第 26 卷第 1 期
2009 年 1 月

控 制 理 论 与 应 用
Control Theory & Applications

Vol. 26 No. 1
Jan. 2009

文章编号: 1000−8152(2009)01−0039−07

具有轮盘反转算子的多Agent算法用于线性系统逼近

张俊岭, 梁昌勇, 杨善林

(合肥工业大学 计算机网络系统研究所, 安徽 合肥 230009)

摘要: 针对John Holland的反转算子在数值优化中的不合理性, 提出了一种轮盘反转算子来克服这种不合理性, 并结合该算子提出了一种多Agent进化算法(RAER), 证明了算法的全局收敛性. 无约束优化仿真实验表明, 该算法性能好于其他算法. 在求解线性系统逼近工程优化问题时, 无论在固定区域还是动态扩展区域搜索, 算法都能得到更好的模型, 较其他算法能够对搜索区域进行更为充分的探索和求精. RAER算法是实际有效的.

关键词: 多智能体; 无约束最优化; 线性系统逼近; 反转算子

中图分类号: TP18　　**文献标识码:** A

Effective multi-Agent algorithm with roulette inversion operator for approximating linear systems

ZHANG Jun-ling, LIANG Chang-yong, YANG Shan-lin

(Institute of Computer Networks, Hefei University of Technology, Hefei Anhui 230009, China)

Abstract: The irrationality of the inversion operator designed by John Holland is analyzed and revealed; and a new roulette inversion operator is proposed to cope with this problem. A new multi-agent evolutionary algorithm(RAER) is then developed by integrating the roulette inversion operator. Theoretical analysis shows that RAER converges to the global optimum. Four benchmark functions are used to test the performance of RAER, the results show that RAER achieves a better performance than other algorithms. RAER can be successfully used to solve linear system approximation problems in fixed search areas and dynamically expanded search areas. Especially, in the stable linear system approximation in several enlarged search areas, RAER can find the typical and optimal solutions in one specified area. This demonstrates the efficacy of RAER in practical applications.

Key words: multi-Agent; unconstrained optimization; approximation of linear system; inversion operator

1 引言(Introduction)

由复杂性研究刺激出来的多Agent系统已获得成功应用. 如文献[1]通过多Agent协作设计了一种智能控制系统; 文献[2]通过多Agent系统感受图像灰度的刺激来标识特征, 来抽取图像特征; 文献[3]创建了AER多智能体模型, 并利用该模型成功解决了7000皇后等问题; 文献[4]提出了一种多智能体遗传算法, 求解了复杂多峰标杆函数等.

本文分析了John Holland遗传算法理论中提出的反转算子在数值优化应用中的不合理性, 定义了一种新的轮盘反转算子(roulette inversion operator)来克服这种不合理性, 并结合轮盘反转算子与均匀设计实验方法设计了一种更加高效的多Agent进化算法(RAER), 无约束优化仿真试验和典型的线性系统逼近工程优化问题实验都表明, RAER是快速有效的, 优于其他算法, 具有较高的实用价值和应用潜力.

2 John Holland 反转算子与轮盘反转算子(John Holland's inversion operator and roulette inversion operator)

John Holland反转算子[5]有如下定义:

定义 1　在遗传算法中, 设W为种群中一个体, $W=(x_1,\cdots,x_l)$, l为编码长度, i_1与i_2为在$[1,l]$间随机整数, 如果i_1=i_2那么对W不做任何操作转入遗传算法主循环中的下一步; 如果$i_1>i_2$则$i_2\leftrightarrow i_1$, 使得$i_1<i_2$, 然后进行如下操作:

for　$k=i_1$　to　$i_1+\lceil(i_2-i_1+1)/2\rceil-1$,

收稿日期: 2007−06−19; 收修改稿日期: 2008−04−30.
基金项目: 国家自然科学基金资助项目(70631003, 70771037).

交换W编码中位置为k与i_1+i_2-k上的数值;

end

John Holland反转算子在数值优化中得到了广泛成功应用, 如文献[4, 6]将该算子与免疫方法、AER模型结合来解决高维函数优化问题, 文献[7]结合该算子来求解传统方法难以解决的线性系统逼近工程优化问题等. 在数值优化中, 反转算子通过与实验设计方法结合可以加强算法对搜索区域代表性点集的抽样, 帮助算法在较优个体周围找到可改进方向, 快速逼近最优解[4,6,7]. 在实数编码数值优化应用中, Holland反转算子常作以下变化定义[4]:

定义 2 Holland反转算子$^{\mathrm{Real}}$: 在遗传算法中, 设$W=(x_1,\cdots,x_l)$为实数编码种群中的一个体, $x_i \in [\overline{x}_i, \underline{x}_i]$, n为其编码长度, i_1与i_2为在$[1,n]$之间随机产生的整数, 如果$i_1=i_2$那么对W不做任何操作转入遗传算法主循环中的下一步操作; 如果$i_1>i_2$则$i_2 \leftrightarrow i_1$, 使得$i_1<i_2$, 然后进行如下操作:

1) 依据下式将W的每一维映射到$[0,1]$空间内, 得$W'=(e_1,e_2,\cdots,e_n)$.

$$e_k=\frac{x_k-\underline{x}_k}{\overline{x}_k-\underline{x}_k}, k=1,\cdots,n.$$

2) 对$W'=(e_1,e_2,\cdots,e_n)$进行反转操作, 得$W''=(e_1,e_2,\cdots,e_{i_1-1},e_{i_2},e_{i_2-1},\cdots,e_{i_1+1},e_{i_1},e_{i_2+1},\cdots,e_n)$.

3) 根据$x'_k=\underline{x}_k+e'_k\times(\overline{x}_k-\underline{x}_k), k=1,\cdots,n$,将$W''$映射回原始定义域空间, 得$W'''=(x'_1,x'_2,\cdots,x'_n)$.

对于形如定义2的John Holland反转算子, 本文先给出如下定理1.

定理 1 设$W=(x_1,x_2,\cdots,x_{i-1},x_i,\cdots,x_j,x_{j+1},\cdots,x_n)$为一实数编码的个体, n为要优化问题的维数; 设$P(X=i_1)$与$P(X=i_2)(i_1,i_2=1,\cdots,n)$表示在$[1,n]$间随机产生两个整数$i_1$和$i_2$的实验概率, 令$i_1,i_2$取$[1,n]$间整数是独立等概率抽样; 设$PI^H(Z=k)(k=1,\cdots,n)$表示$W$中的第$k$个位置上的变量$x_k$得到与$x_u(u\neq k)$进行反转交换的概率, 则有:

$$\begin{cases} PI^H(Z=k_1)=2\times[k_1/n-(k_1/n)^2], \\ k_1=1,\cdots,n/2, \qquad \text{当}n\text{是偶数}, \\ k_1=1,\cdots,(n+1)/2, \ \text{当}n\text{是奇数}, \\ PI^H(Z=n+1-k_1)=PI^H(Z=k_1). \end{cases} \tag{1}$$

证 略.

由定理1可知, 对于$W=(x_1,x_2,\cdots,x_{i-1},x_i,\cdots,x_t,\cdots,x_j,x_{j+1},\cdots,x_n)$, 当$t=n/2$($n$为偶数)时, $PI^H(Z=t)=PI^H(Z=t+1)$为最大, 当$u=t\to 1$和$u=(t+1)\to n$时,$PI^H(Z=u)$皆单调下降, $PI^H(Z=1)=PI^H(Z=n)$为最小; 当$t=(n+1)/2$(n为奇数), $PI^H(Z=t-1)=PI^H(Z=t)=PI^H(Z=t+1)$为最大, 当$u=(t-1)\to 1$和$u=(t+1)\to n$时, $PI^H(Z=u)$皆单调下降, $PI^H(Z=1)=PI^H(Z=n)$为最小. 可见算法应用Holland 反转算子在搜索过程中, 每一维得到探索的概率不同, 且随着n的增大编码边缘维方向得到探索的概率会变得很小, 方法显然是不合理的, 不利算法搜索效率的提高, 因此, 本文提出轮盘反转算子(RIO, roulette inversion operator)来克服这种不合理性.

定义 3 定义轮盘反转算子(roulette inversion operator)为: 设$W=(x_1,x_2,\cdots,x_{i-1},x_i,\cdots,x_j,x_{j+1},\cdots,x_n)$为一实数编码的个体, n为要优化问题的维数; 将这n个数顺序置于n等份的圆盘上, 顺时针转动轮盘先后随机产生两个整数a和b, 对W进行如下操作:

1) 依据下式将W的每一维映射到$[0,1]$空间内, 得$W'=(e_1,e_2,\cdots,e_n)$.

$$e_k=\frac{x_k-\underline{x}_k}{\overline{x}_k-\underline{x}_k}, \quad k=1,\cdots,n.$$

2) 对$W'=(e_1,e_2,\cdots,e_n)$依据以下规则进行反转操作得到W'':

当$a=b$时, 保持W的各分量位置不变;

当$a<b$时, 对$W'=(e_1,e_2,\cdots,e_{a-1},e_a,\cdots,e_b,e_{b+1},\cdots,e_n)$进行反转操作, 得到$W''=(e_1,e_2,\cdots,e_{a-1},e_b,e_{b-1},\cdots,e_{a+1},e_a,e_{b+1},\cdots,e_n)$;

当$a>b$时, 对$W'=(e_1,e_2,\cdots,e_{b-1},e_b,e_{b+1},\cdots,e_{a-1},e_a,e_{a+1},\cdots,e_n)$ 进行反转操作, 得到$W''=(e_{a+b-1},e_{a+b-2},\cdots,e_{a+1},e_a,e_{b+1},\cdots,e_{a-1},e_b,e_{b-1},\cdots,e_2,e_1,e_n,\cdots,e_{a+b})$ 或者$W''=(e_{a+b-n-1},\cdots,e_2,e_1,e_n,\cdots,e_{a+1},e_a,e_{b+1},\cdots,e_{a-1},e_b,e_{b-1},\cdots,e_{a+b-n})$.

3) 根据$x'_k=\underline{x}_k+e'_k\times(\overline{x}_k-\underline{x}_k), k=1,\cdots,n$,将$W''$映射回原始定义域空间, 得$W'''=(x'_1,x'_2,\cdots,x'_n)$.

对于轮盘反转算子有以下定理2.

定理 2 设$W=(x_1,x_2,\cdots,x_{i-1},x_i,\cdots,x_j,x_{j+1},\cdots,x_n)$为一实数编码的个体, n为要优化问题的维数; 对W进行轮盘反转操作, 设$P(X=a)$与$P(X=b)(a,b=1,\cdots,n)$表示转动轮盘随机产生两个整数a和b的实验概率; 设$PI^Z(Z=k)(k=1,\cdots,n)$表示W中的第k个位置上的变量x_k得到

与$x_u(u \neq k)$进行反转交换的概率, 则有:

$$PI^Z(Z=k)=\begin{cases}1/2\times[1-(1/n)^2], k=1,\cdots,n,\\ \quad 当n是奇数;\\ 1/2, k=1,\cdots,n,\\ \quad 当n是偶数.\end{cases} \tag{2}$$

证　略.

由定理1和定理2可知, 对于$W=(x_1,x_2,\cdots,x_{i-1},x_i,\cdots,x_k,\cdots,x_j,x_{j+1},\cdots,x_n), k=1,2,\cdots,n$, 当$t=n/2$($n$为偶数)时, $PI^H(Z=t)=PI^H(Z=t+1)=PI^Z(Z=k)$; 当$t=(n+1)/2$($n$为奇数), $PI^H(Z=t-1)=PI^H(Z=t)=PI^H(Z=t+1)=PI^Z(Z=k)$; 由定理2可知, 算法在应用轮盘反转算子进行随机搜索时, 每一维得到探索的概率相同, 不会因为其编码位置而有所降低, 克服了Holland反转算子的不合理性.

3　RAER算法及其收敛性分析(RAER algorithm and its convergence analysis)

3.1　RAER算法(RAER alogrithm)

本节结合轮盘反转算子从AER模型的Agents、竞争环境与交互规则3个组成部分[3]对RAER算法进行设计.

1) 竞争环境与Agents初始化.

这里本文引入文献[4]定义的$L_{\text{size}}\times L_{\text{size}}$二维连通Agents竞争环境, 其描述如下定义4.

定义 4　令$L_{\text{size}}\times L_{\text{size}}$二维连通Agents网格中位置$(i,j)$的智能体表示成$L_{ij}, i,j=1,2,\cdots,L_{\text{size}}$, L_{ij}的邻域为$Nbs_{ij}=\{L_{i_1,j},L_{i,j_1},L_{i_2,j},L_{i,j_2}\}$, 其中:

$$i_1=\begin{cases}i-1, & i\neq 1,\\ L_{\text{size}}, & i=1,\end{cases}, j_1=\begin{cases}j-1, & j\neq 1,\\ L_{\text{size}}, & j=1,\end{cases}$$

$$i_2=\begin{cases}i+1, & i\neq L_{\text{size}},\\ 1, & i=L_{\text{size}},\end{cases}, j_2=\begin{cases}j+1, & j\neq L_{\text{size}},\\ 1, & j=L_{\text{size}},\end{cases}$$

这样$\{Nbs_{ij},L_{ij}\}$就构成了L_{ij}的局部竞争环境.

$L_{\text{size}}\times L_{\text{size}}$二维连通Agents以下式初始化:

$$x_{ij}=\underline{x}_j+\text{rand}\times(\overline{x}_j-\underline{x}_j)\ \ j=1,\cdots,n, \tag{3}$$

其中rand为$[0,1]$间均匀分布的随机数.

2) 竞争交互规则.

在RAER算法中, 本文设计以下竞争交互规则: 竞争死亡、局部扩散再生、局部再扩张和传播.

R1) 竞争死亡: 在个体L_{ij}的$\{Nbs_{ij},L_{ij}\}$中, 如果L_{ij}的适应性度量最好, 那么L_{ij}竞争排除其余$\{Nbs_{ij},L_{ij}\}$中个体, 抢占资源.

R2) 局部扩散与再生: 假设W_1的局部竞争环境由$W_1\sim W_5$组成, 且W_1的适应性度量最好, W_1有能力向周围进行扩散, 占据$W_2\sim W_5$, 本文将局部扩散分为局部较近与较远距离扩散.

① 局部较近距离扩散　在竞争死亡规则下空出的格点$W_2\sim W_5$以较大概率在W_1局部较近范围内随机扰动生成, 即设$W_1=(x_{11},x_{12},\cdots,x_{1n})$, $W_1\in[\underline{X},\overline{X}]$, 则$W_3\sim W_5$由下两式产生:

$$W_i=\max(\min(W_1+\text{sgn}(\text{rand}-0.5)\times\text{rand}\times(W_1-W_i),\overline{X}),\underline{X})\ i=3,4,5; \tag{4}$$

$$W_i=\max(\min(W_1+\text{random}(-2qr_n,2qr_n),\overline{X}),\underline{X})\ i=3,4,5; \tag{5}$$

其中sgn(·)为符号函数, rand为[0,1]间均匀分布随机数, q为均匀表$U_m(q^n)$中水平数, r_n为小生境半径, 一般取$r_n=0.1$, $\text{random}(-2qr_n,2qr_n)$为$[-2qr_n,2qr_n]$间均匀分布随机数.

② 局部较远距离扩散　W_1有向局部区域较远距离进行扩散的可能, 在此利用轮盘反转算子对W_1进行操作且只填充格点W_2, 如下式所示:

$$W_2=RIO(W_1). \tag{6}$$

通过对较优个体W_1进行轮盘反转操作, 所得个体填充W_2, 那么在W_1和W_2构成的较少维空间中通过试验设计方法进行抽样, 便可在W_1的周围各维方向上进行概率均等的探索, 促使算法找到可能的寻优方向, 提高算法的随机搜索能力.

③ 再生结合均匀设计方法对W_1与新的$W_i(i=2,\cdots,5)$构成区域进行抽样, 所得较优个体替代W_1与W_i. 常用下式两方法生成均匀表$U_m(q^n)$.

$$\begin{cases}U^1_{i,j}=(i\sigma_{j-1}\bmod q)+1,\\ U^2_{i,j}=(ih_j-1)[\bmod m]+1.\end{cases} \tag{7}$$

对于W_1与$W_i(i=3,4,5)$构成的搜索区域可以作以下处理: 令$W_1=(x_{11},x_{12},\cdots,x_{1n})$, $W_i=(x_{i1},x_{i2},\cdots,x_{in}), i=3,4,5$, 则搜索区域$[l_p,u_p]$由下式确定.

$$\begin{cases}l_p=[\min(x_{1,1},x_{i,1}),\cdots,\min(x_{1,n},x_{i,n})],\\ u_p=[\max(x_{1,1},x_{i,1}),\cdots,\max(x_{1,n},x_{i,n})].\end{cases} \tag{8}$$

对由W_1与W_2构成搜索区域, 本文设计下列方法确定: 令$W=(x_1,\cdots,x_{i-1},x_i,\cdots,x_j,x_{j+1},\cdots,x_n)$为一实数编码的个体, 设选中子串$(x_i,\cdots,x_j)$进行反转操作, 其中$x'_k(k=1,\cdots,K,K=j-i+1)$得到与$x'_{K+1-k}$进行反转交换, $x'_k\in[\underline{x}'_k,\overline{x}'_k]$, $x'_{K+1-k}\in[\underline{x}'_{K+1-k},\overline{x}'_{K+1-k}]$, 令$d_k=(x'_k-\underline{x}'_k)/(\overline{x}'_k-\underline{x}'_k)$, $d_{K+1-k}=(x'_{K+1-k}-\underline{x}'_{K+1-k})/(\overline{x}'_{K+1-k}-\underline{x}'_{K+1-k})$, $d=|d_k-d_{K+1-k}|$,

则第x'_k维搜索区间为$[\underline{x}''_k, \overline{x}''_k]$, 其中

$$\begin{cases} \underline{x}''_k = \max((d_k - d) \times (\overline{x}'_k - \underline{x}'_k) + \underline{x}'_k, \underline{x}'_k), \\ \overline{x}''_k = \min((d_k + d) \times (\overline{x}'_k - \underline{x}'_k) + \underline{x}'_k, \overline{x}'_k), \end{cases} \tag{9}$$

第x'_{K+1-k}维搜索区间为$[\underline{x}''_{K+1-k}, \overline{x}''_{K+1-k}]$, 其中:

$$\begin{cases} \underline{x}''_{K+1-k} = \max((d_{K+1-k} - d) \times (\overline{x}'_{K+1-k} - \\ \qquad \underline{x}'_{K+1-k}) + \underline{x}'_{K+1-k}, \underline{x}'_{K+1-k}), \\ \overline{x}''_{K+1-k} = \min((d_{K+1-k} + d) \times (\overline{x}'_{K+1-k} - \\ \qquad \underline{x}'_{K+1-k}) + \underline{x}'_{K+1-k}, \overline{x}'_{K+1-k}). \end{cases} \tag{10}$$

R3) 局部再扩张: 本文采用微智能体网格$RLm\times n(m, n$取得较小, 一般取$m = n = 3)$来实现$\{Nbs_{ij}, L_{ij}\}$中最优个体的再扩张能力, 且设计了自适应网格生成策略: 设要进行再扩张的个体$W_i = (x_{i1}, x_{i2}, \cdots, x_{iu})$, $x_{ij} \in [\underline{x}_i, \overline{x}_i]$, $j = 1, \cdots, u$, 则RL中个体$RW_t = (x_{t1}, x_{t2}, \cdots, !x_{tu})$除$W_i$外由下式生成:

$$x_{tj} = \min(\max(x_{ij} \times \text{random}(1 - sR, 1 + sR), \underline{x}_j), \overline{x}_j), j = 1, \cdots, u, t = 1, \cdots, m \times n - 1. \tag{11}$$

如果$(x_{tj} - x_{ij}) > r_n$则$x_{tj} = x_{ij} + r_n$, 如果$(x_{tj} - x_{ij}) > -r_n$则$(x_{tj} = x_{ij} + r_n)$, 再作边界处理, 其中sR为扩张因子, 一般取sR=0.2, $\text{random}(1 - sR, 1 + sR)$为$[1 - sR, 1 + sR]$间的均匀随机数, r_n为扩张半径, 一般取r_n=0.1. 这样W_i与RW_t便构成微智能体网格RL. 在RL中局部较近距离扩散方法的式(5)由下列式(12)[8]代替:

$$x_{ik} = \begin{cases} \underline{x}_k + \text{rand}^1 \times (\overline{x}_k - \overline{x}_k), \text{rand} < 1/n; \\ x_{ik}, \text{others}; \quad k = 1, \cdots, n, i = 3, 4, 5. \end{cases} \tag{12}$$

R4) 传播: 通过式(3)生成RP×(M×N)个体来代替每代网格中适应性较差的个体, 并将所有个体随机分布到下一代网格中以维持网格多样性, 其中M, N为二维联通Agents网格行列值, RP是多样性控制系数, 一般取RP=0.05∼0.5, 本文取RP=0.1.

与RAER相似, MAGA[4]也是基于AER模型构建的进化算法. 在MAGA中, Holland反转算子被赋予较大执行概率(0.8), 其作用不仅在于利用较优个体自身信息, 也在于保持种群多样性, 相应的, 在MAGA中采用了高斯变异算子, 且变异概率较小. 而在RAER中, 以较小的概率(0.25)执行轮盘反转算子以有效利用较优个体信息, 提高算法的随机搜索能力, 规则R2)体现了这一点, 同时这也与反转算子在IMCPA[6]等算法中作用相同(反转算子执行概率都为0.3), 规则R4)旨在保持种群多样性以使算法更加稳定不易早熟.

综上所述, RAER的详细描述如下算法1.

算法 1 RAER多Agent进化算法

按照式(3)初始化$L_{M\times N}$; 设定局部较近距离扩散概率$P_{nd}, P^4_{nd}, P^5_{nd} = 1 - P^4_{nd}$, 局部较远距离扩散概率$P_{fd} = 1 - P_{nd}$; 选定均匀表$U_m(q^n)$; 设定局部抽样概率$P_s$; 进化周期计数$UGen = 1$; 算法终止信号量done = false;

while !done

MAX=[ ];

for L_{ij} in $L_{M\times N}$

if L_{ij} == MAX_{ij} do nothing! else

//MAX_{ij}是$\{Nbs_{ij}, L_{ij}\}$中适应性值最好个体;

if$U \leqslant P_{nd}$ // U为[0,1]间均匀随机数;

if$U \leqslant P^4_{nd}$ 对L_{ij}执行式(4); else对L_{ij}执行式(5);

end

else

对L_{ij}执行式(6);MAX=[MAX; L_{ij} MAX_{ij}];

end end end

for L_{ij} in $L_{M\times N}$

if L_{ij} == MAX_{ij} do nothing! else

if$U \leqslant Ps$

ifL_{ij} in $\text{MAX}(:, 1)$

对MAX中的$[L_{ij}$ $\text{MAX}_{ij}]$构成区域按照(9)(10)两式进行抽样, 得到适应值较好的个体代替L_{ij}与MAX_{ij}, 否则保持L_{ij}与MAX_{ij};

else

对L_{ij}与MAX_{ij}构成区域按照式(8)进行抽样, 得到适应值较好的个体代替L_{ij}与MAX_{ij}, 否则保持L_{ij}与MAX_{ij};

end end end end

对L中适应性值最好个体进行局部再扩张操作;

对$L_{M\times N}$进行传播操作;

检查信号量done;

$UGen = UGen + 1$;

end.

其中局部再扩张操作是通过式(11)生成微智能体网格$RL_{m\times n}$之进化来达到求精目的, 其主循环与算法1相同, 初始化设置为: 扰动复制概率RP_R(其中执行式(4)的概率为RP^4_R, 执行式(12)的概率RP^{12}_R=1 − RP^4_R), 轮盘反转操作概率RP_I=1 − RP_R, 局部抽样概率RP_s, 算法终止条件为进化代数大于$MGen$.

3.2 RAER算法的收敛性分析 (Convergence analysis of RAER)

设$S = [\underline{X}, \overline{X}]$为RAER算法要搜索的$n$维区域, 算法对$S$的抽样点集为$M$, 算法求解精度为$\varepsilon$, 将搜

索空间离散化为空间S', 即$|S'|=\prod_{i=1}^{n}\lceil(\overline{x}_i-\underline{x}_i)/\varepsilon\rceil$, 根据某种适应性度量可以将$S'$划分为集合$ST_j=\{\text{Fitness}(x)=E_j|x\in S'\}$, $E_j\in E=\{E_1,E_2,\cdots,E_{|E|}\}$, E为空间S'具有的所有适应性度量等级集合, 且$E_1>E_2>\cdots>E_{|E|}$, 显然E_1就是空间S'中最优解所属等级, 这样抽样点集M便是有等级的, 令M_t表示具有适应性度量等级t的点集M, $t=\min\{j|M\bigcap ST_i\neq\emptyset\}$, 令$P_{t,k}(M_t\to M_k,k>t)$与$P_{t,k}(M_t\to M_k,k<t)$分别表示点集M在算法的搜索策略作用下从高等级向低等级与从低等级向高等级转移的概率. 在RAER算法中采用了保优策略, 所以有$\forall i,t\in\{1,2,\cdots,|E|\}$, 有$P_{t,k}(M_t\to M_k,k>t)$=0, 且有如变异等策略使得$P_{t,k}(M_t\to M_k,k<t)>0$, 这样作为有限状态马尔可夫链的$M_t$其转移矩阵是一稳定随机矩阵, 且有$\lim\limits_{k\to\infty}P(M_k=M_1)$ =1, 故RAER算法是全局收敛的.

4　无约束优化仿真实验(Unconstrained optimization experiments)

本节实验旨在通过求解如下函数来测试RAER算法的性能, 并与其他优秀算法进行比较:

$$\text{F01}:\min f(x_i)=\sum_{i=1}^{N}(x_i-i)^2, x_i\in[-100,100].$$

$$\text{F02}:\max f(x_i)=\sum_{i=1}^{N}(-1)^{i+1}x_i^2, x_i\in[-100,100].$$

$$\text{F03}:\min f(x_i)=\sum_{i=1}^{N}(x_i)^4+\text{rand}(0,1), x_i\in[\pm1.28].$$

$$\text{F04}:\min f(x_i)=\sum_{i=1}^{N}(x_i)^2, x_i\in[-100,100].$$

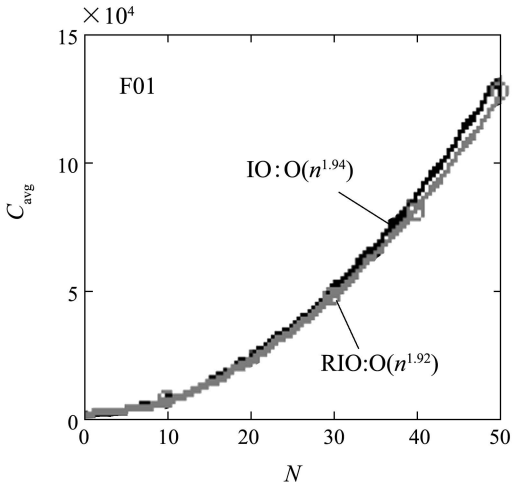
为了说明轮盘反转算子(RIO算子)在RAER算法中的作用和相对于John Holland反转算子(IO算子)优势, 这里将具有IO算子的RAER算法标记为RAER_IO, 并将之与具有RIO算子的RAER算法就优化困难的函数F01和F02 在维数为10∼50的情况下进行测试. 试验参数设置如下: 智能体网格L中M=N=10, P_{nd}=0.75, P_{nd}^4=0.5, P_s=0.2, RL中m=n=3,RP_R=0.2,RP_R^4=0.9,RP_s=0.1,*MGen*=6, 均匀表采用$U_1(5,4)$和$U^2(5,4)$组合使用. 每组实验随机独立运行20次, 两种算法所需平均函数评价次数(C_{avg})与取得函数值的标准差(std)列于表1中.

表 1　RAER与RAER_IO对于F01,F02的结果对比

Table 1　Results comparison between RAER and RAER_IO on F01and F02

N	F01				F02			
	RAER		RAER_IO		RAER		RAER_IO	
	std	C_{avg}	std	C_{avg}	std	C_{avg}	std	C_{avg}
10	1.8e−3	6131	1.4e−3	6157	2.4e−3	9595	2.5e−3	11967
20	7.2e−3	18870	8.9e−3	22446	1.7e−3	17483	1.9e−3	20403
30	3.5e−3	46327	7.3e−3	47485	2.8e−3	22636	1.9e−3	29505
40	1.9e−3	79920	3.3e−3	84478	1.7e−3	25108	1.9e−3	32628
50	2.4e−3	127452	2.8e−3	134629	1.7e−3	27272	1.9e−3	34379

为了更加清楚的表明RAER_IO和RAER的性能差别, 本文用O(Na)来逼近Cavg与维数N之间的关系, 所得两种算法不同逼近曲线如图1所示, RAER_IO算法对于F01&F02的逼近曲线为O(N1.94)和O(N0.58), 而RAER算法对于F01&F02的逼近曲线为O(N1.92)和O(N0.53), 可见, RAER算法对于F01&F02的复杂度均小于RAER_IO算法. 从表1和图1均可以看出RAER算法的性能好于RAER_IO算法, 亦即RIO算子的性能好于IO算子的性能, 且随着求解问题维数的增加, 优势越明显, 这与本文定理1与定理2所得结论相符.



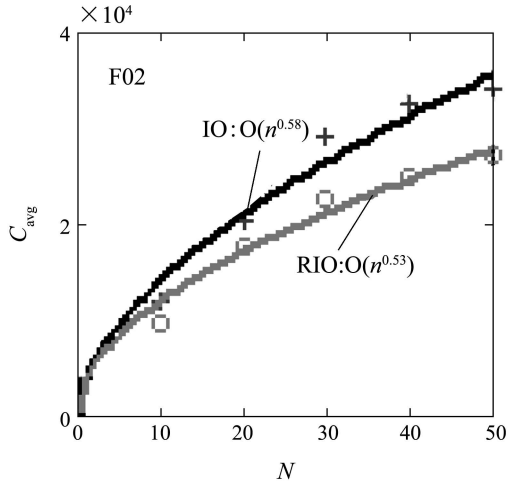


图 1 算法RAER和REAR_IO求解F01＆F02所需$C_{\rm avg}$与N的关系曲线

Fig. 1 The relationship between $C_{\rm avg}$ and N of RAER, REAR_IO on F01＆F02

表2对比了算法ARSAGA[9]、RAER_IO与RAER在N=30时对全部函数的优化性能. 从表2中可以看出RAER_IO与RAER的性能明显好于ARSAGA, 且解的优化质量较好. 尤其对于F04, 由于求解精度很高, 要求算法对每一维的逼近精度相应也很高, 由于RIO算子使得算法对每一维方向的探索概率均等克服了IO算子的不合理性, 所以在求解精度较高的情况下, RAER算法优势明显, 试验中, RAER只需RAER_IO计算量的约1/4就能达到e−15的精度.

表 2 在$N=30$时算法ARSAGA, RAER_IO和RAER对于F01∼F04的结果对比

Table 2 Results comparison between ARSAGA, RAER_IO and RAER on F01∼F04 when $N=30$

F	$C_{\rm avg}$			std		
	ARSAGA	RAER_IO	RAER	ARSAGA	RAER_IO	RAER
F01	363979	47485	46327	2.7e−2	7.3e−3	3.5e−3
F02	254541	29505	22636	6.6e−2	1.9e−3	2.8e−3
F03	15406	3396	2972	4.7e−3	1.1e−3	1.0e−3
F04	107380	49284	11953	0	0(e−15)	0(e−15)

5 RAER算法用于线性系统逼近 (Approximation experiments of linear systems)

本节应用RAER算法求解线性系统逼近问题来验证实际应用的有效性. 首先所要求解的稳定线性系统逼近问题来自文献[10,11], 需要确定的参数为$a_{2,1}$, $k_{2,p}$, $\tau_{2,z}$, $\tau_{2,d}$, 每个参数都位于$[0,+\infty)$上. 实验中, 分别在固定区域以及文献[10]采用的动态扩展区域运行算法. 在固定区域搜索时, 均匀表采用$U^1(11,4)$和$U^2(11,4)$来组合使用, 其余参数设置同第4节; 在动态扩展区域的搜索时, RAER算法的参数设置如下: 智能体网格L中$M=N=10, P_{nd}=0.9, P_{nd}^4=0.5, P_s=0.1, RL$中$m=n=3, RP_R=0.9, RP_R^4=0.9, RP_s=0.05$, *MGen*=8, 扩展周期$N_E$=8, 扩展参数为2, 均匀表为$U^1(5,4)$.

固定区域搜索时, 同文献[11], 令搜索区域为[0,10]^4, [0,50]^4, [0,100]^4, [0,150]^4, 算法终止条件设为$UGen$最大为200或者满足J <8e−5([0,10]^4), J <1e−5([0,50]^4) 和J<6.5e−6([0,100]^4, [0,150]^4); 动态扩展区域搜索时, 初始搜索区域设为[0,0.1]^4, 算法终止条件设为(J <6.5e−6 $\bigcup$ 函数评价次数>18000次). 每一类实验随机运行算法30次, 计算结果如表3, 与其他算法结果比较如表4.

表3中$J_{\rm min}$表示最优解的J值. 对比表3与表4中算法RAER, DEA和AIRA所得最优逼近模型可以看出无论在固定区域还是动态扩展区域搜索, RAER都能找到好于其他算法的逼近模型, 尤其从参数$\tau_{2,z}$的取值可以看出, 对固定区域搜索而言, 随着搜索区域的扩大, RAER都能在扩充的区域内找到相应该区域的典型解和最优解且优于其他算法, 说明RAER较之与其他算法能够对搜索区域进行更为充分的探索, 从而找到更好的模型; RAER算法在动态扩展区域搜索时所需$C_{\rm avg}$为16805, 也均小于其他算法. 此外, 应用本文算法求解非稳定线性系统逼近问题亦取得了较好效果, 限于篇幅, 这里不加赘述. 可见RAER算法优良的逼近性能, 优于其他算法.

表 3 RAER算法在不同搜索区域得到的稳定线性系统逼近模型参数

Table 3 Results of RAER for approximation of the stable linear system

搜索区域	解	$K_{2,p}$	$\tau_{2,z}$	$\tau_{2,d}$	$a_{2,1}$	$J_{\rm min}$ (标准差)
$[0,10]^4$	最好解	0.021	6.269	0.461	1.242	4.41e−05
	典型解	0.014	9.709	0.439	1.308	
$[0,50]^4$	最好解	0.004	34.837	0.348	1.273	6.76e−06
	典型解	0.007	20.080	0.363	1.262	
		0.004	36.912	0.344	1.271	
		0.003	49.247	0.333	1.256	
$[0,100]^4$	最好解	0.001	99.904	0.328	1.266	5.80e−06
	典型解	0.002	65.695	0.332	1.264	
$[0,150]^4$	最好解	0.001	148.325	0.325	1.267	5.74e−06
	典型解	0.001	113.660	0.328	1.269	
		0.001	139.683	0.322	1.262	
动态扩	最好解	0.002	61.579	0.333	1.267	5.97e−06
展区域	最差解	0.016	8.520	0.423	1.249	(3.87e−06)

表 4 不同算法逼近稳定线性系统的最优结果比较

Table 4 Comparative best results of different algorithms for approximation of the stable system

搜索区域	方法	$K_{2,p}$	$\tau_{2,z}$	$\tau_{2,d}$	$a_{2,1}$	$J_{\min}$	C_{avg}
$[0,10]^4$	DEA[10]	0.027	6.106	0.489	1.651	1.43e−04	
	AIRA[11]	0.021	6.130	0.454	1.226	4.49e−05	—
	RAER	0.021	6.269	0.461	1.242	4.41e−05	
$[0,50]^4$	DEA	0.005	35.051	0.391	1.761	6.26e−05	
	AIRA	0.006	20.836	0.360	1.255	8.23e−06	—
	RAER	0.004	34.837	0.348	1.273	6.76e−06	
$[0,100]^4$	DEA	0.004	65.911	0.591	2.714	1.00e−03	
	AIRA	0.003	39.803	0.332	1.251	8.52e−06	—
	RAER	0.001	99.904	0.328	1.266	5.81e−06	
$[0,150]^4$	DEA	0.003	68.171	0.379	1.761	6.11e−05	
	AIRA	0.006	21.479	0.357	1.257	7.74e−06	—
	RAER	0.001	148.325	0.325	1.267	5.74e−06	
动态扩展区域	DEA	0.003	68.171	0.379	1.761	6.11e−05	19800
	AIRA	0.005	28.823	0.351	1.265	6.55e−06	18015
	MAGA[6]	0.017	7.767	0.435	1.248	2.72e−05	19735
	RAER	0.002	61.579	0.333	1.267	5.97e−06	16805

6 结论(Conclusion)

本文RAER算法较其他算法能够对搜索区域进行更为充分的探索和求精, 是实际有效的, 提出的的轮盘反转算子, 克服了John Holland反转算子在数值优化中的不合理性. 进一步研究可使算法适应于非线性动态系统逼近、供应商选择等问题, 使得算法体现出更多的应用价值.

参考文献(References):

[1] 方义, 熊璋, 王剑昆. 智能控制中的多Agent系统[J]. 控制理论与应用, 2006, 23(5): 810 – 814.
(FANG Yi, XIONG Zhang, WANG Jiankun. Multi-agent system in intelligent control[J]. *Control Theory & Applications*, 2006, 23(5): 810 – 814.)

[2] LIU J, TANG Y Y. An evolutionary autonomous agents approach to image feature extraction[J]. *IEEE Transactions on Evolutionary Computation*, 1997, 1(2): 141 – 158.

[3] 韩靖, 蔡庆生. AER模型中的智能涌现[J]. 模式识别与人工智能, 2002, 15(2): 134 – 142.
(HAN Jing, CAI Qingsheng. Emergent intelligence in AER model[J]. *Pattern Recognition and Artificial Intelligence*, 2002, 15(2): 134 – 142.)

[4] ZHONG W C, LIU J, XUE M Z, et al. A multiagent genetic algorithm for global numerical optimization[J]. *IEEE Transactions on Systems, Man, and Cybernetics-part B*, 2004, 34(2): 1128 – 1141.

[5] HOLLAND J H. *Adaptation in Natural and Artificial Systems: an Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*[M]. Cambridge: MIT Press, 1992.

[6] DU H F, GONG M G, JIAO L C, et al. A novel algorithm of artificial immune system for high-dimensional function numerical optimization[J]. *Progress in Natural Science*, 2005, 15(5): 463 – 471.

[7] 钟伟才, 刘静, 焦李成. 多智能体遗传算法用于线性系统逼近[J]. 自动化学报, 2004, 30(6): 933 – 938.
(ZHONG Weicai, LIU Jing, JIAO Licheng. Optimal approximation of linear systems by multi-agent genetic algorithm[J]. *Acta Automatica Sinica*, 2004, 30(6): 933 – 938.)

[8] 刘静, 钟伟才, 刘芳, 等. 组织进化数值优化算法[J]. 计算机学报, 2004, 27(2): 157 – 167.
(LIU Jing, ZHONG Weicai, LIU Fang, et al. An organizational evolutionary algorithm for constrained and unconstrained optimization problems[J]. *Acta Automatica Sinica*, 2004, 27(2): 157 – 167.)

[9] HWANG S F, HE R S. A hybrid real-parameter genetic algorithm for function optimization[J]. *Advanced Engineering Informatics*, 2006, 20(1): 7 – 21.

[10] CHENG S L, HUANG C Y. Optimal approximation of linear systems by a differential evolution algorithm[J]. *IEEE Transactions on Systems, Man, and Cybernetics-part A*, 2001, 31(6): 698 – 707.

[11] GONG M G, DU H F, JIAO L C. Optimal approximation of linear systems by artificial immune response[J]. *Science in China: Series F Information Science*, 2006, 49(1): 63 – 79.

作者简介:

张俊岭　(1981—), 男, 博士研究生, 研究领域为进化计算与智能控制, E-mail: zjllogic@126.com;

梁昌勇　(1965—), 男, 教授, 博士生导师, 研究领域为智能决策与控制;

杨善林　(1948—), 男, 教授, 博士生导师, 研究领域为Agent建模与智能决策.