一种快速的双目标非支配排序算法*

刘 敏[1,2,3] 曾文华[4] 赵建峰[1,2]

[1](厦门大学 智能科学与技术系 厦门 361005)
[2](厦门大学 福建省仿脑智能系统重点实验室 厦门 361005)
[3](漳州师范学院 计算机科学与工程系 漳州 363000)
[4](厦门大学 软件学院 厦门 361005)

摘 要 提出一种快速的双目标非支配排序算法(BNSA). 设计了前向比较操作,以便快速识别非支配个体. 提出了按需排序策略,避免生成多余的非支配前沿. 论证 BNSA 算法的正确性,分析其时间复杂度为 $O(NlogN)$. 在 9 个标准的双目标优化测试问题上进行了比较实验. 实验结果表明与其它 3 种非支配排序算法相比,BNSA 算法在大多数测试问题上具有更快速的性能. 当进化代数超过 400 代时,BNSA 在所有的测试问题上都具有最好的加速效果. 此外,BNSA 算法简明、易于编程实现,可集成到任何基于非支配排序的多目标进化算法中,能较大程度地提高双目标优化的运行速度.

关键词 多目标进化算法,非支配排序,前向比较,按需排序
中图法分类号 TP 301.6

A Fast Bi-Objective Non-Dominated Sorting Algorithm

LIU Min[1,2,3], ZENG Wen-Hua[4], ZHAO Jian-Feng[1,2]

[1](*Cognitive Science Department, Xiamen University, Xiamen* 361005)
[2](*Fujian Key Laboratory of the Brain-like Intelligent Systems, Xiamen University, Xiamen* 361005)
[3](*Department of Computer Science and Engineering, Zhangzhou Normal University, Zhangzhou* 363000)
[4](*Software School, Xiamen University, Xiamen* 361005)

ABSTRACT

A fast bi-objective non-dominated sorting algorithm (BNSA) is proposed. An operator of forward comparison is designed to identify non-dominated individuals quickly. A sorting strategy according to need is proposed to avoid generating unnecessary non-dominated fronts. Then, the correctness of BNSA is proved and its time complexity is analyzed to be $O(NlogN)$. Next, some comparable experiments are carried out on nine benchmark test problems for bi-objective optimization. Results of the experiments indicate that the proposed BNSA, for the most test problems, is faster than the other three non-dominated sorting algorithms. Furthermore, the BNSA, on all the test problems, has the best of accelerative effect, particularly when the number of evolutionary generations exceeds 400. In addition, the BNSA is concise and easy to be implemented. It can be incorporated into any multi-objective evolutionary algorithms based

* 国家自然科学基金项目资助(NO. 60672018, 40774065)
收稿日期:2010-11-30;修回日期:2011-01-25
作者简介 刘敏,男,1974 年生,博士研究生,讲师,主要研究方向为进化计算、多目标优化. E-mail: liumin_xt@163.com. 曾文华,男,1964 年生,教授,博士生导师,主要研究方向为并行进化计算、神经网络、网格计算等. E-mail: whzeng@xmu.edu.cn. 赵建峰,男,1982 年生,博士研究生,主要研究方向为网格计算.

on non-dominated sorting to improve the running speed of bi-objective optimization.

Key Words　Multi-Objective Evolutionary Algorithm, Non-Dominated Sorting, Forward Comparison, Sorting According to Need

1　引　言

现实世界中的许多实际问题是高度复杂的,通常需要对多个彼此冲突的目标进行同时优化,人们将其称为多目标优化问题(Multi-objective Optimization Problem, MOP). 近年来,多目标进化算法(Multi-Objective Evolutionary Algorithm, MOEA)非常适合于求解该类问题,在科学研究和实际应用领域引起了广大学者浓厚的研究兴趣[1-2]. 目前,在众多的 MOEA 中,2002 年 Deb 等[3] 提出的 NSGA-II (Non-Dominated Sorting Genetic Algorithm II)算法最为著名,应用也最为广泛,至今被 SCI 引用已达 2 012次. NSGA-II 通过一个多目标非支配排序算法(Fast Non-Dominated Sorting Approach, FNSA)将组合种群分成多层的非支配前沿(Non-Dominated Front),为生成下一代种群以及完成选择操作提供重要的个体等级信息. NSGA-II 的时间复杂度取决于 FNSA 的时间复杂度——$O(mN^2)$,m 为目标数,N 为组合种群规模[3]. 为了降低 FNSA 的时间复杂度以便提高 NSGA-II 的运行速度,2003 年 Jensen[4] 提出一种时间复杂度为 $O(NlogN)$ 的双目标非支配排序(Non-dominated Sorting on Two Objectives, NSTO)算法,并在 NSTO 的基础上通过递归与分治提出了一种时间复杂度为 $O(Nlog^{m-1}N)$ 的多目标非支配排序算法[4]. 2007 年郑金华等[5] 提出一种名为擂台赛法则(Arena's Principle, AP)的构造非支配集的方法,并基于 AP 改进了 FNSA. 这种基于 AP 的多目标非支配排序(AP-Based Non-Dominated Sorting, APNS)算法的时间复杂度为 $O(rmN)$,$(0 < r/N < 1)$,r 为非支配个体数.

尽管高维多目标优化(Many Objective Optimization)是目前多目标优化的一个研究热点,但是在现实应用中高维问题相对较少,而且有些高维问题可通过降维的方法将其转化为低维(2 ~ 3 个目标)问题[2]. 相反,双目标优化是多目标优化问题的一种常见的优化形式,在车间作业调度、物流配送、电力调度等领域有广泛的应用[6-8]. 此外,多目标优化领域最权威的专家——Deb 教授的部分最新研究成果也与双目标优化有关[9-10]. 因此对双目标优化问题进行研究具有重要的学术意义和实际应用价值. 针对双目标优化问题,在以上的非支配排序算法中,NSTO 的时间复杂度最低. 但是 NSTO 算法还存在以下缺点: 1) 存在多余的非支配前沿分层,运行速度有待提高; 2) 需要动态地建立查找表,并反复调用折半查找,编程实现难度有待降低. 因此,本文研究目的是希望克服 NSTO 算法的缺点,提出一种快速简明的双目标非支配排序算法,设计了前向比较操作来快速识别非支配个体,提出了按需排序策略,以避免生成不必要的非支配前沿.

2　相关定义、性质与非支配排序算法

2.1　相关的定义与性质

不失一般性,一个具有 n 个决策变量,m 个目标函数的多目标优化问题[1] 可以描述为

$$\min\ y = F(x) = (f_1(x), f_2(x), \cdots, f_m(x)),$$
$$s.t.\ g_i(x) \leqslant 0,\ i = 1,2,\cdots,p,$$
$$h_j(x) = 0,\ j = 1,2,\cdots,q,$$

其中,$\boldsymbol{x} = (x_1, x_2, \cdots, x_n) \in X \subset R^n$ 为 n 维决策变量向量,X 为 n 维决策空间. $\boldsymbol{y} = (f_1, f_2, \cdots, f_m) \in Y \subset R^m$ 为 m 维目标函数向量,Y 为 m 维目标空间,目标函数 $F(x): X \to Y$ 定义了 m 个由决策空间到目标空间的映射. $g_i(x) \leqslant 0$, $i = 1,2,\cdots,p$ 定义了 p 个不等式约束; $h_j(x) = 0$, $j = 1,2,\cdots,q$ 定义了 q 个等式约束.

此外,设 R_t 为一个多目标解集(在 NSGA - II 中它对应为一个组合种群,即 $R_t = P_t \cup Q_t$. 其中 P_t 为父种群,Q_t 为子种群,单个种群规模设为 POPSIZE),R_t 的规模为 N,且 $N = 2 \times$ POPSIZE.

定义 1(Pareto 支配(dominance)关系)　$\forall u, v \in X$,若 $f_k(u) \leqslant f_k(v)$, $k = 1,2,\cdots,m$; 且 $\exists l \in \{1,2,\cdots,m\}$,使 $f_l(u) < f_l(v)$,则称 u 支配 v,记为 $u \prec v$. 其中"$\prec$"是 Pareto 支配关系. 此时 v 是被支配的(dominated),称 v 为支配个体[1)].

定义 2(非支配集)　$\forall u \in R_t$,若 $\nexists v \in R_t$,使 $v \prec u$,则称 u 为集合 R_t 的非支配个体. 由 R_t 的所有非

1) 相对于解集中的元素而言,在种群中称之为个体,本文为了表述方便有时将元素称为个体.

支配个体组成的集合，称之为 R_t 的非支配集(Non-dominated Set，Nds).

定义 3(非支配前沿) 对于给定 R_t，称其非支配集为 R_t 的第 1 层非支配前沿 F_1；称集合 $R_t - F_1$ 的非支配集为 R_t 的第 2 层非支配前沿 F_2；称集合 $R_t - (F_1 \cup F_2)$ 的非支配集为 R_t 的第 3 层非支配前沿 F_3；第 i 层非支配前沿 F_i 的定义以此类推.

定义 4(预排序) 按照下面的比较规则将 R_t 排序成序列 $s_1, s_2, \cdots, s_N$：

$$i < j \rightarrow (f_1(s_i) < f_1(s_j)) \vee (f_1(s_i) = f_1(s_j) \wedge f_2(s_i) \leqslant f_2(s_j)),$$

即先按第 1 个目标值进行升序排序，当第 1 个目标值相同时，再按照第 2 个目标值进行升序排序.

假设 L 为构造非支配集的集合(简称为构造集)，根据定义 4 对 R_t 进行预排序，将排序后的结果保存在构造集 L 中. Jensen[14] 对预排序后的构造集 L 给出了一些描述，本文将其归纳整理为以下性质.

性质 1 构造集 L 的所有非支配个体的第 2 个目标函数值的大小是依次递减的.

说明 如图 1 所示，预排序后，构造集 $L = \{s_1, s_2, \cdots, s_{10}\}$. L 的非支配集 $Nds = F_1 = \{s_1, s_3, s_5, s_8\}$. 容易看出所有非支配个体 s_1, s_3, s_5, s_8 的第 2 个目标值是依次递减的，即

$$f_2(s_1) > f_2(s_3) > f_2(s_5) > f_2(s_8).$$

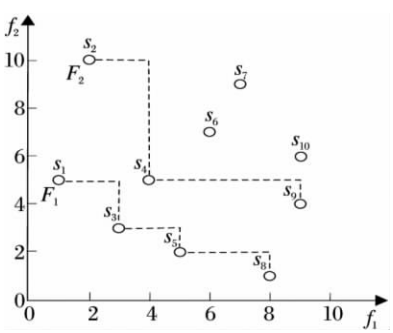


图 1 双目标非支配排序示例

Fig. 1 Illustration of bi-objective non-dominated sorting

性质 2 构造集 L 中任意一个元素都不被它后面的元素支配.

2.2 NSGA - II 的非支配排序算法

NSGA - II 的非支配排序算法(FNSA)需要将组合种群 R_t 分成多层的非支配前沿 $F_i(i = 1, 2, \cdots, k$，k 为非支配前沿的层数). FNSA 的基本步骤如下.

step 1 令 $i = 1$；R_t 中任一个体 p，都将与其它个体进行支配关系比较，并计算支配个体 p 的个体数目 n_p 和被个体 p 支配的个体集合 S_p. 凡是支配计数 n_p 为 0 的个体都被分入第 1 层非支配前沿 F_1.

step 2 对于 F_i 中的任一个体 p，将 S_p 集合中的每一个成员 q 的支配计数 n_q 减 1. 如果 q 的支配计数 n_q 等于零，将 q 分入下一层非支配前沿 F_{i+1}.

step 3 令 $i = i + 1$，如果 F_i 非空，则跳转至 step 2. 否则，返回前沿 $F_1, F_2, \cdots, F_k$ 并结束非支配排序.

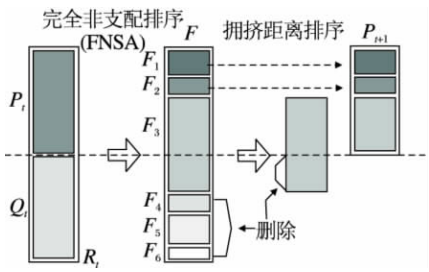


图 2 NSGA - II 过程

Fig. 2 Procedure of NSGA - II

FNSA 的排序结果为非支配前沿集 $F(F = F_1 \cup F_2 \cup \cdots \cup F_k)$，如图 2 所示. FNSA 的缺点在于：1) 时间复杂度高，为 $O(mN^2)$，这里的 m 为目标数，N 为组合种群规模；空间复杂度高，为 $O(N^2)$；编程实现较难；2) 因采用完全非支配排序(即划分所有的非支配前沿)而存在多余的分层. 如图 2 所示，F_3 分层完毕后，前沿 F_1, F_2, F_3 中的累计个体数目已经超出了单个种群规模(POPSIZE)，结合后面的拥挤距离排序和填充下一代种群 P_{t+1} 可以看出，无须再对 F_4, F_5, F_6 进行分层.

2.3 Jensen 的双目标非支配排序算法

2003 年 Jensen[14] 提出一种双目标非支配排序算法(NSTO)，该算法的基本步骤如下.

step 1 按照定义 4 对组合种群 R_t 进行预排序.

step 2 对 R_t 进行顺序扫描，同时构造所有的非支配前沿. 如果当前的被扫描个体 $s_i(i = 1, 2, \cdots, N)$ 不被最末层前沿 F_k 中的个体支配时，则在已分层的前沿($F_1, F_2, \cdots, F_k$)中通过折半查找将 s_i 分到一个合适的前沿中. 否则，执行 $k = k + 1$；set $F_k = \{s_i\}$.

step 3 如果没有扫描完，则跳转至 step 2. 否则，返回前沿层 $F_1, F_2, \cdots, F_k$ 并结束非支配排序.

NSTO 算法的时间复杂度为 $O(NlogN)$，空间复杂度为 $O(N)$. 但该算法还存在引言所提及的缺点.

2.4 基于擂台赛法则的非支配排序算法

作为一种构造非支配集的方法，擂台赛法则

(AP) 的基本思想是: 从构造集中任选一个体出任擂主,由擂主与其它个体进行支配比较. 败者被淘汰出局,胜者成为新擂主,并继续与剩下的其它个体进行比较,最后的擂主即为非支配个体. 按照这种方法继续找出其它非支配个体,直至构造集为空.

AP 构造的非支配集其实是第一层非支配前沿. 要完成非支配排序,需要利用 AP 改进 FNSA. 这种基于 AP 的非支配排序(APNS) 算法的时间复杂度为 $O(rmN)$,空间复杂度为 $O(N)$,r 为非支配个体的数目,m 为目标个数,N 为组合种群规模,且 $0 < r/N < 1$. APNS 的缺点在于: 1) APNS 同 FNSA 一样存在多余的非支配前沿分层; 2) AP 每次产生的最后一个擂主需要一个回溯比较过程,将没有被老擂主打败的个体淘汰出局,增加了编程实现难度.

对于双目标优化,以上 3 种非支配排序算法不是时间复杂度较高,就是算法不够简明. 本文为此提出一种快速简明的双目标非支配排序算法(Bi-objective Non-dominated Sorting Algorithm, BNSA) .

3　双目标非支配排序算法(BNSA)

虽然本文的 BNSA 算法初始阶段将使用与 NSTO 算法中相同的预排序,但是 BNSA 中预排序之后的主体部分与 NSTO 的主体部分截然不同. NSTO 主体部分的主要思想是: 对组合种群中的个体进行顺序扫描,对每一个体都不断地通过折半查找的方法来寻找与该个体等级恰好匹配的非支配前沿,并将该个体装入此前沿,直至所有的个体扫描分层完毕. NSTO 其实采用的是“根据个体找非支配前沿”的完全排序思路,而 BNSA 主体部分的设计思路恰好与之相反. BNSA 采取的是“根据非支配前沿找个体”的按需排序思路: 首先通过快速简易的前向比较操作将组合种群中所有等级最高的个体快速装入等级最高的非支配前沿,然后将等级次之的个体装入到等级次高的非支配前沿,……,直至已分层非支配前沿中的个体总数达到单个种群规模(POPSIZE) 为止.

首先假设 R_t 为组合种群,L 为构造集,Q 为 L 中的支配个体所组成的集合(集合 L 和 Q 适宜采用线性表类型的数据结构) ,F_i 为第 i 层非支配前沿. 根据定义 4 对 R_t 进行预排序,将排序后的结果保存在构造集 L 中.

3.1　判定某个体为非支配个体的快速识别准则

快速识别准则　构造集 L 中,设 u 是 v 之前最近的一个非支配个体,如果 u 不支配 v,则 v 是非支配个体.

本文提出的快速识别准则能快速简易地判别预排序后的构造集 L 中的个体是否为非支配个体,该准则为 3.2 节的前向比较操作提供了理论基础,证明过程如下.

快速识别准则的证明　$\forall u, v \in L$,设 u 是 v 之前最近的一个非支配个体,下面分为 $u = v$(如果种群中存在相同的个体时) 与 $u \neq v$ 两种情行加以证明.

1) 当 $u = v$ 时,已知 u 是一个非支配个体,显然 v 也是一个非支配个体,准则得证.

2) 当 $u \neq v$ 时,已知 u 是 v 之前最近的一个非支配个体,根据预排序定义有 $f_1(u) < f_1(v) \vee (f_1(u) = f_1(v) \wedge f_2(u) \leqslant f_2(v))$.

下面用反证法证明 $f_1(u) < f_1(v)$. 假设 $f_1(u) = f_1(v) \wedge f_2(u) \leqslant f_2(v)$ 成立,则或者是 $u \prec v$,或者 $u = v$. 然而 $u \prec v$ 与已知条件 u 不支配 v 矛盾,$u = v$ 也与已知条件 $u \neq v$ 矛盾,故原假设 $f_1(u) = f_1(v) \wedge f_2(u) \leqslant f_2(v)$ 不成立,所以 $f_1(u) < f_1(v)$.

同理易用反证法证得 $f_2(v) < f_2(u)$.

下面证明 v 的第 2 个目标值 $f_2(v)$ 比它前面所有个体的第 2 个目标值都要小.

(a) 先证个体 v 的 $f_2(v)$ 比它前面所有的非支配个体的第 2 个目标值都要小.

已知 u 是非支配个体,由性质 1 可知,u 的第 2 个目标值 $f_2(u)$ 比它前面所有非支配个体的第二个目标值都要小. 因已证明 $f_2(v) < f_2(u)$,所以(a) 得证.

(b) 再证个体 v 的 $f_2(v)$ 比它前面所有的支配个体的第 2 个目标值都要小.

设个体 v 之前还存在若干支配个体,令 w 为其中的任一个体. 假设 $f_2(w) < f_2(u)$ 成立,则易知个体 w 不会被非支配个体 u 支配,进而 w 更不会被 u 之前的非支配个体支配. 这与个体 w 是支配个体相矛盾,因此 $f_2(w) \geqslant f_2(u)$. 又因为已证得 $f_2(v) < f_2(u)$,则 $f_2(v) < f_2(w)$,所以(b) 得证.

综合(a) ,(b) 的结论,易证得 v 不会被它前面所有的个体支配. 再由性质 2 可知,v 也不会被它后面的所有个体支配,所以 v 是非支配个体,准则得证.

3.2　前向比较操作

根据快速识别准则,前向比较操作可描述为: 将 L 中当前被访问的个体同离其最近的前一个非支配个体进行支配关系比较. 若当前个体不被该非支配

个体支配,则当前个体也是非支配个体,并将其分入到当前前沿 F_i 中. 否则,该当前个体是被支配的,将其复制到支配个体集 Q 当中.

前向比较操作能快速简易地判别构造集 L 中的个体是否为非支配个体. 因此通过顺序地扫描构造集 L,并对每个个体执行前向比较操作,可快速完成当前前沿 F_i 的分层. 最后的集合 Q 就是下一层非支配前沿的构造集.

为了识别一个非支配个体,FNSA 需要将该个体与种群中其它个体都进行一次支配比较; APNS 需要一轮擂台赛的支配比较; NSTO 需要一次支配比较和若干次折半查找; BNSA 使用前向比较操作仅需一次支配比较,比其它 3 种算法更快速简易.

3.3 按需排序策略

按需排序策略可描述为: 在非支配排序过程中,当已分层的非支配前沿中的累计个体数超过单个种群规模(POPSIZE) 便停止非支配排序. 如图3 所示,前 3 层非支配前沿分层完,便停止非支配排序.

本文称集成了 BNSA(用 BNSA 替代 NSGA - II 的 FNSA) 的 NSGA - II 为 BNSA + NSGA - II,如图 3 所示. BNSA + NSGA - II 不仅能有效减少非支配前沿的分层数目,而且它仍能利用 NSGA - II 的拥挤距离方法来保持种群的多样性,它所获得的下一代种群 P_{t+1} 与 NSGA - II 所获得的下一代种群相同.

BNSA 克服了 NSTO 在引言中提及的两个缺点: 1) BNSA 应用按需排序的策略避免了不必要的分层; 2) BNSA 对每个个体采用快速简易的前向比较操作,不必反复折半查找,比 NSTO 更简易.

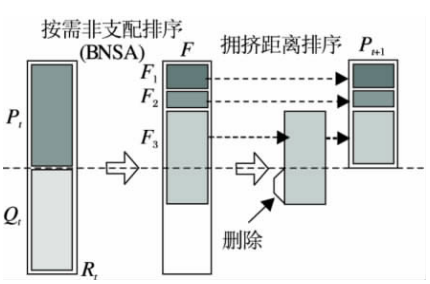


图 3 BNSA + NSGA - II 过程

Fig. 3 Procedure of BNSA + NSGA - II

3.4 BNSA 算法的基本步骤

step 1 对 R_t 预排序,结果保存至构造集 L.

step 2 从 L 中的第 2 个个体开始,依次对每个个体执行前向比较操作.

step 3 按需排序检测. 如果已分层前沿中的累计个体数目小于单个种群规模,则由支配个体集 Q 生成新的构造集 L 并跳转至 step 2. 否则,返回已分层前沿 $F_1, F_2, \cdots, F_k$ 并结束非支配排序.

3.5 BNSA 算法的详细描述

```
BNSA( R_t ) {
/* 对组合种群 R_t 进行双目标非支配排序,分成 k 层非支配前沿集{ F_i | i = 1,2,…,k} ,初始时F_i = Φ* /
    L = presort( R_t ) ; /* 预排序 * /
    i = 1; /* i 为当前的非支配前沿层数 * /
    count = 0; /* count 为已分层的个体数 * /
    do {
      F_i = {L [0]}; /* 第一个元素是非支配个体 * /
      count = count + 1;
      Q = Φ; /* Q 暂存所有被支配的个体 * /
      pre = 0; /* pre 指向前一个非支配个体 * /
      for (j = 1; j < L. size( ) ; j ++ ) { /* 扫描 L* /
        if ( L [pre] ≺ L [j] ) /* 执行前向比较 * /
          Q = Q ∪ { L [j] } ;
        else {
          F_i = F_i ∪ { L [j] } ;
          count = count + 1;
          pre = j;
        }
      } /* 结束 F_i 层非支配前沿的分层 * /
      i = i + 1;
      L = Q; /* 生成下一层前沿的构造集 L* /
    } while( count < POPSIZE) /* 按需排序检测 * /
    k = i - 1; /* 修正层数, k 为最末层的层数 * /
    return F_1 ,F_2 ,…,F_k ;
}
```

3.6 BNSA 算法的正确性证明

BNSA 的正确性证明是指对于任意输入的组合种群,BNSA 都能正确地完成非支配排序,即需要证明 BNSA 的返回结果 $F_1, F_2, \cdots, F_k$ 都是正确的非支配前沿.

以下采用数学归纳法进行证明.

1) 基础步骤. 证明返回的 F_1 是正确的第一层非支配前沿.

(1) 已知算法中的第 2 行代码将当前的前沿层数设置为 1. 再由性质 2 可知,第一次执行完第 5 行代码能正确地将 L 中第一个非支配个体分入第一层非支配前沿 F_1.

(2) for 循环体内的代码完成前向比较操作,根据快速识别准则可知前向比较操作能正确地将 L 中当前扫描到的非支配个体分入前沿 F_1. 又因为算法中第 9 ~ 17 行的 for 语句实际是从第 2 个元素一直到最后一个元素顺序地扫描 L,执行前向比较,所以能保证 L 中其余的非支配个体都被分入前沿 F_1.

综合(1)(2),可证得第一次执行 for 语句后,F_1 是正确的第 1 层非支配前沿.

2) 归纳步骤. 必须证明对于任意的前沿层 F_i, $i = 1,2,\cdots,k-1$,如果 F_i 是正确的第 i 层前沿,则 F_{i+1} 是正确的第 $i+1$ 层前沿. 先假设 F_i 是正确的第 i 层前沿. 因为第 11 行代码可通过线性表末端插入的方式来保证支配个体依次保存到支配集 Q,且 Q 中的个体仍维持预排序后的顺序. 所以第 19 行代码生成下一层非支配前沿的构造集 L 依旧满足 2.1 节中的性质和 3.1 节的快速识别准则. 因此再次执行 do while 循环体后,能保证所产生的下一层非支配前沿 F_{i+1} 也是正确的第 $i+1$ 层前沿.

综合 1) 和 2),由数学归纳的原理证明了 BNSA 的返回结果 $F_1,F_2,\cdots,F_k$ 都是正确的非支配前沿.

3.7　BNSA 算法的时间与空间复杂度分析

设组合种群 R_t 的规模为 $N(N = 2 \times \text{POPSIZE}$, POPSIZE 为单个种群的规模),非支配排序后的前沿层数为 k. 下面先进行时间复杂度分析.

1) 第 1 行代码中的预排序若采用堆排序或快速排序等高级排序,其时间复杂度为 $O(N\log N)$.

2) do while 循环(即按需排序)的时间复杂性: 生成第 1 层非支配前沿需要进行 $N-1$ 次支配比较; 生成第 2 层非支配前沿最多要进行 $N-2$ 次支配比较; ……; 生成第 k 层非支配前沿最多要进行 $N-k$ 次支配比较,所以按需排序的最坏时间复杂度为

$$\begin{aligned} T(N) &= \sum_{i=1}^{k} \text{第 } i \text{ 层的最多比较次数} \\ &= \sum_{i=1}^{k} N - i \\ &= N-1+N-2+\cdots+N-k < kN. \end{aligned}$$

最好情况下,当 R_t 中一半以上的个体为非支配个体时,$|F_1| \geqslant \text{POPSIZE}$,算法仅需分一层非支配前沿,则 $k = 1$. 最坏情况下,当每层非支配前沿恰好仅有一个个体时,则 $k = N/2$,但该情形出现的概率非常小. 一般来说,参数 k 的取值与 R_t 中非支配个体的比例(rate)密切相关. 当 rate 较大时,k 的值较小; 当 rate 较小时,k 值较大. 随着多目标进化算法进化代数的增加,种群中的 rate 值将快速地增大[5,11]. 相应地,k 的取值也将快速减小. 平均而言,一般有 $k < \log N$,则此时按需排序的时间复杂度 $T(N) < kN < N\log N$.

综合 1) 和 2),BNSA 的时间复杂度为 $O(N\log N)$.

下面再进行空间复杂度分析: 第 1 行代码中的预排序的空间复杂度一般不会超过 $O(N)$. 构造集 L 的最大空间需求为 N,支配个体集 Q 的最大空间需求为 $N-1$,非支配前沿 $F_1,F_2,\cdots,F_k$ 共需要的最大空间为 N. 综上所述,BNSA 算法的空间复杂度为 $O(N)$.

表 1　4 种算法的对比

Table 1　Comparison of 4 algorithms

	FNSA	APNS	NSTO	BNSA
时间复杂度	$O(N^2)$	$O(rN)$	$O(N\log N)$	$O(N\log N)$
空间复杂度	$O(N^2)$	$O(N)$	$O(N)$	$O(N)$
实现难易度	较难	较易	较易	容易

针对双目标优化问题,下面从时间复杂度、空间复杂度和编程实现难易程度这 3 个方面将以上 4 种非支配排序算法进行对比(难易度分析参见 FNSA、APNS、NSTO 的缺点以及 3.2 节的描述). 结果如表 1 所示,BNSA 在各项指标上都非常理想.

3.8　BNSA 算法的实例说明

例 1　考虑一个由 10 个个体组成的组合种群 R_t(R_t 的规模 $N = 10$,单个种群规模 POPSIZE = 5): $s_1 = (1,5)$, $s_2 = (2,10)$, $s_3 = (3,3)$, $s_4 = (4,5)$, $s_5 = (5,2)$, $s_6 = (6,7)$, $s_7 = (7,9)$, $s_8 = (8,1)$, $s_9 = (9,4)$, $s_{10} = (9,6)$.

个体 $s_i = (f_1, f_2)$ 具有两个目标,目标值分别为 f_1 和 f_2, $i = 1,2,\cdots,10$.

假设初始时

$$R_t = \{s_5, s_1, s_6, s_8, s_3, s_9, s_7, s_{10}, s_2, s_4\},$$

如图 1 所示,BNSA 的双目标非支配排序过程可描述如下.

预排序后构造集

$$L = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9, s_{10}\}.$$

第一次执行完 do while 循环体内代码后,生成前沿 $F_1 = \{s_1, s_3, s_5, s_8\}$,新的 $L = \{s_2, s_4, s_6, s_7, s_9, s_{10}\}$,进行了 9 次支配比较,已分层个体数 count = 4 < POPSIZE.

第二次执行完 do while 循环体内代码后,生成前沿 $F_2 = \{s_2, s_4, s_9\}$,新的 $L = \{s_6, s_7, s_{10}\}$,进行了 5 次支配比较. 此时 count = 7 > POPSIZE,返回非支配前沿 F_1, F_2,非支配排序结束.

两次循环共执行 14 次支配比较,小于 $kN(2 \times 10 = 20$,k 为分层数目),更小于 $N\log_2 N$ (> 30).

4　实验结果与分析

首先,本文将 NSTO、APNS、BNSA 这 3 种非支配排序算法集成到 NSGA - II 中(FNSA 本身已在 NSGA - II 中,无须集成),即分别将这 3 种算法替代

表 2 文献[3]中的 9 个标准双目标测试问题

Table 2 Nine bi-objective benchmark test problems used in reference [3]

问题	n	变量边界	目标函数
SCH	1	$[-10^3, 10^3]$	$f_1(x) = x^2$ $f_2(x) = (x-2)^2$
POL	2	$[-\pi, \pi]$	$f_1(x) = 1 + (A_1 - B_1)^2 + (A_1 - B_2)^2$ $f_2(x) = (x_1 + 3)^2 + (x_2 + 1)^2$ $A_1 = 0.5\sin 1 - 2\cos 1 + \sin 2 - 1.5\cos 2$ $A_2 = 1.5\sin 1 - \cos 1 + 2\sin 2 - 0.5\cos 2$ $B_1 = 0.5\sin x_1 - 2\cos x_1 + \sin x_2 - 1.5\cos x_2$ $B_2 = 1.5\sin x_1 - \cos x_1 + 2\sin x_2 - 0.5\cos x_2$
FON	3	$[-4, 4]$	$f_1(x) = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i - \frac{1}{\sqrt{3}}\right)^2\right)$ $f_2(x) = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i + \frac{1}{\sqrt{3}}\right)^2\right)$
KUR	3	$[-5, 5]$	$f_1(x) = \sum_{i=1}^{n-1}\left(-10\exp\left(-0.2\sqrt{x_i^2 + x_{i+1}^2}\right)\right)$ $f_2(x) = \sum_{i=1}^{n}\left(\lvert x_i\rvert^{0.8} + 5\sin x_i^3\right)$
ZDT 1	30	$[0, 1]$	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \sqrt{\frac{x_1}{g(x)}}\right]$ $g(x) = 1 + 9\frac{\left(\sum_{i=2}^{n} x_i\right)}{n-1}$
ZDT 2	30	$[0, 1]$	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \left(\frac{x_1}{g(x)}\right)^2\right]$ $g(x) = 1 + 9\frac{\left(\sum_{i=2}^{n} x_i\right)}{n-1}$
ZDT 3	30	$[0, 1]$	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \sqrt{\frac{x_1}{g(x)}} - \frac{x_1}{g(x)}\sin(10\pi x_1)\right]$ $g(x) = 1 + 9\frac{\left(\sum_{i=2}^{n} x_i\right)}{n-1}$
ZDT 4	10	$x_1 \in [0, 1]$ $x_i \in [-5, 5]$ $i = 2, \cdots, n$	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \sqrt{\frac{x_1}{g(x)}}\right]$ $g(x) = 1 + 10(n-1) + \sum_{i=2}^{n}\left[x_i^2 - 10\cos(4\pi x_i)\right]$
ZDT 6	10	$[0, 1]$	$f_1(x) = 1 - \exp(-4x_1)\sin^6(6\pi x_i)$ $f_2(x) = g(x)\left[1 - \left(\frac{f_1(x)}{g(x)}\right)^2\right]$ $g(x) = 1 + 9\left[\frac{\left(\sum_{i=2}^{n} x_i\right)}{n-1}\right]^{0.25}$

NSGA - II 的 FNSA. 然后比较 NSTO + NSGA - II、APNS + NSGA - II、BNSA + NSGA - II 和 NSGA - II 在非支配排序时的实验结果. 针对双目标优化问题，本文做了以下 3 类比较实验: 1) 最大进化代数和种群规模都不变时，计算 4 种算法双目标非支配排序时的关键比较次数[1)]、分层数目和运行时间; 2) 种

1) 关键比较主要是指支配比较，但是由于 NSTO 和 BNSA 算法中存在预排序，为公平起见，本文将预排序中的比较也视为关键比较. 此外，本文以关键比较次数作为评测标准，评估各种算法执行时的计算工作量.

群规模固定而最大进化代数变化时，计算 BNSA、NSTO 和 APNS 这 3 种算法相对于 FNSA 的加速比; 3) 最大进化代数固定而种群规模变化时，验证并对比 4 种算法的时间复杂度.

本文的实验使用文献 [6] 中的 9 个标准的双目标测试问题(问题描述参见表 2) . 这些测试问题的 Pareto 最优前沿的属性可以是凸和非凸、连续和非连续、均匀和非均匀等类型的组合，具有广泛的代表性. 所有的实验结果均为 25 次重复实验的统计结果. 实验参数按照文献 [6] 进行设置: 采用实数制染色体编码，模拟的二进制交叉和多项式变异，交叉概率 $p_c = 0.9$，变异概率 $p_m = 1/n$，n 为决策变量向量的维数，交叉的分布指数 η_c 和变异分布指数率 η_m 同为 20. 实验硬件环境为 Intel Core i3 2.93GHz CPU，4GB 内存的个人电脑，软件环境为 Windows XP 操作系统和 Visual C ++ 6.0 开发工具.

4.1　最大进化代数和种群规模不变时的实验对比

按照文献 [6] 设置最大进化代数为 250，单个种群规模(POPSIZE) 为 100，结果如表 3 所示. 表中的 rate 为组合种群的平均非支配个体比例，粗体数据表示该项取得最优结果. 从表 3 可以看出:

1) APNS 的比较次数和运行时间都要稍好于 FNSA 的结果，这与文献 [6] 的结论是一致的.

2) NSTO 与 BNSA 的比较次数和运行时间都要远好于 FNSA 和 APNS 的结果.

3) FNSA、APNS 和 NSTO 因同采用完全排序而分层数目基本相同. BNSA 因采用按需排序策略，其分层数目最少. BNSA 的分层数目和 rate 基本上是负相关的，譬如 rate 值越高时，其分层数目越少.

4) 从前 7 个测试问题来看，当 rate 超过 40% 时，BNSA 的分层数目都在 3 层以内，BNSA 的关键比较次数和运行时间都明显好于 NSTO 的结果. 从最后 2 个测试问题来看，当 rate 值在 30% 左右时，BNSA 分层数目大约在 3 ~ 4 层之间，NSTO 的关键比较次数和运行时间好于 BNSA 的结果. 可见，在比较 BNSA 和 NSTO 的性能时，BNSA 的分层数目(k 值) 是一个关键指标. 抛开两种算法中相同的预排序，NSTO 算法的主体部分的时间复杂度为 $O(N\log N)$; 而 BNSA 算法的主体部分的时间复杂度为 $O(kN)$. 显然，一般 rate 值越大时，k 值越小，$kN < N\log N$，BNSA 的结果比 NSTO 的结果更好. 总之，对于表 2 中大多数测试问题来说，BNSA 的结果都要好于 NSTO 的结果.

表 3　4 种算法在 9 个标准测试问题上的实验结果比较

Table 3　Comparison of results provided by 4 algorithms on 9 benchmark test problems

测试问题	rate/%	平均关键比较次数				平均分层数目				平均运行时间/ms			
		FNSA	APNS	NSTO	BNSA	FNSA	APNS	NSTO	BNSA	FNSA	APNS	NSTO	BNSA
SCH	88.58	16486.4	15974.7	2158.7	**1999.1**	14.16	14.17	14.16	**1.19**	4.46	3.45	0.35	**0.27**
POL	72.42	13534.6	12544.7	2156.9	**1981.7**	10.57	10.59	10.61	**1.12**	3.84	2.85	0.33	**0.30**
FON	64.91	12494.2	11429.3	2189.3	**2038.3**	8.70	8.72	8.71	**1.21**	3.60	2.64	0.40	**0.30**
KUR	58.14	11120.6	9917.1	2180.1	**2009.2**	12.14	12.15	12.12	**1.26**	3.44	2.57	0.42	**0.29**
ZDT1	52.76	15789.4	13589.8	2140.7	**2022.3**	5.27	5.29	5.26	**1.57**	4.50	2.98	0.42	**0.37**
ZDT2	45.46	14757.7	12632.8	2154.2	**2149.7**	7.30	7.26	7.67	**2.47**	4.43	3.08	0.43	**0.42**
ZDT3	52.02	15472.0	13194.4	2137.9	**2009.7**	6.17	6.18	6.17	**1.49**	4.29	2.95	0.37	**0.34**
ZDT4	30.85	10241.5	8642.5	**2169.0**	2272.9	16.84	16.67	16.72	**3.60**	3.94	3.29	**0.38**	0.43
ZDT6	34.37	12413.2	10134.2	**2133.9**	2212.1	14.65	14.60	14.59	**3.30**	4.35	3.27	**0.42**	0.44

4.2　种群规模固定而进化代数变化时的实验对比

根据平均非支配个体比例(rate) 为高、中、低 3 档，分别以 SCH、ZDT1 和 ZDT4 这 3 个测试问题为例，单个种群规模(POPSIZE) 固定为 100，最大进化代数的变化范围为 200 ~ 1 000，进行实验. 结果如图 4 所示. 图中的加速比定义为 FNSA 的关键比较次数除以相应算法的关键比较次数. 如果某算法加速比越高，则反映其加速性能越好.

总体来看，APNS 的加速比都稍高于 1 且变化幅度不大，即 APNS 算法的加速性能比 FNSA 算法稍好. 但是，APNS 算法的加速比远不及 BNSA 与 NSTO 的结果.

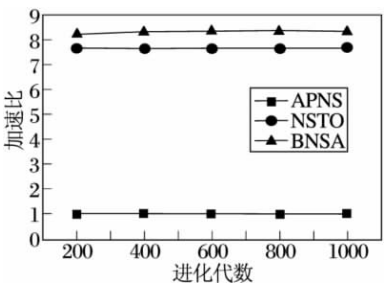
对于 rate 值最高的测试问题 SCH 而言，如图 4(a) 所示，BNSA 的加速比明显好于 NSTO 的结果. 随着进化代数的增加，两者的加速比变化不大，这是因为 20 代以后，SCH 问题的 rate 值基本上保持在 88% 左右，不再发生大的变化.

对于 rate 值适中的 ZDT1 问题而言，如图 4(b) 所示，BNSA 的加速比都要好于 NSTO 的加速比. 此外，NSTO 的加速比呈现略微下降的趋势，而 BNSA

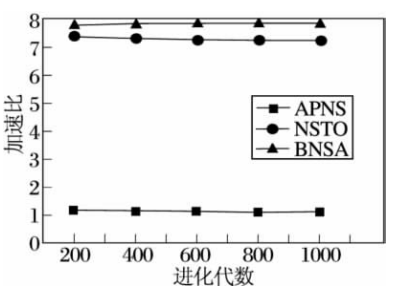
算法的加速比却呈略微上升的趋势.

对于 rate 值最低的测试问题 ZDT4 而言,如图 4(c)所示,尽管进化初期 NSTO 算法的加速效果比 BNSA 稍好,但是随着进化代数的增加,BNSA 算法的加速比却呈快速上升趋势. 当进化代数超过 400 代后,BNSA 的加速比明显好于 NSTO 的结果.

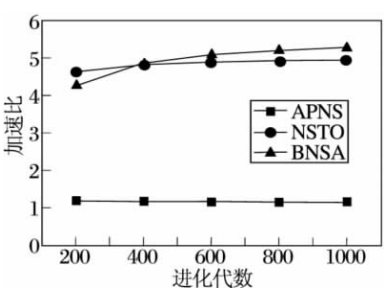
总之,随着进化代数增长,BNSA 在所有的 9 个测试问题上都具有最好的加速效果.



(a) SCH



(b) ZDT1



(c) ZDT4

图 4 3 种算法在 3 个标准测试问题上的加速比

Fig. 4 Speedups comparison of 3 algorithms on 3 benchmark test problems

4.3 进化代数固定而种群规模变化时的实验对比

为了验证各种算法的时间复杂度,以 ZDT1 为例,进化代数固定为 250 代,组合种群规模(N)的变化范围为 200 ~ 1 000,进行实验,结果如图 5 所示. 各种算法的实验数据位于 $O(N^2)$ 和理论下界 $O(N\log N)$ 之间. 其中 FNSA 增长幅度最大,验证了其时间复杂度最高; APNS 的增幅略低于 FNSA; 而 NSTO 和 BNSA 的结果几乎与下界 $O(N\log N)$ 重合,验证了两者的时间复杂度同为 $O(N\log N)$.

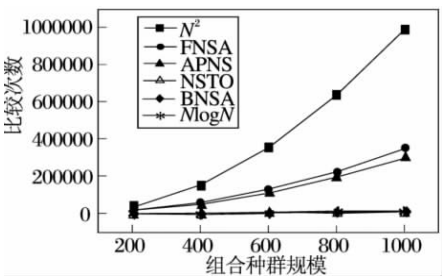


图 5 4 种算法时间复杂度的实验结果

Fig. 5 Time complexity of 4 algorithms by experimental data

5 结 束 语

非支配排序是基于 Pareto 支配的 MOEA 中最为重要的一种个体等级分类方法. 快速的非支配排序算法往往能极大地提高 MOEA 的运行速度. 本文针对双目标优化这一常见的多目标优化形式,提出了一种快速的双目标非支配排序算法(BNSA). 设计了前向比较操作以便快速识别非支配个体. 提出了按需排序的策略以避免生成多余的非支配前沿. 论证了 BNSA 的正确性,分析了其时间复杂度为 $O(N\log N)$. 实验结果表明: 当最大进化代数在 250 代以内时,BNSA 在大多数测试问题上比其它三种非支配排序算法具有更快速的性能. 当进化代数超过 400 代时,BNSA 在所有的测试问题上都具有最好的加速效果. 此外,BNSA 比其它三种算法简明,易于实现,更有利于实际双目标优化问题的求解. 但是,最后需要指出 BNSA 目前仅局限于双目标优化问题. 至于为三个目标的优化问题设计出快速的非支配排序算法,以及通过引入用户偏好为高维优化问题设计出相应的非支配排序算法是我们未来研究工作的重点.

参 考 文 献

[1] Coello C A, Lamont G B, Van Veldhuizen D A. Evolutionary Algorithms for Solving Multi-Objective Problems. 2nd Edition. New York, USA: Springer-Verlag, 2007

[2] Gong Maoguo, Jiao Licheng, Yang Dongdong, *et al*. Research on Evolutionary Multi-Objective Optimization Algorithms. Journal of

Software, 2009, 20(2): 271 – 289 (in Chinese)
(公茂果,焦李成,杨咚咚,等. 进化多目标优化算法研究. 软件学报, 2009, 20(2): 271 – 289)

[3] Deb K, Pratap A, Agarwal S, *et al*. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA – II. IEEE Trans on Evolutionary Computation, 2002, 6(2): 182 – 197

[4] Jensen M T. Reducing the Run-Time Complexity of Multiobjective EAs: The NSGA – II and Other Algorithms. IEEE Trans on Evolutionary Computation, 2003, 7(5): 503 – 515

[5] Zheng Jinhua, Jiang Hao, Kuang Da, *et al*. An Approach of Constructing Multi-Objective Pareto Optimal Solutions Using Arena´s Principle. Journal of Software, 2007, 18(6): 1287 – 1297 (in Chinese)
(郑金华,蒋浩,邝达,等. 用擂台赛法则构造多目标 Pareto 最优解集的方法. 软件学报, 2007, 18(6): 1287 – 1297)

[6] Berrichi A, Amodeo L, Yalaoui F, *et al*. Bi-Objective Optimization Algorithms for Joint Production and Maintenance Scheduling: Application to the Parallel Machine Problem. Journal of Intelligent Manufacturing, 2009, 20(4): 389 – 400

[7] Jozefowiez N, Semet F, Talbi E G. An Evolutionary Algorithm for the Vehicle Routing Problem with Route Balancing. European Journal of Operational Research, 2009, 195(3): 761 – 769

[8] Abido M A. Multiobjective Evolutionary Algorithms for Electric Power Dispatch Problem. IEEE Trans on Evolutionary Computation, 2006, 10(3): 315 – 329

[9] Deb K, Datta R. A Hybrid Bi-Objective Evolutionary-Penalty Approach for Computationally Fast and Accurate Constrained Optimization. Technical Report, 2010004. Kanpur Genetic Algorithms Laboratory, Indian Institute of Technology. Kanpur, India, 2010

[10] Deb K, Saha A. Multimodal Optimization Using a Bi-Objective Evolutionary Algorithm. Technical Report, 2009006. Kanpur Genetic Algorithms Laboratory, Indian Institute of Technology. Kanpur, India, 2009

[11] Deb K, Goel T. Controlled Elitist Non-Dominated Sorting Genetic Algorithms for Better Convergence // Zitzler E, Deb K, Thiele L, *et al*, eds. Proc of the 1st International Conference on Evolutionary Multi-Criterion Optimization. Zurich, Switzerland, 2001: 67 – 81