找到全局最优值。

(2)固定 c_1 和 c_2 值，p_c 值越小，收敛速度越快，全局寻优能力会稍微变弱。

根据以上观点，本文提出自适应参数调整方案如下：

(1)算法初始时，设置较大参数值，较大 p_c 值使算法专注于全局搜索，保持团队阵型多样性，较大 c_1、c_2 值使算法收敛速度不易过慢(本文实验设 $p_c=0.8$，$c_1=c_2=1$)；

(2)当算法收敛过慢时，减小 p_c 以加快收敛速度；当算法陷入局部极值时，减小 c_1 和 c_2 值以减慢收敛速度，保持团队阵型多样性(本文实验 $p_c=p_c\times0.8$、$c_1=c_1\times0.8$ 和 $c_2=c_2\times0.8$)。

2.3 淘汰机制

由表 1 可见，算法设置同一种参数组合时，针对不同优化问题，算法收敛到全局最优的比例差别较大，如当 p_c 为 0.01，c_1 和 c_2 为 1 时，优化 Ackley 时收敛到全局最优的比例仅 55%，而其他函数收敛到全局最优的比例都较高；当算法优化同一函数时，设置不同的参数组合收敛到全局最优的比例差距也较大，如当优化 Ackley 时，收敛到全局最优的比例最高有 99%，最低仅 47%。由此可见算法的鲁棒性较低。

为提高算法的计算效果，本文将淘汰机制引入算法。在体育团队联赛中一般有淘汰机制，如从 32 强到 16 强，这个过程必须淘汰一半的团队。本文借鉴淘汰赛思想提出淘汰策略：将所有团队按照团队适应值从优到差进行排序，选择排在前面的一半团队进行重新随机初始化。

首先算法陷入局部极值很可能是由最优适应值的团队已经陷入局部极值，且较差适应值的团队向较优适应值团队靠拢导致，所以只有将较优适应值团队淘汰后才能更好地跳出局部极值；其次为了保持参赛团队总数不变，直接对要淘汰的团队进行初始化，可解释为加入新团队参与比赛。

如果算法每轮迭代都使用该策略，将导致算法收敛过慢或无法收敛。因此本文提出基于收敛速度的淘汰机制：

(1)初始化开关变量 Kswitch=false，值为 true 表示淘汰状态开启，该状态下算法每完成一个赛季都会执行淘汰策略；值为 false 表示淘汰状态关闭，不执行淘汰策略；

(2)当发现算法陷入局部极值时，令 Kswitch=true；当发现算法收敛过慢时，令 Kswitch=false；

2.4 带淘汰制的自适应联盟竞赛算法流程

综上所述，改进的联盟竞赛算法具体步骤如下：

(1)初始化团队数量、阵型、最大比赛轮数 S、窗口大小 w、Gbest、Tbest、阈值 λ_s 等；Kswitch=false、t=0；

(2)产生一个赛季的比赛赛程；

(3)将最优适应值的团队阵型保存到 $\hat{X}^t$；

(4)根据第 t 轮的赛程进行比赛，并判断比赛输赢；

(5)$t=t+1$，若 $t>S$，执行步骤(11)；

(6)更新团队阵型和 Gbest、Tbest；

(7)按式(8)检测算法是否收敛过慢或陷入局部极值；若算法收敛过慢，则按 2.2 节方案减小 P_c 值并令 Kswitch=false；若算法陷入局部极值，则按 2.2 节方案减小 C_1 和 C_2 值并令 Kswitch=true；

(8)若 Kswitch=true，调用 2.3 节的淘汰策略；

(9)判断是否要产生新赛程，若是则根据某种规则产生新赛程并替换旧赛程，否则继续使用旧赛程；

(10)转步骤(3)；

(11)输出全局最优解 $\hat{X}^s$。

3 实验

本文通过 5 个经典的函数优化问题来测试改进后的联盟竞赛算法和原始联盟竞赛算法的性能。测试函数包括：Sphere、Rosenbrock、Rostrigin、Griewank 和 Ackley，其自变量取值范围分别为：[-100，100]、[-2.048，2.048]、[-5.12，5.12]、[-32.76，32.76]、[-5.12，5.12]。

实验采用随机初始化团队阵型，函数值作为适应值，结束条件为最大迭代次数，由于 5 个测试函数的全局最优值都是 0，假设最优值<λ_s 时，认为算法收敛到全

局最优。设团队数量 $L=10$、迭代次数 $S=2\ 000$、函数维数 $n=5$、$\lambda_s=1E-06$，$w=40$，对 LCA 采用 12 种不同参数组合。

将 ALCAKS 和 LCA 的不同参数组合对每个测试函数分别试验 100 次。实验结果如表 2 和表 3 所示，对 LCA 只列出效果较好的 3 种参数组合。表 2 列出算法对每个测试函数在 100 次实验中收敛到全局最优的比例。表 3 列出算法对测试函数在 100 次实验中收敛到全局最优的最优值平均值。

表 2 收敛到全局最优值的比例

算法	p_c	c_1c_2	Sphere/%	Rosenbrock/%	Rostrigin/%	Griewank/%	Ackley/%
ALCAKS			100	100	100	100	97
LCA	0.5	1	100	96	93	99	72
LCA	0.9	1	100	0	99	100	53
LCA	0.01	0.5	100	85	8	51	94

表 3 收敛到全局最优值的平均值

算法	p_c	c_1c_2	Sphere	Rosenbrock	Rostrigin	Griewank	Ackley
ALCAKS			6.5e-115	1.2e-16	5.3e-17	5.0e-15	1.6e-8
LCA	0.5	1	3.2e-118	7.0e-09	0	2.4e-15	4.9e-9
LCA	0.9	1	2.8e-102	NAN	0	3.7e-15	8.5e-8
LCA	0.01	0.5	1.9e-039	3.1e-12	4.1e-08	3.0e-12	2.0e-8

从表 2、表 3 可以看出相比 LCA 算法，本文 ALCAKS 算法鲁棒性和全局搜索能力有一定程度的提高，因此本文提出的改进方案是可行的。

参考文献

[1] KASHAN A H. League championship algorithm: a new algorithm for numerical function optimization [C]. International Conference of Soft Computing and Pattern Recognition, 2009.

[2] KASHAN A H. A new algorithm for constrained optimization inspired by the sport league championships [C]. Evolutionary Computation(CEC), 2010.

[3] KASHAN A H. An efficient algorithm for constrained global optimization and application to mechanical engineering design: League championship algorithm (LCA)[J]. Computer-Aided Design, 2011, 43: 1769-1792.

[4] POURALI Z, AMINNAYERI M. A novel discrete league championship algorithm for minimizing earliness/tardiness penalties with distinct due dates and batch delivery consideration[C]. Advanced Intelligent Computing, 2012.

[5] KASHAN A H, KARIMI B, JOLAI F. Effective hybrid genetic algorithm for minimizing makespan on a single-batch-processing machine with non-identical job sizes [C]. Int J Prod Res, 2006.

（收稿日期：2013-04-10）

作者简介：

卢协平，男，1989 年，硕士研究生，主要研究方向：数据挖掘。

刘秉瀚，女，1963 年，硕士，教授，硕士生导师，主要研究方向：模式识别。