多目标进化算法中多样性策略的研究

谢承旺 丁立新
(武汉大学软件工程国家重点实验室 武汉 430072)

摘 要 进化多目标优化中由于进化算子固有的随机误差以及进化过程中选择压力和选择噪音的影响使得进化群体容易丧失多样性,而保持进化群体的多样性不仅有利于进化群体搜索,而且也是多目标优化的重要目标。对多目标进化算法的多样性策略进行了分类,在统一的框架下描述了各种策略的机制,并分析了各自的特性。随后,分析并比较了多样性保持算子的复杂度。最后,证明了一般意义下多目标进化算法的收敛性,指出在设计新的多样性策略中需要保证进化世代间的单调性,避免出现退化现象。

关键词 多目标进化算法,多样性策略,算子复杂度,收敛性
中图法分类号 TP301 **文献标识码** A

Diversity Strategies on Multiobjective Evolutionary Algorithms

XIE Cheng-wang DING Li-xin
(State Key Laboratory of Software Engineering, Wuhan University, Wuhan 430072, China)

Abstract The intrinsic random errors on the evolutionary operators and the pressure of selection and the noise of selection in the evolutionary process easily make the loss of diversity on the evolutionary population. But the maintenance of diversity on the population is very important because it is not only benificial to the search process but also becomes the essential objective in multiobjective optimization. With the unified framework, this paper first classifiesd the diversity strategy on the MOEAs and described the principles and mechanisms on different types of diversity strategies and analyzed their characteristics. Then this paper analyzed the complexity of these diversity operators. At last, this paper proved the convergence of MOEAs in the general sense and pointed out that it is necessary to keep the monotonicity in the evolutionary population and avoid the degradation of population as the design of new diversity strategy.

Keywords Multiobjective evolutionary algorithms, Diversity strategies, Complexity, Convergence

1 引言

多目标优化问题(Multiobjective Optimization Problem, MOP)的解通常是一组解集合,且集合中的解彼此非劣。进化多目标算法(Multiobjective Evolutionary Algorithms, MOEAs)的目的在于搜索出问题的整个 Pareto 曲面上尽可能均匀分布的一组解,而不仅是某一个非劣解。在多目标优化中,进化算子由于存在随机误差,容易出现"遗传漂移"现象,而且存在选择压力、选择噪音或搜索空间粒度的问题,使得进化群体容易丧失多样性。但是保持进化群体的多样性不仅有利于进化过程的搜索,而且更是多目标优化的目的所在。因此,采取一定的策略保持群体的多样性,既避免"早熟"产生的局部非劣解,又使进化结果呈现为一组不相同的 Pareto 最优解显得尤为重要。

根据进化计算中的模式(Schema)定理[1],若群体的规模为 M,则遗传操作可以从这 M 个个体中生成和检测 M^3 个模式,并在此基础上不断形成和优化积木块,直至找到最优解。显然 M 取值越大,遗传操作所处理的模式就越多,那么生成有意义的积木块以及逐渐进化到最优解的机会就越高。如果单纯从群体的多样性出发,群体规模应越大越好。但是群体规模太大会带来若干弊病:一是群体越大,导致其适应度评估的次数增加,引起计算量的增加,从而影响算法效能;二是群体中个体生存下来的选择概率大多采用与适应度成比例的方法,当群体中个体非常多时,少量适应度很高的个体会被选择而生成下来,但大多数个体会被淘汰,这会影响配对库的正常形成,从而影响交叉操作。因此群体规模只能维持在一定数量上,它并不能成为解决进化算法多样性的途径。

本文讨论 MOEAs 中保持群体多样性的策略,描述这些策略的机制,分析这些策略的复杂度,最后,从一般意义上分析 MOEAs 的收敛性。

2 MOEAs 的多样性策略

为不失一般性,考虑下面 n 个自变量和 k 个目标函数的多目标函数最大化问题:

到稿日期:2009-03-05 返修日期:2009-05-15 本文受高等学校博士点基金项目(No. 20070486081),湖北省杰出青年人才基金(No. 2005ABB017)资助。

谢承旺(1974—),男,博士生,主要研究领域为智能计算、数据挖掘和分布式计算技术,E-mail:chengwangxie@163.com;丁立新(1967—),男,教授,博士生导师,主要研究领域为智能计算、智能信息处理。

$$(MOP)\ \text{Max}: y=f(x)=(f_1(x),f_2(x),\cdots,f_k(x))$$
$$S.T.\ e(x)=(e_1(x),e_2(x),\cdots,e_m(x))\leqslant 0 \quad (1)$$

其中，$x=(x_1,x_2,\cdots,x_n)\in X\subset R^n$，$y=(y_1,y_2,\cdots,y_k)\in Y\subset R^k$，$x$ 表示决策向量，y 表示目标向量，X 表示决策向量 x 形成的决策空间，Y 表示目标向量 y 形成的目标空间，约束条件 $e(x)\leqslant 0$ 确定决策变量 x 的可行取值范围。本文仅讨论无约束 MOP 问题。

在 MOEAs 中，对于某些问题，Pareto 最优解集可能很大，也可能包含无穷多个解，把所有这些解都列入非支配集中有时比较困难，同时也无多大实际意义。因此有必要使非支配解集的大小保持在一个合理的界限内。迄今为止，已有多种方法和技术应用在一定规模的非支配解集上并能保持其中解的多样性。

2.1 小生境技术

生物学上，生物总是喜欢与自己特征、性状相类似的生物生活在一起，即所谓的“物以类聚”。在进化算法中，为了保持进化群体的多样性，模拟自然界生物的这种现象，提出了小生境技术，其中以基于共享机制的小生境技术[2]最具代表性。

Goldberg 等人定义了一个共享函数来表示个体在表现型空间或基因型空间的相似程度。设 $d(i,j)$ 为个体 i 和 j 之间的距离或称相似程度，S_i 表示个体在群体中的共享度，则 $S_i=\sum_{j\in Pop} sh[d(i,j)]$，其中 Pop 为进化群体，$sh[d]$ 为共享函数。这样，个体 i 共享后的适应度为 $fitness(i)/S_i$。这种共享方法考虑了一个个体与群体中所有其它个体间的相似程度，时间开销较大。目前，应用较多的是设置一个共享半径（亦称小生境半径），只需计算共享半径内个体的相似程度。

设个体 i 的适应度为 $fitness(i)$，个体 i 的小生境计数为 m_i，其中 $m_i=\sum_{j\in Pop} sh[d(i,j)]$，$Pop$ 和 $d(i,j)$ 的意义同上文。定义共享函数 $sh[d]$ 如下：

$$sh[d]=\begin{cases}1-(d/\sigma_{share})^{\alpha} & d<\sigma_{share}\\ 0 & \text{其它}\end{cases}$$

通常，$\alpha=1$，σ_{share} 是控制允许共享程度的共享参数。个体 i 共享适应度即为 $fitness(i)/m_i$，此处 m_i 实质是个体 i 的小生境聚集度。同一小生境内的个体相互降低对方的共享适应值，这样有利于以适当的选择压力维持一个整体稳定的群体适应度[2]。如在 MOGA[3]，NSGA[4] 和 NPGA[2] 算法中就使用了这种共享的多样性策略。

运用小生境技术保持解群的多样性策略存在的主要问题是 σ_{share} 参数难以确定，而且 MOEAs 的性能对 σ_{share} 很敏感，因此确定合适的小生境半径就显得十分重要了。

2.2 群体信息熵

利用信息熵来刻画进化群体的多样性与分布性可以从宏观或整体上反映进化群体是否具有良好的多样性。为方便问题的描述，现定义如下概念[5]：

定义 1（解群体方差） 群体 $Pop=\{X_1,X_2,\cdots,X_N\}$ 的规模为 N，个体 X_i 由 L 个基因构成，即 $X_i=[x_i^{(1)},x_i^{(2)},\cdots,x_i^{(L)}]$，$i\in(1:N)$。群体 Pop 中的个体均值定义为 $\bar{X}=\{\bar{x}^{(1)},\bar{x}^{(2)},\cdots,\bar{x}^{(L)}\}$，其中 $\bar{x}^{(j)}=\sum_{i=1}^{N} x_i^{(j)}/N$，则群体的方差定义为 $D=[D^{(1)},D^{(2)},\cdots,D^{(L)}]$，其中 $D^{(j)}=\sum_{i=1}^{N}(x_i^{(j)}-\bar{x}^{(j)})^2/N$，$j=(1:L)$。

从定义 1 可看出，方差 D 是 L 维的行向量，每一个分量表示解群在该维坐标的空间分布。显然 $D^{(i)}$ 越大，则解群在某维坐标轴的空间分布就越大。

方差概念仅反映解群的空间偏离程度，还不能完全刻画解群的多样性。为此引入解群体熵的概念。

定义 2（解群体的熵） 若进化群体 Pop 的规模为 N，将它划分为 m 个子集 $P_1,P_2,\cdots,P_m$，且满足下列性质：$\bigcup_{p\in\{P_1,P_2,\cdots,P_m\}} P=Pop$；$\forall i,j\in(1:m)$ 且 $i\neq j$：$P_i\cap P_j=\emptyset$。则定义解群体的熵 $E=-\sum_{i=1}^{m} q_i\log q_i$，其中 $q_i=|P_i|/N$，$|P_i|$ 为 P_i 的规模大小。

由定义 2 可知，当解群中所有个体都相同，即 $m=1$ 时，熵取最小值 $E=0$；当 $m=N$ 时，熵取最大值 $E=\log N$。个体在解群中分布得越平均，个体类型越多，则其熵就越大。对于十进制编码，熵的最大值 $E^D=\log N$；对于二进制编码，熵的最大值 $E^B=\log(\min(N,2^L))$。

对多目标优化问题而言，当非支配集收敛到 Pareto 最优解集时，方差和熵都达到较大值。最理想的情况是非支配集中有 N 个个体，且都均匀分布在 Pareto 最优前端上，此时其熵达到最大值，同时也会有较大的方差。由于在刻画群体多样性方面，群体的平均信息熵与群体熵具有相同的能力或效果，而且在群体进化过程中，群体的熵不易求取，而群体的平均信息熵则比较容易求，故下面定义群体平均信息熵的概念。

定义 3（群体平均信息熵） $\forall X\in Pop$，Pop 为进化群体，$|Pop|=N$，$X=[x_1,x_2,\cdots,x_L]$，设 D 为一个符号集，$|D|=s$，$x_j\in D$，在 D 中的取值概率分别为 $P=\{P_{j1},P_{j2},\cdots,P_{js}\}$，其中 $j\in\{1,2,\cdots,L\}$，则对应于基因座 j 的信息熵定义为 $H_j(N)=-\sum_{k=1}^{s} P_{jk}\log(P_{jk})$，其中 P_{jk} 为 D 中第 k 个符号出现在基因座 j 上的概率，即 $P_{jk}=$（基因座 j 上出现的第 k 个符号的总数）$/N$；定义群体的平均信息熵为：$H=\frac{1}{L}\sum_{j=1}^{L} H_j(N)$。

崔逊学等人[6]将免疫原理中的浓度机制引入多目标进化的适应度函数设计中，通过定义聚合适应度函数对群体内个体的适应度值进行一种特殊的变换处理来减少相似个体的选择压力，从而有效地保持了群体的多样性。

2.3 聚集密度方法

虽然从宏观上进化群体的熵或群体的平均信息熵能够较好地刻画群体中个体的分布性，但这种方法缺乏对群体内部个体之间关系的刻画，不便于调控群体进化过程中的多样性。采用聚集密度的方法不仅可从宏观上刻画群体的多样性，而且也能较好地刻画个体之间内在的关系，可用于进化过程中对群体的调控。

一般地，在 MOEAs 中常常通过计算个体间相似度、个体间的影响因子或聚集距离的方法来计算个体的聚集密度[5]。

定义 4（个体间相似度） 群体 $Pop=\{X_1,X_2,\cdots,X_N\}$ 中的个体 $X_i=[x_i^{(1)},x_i^{(2)},\cdots,x_i^{(L)}]$，$X_j=[x_j^{(1)},x_j^{(2)},\cdots,x_j^{(L)}]$，定义个体 X_i 和 X_j 之间的相异程度为 $A_{ij}=\frac{1}{L}\sum_{k=1}^{L} C_k(x_i^{(k)}-x_j^{(k)})$，其中 $i,j\in(1:N)$，C_k 为对应于基因座 k 的常数因子，且通常有 $C_k=BC_{k+1}$，B 为一常量。则定义个体 X_i 和 X_j 间的相似度为 $1-A_{ij}$。

定义 5（个体聚集度） 定义个体 p 的聚集度为与个体 p 相似的个体在群体中所占的比重，即 $crowds(p)=$ 与个体 p 相似度大于 γ 的个体的总数$/N$。其中 γ 为一常数，一般取值

为 $\gamma\in[0.9,1]$。

定义 6(个体影响因子) 定义目标空间中第 i 个解个体对另一个解个体 y 的影响函数为 $\varphi(l_{i\to y}):R\to R$。其中 $l_{i\to y}$ 表示个体 i 与个体 y 的 Euclid 距离,$\varphi(l_{i\to y})$ 为两个个体之间距离的减函数,这里采用较多的有高斯影响函数 $\varphi(r)=(1/(\sigma\sqrt{2\pi}))e^{-r^2/2\sigma^2}$ 等。

当有 k 个目标时,其可行解区域可以表示为一个有 $a_1\times a_2\times\cdots\times a_k$ 个网格的超网格面。设任意一个解个体 $y\in F^k$,其密度定义为 $D(y)=\sum_{i=1}^{N}\varphi(l(i,y))=\sum_{i_1=1}^{a_1}\sum_{i_2=1}^{a_2}\cdots\sum_{i_k}^{a_k}\varphi(l(\langle i_1,i_2,\cdots,i_k\rangle,y))$。

Deb 等人在 NSGA2[7] 算法中使用了聚集距离来评估个体的聚集密度,其密度估计机制是定义 $P[i]_{distance}$ 为个体 i 的聚集距离,这个距离为与 i 最近的邻点为顶点形成的矩形的周长。设 $P[i]_r$ 为个体 i 在目标 r 的函数值,则个体 i 的聚集距离的计算方法为 $P[i]_{distance}=\sum_{r=1}^{k}(P[i+1].f_r-P[i-1].f_r)$。这种评估个体密度的方法相对于共享函数法的优势在于至少省去了选择参数 σ_{share} 的不便。

2.4 网格技术

在 PESA,PAES 以及 MGAMOO 等算法中均运用了网格技术来保持进化群体的多样性。其基本思想是将目标空间划分割为若干超网格(hyper-cube),通过跟踪网格中个体的数目来评估个体的密度信息,并调整群体中的个体,以此来保持群体的多样性。其中以 Knowles 等人的 PAES 算法最具代表性[5,8]。

PAES 算法采用了自适应网格技术来保持群体的多样性,这种技术与一般的网格技术相比独特之处在于:第一,网格的边界不是固定的,可能在每一代都不同;第二,在每代进化过程中,需要根据当前代的个体分布情况自适应地调整边界;第三,如果新产生的个体落在上一代确定的边界以外,只要它是非支配的,则一定将该个体加入到档案集合中。此时,若档案集合已满,则考虑将其它个体从档案中删除。

设第 t 代进化时所得到的非支配集为 $ARCH_t$,在每一维上的域宽 $range_i$ 设置为 $\forall i\in(1:k)$,存在:$range_{i,t}=\max\{z_i\mid z\in ARCH_t\}-\min\{z_i\mid z\in ARCH_t\}$。

第 t 代的边界设为:下界($lb_{i,t}$)和上界($ub_{i,t}$)($i\in(1:k)$;$t>0$)。同时要求边界值满足 $\forall t,\forall i:lb_{i,t}<\min\{z_i\mid z\in ARCH_t\}$ 且 $ub_{i,t}>\max\{z_i\mid z\in ARCH_t\}$。

在此,将边界值进行拓宽,可将其设为:$lb_{i,t}=\min\{z_i\mid z\in ARCH_t\}-(1/2d)range_{i,t}$ 以及 $ub_{i,t}=\max\{z_i\mid z\in ARCH_t\}+(1/2d)range_{i,t}$,这里 d 是一常数,表示在每一维上的分割次数。

2.5 聚类分析方法

在 MOEAs 中,聚类分析的目的是维持和增加进化群体的多样性与分布性,它将群体中的个体划分为若干由类似的个体组成集合(子类),并通过计算个体或类之间的相似度(或相异度)来实现。

实际上,聚类分析方法可细分为划分法、层次法、基于密度的方法、基于网格的方法和基于模型的方法等[5],这些方法在应用中通常是结合起来使用的。在 Zitzler 的 SPEA[9] 中就应用了层次聚类方法,这是一种按照自底向上的策略来聚类个体的方法,初始时把外部非支配集合中每一个个体分别作为一个子类,然后通过计算个体之间的相似度,逐步将具有最大相似度的个体聚集到同一类中,直至档案的规模不超过规定值为止。而且在 SPEA 算法中,当外部种群大小超过了规定值时,将新解和保留解共同进行聚类处理。文献[10]通过实验指出:聚类的方法一般会提高多目标进化算法的性能,而且聚类方法降低了算法对一些次优参数设置的敏感性。

在聚类分析中要注意保留极点,所谓极点[5]指在 MOP 中,单个目标最优的点称为极端点(简称极点)。在本文 MOP 问题定义中,有 k 个优化目标,若这 k 个优化目标是相互冲突的,现假定 x^* 使 $f_i(k)(i\in(1:k))$ 达到最优,且满足约束式(1)($e(x)\leqslant 0$),则 $x^*=(x_1^*,x_2^*,\cdots,x_n^*)$ 为极点。极点对保持群体的多样性与分布性有重要作用。例如在 SPEA2[11] 算法中,通过给边界解点赋予无穷大的距离值来保留那些边界解,从而获得解群的良好分布。至于如何寻找极点则超出了本文的讨论范围。

2.6 约束配对法

在群体中对任意两个个体进行配对时要施加约束条件,要求它们彼此之间的基因型或表现型的距离满足一定的要求。约束配对过程可以在群体内产生更多不同的种群,有利于提高群体的多样性,避免出现同化现象。

MOGA[3] 算法中采用了杂交限制的策略以避免种群中个体间的过度竞争。其杂交限制的思想是假定相邻的适合的配对个体的基因型应该相似,以利于形成稳定的小生境。因此在 MOGA 算法中采用了 Gray 编码技术而非二进制编码方式。这里关键的问题是约束配对区域的半径 σ_{mating} 的计算,其方法类似于计算 σ_{share}。σ_{mating} 的计算可利用公式:$N=\frac{\prod_{i=1}^{k}(M_i-m_i+\sigma_{muting})-\prod_{i=1}^{k}(M_i-m_i)}{\sigma_{mating}^{k}}$ 求取。这里 N 表示均衡面上得到的互不干涉的非支配解点的最大个数,其中,$M=(\max_y y_1,\max_y y_2,\cdots,\max_y y_k)=(M_1,M_2,\cdots,M_k)$,$m=(\min_y y_1,\min_y y_2,\cdots,\min_y y_k)=(m_1,m_2,\cdots,m_k)$,$k$ 是求解问题的目标数目。如果已知了均衡面上解点数目 N,通过上面公式可以得到一个 $(k-1)$ 阶的多项式方程而求得 σ_{mating}($\sigma_{mating}>0$)。

不过也有研究人员如 Eshelman 等人采用相反的做法,他们不允许非常相似的个体配对,以此来避免近亲繁殖[1]。

3 算子的复杂度

利用聚集距离来评估个体聚集密度的方法比较简单,在有些情况下不能较好地反映实际情况,特别是当一个个体与其邻近个体之间的距离相等时。为此,需要提供这个个体与其更多邻近个体的聚集信息。现讨论与一个个体邻近的 $2l$ 个点的情形,算法如下:

计算个体与 $2l$ 个邻近点聚集距离的均值[5]

```
Crowding-distance-assignment-for-multi-point(P)
{ N=|P|;//N 为群体大小
for each i,P[i]distance=0;//初始化各个体聚集距离
for each objective m //对各目标进行如下操作
  {P=sort(P,m);//对目标 m 的函数值进行排序
    for i=2 to (N-1) //针对边界点之外的解
      for j=1 to l
        P[i]distance=P[i]distance+(P[i+j]m-P[i-j]m);
        P[i]distance=P[i]distance/l;
  }//end for objective m
```

$P[0]_{distance}=P[N]=\infty$;//给边界点一个无穷大值

}

上述算法在最坏情况下对 k 个目标分别进行排序的时间为 $O(kN\log N)$，计算每个个体的聚集距离的时间为 $O(klN)$，由于 l 通常比 N 小得多，一般取 $l\leqslant\log N$，因此算法的时间复杂度仍为 $O(kN\log N)$。

在 PAES[8] 算法中采用了自适应的网格算子，该算子采用反复划分 k 维目标空间的方法进行拥挤程度评估。算子将表现型空间反复划分成若干超立方体，超体的每一维的宽度是 $k_r/2^l$，这里的 k_r 是档案集合中当前解在目标 k 上的范围，l 是表现型空间划分的层次。每产生一个解就通过反复划分空间来确定解在档案中的位置，并用网格图来标示档案中的解及其所处的网格情况。目标空间划分成超体的数目为 $n=(2^l)^k$，随着进化过程不断逼近 Pareto 前端，最后档案中个体可能只会占据其中很少数目的超体。

对于自适应网格算子，其时间复杂度需要考虑进行比较的次数。假设群体规模为 N，档案中当前解个体的数目为 N^*，l 表示划分的层次，k 为问题的目标数目，那么找到一个解个体的网格位置需要 $l\times k$ 次比较，因此找到整个档案的网格位置需要 $N^*\times l\times k$ 次比较，然后更新 quadtree 的范围。在最坏的情况下，整个种群 N 加到档案中，因此整个种群必须与 k 维目标的每一维比较当前的最大值和最小值，因此共需 $2k\times N+N\times l\times d$ 次比较，最坏情况下每次迭代的时间复杂度是 $O((N^*+N)k)$，这里忽略了 l，因为 l 通常很小。实际上，档案中的网格位置并非经常更新，因为每代中找到位于档案当前范围之外的解极少，所以平均情况下的比较次数远少于 $2k\times N+N\times l\times d$。

相比之下，小生境方法[2] 每代需要 $N(N-1)$ 次比较，而且计算每对解点之间的距离也相当耗时。如果小生境的计算是基于下一代持续更新的种群，那么小生境方法仍然需要 $\frac{N}{2}(N+1)$ 次比较，时间复杂度仍为 $O(N^2)$。限于篇幅，其它未及分析的多样性算子此处略去。

4 算法收敛性

MOEAs 的目的在于既要使求得的解接近真正的 Pareto 前端，又要保持 Pareto 前端的多样性。在 MOEAs 的设计中，多样性策略会影响到算法的收敛性。理解 MOEAs 的收敛性有利于设计新的多样性策略，本文从一般意义上讨论 MOEA 的收敛性。

在多目标情况下，一方面，可行解集是偏序的；另一方面，基于 Pareto 的适应度的计算在不同的进化代可能具有不同的值；第三方面，多目标优化的解个体是一个向量，而且最后的结果是一个解集。

MOEA 的收敛过程是通过 PF_{known} 不断逼近 PF_{true} 实现的，为描述不同进化代之间的关系，下面给出有关定义[5]。

定义 7 给定一个 MOP，以及进化群体 PA 和 PB，定义 PA 和 PB 的关系为 $PA\geqslant PB$，若满足条件：$\forall x\in PA,\neg\exists y\in PB$，使 $y\succ x$。

定义 7 表明，若 $PA\geqslant PB$，则在 PA 中不存在任何受 PB 所支配的向量，但在 PB 中可能存在受 PA 所支配的向量。由此可知 $P_{known}(t+1)\geqslant P_{known}(t)$，因为 $P_{known}(t+1)=\{x\in P_{known}(t)\cup P_{current}(t+1)\mid \exists y\in P_{known}(t)\cup P_{current}(t+1)$，使 $y\succ x\}$。

定理 1 给定 MOP 和 MOEA，若满足条件：

(1) Pareto 最优边界的 Box 计数维数不大于 $k-1$，k 为目标数；

(2) $P_{known}(0)$，$P_{known}(1)$，…是单调的，即 $\forall t: P_{known}(t+1)\geqslant P_{known}(t)$，则 MOEA 以概率 1 收敛，即 $Prob(\lim_{t\to\infty}\{P_{true}=P(t)\})=1$。其中 $Prob()$ 为概率，$P(t)=P_{known}(t)$，P_{true} 为全局最优解。

证明[5]：可以将 MOEA 的执行过程看作一个 Markov 链，包含两个状态。第一个状态是 $P_{known}(t)=P_{true}$ 成立，第二个状态是 $P_{known}(t)=P_{true}$ 不成立。从条件(2)可知，从第一个状态到第二个状态的转移概率为 0，这表明第一个状态是吸引的。由条件(1)可知，从第二个状态到第一个状态的转移概率大于 0，这表明第二个状态是瞬时的。这样，MOEA 的执行状态完全满足 Markov 定理，即 MOEA 以概率 1 收敛。

从 MOEA 的收敛性分析可以看出，保证代之间的单调性是算法收敛的必要条件，非支配解集中个体的调整亦应满足单调性要求，避免出现退化现象，使得进化过程不断朝 Pareto 最优前端逼近。

结束语 本文在统一的框架下依据 MOEAs 中多样性策略机制，将其分成小生境方法、群体信息熵、聚集密度方法、网格技术、聚类分析方法以及约束配对方法。本文不仅描述了多样性保持方法的机制，而且分析了这些策略各自的特性。随后，根据这些多样性方法执行时所需的比较次数对这些算子的时间复杂度进行了分析，其中以 PAES 的自适应网格算子的时间开销更胜一筹。最后，从一般意义上分析 MOEAs 收敛性，指出对非劣解集中的个体进行多样性调整时必须保证世代间的单调性，做到一代更比一代强，这一点在设计新的多样性保持策略时务必要考虑。

参 考 文 献

[1] 崔逊学. 多目标进化算法及其应用[M]. 北京：国防工业出版社，2006

[2] Horn J, Nafpliotis N, Goldberg D E. A niched Pareto genetic algorithm for multiobjective optimization[C]// IEEE World Congress on Computation. Piscataway, NJ, 1994

[3] Fonseca C M, Fleming P J. Genetic algorithms for Multiobjective optimization: Formulation, discussion and generalization[C]// Proceedings of the 5th International Conference on Genetic Algorithms. San Mateo, California, 1993

[4] Srinivas N, Deb K. Multiobjective optimization using nondominated sorting in genetic algorithms[R]. Dept. Mechanical Engineering, Kanput, India, 1993

[5] 郑金华. 多目标进化算法及其应用[M]. 北京：科学出版社，2007

[6] Cui X X, Li M, Fang T. Study of Population Diversity of Multiobjective Evolutionary Algorithm Based on Immune and Entropy Principles[C]// Proceedings of the 2001 International Congress on Evolutionary Computation. IEEE Neural Networks Council, Seoul, Korea, 2001

[7] Deb K, Agrawal S, Pratap A, et al. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA II[C]// Parallel Problem Solving from Nature (PPSN VI). Berlin, 2000

[8] Knowles J D, Corne D W. The Pareto archived evolution strategy: A new baseline algorithm for Pareto multiobjective optimization[C]// Congress on Evolutionary Computation (CEC99). Piscataway, NJ, 1999

[9] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: A comparative case study and strength Pareto approach[J]. IEEE Transactions on Evolutionary Computation, 1999, 3: 257-271

[10] Laumanns M, Zitzler E, Thiele L. On the effects of archiving, elitism, and density based selection in evolutionary multi-objective optimization[C]// Evolutionary Multi-Criterion Optimization (EMO 2001). 2001

[11] Zitzler E, Laumanns M, Thiele L. Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization[C]// EUROGEN 2001, Evolutionary Methods for Design, Optimization and Contorl with Applications to Industrial Problems. September 2001

(上接第 140 页)

WP33 一般合并结构(Generalized AND-Join):两个或多个分支被合并到后继分支,当所有分支都被激发后,则控制线程被传递到后继分支。在合并期间,各个分支仍可接受激发(每个分支接受的激发数目要相同),并保存这些激发以将来使用。

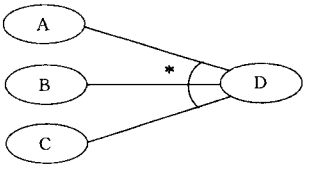

图 6 一般合并结构

$A=!\ r_a.\bar{a}.0$

$B=!\ r_b.\bar{b}.0$

$C=!\ r_c.\bar{c}.0$

$D_0=(\nu h)(D_1|D_2)$

$D_1=a.\bar{h}.0|b.\bar{h}.0|c.\bar{h}.0$

$D_2=h.h.h.\bar{r}_d.D_0$

$D=!\ r_d.\sigma_d.D'$

$WP33=A|B|C|D_0|D$

WP37 无循环同步合并(Acyclic Synchronizing Merge):早期被分解的两个或多个分支合并到一个后继分支上,当每一个活动的输入分支被激活时,控制线程都会被传递给后续分支。至于多少个分支需要同步则取决于合并结构能够获得的信息。如图 7 所示,A,B,D 3 个分支是从同一个分支分解出来的且都处于活动状态,当它们中的任意一个完成时,都将会引起 C 的执行,而且 C 中还需要 B,D 同步。

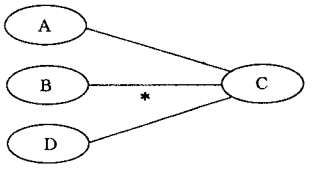

图 7 无循环同步合并

$A=!\ r_a.\bar{a}.0$

$B=!\ r_b.\bar{b}.0$

$D=!\ r_d.\bar{d}.0$

$C=(\nu h)(C_1|C_2)$

$C_1=b.\bar{h}.0|c.\bar{h}.0$

$C_2=a.\sigma_c.C'|h.h.\sigma_c.C'$

$WP37=A|B|C|D$

WP38 通用的同步合并(General Synchronizing Merge):早期被分解的两个或多个分支合并到一个后继分支上,当满足下面两个条件之一时:(1)每一个活动的输入分支被激活时,(2)未被激活的分支在将来也不可能被激活,则控制线程被传递给后续分支。如图 8 所示,A,B,D 是从同一个分支分解出来的,这里 A,B 是两个活动的输入分支,D 代表不被激活的分支。

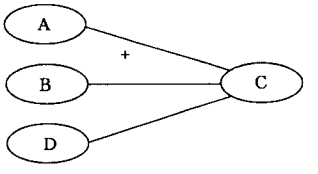

图 8 通用同步合并

$A=!\ r_a.\bar{a}.0$

$B=!\ r_b.\bar{b}.0$

$D=\bar{d}.0$

$C=a.\sigma_c.C'|b.\sigma_c.C'|d.\sigma_c.C$

$WP38=A|B|C|D$

结束语 本文利用 Pi 演算对复杂分支和同步工作流模式进行了详细描述,所用描述完全是形式化的,它不但使过程模型的语义更加精确,而且在模型建立后,还可利用 Pi 演算来推演系统的行为,验证模型的正确性,譬如发现系统行为不完整、死锁或缺少同步等。

参 考 文 献

[1] Aalst W M P, Barros A P, Hofstede A H M, et al. Advanced workflow patterns[C]// Proceedings of the Fifth IFCIS International Conference on Cooperative Information Systems (CoopIS' 2000). volume 1901 of LNCS. Eilat, Israel, Springer, 2000: 18-26

[2] Russell N. Foundations of Process-Aware Information Systems Faculty of Information Technology, Queensland University of Technology[J]. Brisbane, Australia, 2007

[3] van der Aalst W M P. Pi Calculus Tjersus Petri Nets: Let Us Eat Humble Pie Rather Than Further inflate the Pi Hype[J]. BPTrends, 2005, 3(5): 1-11

[4] Puhlmann F, Weske M. Using the Pi-Calculus for Formalizing Workflow Patterns[C]// BPM 2005. volume 3649 of LNCS. Berlin: Springer-Verlag, 2005: 153-168

[5] Milner R, Parrow J, Walker D J. A calculus of Mobile Process Part Ⅰ[R]. Report ECS-LFCS-89-85. Laboratory for Foundation of Computer Science, Computer Science Department, Edinburgh University, 1989

[6] Milner R, Parrow J, Walker D J. A calculus of Mobile Process Part Ⅱ[R]. Report ECS-LFCS-89-85. Laboratory for Foundation of Computer Science, Computer Science Department, Edinburgh University, 1989