文章编号: 1001-0920(2007)09-1011-06

一种子群体个数动态变化的多目标优化协同进化算法

申晓宁, 郭 毓, 陈庆伟, 胡维礼
(南京理工大学 自动化学院, 南京 210094)

摘 要: 给出一种新型的在多目标优化条件下的进化算法群体停滞判别准则,并基于该准则提出一种合作型多目标优化协同进化算法. 该算法在运行过程中自适应地决定子群体的新增和灭绝,使得子群体个数依据需要动态变化,减小了对计算资源的消耗,并解决了对复杂多目标优化问题难以事先进行分解的问题. 对所提算法的计算复杂度进行了理论分析,并把它与已有的多目标进化算法进行了比较,结果表明所提算法具有较高的搜索性能.

关键词: 自适应; 协同进化; 多目标优化; 进化算法; 精英保留
中图分类号: TP18 **文献标识码**: A

A multi-objective optimization co-evolutionary algorithm with dynamically varying number of subpopulations

SHEN Xiao-ning, GUO Yu, CHEN Qing-wei, HU Wei-li
(School of Automation, Nanjing University of Science and Technology, Nanjing 210094, China. Correspondent: SHEN Xiao-ning, E-mail: sxnystsyt@sina.com.cn)

Abstract: A new kind of criterion judging the stagnation of the population in evolutionary algorithms in the existence of multiple objectives is presented, and a cooperative multi-objective optimization co-evolutionary algorithm based on this criterion is proposed. The sub-population is adaptively added or deleted during the running of the algorithm, which makes the number of the sub-populations vary dynamically so that the computational cost is reduced and the difficulty of the decomposition of the complicated optimization problem is overcome. The computational complexity of the proposed algorithm is analyzed theoretically and it is compared with existed multi-objective evolutionary algorithms. Results indicate that the proposed algorithm can search more effectively.

Key words: Adaptation; Co-evolution; Multi-objective optimization; Evolutionary algorithms; Elitism preservation

1 引 言

协同进化算法是一种模拟自然界生物种群之间相互作用、相互影响、协同进化的过程而产生的智能优化算法. 它把一个复杂的问题分解为若干个相对简单的子问题,分配给多个子群体分别进行进化操作,子群体间定期进行信息的交互,通过合作或竞争的关系,共同完成对优化问题的求解. 因而,协同进化算法具有较高的优化效率,特别适用于解决具有多个变量或多个目标的复杂优化问题.

近年来,国内外涌现出大量的基于进化算法求解多目标优化问题的文献,但在其中引入协同进化机制的研究还比较少见,且这类研究大多为了简化问题,在进化开始之前便已确定好子群体的个数及其负责搜索的范围. 如文献[1,2]以合作型协同进化模型[3]为基础,设置 n 个子群体分别独立地基于已有的多目标优化算法进化问题的 n 个变量;文献[4]基于捕食者-猎物模型,采用两个相互竞争的子群体分别进化候选解和目标向量,随着进化过程的推进,目标向量的求解难度逐渐增加,而候选解的"竞争力"也不断加强;文献[5]令一个目标对应一个子群体,依据生态种群密度竞争方程调整各子群体的规模,以描述多个目标之间的关系.

在许多实际的多目标优化问题中,由于种种原因(如变量之间存在强耦合关系),事先难以对问题确定适当的分解方法以及所需的子群体个数. 目前,仅有文献[6]对多目标协同进化算法的子群体确定

收稿日期: 2006-05-11; **修回日期**: 2006-07-14.
基金项目: 国家自然科学基金项目(60174019,60474034); 江苏省自然科学基金项目(BK2007210).
作者简介: 申晓宁(1981—),女,南京人,博士生,从事进化算法、多目标优化的研究; 胡维礼(1941—),男,江苏东台人,教授,博士生导师,从事智能控制、网路控制系统等研究.

问题作了初步的探讨. 该文依据在进化过程中获取的信息划分搜索空间,为划分的每个子区域分配一个子群体,并依据对当前 Pareto 前沿所做的贡献决定是否应灭绝该子群体还是应对其作进一步的分解. 该方法的不足之处是划分搜索空间之后可能产生大量的子群体,导致计算资源的过度消耗.

针对上述问题,本文给出一种在多目标优化条件下的进化算法群体停滞判别准则,并由此引出多目标协同进化算法中子群体新增和灭绝的条件以及算法的终止准则. 在此基础上,提出一种子群体个数动态变化的合作型多目标优化协同进化算法,简称 DCMOCEA. 对 DCMOCEA 的计算复杂度进行了理论分析,并把它与两种已有的多目标进化算法在一组标准测试函数上进行了对比.

2 多目标条件下群体的停滞判别准则

在同时具有多个优化目标的问题中,解之间的优劣程度往往是无法比较的(互不支配),因而在多目标进化算法中判断群体是否已经达到停滞的状态比较复杂. 为此,本文提出一种新型的在多目标优化条件下的进化算法群体停滞判别准则.

设置一个外部集合 archive 存储群体搜索到的非支配解. 群体在每一代进化过程中都要对 archive 进行更新:把当前群体 pop 中的非支配解加入 archive,archive 中便可能存在若干解被 archive 中其余解支配,把这些受支配的解从 archive 中删除,同时判断 pop 是否对 archive 产生了“推进作用”. 这里对“推进作用”定义如下:

定义 1 若当前群体 pop 中存在若干个解支配外部集合 archive 中原已储存的解,则称 pop 对 archive 产生了“推进作用”.

依据“推进作用”的定义,可引出多目标优化条件下进化算法群体的停滞判别准则:

定义 2 若在连续的 gen 代(gen 的取值依据问题而定)中,群体均未对外部集合 archive 产生“推进作用”,则说明其已处于停滞状态.

当群体发生停滞后,它虽然也能向外部集合提供一些新的非支配解,但却无法为整个外部集合收敛性能的进一步提高做出贡献.

3 动态的合作型多目标优化协同进化算法

所提算法 DCMOCEA 基于 Pareto 支配的概念和精英保留的策略,把协同进化机制引入对多目标优化问题的求解中. DCMOCEA 直接采用实数编码,子群体中的每个个体表示问题的一个完整的解;每个子群体独立地进行进化繁殖操作,再通过合作的关系,共同求得问题最终的 Pareto 最优解集.

在很多实际问题中,对目标进行评价比较困难,需要占用算法绝大多数的运行时间,因而如何有效减少目标的评价次数成为提高算法效率和实用性的关键因素. 在协同进化算法中,子群体的个数与规模是影响目标评价次数的主要原因[7]. 基于此,DCMOCEA 采用小规模的子群体,且不事先指定子群体的个数,而是随着进化过程的推进依据子群体的新增和灭绝条件动态地对其个数进行调整,从而在保证算法搜索效率与多样性的同时最大限度地节约计算资源.

3.1 精英保留机制

在 DCMOCEA 中,为当前存活的每个子群体 pop(i) 均分配一个子外部集合 archive(i) 以存储 pop(i) 搜索到的非支配解,同时设置一个总的外部集合 totalarchive 保存所有子群体搜索到的非支配解. 外部集合的使用将有效防止算法在搜索过程中由于随机因素或群体规模的限制而丢失优良解,并能够加快算法的收敛速度.

多目标优化问题中非支配解的个数可能相当大,由于计算资源的限制,DCMOCEA 固定 totalarchive 和 archive(i) 的最大容量分别为常数 M 和 M_{sub}. 以 totalarchive 为例,一旦当前存储的非支配解的个数大于 M,则移除若干个密度较大的解. 本文使用排挤距离[8]的概念估计解的密度信息. 把当前 totalarchive 中的所有解分别依据各目标进行排序,则解 x 的排挤距离定义为在各个目标上,排列在 x 的左侧和右侧的两个解的距离的平均值. 解 x 的排挤距离越小,说明解 x 周围的密度越大.

3.2 新增子群体的条件

DCMOCEA 在进化过程的初始阶段只采用一个规模较小的子群体,因而算法具有较快的运行速度. 然而,由于规模的限制,该子群体难以对整个决策空间进行大范围的搜索,因此随着进化的推进,应根据需要动态地添加新的子群体,使得多个子群体间相互合作,协同进化,以增强算法搜索的多样性. 在 DCMOCEA 中,新增子群体的条件是:

1) 到当前代为止,现有的所有子群体均已连续 $\text{gen1}_{\text{live}}$ 代以上未对总的外部集合 totalarchive 产生“推进作用”;

2) 现有的所有子群体中,最近一个产生的子群体从生成开始直到当前代为止至少曾经对 totalarchive 产生过 $\text{gen2}_{\text{live}}$ 次“推进作用”.

以上两个条件需同时满足,因为如果没有条件 2) 的限制,当 totalarchive 已接近于问题真实的 Pareto 最优解集时,条件 1) 易于满足,算法将频繁地新增子群体,这些新的子群体耗费了大量的计算资源,却难以对 totalarchive 的推进产生显著的作

用.

3.3　灭绝子群体的条件

由于 DCMOCEA 中子群体的规模较小,某个子群体在进化到一定的程度之后将难以进一步搜索到质量更优或范围更广的解.这时,应把该子群体灭绝,以免造成对计算资源的浪费.在 DCMOCEA 中,认为子群体 pop(*i*)需要灭绝的条件为:pop(*i*)已连续 $gen1_{kill}$ 代以上未对其相应的子外部集合 archive(*i*)产生"推进作用",且已连续 $gen2_{kill}$ 代以上未对总的外部集合 totalarchive 产生"推进作用".需要说明的是,若当前只有一个存活的子群体,则无论它是否满足上述灭绝条件,都不把它灭绝.

3.4　算法的终止准则

适当的终止准则是确保算法在消耗尽可能少的计算资源的同时,有效地搜索到一组质量较优的解的重要因素. DCMOCEA 采用的终止准则为:1)到当前代为止,现存的所有子群体均已连续 $gen1_{stop}$ 代以上未对总的外部集合 totalarchive 产生"推进作用";2)最近一个产生的子群体已经历了 $gen2_{stop}$ 代以上,且从未对 totalarchive 产生过"推进作用".以上两个条件均需满足,但如果当前进化的总代数 t 超过了某个设定的较大值 T,则强迫算法终止.

3.5　算法的步骤

所提算法 DCMOCEA 的详细步骤如下:

Step1: 令进化代数 $t = 1$,子群体个数 $k = 1$,子群体规模 $= N$;随机生成 k 个初始父代子群体 $pop_t(i)$,并创建 k 个空的子外部集合 archive$(i) = \Phi$ 和一个总的外部集合 totalarchive $= \Phi$.

Step2: 令子群体计数器 $i = 1$,对当前存活的所有子群体进行进化操作:

Step2.1: 对父代子群体 $pop_t(i)$ 依概率进行交叉和变异操作,生成子代子群体 $newpop_t(i)$;

Step2.2: 取出 $newpop_t(i)$ 中所有的非支配解对 archive(i) 和 totalarchive 进行更新,并分别判断其是否对 archive(i) 和 totalarchive 产生了"推进作用";此后,如果 archive(i) 或 totalarchive 中解的个数超过了其最大容量 M_{sub} 或 M,则对其进行裁减操作,移除那些排挤距离较小的解;

Step2.3: 若 $|\text{archive}(i)| \leq N$($|\cdot|$ 表示集合中元素的个数),则从 $newpop_t(i)$ 中随机选取 $N - |\text{archive}(i)|$ 个个体,与 archive(i) 中的所有个体共同构成下一代的父代子群体 $pop_{t+1}(i)$;否则,若 $|\text{archive}(i)| > N$,则从 archive(i) 中随机挑选 N 个解作为 $pop_{t+1}(i)$;

Step2.4: $i = i + 1$,如果 $i \leq k$,则转至 Step2.1.

Step3: 对当前存活的每个子群体,依次判断其是否满足 3.3 节中子群体的灭绝条件.若满足,则把它及相应的子外部集合删除,$k = k - 1$,直至处理完所有的子群体(需确保 $k > 0$).

Step4: 若当前存活的子群体满足 3.2 节中新增子群体的条件,则随机生成一个新的规模为 N 的子群体,并为其设置一个空的子外部集合,$k = k + 1$.

Step5: 增加进化代数,$t = t + 1$.

Step6: 依据 3.4 节确定算法是否满足终止准则.若满足,则把 totalarchive 作为最终求得的非支配解集输出,算法终止;否则,转至 Step2.

3.6　算法的计算复杂度

3.6.1　时间复杂度

DCMOCEA 的时间复杂度主要取决于对外部集合的更新与裁减操作.设优化问题有 m 个目标,对 DCMOCEA 在最坏情况下的时间复杂度分析如下:

1) 子代子群体对子外部集合的更新:从规模为 N 的子群体中确定出所有的非支配解最多需进行 $mN(N-1)$ 次比较;把它们加入子外部集合后,判断它们是否受子外部集合中原已保存的解支配,最多需进行 mNM_{sub} 次比较;由于子外部集合中原已保存的解之间互不支配,最多只需比较 mNM_{sub} 次,判断它们是否受新加入的解支配.当算法中存在 k 个子群体时,该步骤的时间复杂度为 $O(kmN(N + 2M_{sub}))$.

2) 子代子群体对总的外部集合的更新:类似 1)的分析,该步骤的时间复杂度为 $O(kmN(N + 2M))$.

3) 对子外部集合进行裁减:在子群体对子外部集合进行更新之后,每个子外部集合中最多有 $N + M_{sub}$ 个个体,把这些个体分别依 m 个目标进行排序,以计算个体间的排挤距离.采用快速排序法,并考虑到共有 k 个子外部集合,该步骤的时间复杂度为 $O(km(N + M_{sub})\log(N + M_{sub}))$.

4) 对总的外部集合进行裁减:类似 3)的分析,由于只有一个总的外部集合,该步骤的时间复杂度为 $O(m(N + M)\log(N + M))$.

DCMOCEA 总的时间复杂度取上述分析结果中的最大值 $O(kmN(N + 2M))$.若 $2M \gg N$,则可简化为 $O(kmNM)$.

3.6.2　空间复杂度

DCMOCEA 的空间复杂度主要取决于对外部集合进行裁减时,存储解的排挤距离所占用的空间:

1) 对子外部集合进行裁减:最多需存储 k 个子外部集合中共 $k(N + M_{sub})$ 个个体的排挤距离,因此该步骤的空间复杂度为 $O(k(N + M_{sub}))$.

2) 对总的外部集合进行裁减:总的外部集合只有一个,所以该步骤的空间复杂度为 $O(N+M)$.

DCMOCEA 总的空间复杂度取上述分析结果中的最大值 $O(k(N+M_{sub}))$.

3.6.3 与其他算法的比较

NSGA-Ⅱ[8] 是仅对单个群体进行操作的快速非支配排序多目标遗传算法,设其群体规模为 L,则其时间复杂度为 $O(mL^2)$,空间复杂度为 $O(L^2)$;NSCCGA[1] 使用 n 个子群体分别独立地进化问题的 n 个变量,设其子群体规模为 N',则其时间复杂度为 $O(m(nN')^2)$,空间复杂度为 $O((nN')^2)$. 在所提算法DCMOCEA的参数设置中,令 $N+M=L$, $N+M_{sub}=L/2$, $N=N'/2$,则 DCMOCEA 的时间复杂度又可表示为 $O(kmN(L-N))$ 或 $O(kmN^2)$,空间复杂度又可表示为 $O(kL)$ 或 $O(k(N'/2+M_{sub}))$. 把 3 种算法的时间和空间复杂度进行对比,由于 DCMOCEA 中的子群体规模 N 和子群体个数 k 均较小(k 从 1 开始自适应变化),而 NSGA-Ⅱ中的群体规模 L 一般较大,NSCCGA 具有较多的子群体个数 n,因而在 3 种算法中,DCMOCEA 具有最小的时间和空间复杂度.

4 仿真实验

4.1 测试函数

使用 4 种具有不同特征的测试函数对所提算法的有效性进行验证. 测试函数定义如下:

$$\text{ZDT1}^{[9]}:\begin{cases}\text{Min } f_1=x_1,\\ \text{Min } f_2=g\cdot h.\end{cases}$$

其中

$$g=1+9\cdot\sum_{i=2}^{n}x_i/(n-1),$$

$$h=1-\sqrt{f_1/g},$$

$$n=30,\ x_i\in[0,1]. \tag{1}$$

ZDT1 的 Pareto 前沿为凸曲线.

$$\text{ZDT2}^{[9]}:\begin{cases}\text{Min } f_1=x_1,\\ \text{Min } f_2=g\cdot h.\end{cases}$$

其中

$$g=1+9\cdot\sum_{i=2}^{n}x_i/(n-1),$$

$$h=1-(f_1/g)^2,$$

$$n=30,\ x_i\in[0,1]. \tag{2}$$

ZDT2 的 Pareto 前沿为非凸曲线.

$$\text{ZDT3}^{[9]}:\begin{cases}\text{Min } f_1=x_1,\\ \text{Min } f_2=g\cdot h.\end{cases}$$

其中

$$g=1+9\cdot\sum_{i=2}^{n}x_i/(n-1),$$

$$h=1-\sqrt{f_1/g}-(f_1/g)\sin(10\pi f_1),$$

$$n=30,\ x_i\in[0,1]. \tag{3}$$

ZDT3 的 Pareto 前沿具有 5 段离散的凸曲线.

$$\text{Vie}:\begin{cases}\text{Min } f_1=(x_1-2)^2/2+(x_2+1)^2/13+3,\\ \text{Min } f_2=(x_1+x_2-3)^2/175+(2x_2-x_1)^2/17-13,\\ \text{Min } f_3=(3x_1-2x_2+4)^2/8+(x_1-x_2+1)^2/27/15,\\ x_1,x_2\in[-4,4]. \end{cases}\tag{4}$$

Vie 具有 3 个目标函数.

4.2 实验结果

使用收敛性测度 M_1[9] 和分布性能测度 SP[6] 及作图法把所提算法 DCMOCEA 与两种已有的多目标优化进化算法 NSGA-Ⅱ[8] 和 NSCCGA[1] 在 4.1 节中的测试函数上进行对比实验. 测度 M_1 和 SP 的值越小,说明算法的收敛性能及解的分布性能越好.

3 种算法均采用实数编码,使用模拟二进制交叉算子(SBX) 和多项式变异算子[10],交叉概率取 0.9,变异概率取 1/n(n为变量的个数). DCMOCEA 中,$N=10$, $M=90$, $M_{sub}=40$, $gen1_{live}=5$, $gen2_{live}=1$, $gen1_{kill}=5$, $gen2_{kill}=20$, $gen1_{stop}=5$, $gen2_{stop}=20$, $T=1\,000$;NSGA-Ⅱ中群体规模 L 取 100;NSCCGA 中子群体规模 N 取 20.

为了比较的公正性,所有实验均在同一台计算机上使用 Matlab 7.0 软件完成,算法的计算精度设为 10^{-10}. 实验分为两组,在每组实验中,3 种算法在各个测试函数上均独立地运行 10 次.

第 1 组实验求取 3 种算法在相同的计算量下产生的非支配解集,以比较其在 M_1 和 SP 测度上的性能. 令各算法在每次运行中执行相同次数的目标评价,表 1 列出了在各测试函数中 3 种算法 10 次运行结果的平均值,图 1 ~ 图 4 分别给出了各算法产生的 Pareto 前沿与真实的 Pareto 前沿的比较.

第 2 组实验求取 3 种算法在 M_1 测度上的性能达到相同的要求时所需进行的目标评价次数,以比

表 1 3 种算法在 M_1 和 SP 测度上的比较

		ZDT1	ZDT2	ZDT3	Vie
M_1	DCMOCEA	0.002 2	0.002 1	0.001 4	0.039 6
	NSGA-Ⅱ	0.002 9	0.003 3	0.001 5	0.043 1
	NSCCGA	0.009 4	0.141 1	0.002 6	0.044 7
SP	DCMOCEA	0.003 8	0.003 6	0.005 1	0.061 3
	NSGA-Ⅱ	0.005 1	0.007 0	0.007 3	0.134 3
	NSCCGA	0.009 9	0.010 4	0.029 4	0.137 0

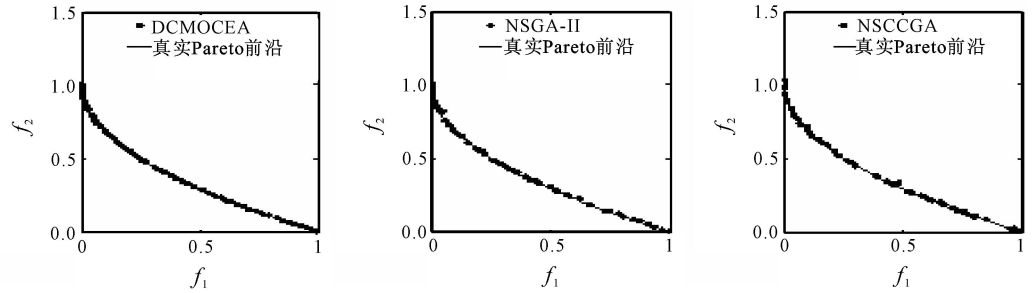

图 1　ZDT1 中 Pareto 前沿的比较

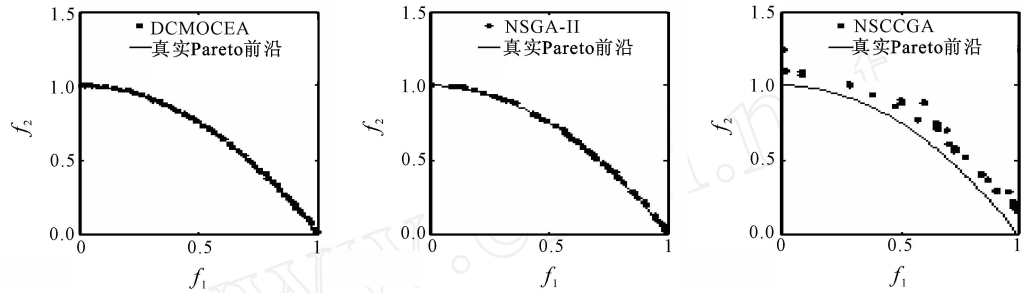

图 2　ZDT2 中 Pareto 前沿的比较

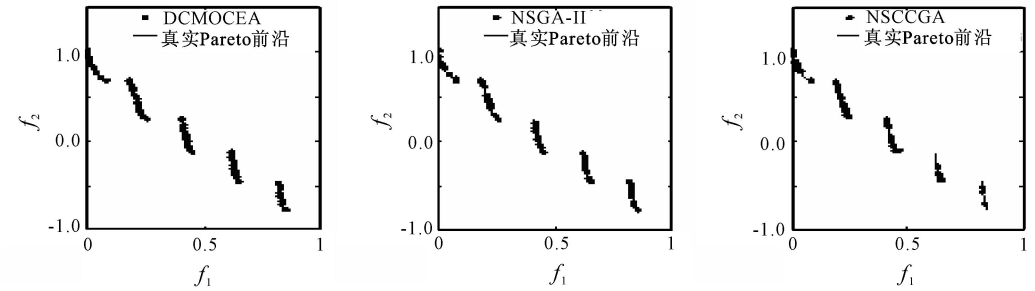

图 3　ZDT3 中 Pareto 前沿的比较

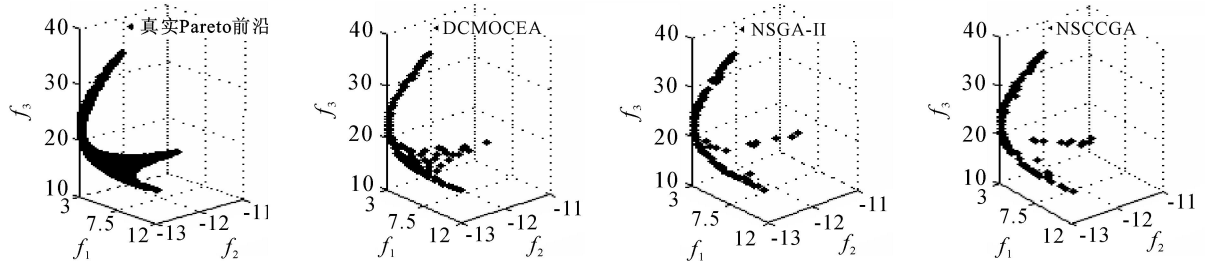

图 4　Vie 中 Pareto 前沿的比较

较各算法计算的复杂性. 本文取表 1 中 DCMOCEA 在 M_1 上的值作为对各算法 M_1 性能的要求,表 2 列出了各算法运行 10 次的平均目标评价次数.

由表 1 和图 1 ～ 图 4 可见,在 4 个测试函数中,所提算法DCMOCEA 在收敛性能(M_1 测度) 和分布特性(SP 测度) 上均优于 NSGA- Ⅱ和 NSCCGA ,因而 ,在计算量相同的情况下 ,DCMOCEA 可以产生一组质量更优的解.

由表 2 可见, 在相同的 M_1 性能要求下, DCMOCEA 所需执行的目标评价次数少于 NSGA- Ⅱ和 NSCCGA ,说明 DCMOCEA 具有较小的计算复杂性.

表 2　3 种算法的目标评价次数

	ZDT1	ZDT2	ZDT3	Vie
DCMOCEA	9 210	5 030	10 130	13 870
NSGA- Ⅱ	14 200	7 080	10 240	16 300
NSCCGA	18 450	48 900	22 200	19 800

在DCMOCEA 中,子群体新增和灭绝条件中的参数决定了子群体个数在算法运行过程中的变化情况,影响算法搜索的多样性和计算的复杂度;而终止准则中的参数判定算法决定是否应继续运行,以避

免对计算资源不必要的耗费. 因而,这些参数的取值是 DCMOCEA 的关键因素. 本文对以上参数值的选取是经多次实验反复调试得到的, 图 5 给出了 DCMOCEA 在函数 ZDT1 的某一次运行中子群体个数随进化代数变化的曲线图.

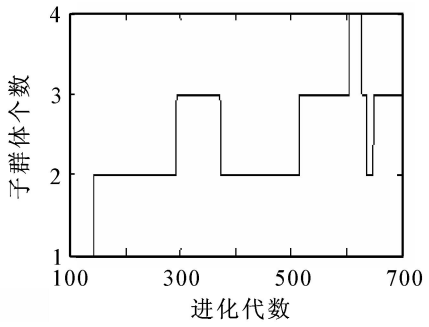

图 5 DCMOCEA 的子群体个数随进化代数的变化

以上实验针对的是 2 目标或 3 目标优化问题, 其结果也可以推广到具有多个(≥4) 优化目标的问题中. 在本文中,DCMOCEA 采用实数编码,以便于解决连续优化问题. 若改变 DCMOCEA 的编码方式,并采用适当的交叉和变异算子(如在 TSP 问题中采用城市序号顺序编码、部分匹配交叉和对换变异算子) ,则 DCMOCEA 也同样适用于求解离散组合多目标优化问题(如 TSP ,Flow-shop 等问题).

5 结 论

本文提出一种新型的合作型多目标优化协同进化算法,该算法依据子群体新增和灭绝的条件动态地调整子群体的个数. 把所提算法和其余两种有效的多目标进化算法在具有不同特性的标准测试函数上进行了比较,结果表明所提算法具有更高的搜索效率,它能够在较少的目标评价次数下,产生一组收敛性能与分布性能均较优的非劣解. 今后的工作将进一步研究所提算法中参数的自适应控制方法,并把它应用于实际优化问题.

参考文献(References)

[1] Iorio A W, Li X D. A cooperative coevolutionary multiobjective algorithm using non-dominated sorting [C]. Proc of the Genetic and Evolutionary Computation Conf , Part I. Washington: Springer-Verlag , 2004: 537-548.

[2] Kuntinee M , Kittipong B , Nachol C. Multi-objective optimisation by co- operative co-evolution[C]. Parallel Problem Solving from Nature. Birmingham: Springer-Verlag , 2004: 772-781.

[3] Potter M A , De J K. A cooperative coevolutionary approach to function optimization[C]. Proc of Parallel Problem Solving from Nature III. Germany : Springer-Verlag , 1995: 249-257.

[4] Lohn J D , Kraus W F , Haith G L. Comparing a coevolutionary genetic algorithm for multiobjective optimization [C]. Congress on Evolutionary Computation. Piscataway: IEEE Service Center , 2002: 1157-1162.

[5] 曹先彬, 李金龙, 王煦法. 基于生态协同的多目标优化研究[J]. 软件学报, 2001 , 12(4) : 521-528.
(Cao Xian-bin , Li Jin-long , Wang Xu-fa. Research on multiobjective optimization based on ecological cooperation[J]. J of Software , 2001 , 12(4) : 521-528.)

[6] Coello C A , Sierra M R. A coevolutionary multi-objective evolutionary algorithm[C]. Proc of the 2003 Congress on Evolutionary Computation. Australia: IEEE Press , 2003: 482-489.

[7] 孙晓燕, 巩敦卫. 变种群规模合作型协同进化遗传算法及其在优化中的应用[J]. 控制与决策, 2004 , 19(12) : 1437-1440.
(Sun Xiao-yan , Gong Dun-wei. Varying population size cooperative coevolutionary genetic algorithm and its application in optimization[J]. Control and Decision , 2004 , 19(12) : 1437-1440.)

[8] Deb K, Pratap A , Agarwal S , et al. A fast and elitist multiobjective genetic algorithm: NSGA-II[J]. IEEE Trans on Evolutionary Computation , 2002 , 6(2) : 182-197.

[9] Zitzler E , Deb K, Thiele L. Comparison of multiobjective evolutionary algorithms: Empirical results [J]. Evolutionary Computation , 2000 , 8(2) : 173-195.

[10] Deb K, Agrawal R B. Simulated binary crossover for continuous search space[J]. Complex Systems , 1995 , 9: 115-148.

(上接第 1004 页)

[11] Chiang Kao , Wen-Kai Hsu. A single-period inventory model with fuzzy demand [J]. Computers and Mathematics with Application , 2002 , 43 (6-7) : 841-848.

[12] Yager R R. A procedure for ordering fuzzy subsets of the unit interval[J]. Information Sciences , 1981 , 24 (4) : 143-161.

[13] Lious T S , Wang M J. Ranking fuzzy numbers with integral values[J]. Fuzzy Sets and Systems , 1992 , 50 (3) : 247-255.