一个快速和精英机制的多目标遗传算法

Kalyanmoy Deb, Associate Member, IEEE, Amrit Pratap, Sameer Agarwal, T. Meyarivan

***摘要:*应用非支配排序的多目标进化算法被广为评判，主要是因为：1）$O(MN^3)$计算复杂度（其中 M 代表目标个数，N 代表种群个数）2）非精英机制方法；还有 3）需要指定一个共享参数。本文中，我们提出了一个基于非支配排序的多目标进化算法（MOEA），称为第二代非支配排序进化算法（NSGA-II），它缓解了以上三个难点。特别要明确指出的是，一个计算复杂度只有$O(MN^2)$的快速非支配排序方法被提出。还有，一个通过结合父代和子代种群以及选择最佳解决方法（根据适应度和扩展性）创建交配池的选择算子被提出。对不同的测试问题进行的模拟仿真结果表明所提出的 NSGA-II，在大多数问题中，与其他进化策略和强性 Pareto 进化算法——两个注重创造具有多样性 Pareto 占优前沿面的精英机制的多目标进化算法——相比能找到相对扩展性较好的解以及更能收敛于实际 Pareto 占优的前沿面。另外，为了高效解决约束多目标优化问题，我们修改了支配的定义。对一些测试问题，包括 5 目标、7 约束的非线性问题，将约束 NSGA-II 算法的模拟仿真结果与另一个约束多目标优化算法相比，NSGA-II 的性能明显更好。**

***索引词* — 约束处理，精英机制，遗传算法，多重判据决策，多目标优化，Pareto 占优法**

I. 引言

原则上，在一个问题中存在多个目标会产生一个最优解集合（广范知晓的是 Pareto 占优解法），而不仅仅是一个最优解。没有任何进一步的信息，不可以认为一个 Pareto 占优的解法会比其它解法好。这需要用户找到尽可能多 Pareto 占优的解。经典优化方法（包括多重判据决策方法）建议通过每次强调一个特有的 Pareto 占优解将多目标优化问题转化为单目标优化问题。当这种方法被用来寻找多个解时，在每次模拟运行时它必须运行多次才有希望找到不同的解。

在过去几年，一些多目标进化算法(MOEAs)被相继提出[1],[7], [13],[20],[26].其主要原因是它们能够在一次单一的模拟运行中找到多个 Pareto 占优解。因为进化算法(EAs)以种群方式运行，所以一个简单的进化算法可以被扩展到保持不同解的集合。以向 Pareto 占优区域移动为重点，那么在一次单一模拟运行下一个进化算法可以被用来找到多个 Pareto 占优解。

非支配排序遗传算法(NSGA)[20]正是首批基于这种思想的算法之一。多年来，对 NSGA 方法的主要批判有如下几点。

1） 支配排序的高计算复杂度：当前使用的支配排序算的计算复杂度为$O(MN^3)$（其中 M 代表目标函数的数目，N 代表种群的数目）。对大的种群数目会使得 NSGA 的计算非常昂贵。这种大复杂度的起因是由于在每一子代中都包含非支配排序程序。）

2） 缺少精英机制：近来结果[25], [18]表明精英机制能够很大程度上提高遗传算法的性能，同时还能够防止较优解一旦被发现后丢失的情况。

3） 需要指定共享参数：传统的种群多样性保存机制为了得到多种多样等价的解则非常依赖共享的概念。共享的主要问题是它需要指定共享参数(σ_{share})。即使已经存在一些共享参数动态大小调整的工作[10]，但是一种无参的多样性保持机制仍然很需要。

本文中，我们致力于所有这些问题并提出了一个 NSGA 的改进版本，我们称之为 NSGA-II。从对一些困难测试问题的仿真结果上看，就找到一个多样性解的集合以及收敛于真实 Pareto 最优解集合的性能来看，我们能够发现 NSGA-II 优于同时期的其他两个多目标优化算法 MOEAs：Pareto 归档进化策略(PAES)[14]和强化 Pareto 进化算法(SPEA)[24]。

约束多目标优化对实际要解决的问题来说是非常重要的，但是至今在进化算法领域并没有得到足够的重视。本文中，我们根据 NSGA-II 提出了一个适合任何遗传算法的简单约束处理策略。在文献中选择出来的四个问题测试，NGSA-II 被用来和另一个当前提出的约束处理策略做对比。实验结果表明 NSGA-II 可以被鼓励应用于更多复杂的以及现实世界中的多目标优化问题。

在文章的剩余部分，在第二章节我们简略地提到一些存在精英机制的多目标进化算法。然后，在第三章节，我们详细描述了所提出的 NSGA-II 算法。第四章节展示了 NSGA-II 的仿真结果并与其它两种多目标优化算法(PAES 和 SPEA)对比。在第五章节，我们重点强调了参数相互作用的问题，它是在进化计算方向非常重要的问题。下一章节将 NSGA-II 扩展到处理约束问题并将结果与当前另一个提出的约束处理方法做对比。最后，我们概括了本文的结论。

II. 精英机制的多目标进化算法

1993-1995 年期间，一些不同的进化算法被提出用来解决多目标优化问题。在它们之中，Fonseca 和 Fleming 的 MOGA[7], Srinivas 和 Deb 的 NSGA[20],以及 Horn et al.的 NPGA [13]得到了更多的关注。这些算法展示了将 EA 转化为 MOEA 的附加算子。以上三个算子有如下两个共同特征：i）基于非支配排序给种群成员指定适应度 ii)在同一支配前沿面的解中保持多样性。即使他们已经被证明在很多测试问题以及一些工程设计问题上可以找到多个非支配解，研究者发现需要引进更多有用的算子（在单目标进化算法上已被证明有用）来更好地解决多目标优化问题。特别地，这种想法致使精英机制被引进用来提高 MOEA 的收敛性能。文献[25]表明经营机制能帮助 MOEAs 实现更好地收敛性能。在已经存在的精英机制 MOEAs，Zitzler 和 Thiele 的 SPEA [26], Knowles 和 Corne 的 Pareto 归档 PAES [14],以及 Rudolph 的精英机制 GA [18]得到充分地研究。我们简单描述了这些方法。细节部分，读者可自行参考原研究。

Zitzler 和 Thiele 在他们的 SPEA 中提出了一个带有非支配概念的精英机制多重判据进化算法。他们建议在每一子代中维持一个外部种群用于保存所有从开始至今找到的非支配解。这个外部种群参加每一次的遗传操作。在每一子代中，一个由外部和当前种群合成的种群首先被构造出来。在合成种群中所有非支配解都会根据他们所支配的解的数目被指派一个适应度值，并且被支配的解会被指派一个比当前最坏适应度值更坏的值作为适应度。这种适应度的分配保证搜索是直接朝向非支配解的。一种确定性的聚类技术被用来在非支配解中保证多样性。即使在文献[26]中提出的方法实现需要$O(MN^3)$，但是通过合适的保存方法 SPEA 的复杂度就会较少到$O(MN^2)$。

Knowles 和 Corne [14] 提出了一个使用与 (1+1)-进化策略相似的单父代单子代进化算法的简单的 MOEA。相反不使用实参，而是使用二进制串和按位变异来产生子代。在他们只有一个父代和一个子代的 PAES 中，子代被用来与父代比较。如果子代支配父代，那么子代就被接受作为下一次父代并且迭代继续。另一方面，如果父代支配子代，那么子代被抛弃并且一个新的变异解（新子代）被找到。然而，如果子

代和父代都不互相支配，那么通过比较他们的目前找到最优解的存档来决定选择父代或子代。将子代与存档对比来检查它是否支配存档中的成员。如果存在支配情况，那么子代就被接受作为新的父代并且将所有的支配解从存档中剔除掉。如果子代没有支配存档中的成员，那么检查父代和子代与存档中其他解的接近程度。如果子代在存档的成员中处于目标空间最不拥挤的区域，那它就被接受作为父代，并拷贝加入存档中。通过将整个搜索空间准确划分为 d^m 子空间（其中 d 是深度参数，n 是决策变量的数目）以及动态更新子空间来保持拥挤度。研究者已经计算出 PAES 最坏的复杂度为 $O(\alpha MN)$ 对 N 个评估来说，其中 α 是存档的长度。由于存档大小经常按种群大小 N 的比例选择，所以总体上算法的复杂度为 $O(MN^2)$。

Rudolph [18]提出了，但是并没有仿真，一个基于系统的来自父代与子代种群个体比较的简单精英机制 MOEA。将子代非支配解通过与父代解对比建立总体非支配解的集合，并将它作为下一次迭代的父代种群。如果这个集合的大小不比期望的种群大，那么来自子代种群的其他个体就被包括进来。通过这个策略，他证明了这个算法对 Pareto 最优前沿面的收敛性。即使对它自己来说这是一个重要的成功，但是算法缺少对第二个保持 Pareto 多样性解任务的推动性。一个明确的多样性保留机制一定是更加实用的。由于第一个非支配前沿面的确定需要 $O(MN^2)$，所以 Rudolph 算法整体的复杂度也为 $O(MN^2)$。

接下来，我们展示了所提出的非支配排序 GA 方法，它使用了快速非支配排序步骤，精英保留方法以及一个无参小生境算子。

III. 精英机制的非支配排序遗传算法

A.快速非支配排序方法

由于描述清楚的缘故，我们首先描述了一个将种群排序为不同支配等级的简单并且缓慢的步骤。然后，我们描述了一个快速的方法。在这个简单的方法中，为了能在大小为 N 的种群中确定第一非支配前沿面的解，每一个解都被用来和其它种群中的解来对比来判断它是否被支配。对每个解需要 $O(MN)$ 次比较，其中 M 是目标的个数。持续这个程序直到找到种群中所有第一支配等级的成员，全部复杂度为 $O(MN^2)$。现阶段所有在第一非支配前沿面的个体都被找到。为了找到下一支配前沿面的个体，第一前沿面的解被暂时打折并且以上程序被重复进行。在最坏的情况下，找到第二前沿面同样需要 $O(MN^2)$ 次比较，特别当 $O(N)$ 个解属于第二或者更高的非支配等级。这个证据对找到第三和更高等级的非支配面同样正确。因此，最坏的情况是有 N 个前沿面并且每一个前沿面只有一个解。这总共需要 $O(MN^3)$ 次比较。要注意的是程序需要 $O(N)$ 的储存空间。在接下来的段落和页底所示的方程中，我们描述了一个只需要 $O(MN^2)$ 比较的快速非支配排序方法。

首先，对每一个解我门计算两个实体：1）支配计数 np，即支配着解 的解的数量，还有 2）Sp，解所支配的解的集合，这需要 $O(MN^2)$ 次比较。

所有第一非支配前沿面解的支配计数都为零。现在，对每一个解 p 都有 $np=0$，我们访问每一成员（q）和他的集合 Sq 并且减少逐一减少支配计数。通过这样，如果任何成员 q 的支配计数达到 0，我们就把它放进一个单独集合 Q。这些成员属于第二非支配前沿面。现在，以上程序应用 Q 中的每一成员继续执行直到第三前沿面被确定。这一过程持续到所有前沿面被确定。

对每一个在第二或更高非支配等级的解 p 来说，支配计数 np 最多为 N-1。因此，每一个解 p 在它的支配计数达到零时最多被访问 N-1 次。基于这一点，一个被指定非支配等级的解不会被再次访问。由于存在最多这样的解有 N-1 个，所以全部复杂度为 $O(N^2)$。因此整体的程序复杂度为 $O(MN^2)$。另一种计算复杂度的方法，第一内部循环的整体（对每一个 $p \in F_i$）被执行 N 次同时每一个体之多成为一个前沿面的成员，在第二层内循环中（对每一 $q \in Sp$）对每一个体来说被最多执行（N-1）次（每一个体最多支配(N-1)个体并且每次支配检查需要至多 M 比较）。这样的结果就是整体需要 $O(MN^2)$ 次比较。需要注意的是虽然时间复杂度减少到了 $O(MN^2)$，但存储空间需要增加到 $O(N^2)$。）

B.保留多样性

我们之前提到，EA 有收敛于 Pareto 占有集合的性能，在

```
fast-non-dominated-sort(P)
for each p ∈ P
  S_p = ∅
  n_p = 0
  for each q ∈ P
    if (p ≺ q) then                    If p dominates q
      S_p = S_p ∪ {q}                  Add q to the set of solutions dominated by p
    else if (q ≺ p) then
      n_p = n_p + 1                    Increment the domination counter of p
  if n_p = 0 then                      p belongs to the first front
    p_rank = 1
    F_1 = F_1 ∪ {p}
i = 1                                  Initialize the front counter
while F_i ≠ ∅
  Q = ∅                                Used to store the members of the next front
  for each p ∈ F_i
    for each q ∈ S_p
      n_q = n_q − 1
      if n_q = 0 then                  q belongs to the next front
        q_rank = i + 1
        Q = Q ∪ {q}
  i = i + 1
  F_i = Q
```

已获得解的集合中保持很好的扩展性也是 EA 所需要的。NSGA 原型使用了著名的共享函数方法，这种方法已经被证明在相关参数设定合理的情况下能够在种群中找到很好的多样性。共享函数的方法包括一个共享参数 $\sigma share$，它用来设置一个问题中要求的共享程度。这个参数与被选择计算两个种群成员的接近程度的距离度量标准相关。这个参数 $\sigma share$ 表明在任意两个解间最大的距离测度会共享它们的适应度值。这个参数通常由用户来设置，即使在文献[4]中有一些指导方法，但是共享函数方法仍存在两个难点。

1） 共享函数方法在维持解的传播性能上过分依赖于 $\sigma share$ 值的选择。

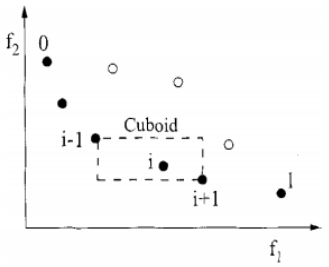

图 1 拥挤距离计算。实心点代表同一非支配前沿面的解

2） 由于每个解必须与种群中其它所有解比较，共享函数方法的总体复杂度为 $O(N^2)$。

在所提出的 NSGA-II 中，我们用拥挤比较的方法替换了共享函数方法，它在一定程度上排除了以上两个难点。这个新方法不需要任何用户定义维持种群多样性的参数。同时，所提出的方法有更好的复杂度。为了描述这种方法，我们定义了密度估计准则并展示了拥挤比较算子。

1) 密度估计：为了得到种群中特定解周围的解的密度估计，我们根据每一目标函数计算这点两侧的两个点的平均距离。这个数值作为以最近邻居作为顶点的长方体周长的估计（称为拥挤系数）。在图 1 中，第 i 个解在它所在前沿面的拥挤系数是它周围长方体的长度（如虚线框所示）。

拥挤系数的计算需要根据每一目标函数值的大小的升序顺序对种群进行排序。因此，对每一目标函数，边界解（拥有最大和最小值的解）被指定为无穷大距离的值。所有其它中间的解都被指定为等于两个相邻解的函数值归一化后的绝对差值。计算方法对其它目标函数也是这样。全部拥挤系数值是通过个体每一目标的距离值的加和计算得到的。每一目标函数在计算拥挤系数前都会进过归一化处理。在页底展示的算法概括了非支配集合 I 中所有解拥挤系数的计算过程。

这里，$\mathrm{I}[i].m$ 代表集合 I 中第 i 个个体的第 m 个目标函数值，参数 $f_m^{\max}$ 和 $f_m^{\min}$ 是第 m 个目标函数的最大和最小值。这个过程的复杂度主要有排序算法占据。由于有 M 次独立的排序，每次最多有 N 个解（当多有种群成员都在一个前沿面中）包括进来，上述算法的计算复杂度为 $O(MN\log N)$。

在集合 I 中所有种群成员都被指定了一个距离度量，我们就能够对比两个解与其它解的接近程度。一个拥有更小的距离度量值的解，在一定程度上，会被其它解挤到。这就是下面描述中我们在拥挤比较算子中所比较的。虽然图 1 展示了两个目标的拥挤距离的比较，但是这个方法同样适用于多于两个目标的情况。

2）拥挤比较算子：拥挤比较算子在算法的不同阶段将选择的过程导向均匀分布的 Pareto 最优前沿面。假设种群中每一个体有两个属性：

1）非支配排名 $(irank)$;

2）拥挤系数 $(idistance)$.

现在我们定义一个偏序 $\prec_n$ 如下：

$$i \prec_n j \quad \text{如果} \quad (i_{rank} < j_{rank})$$
$$\text{或} \quad (i_{rank} = j_{rank})$$
$$\text{并且} (i_{distance} > j_{distance})$$

就是说，在两个有不同非支配排名解中，我们更喜欢拥有更低（更好）排名的解。否则如果两个解属于同一前沿面，那么我们更喜欢处于相对不太拥挤区域的解。

介绍了一这三个创新点——快速非支配排序程序，快速拥挤系数估计程序以及简单的拥挤比较算子，接下来我们准备开始描述 NSGA-II 算法。

C. 主循环

最初随机创建父代种群 P_0。种群根据非支配情况排序。每一个解都会被指定一个等于它自身非支配等级(1 代表最高等级，2 次之，依次类推）的适应度值（或者排名）。就这样，假设了适应度的最小值。首先，常用的二进制锦标赛机制选择，重组以及变异算子被用来生成大小为 N 的子代种群 Q_0。由于精英是通过比较当前种群和先前找到的最优非支配解而引进的，所以这个过程在初始代是不同的。我们首先在页底描述了所提出算法的第 t 代。

这一步接一步的程序说明 NSGA-II 算法是简单直接的。首先，合成种群 $R_t = R_t \bigcup Q_t$ 的形成。种群 R_t 大小为 N。然后种群根据非支配度排序。由于先前和当前种群都在 R_t 中，则精英也就确定了。现在属于最优非支配集合 F_1 的解是合成种群中最优的解，并且相对合成种群中的其它解要更加重视它们。如果的 F_1 大小比 N 小，我们就明确的将集合 F_1 所有成员归为新种群 P_{t+1} 中。种群 P_{t+1} 剩余的成员可以在后来按排名排序的的非支配前沿面选择。因此，接下来 F_2 中的解被选择出来，然后是 F_3 中的解，依次进行。这个过程一直持续到没有更多的集合可以被供应。例如 F_1 是最后的非支配前沿面，再后来就没有其它集合可以被供应。事实上，在 F_1 到 F_l 所有集合解的数目会比种群的大小要大。要选择出刚好 N 个种群成员，我们通过拥挤比较算子将最后前沿面 F_l 中的解按降序排列并且选择最好的姐来填充全部种群的空隙。NSGA-II 的步骤同样如图 2 所示。大小为的 N 新种群现在被用来进行选择

---

crowding-distance-assignment($\mathcal{I}$)
$l = |\mathcal{I}|$ — number of solutions in $\mathcal{I}$
for each i, set $\mathcal{I}[i]_{\text{distance}} = 0$ — initialize distance
for each objective m
  $\mathcal{I} = \text{sort}(\mathcal{I}, m)$ — sort using each objective value
  $\mathcal{I}[1]_{\text{distance}} = \mathcal{I}[l]_{\text{distance}} = \infty$ — so that boundary points are always selected
  for $i = 2$ to $(l-1)$ — for all other points
    $\mathcal{I}[i]_{\text{distance}} = \mathcal{I}[i]_{\text{distance}} + (\mathcal{I}[i+1].m - \mathcal{I}[i-1].m)/(f_m^{\max} - f_m^{\min})$

交叉和变异操作从而产生新的大小为 N 的种群。特别需要注意的是我们使用了二进制锦标赛选择操作但是如今的选择标准是基于拥挤比较算子 $\prec_n$。由于这个算子需要种群中解的排名以及拥挤系数，我们在建立新种群 P_{t+1} 后会计算这些属性值，具体算法如上所示。

考虑整个算法一次迭代的计算复杂度。主要操作以及最坏情况下的复杂度如下所示：

1） 非支配排序 $O(M(2N)^2)$
2） 拥挤系数指定 $O(M(2N)\log(2N))$
3） $\prec_n$ 上的排序 $O(2N\log 2N)$

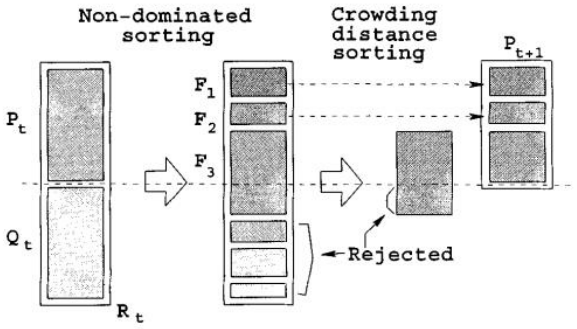

图 2 NSGA-II 步骤

算法整体上的复杂度为 $O(MN^2)$，其中非支配排序占据了主导地位。如果执行仔细得当，大小为 $2N$ 完整种群不需要在根据非支配情况排序。只要排序过程已经找到在前沿面中足够多的解，那么就没有理由继续执行排序操作。）

非支配解的多样性通过使用拥挤比较程序维持，它被应用于竞标赛选择以及种群减少阶段。由于解是通过它们的拥挤系数(一种对解的周围邻居密度的测量值)，不需要额外的小生境系数（例如 NSGA 算法中的 σ_{share}）。即使拥挤系数是在目标函数空间计算的，如果需要的话它同样可以应用于参数空间。然而在所有本次研究的模拟中，我们使用的是目标函数空间小生境。

IV. 模拟仿真结果

这本章节中，我们首先描述了用来比较 NSGA-II 和 PAES 以及 SPEA 性能的测试问题。至于 PAES 和 SPEA，我们依据原始研究来确定参数设置。至于 NSGA-II，我们选择了一组合理值的集合但是并没有试图找到最佳的参数设置。我们将这个任务留作后续研究。

A。测试问题

首先我们描述用来比较个汇总 MOEAs 的测试问题。测试问题是从这个领域过去有重要意义研究中挑选出来的。

表 1
本次研究中用到的测试问题

Problem	n	Variable bounds	Objective functions	Optimal solutions	Comments
SCH	1	$[-10^3, 10^3]$	$f_1(x) = x^2$ $f_2(x) = (x-2)^2$	$x \in [0, 2]$	convex
FON	3	$[-4, 4]$	$f_1(\mathbf{x}) = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i - \frac{1}{\sqrt{3}}\right)^2\right)$ $f_2(\mathbf{x}) = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i + \frac{1}{\sqrt{3}}\right)^2\right)$	$x_1 = x_2 = x_3$ $\in [-1/\sqrt{3}, 1/\sqrt{3}]$	nonconvex
POL	2	$[-\pi, \pi]$	$f_1(\mathbf{x}) = \left[1 + (A_1 - B_1)^2 + (A_2 - B_2)^2\right]$ $f_2(\mathbf{x}) = \left[(x_1+3)^2 + (x_2+1)^2\right]$ $A_1 = 0.5\sin 1 - 2\cos 1 + \sin 2 - 1.5\cos 2$ $A_2 = 1.5\sin 1 - \cos 1 + 2\sin 2 - 0.5\cos 2$ $B_1 = 0.5\sin x_1 - 2\cos x_1 + \sin x_2 - 1.5\cos x_2$ $B_2 = 1.5\sin x_1 - \cos x_1 + 2\sin x_2 - 0.5\cos x_2$	(refer [1])	nonconvex, disconnected
KUR	3	$[-5, 5]$	$f_1(\mathbf{x}) = \sum_{i=1}^{n-1}\left(-10\exp\left(-0.2\sqrt{x_i^2 + x_{i+1}^2}\right)\right)$ $f_2(\mathbf{x}) = \sum_{i=1}^{n}\left(\lvert x_i\rvert^{0.8} + 5\sin x_i^3\right)$	(refer [1])	nonconvex
ZDT1	30	$[0, 1]$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x})\left[1 - \sqrt{x_1/g(\mathbf{x})}\right]$ $g(\mathbf{x}) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)/(n-1)$	$x_1 \in [0, 1]$ $x_i = 0,$ $i = 2, \ldots, n$	convex
ZDT2	30	$[0, 1]$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x})\left[1 - (x_1/g(\mathbf{x}))^2\right]$ $g(\mathbf{x}) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)/(n-1)$	$x_1 \in [0, 1]$ $x_i = 0,$ $i = 2, \ldots, n$	nonconvex
ZDT3	30	$[0, 1]$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x})\left[1 - \sqrt{x_1/g(\mathbf{x})} - \frac{x_1}{g(\mathbf{x})}\sin(10\pi x_1)\right]$ $g(\mathbf{x}) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)/(n-1)$	$x_1 \in [0, 1]$ $x_i = 0,$ $i = 2, \ldots, n$	convex, disconnected
ZDT4	10	$x_1 \in [0, 1]$ $x_i \in [-5, 5],$ $i = 2, \ldots, n$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x})\left[1 - \sqrt{x_1/g(\mathbf{x})}\right]$ $g(\mathbf{x}) = 1 + 10(n-1) + \sum_{i=2}^{n}\left[x_i^2 - 10\cos(4\pi x_i)\right]$	$x_1 \in [0, 1]$ $x_i = 0,$ $i = 2, \ldots, n$	nonconvex
ZDT6	10	$[0, 1]$	$f_1(\mathbf{x}) = 1 - \exp(-4x_1)\sin^6(6\pi x_1)$ $f_2(\mathbf{x}) = g(\mathbf{x})\left[1 - (f_1(\mathbf{x})/g(\mathbf{x}))^2\right]$ $g(\mathbf{x}) = 1 + 9\left[\left(\sum_{i=2}^{n} x_i\right)/(n-1)\right]^{0.25}$	$x_1 \in [0, 1]$ $x_i = 0,$ $i = 2, \ldots, n$	nonconvex, nonuniformly spaced

所有的目标函数都是求最小化。

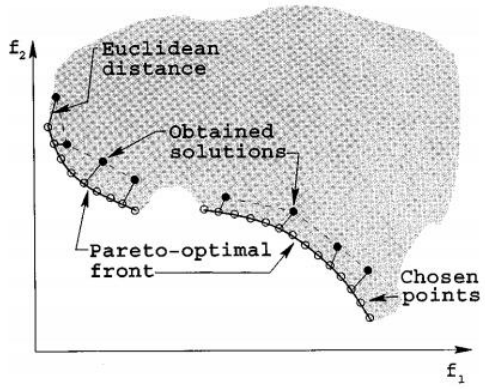

图 3 距离度量标准 ϒ

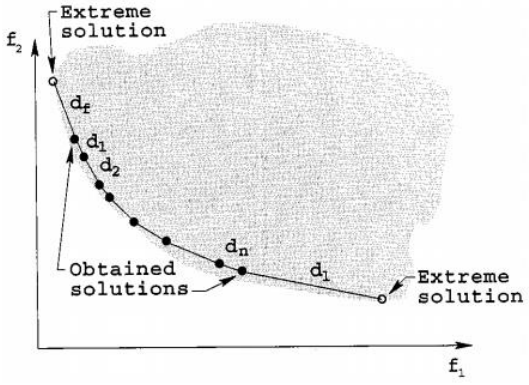

图 4 多样性度量标准 Δ

Veldhuizen [22]引用了一些过去被使用过的测试问题。在他们之中，我们选择了四个问题：Schaffer 的研究 (SCH) [19], Fonseca 和 Fleming 的研究 (FON) [10], Poloni 的研究 (POL) [16]，以及 Kursawe 研究 (KUR) [15]. 1999 年，第一次有作者提出了一种为多目标优化问题开发测试问题的系统方法[3]。Zitzler et al.[25]根据这些指导方针并提出了六个测试问题。我们选择这六个问题其中五个并称之为 ZDT1, ZDT2, ZDT3, ZDT4, 以及 ZDT6。所有这些问题都有两个目标函数。这些问题全都没有约束条件。在表格 1 中我们描述了这些问题。表格中同样给出了每个问题变量的个数，它们的界限，最优 Pareto 解以及原始 Pareto 最优前沿面。

所有的方法运行最多进行 25000 次函数评价。我们使用二进制编码 GAs 的单点交叉和按位变异以及实数编码 GAs 的模拟二进制交叉（SBX）算子和多项式变异[6]。交叉概率设为 $p_c = 0.9$，变异概率设为 $p_m = 1/n$ 或 $1/l$ (其中 n 是实数编码决策变量的数目，l 是二进制编码的字串长度)。对于实数编码的 NSGA-II，我们是使用分别设交叉以及变异算子为 $\eta_c = 20$ 且 $\eta_m = 20$ 分布索引。在经过 250 次迭代后得到的种群（种群经过了精英保留机制）被用来计算两个我们接下来会讨论的性能评价指标。对于 PAES，我们设定深度值 d 为 4，存档长度 a 为 100。我们使用存档中经过 25000 次迭代得到的种群成员来计算性能评价指标。对于 SPEA，我们设定种群大小为 80，外部种群大小为 20（这个 4:1 比例是由开发者提出，用来维持求精英解而需要的足够多的选择压力），所以种群的整体大小变为 100。SPEA 同样运行 25000 函数评价结束。对于 SPEA，我们在最后一代中使用合成 GA 与外部种群的非支配解来计算性能度量指标。对 PAES，SPEA 以及二进制编码 NSGA-II，我们设定决策变量长度为 30。

B. 性能测量

与单目标优化不同，在多目标优化中有两个目的：1）收敛于 Pareto 占优集合；2）维持 Pareto 占优集合解的多样性。这两个任务不能被一个性能评价标准充分测量。许多的性能度量标准在文献[1], [8], [24]中被提出。这里，我们定义了两个性能度量标准并且他们可以非常直接地对经过多目标优化得到的解集合进行评估。

第一个度量值 ϒ 测量了已知 Pareto 占优解集合的收敛程度。由于多目标优化算法会在一些已知 Pareto 占优解集的问题上测试，所以这个度量值得计算是可以实现的。然而我们认为，这样一个度量标准不能够被用于任何问题。首先，我们找到在目标空间中真实 Pareto 占优前沿面上的均匀间隔分布的 $H = 500$ 的集合。对于通过算法获得的每一个解，我们都会计算距离从 Pareto 占优前沿面挑选出的 H 集合中解的最小欧氏距离。这些距离的平均值作为第一个度量标准 ϒ（收敛性标准）。图 3 展示了这个度量值得计算过程。阴影区域是可行解搜索区域，实曲线制定了 Pareto 占优解。带开口圆圈的解是 H 在 Pareto 前沿面挑选出用来计算收敛指标的解。黑色圆圈代表算法得到的解。这个度量值越小，越收敛于 Pareto 占优前沿面。当所有的解刚好在 H 选择的解上，这个度量值的值为零。在这里运行的所有仿真，我们都会呈现经过多次运行之后的平均值 $\bar{\Upsilon}$ 以及方差 σ_Υ。

即使当所有的解都收敛于 Pareto 占优前沿面，以上的收敛度量值也不能达到零。这个度量值只有当每个获得的解刚好落在所选择的解时才能达到零。虽然这个度量值能单独提供一些获得解的传播性信息，我们还定义了一个不同的度量值来直接测量算法所得到解的扩展性。第二个度量值 Δ 能测量已获得解的扩展性能程度。这里我们更希望得到能够跨越整个 Pareto 占优区域的解的集合。我们计算在已获得的非支配解集合的两个连续解间的欧氏距离。计算这些距离的平均值 $\bar{d}$。其后，从获得的非支配解集中，我们首先通过拟合一条平行于真实 Pareto 占优前沿面的曲来计算末端解(在目标空间)。在然后，我们在用下面的度量值计算分布的不均匀性。

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} \left| d_i - \bar{d} \right|}{d_f + d_l + (N-1)\bar{d}}. \qquad (1)$$

这里的参数 d_f 和 d_l 是极值解与所获得的边界解的欧氏距离，如图 4 所示。图中阐释了上述方程所提到的所有距离。参数 $\bar{d}$ 是所有距离 d_i，$i = 1, 2, ..., (N-1)$ 的平均值，假设最优非支配前沿面有 N 个解，这里就有 $(N-1)$ 间距。分母是当分子中 N 全部在同一解处情况下的值。值得一提的是这并不是最坏可能解的传播情况。我们可以有一个情形，其中在 d_i 中存在巨大的差异。在这种情形中，度量值可能比 1 大。因此，上述度量标准的最大值可能比 1 大。但是，一个好的分布结果会使得所有的距离 d_i 等于 $\bar{d}$ 并且使得 $d_f = d_l = 0$（当非支配解集合中存在极值解）。因此对于分布最广泛最均匀的非支配解，Δ 的分子是 0，致使度量值也为 0。对于其它的分布，这个度量值的值会比 0 大。对于有 d_f 和 d_l 确定值得两个分布，度量值 Δ 在极值解中会在对坏分布情况下取更高一点的值。鉴于上述多样性度量标准能够用在任何非支配解集合中，包括非 Pareto 占优解集合。使用一种三角化技术或者 Voronoi 图表法[1]计算 d_i，上述步骤能够扩展到估计更高维的解的传播上。

TABLE II
MEAN (FIRST ROWS) AND VARIANCE (SECOND ROWS) OF THE CONVERGENCE METRIC ϒ

Algorithm	SCH	FON	POL	KUR	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
NSGA-II	0.003391	0.001931	0.015553	0.028964	0.033482	0.072391	0.114500	0.513053	0.296564
Real-coded	0	0	0.000001	0.000018	0.004750	0.031689	0.007940	0.118460	0.013135
NSGA-II	0.002833	0.002571	0.017029	0.028951	0.000894	0.000824	0.043411	3.227636	7.806798
Binary-coded	0.000001	0	0.000003	0.000016	0	0	0.000042	7.30763	0.001667
SPEA	0.003403	0.125692	0.037812	0.045617	0.001799	0.001339	0.047517	7.340299	0.221138
	0	0.000038	0.000088	0.00005	0.000001	0	0.000047	6.572516	0.000449
PAES	0.001313	0.151263	0.030864	0.057323	0.082085	0.126276	0.023872	0.854816	0.085469
	0.000003	0.000905	0.000431	0.011989	0.008679	0.036877	0.00001	0.527238	0.006664

TABLE III
MEAN (FIRST ROWS) AND VARIANCE (SECOND ROWS) OF THE DIVERSITY METRIC Δ

Algorithm	SCH	FON	POL	KUR	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
NSGA2R	0.477899	0.378065	0.452150	0.411477	0.390307	0.430776	0.738540	0.702612	0.668025
Real-coded	0.003471	0.000639	0.002868	0.000992	0.001876	0.004721	0.019706	0.064648	0.009923
NSGA-II	0.449265	0.395131	0.503721	0.442195	0.463292	0.435112	0.575606	0.479475	0.644477
Binary-coded	0.002062	0.001314	0.004656	0.001498	0.041622	0.024607	0.005078	0.009841	0.035042
SPEA	1.021110	0.792352	0.972783	0.852990	0.784525	0.755148	0.672938	0.798463	0.849389
	0.004372	0.005546	0.008475	0.002619	0.004440	0.004521	0.003587	0.014616	0.002713
PAES	1.063288	1.162528	1.020007	1.079838	1.229794	1.165942	0.789920	0.870458	1.153052
	0.002868	0.008945	0	0.013772	0.004839	0.007682	0.001653	0.101399	0.003916

C. 结果讨论

表 2 展示了使用 NSGA-II (实数编码), NSGA-II (二进制编码), SPEA,以及 PAES 四种算法所得到的收敛度量值的平均值以及方差。

NSGA-II（实数编码或二进制编码）能够在除了 ZDT3 和 ZDT6 以外所有问题上更好地收敛，其中 PAES 在 ZDT3 和 ZDT6 能更好收敛。NSGA-II 在所有情况中，十次运行的方差也是很小的，除了对 NSGA-II（二进制编码）下的 ZDT4。固定存档策略的 PAES 在九个问题中的两个能更好的收敛。

表 3 展示了使用三个算法所获得的多样性度量标准 Δ 的平均值和方差。

NSGA-II(实数或二进制编码)在九个问题中都表现最好。表现性能最差是 PAES 所得结果。为了进一步说明，我们在图 5 中展示 PAES 十次运行结果的一个以及 NSGA-II（实数编码）仅运行一次的结果。

在大多数问题中，实数编码 NSGA-II 相比其他任何算法（包括二进制编码 NSGA-II）能够找到更好的扩展解。

为了能够展示这些算法的成果，我们展示了 PAES, SPEA 以及 NSGA-II 在 KUP，ZDT2，ZDT4 和 ZDT6 这些问题上经典仿真结果。KUR 问题的 Pareto 占优前沿面有三段不连续的区域。图 6 展示了 NSGA-II（实数编码）经过 250 次迭代后得到的全部非支配解。Pareto 占优区域同样在图中展示出来。图 7 展示了 SPEA 获得的非支配解，在这些问题上它是仅次于最优解（参考表 2，表 3）。

接下来，在图 8，图 9 中我们展示了 ZDT2 问题的非支配解。这个问题有一个非凹面的 Pareto 占优前沿面。我们展示了这个问题上二进制编码 NSGA-II 和 SPEA 的性能。尽管收敛性对每个算法来说都不是难点，但是实数与二进制编码 NSGA-II 相对 SPEA（在这些问题上的次优算法）都能在整个 Pareto 占优区域找到传播性能更好与更多的解。

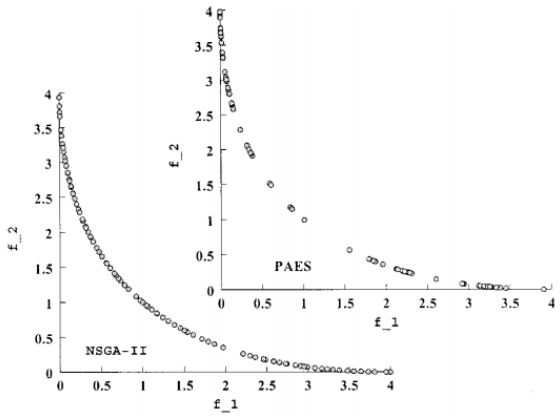

图 5 在 SCH 问题上 NSGA-II 相比 PAES 找到更好地扩展解

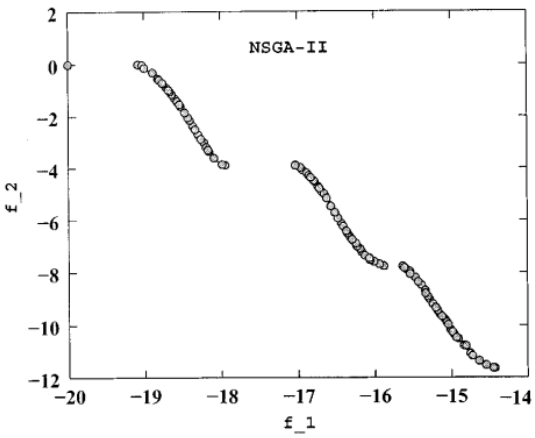

图 6 在 KUR 问题上 NSGA-II（实数编码）找到地非支配解

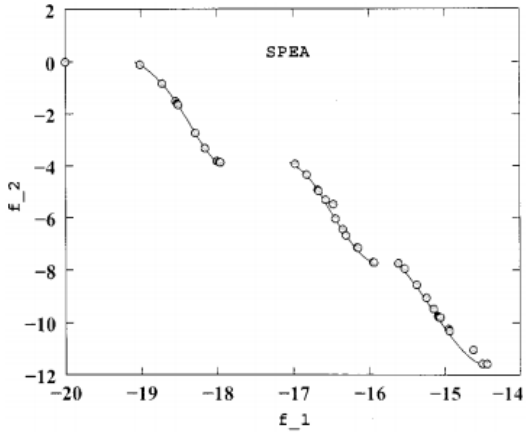

图 7　SPEA 在问题 KUR 上的非支配解

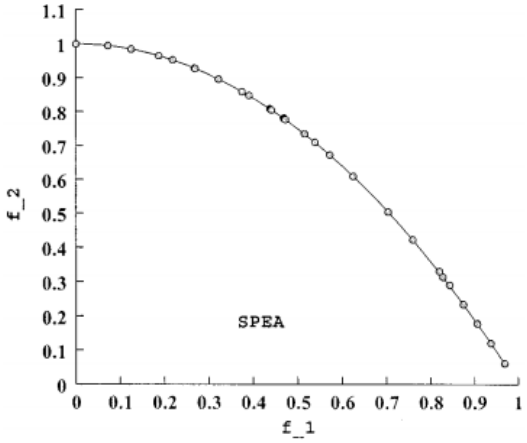

图 9　SPEA 在问题 ZDT2 的非支配解

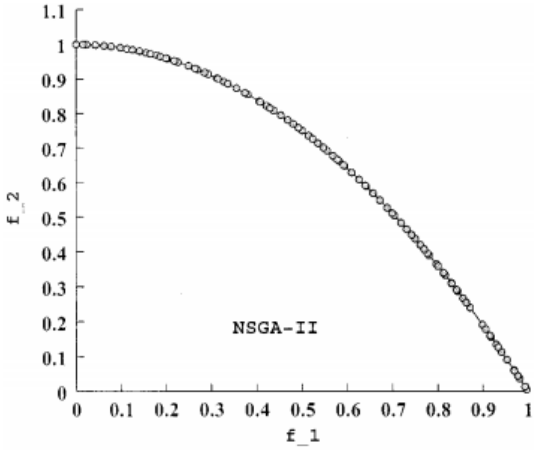

图 8　NSGA-II（二进制编码）在问题 ZDT3 上的非支配解

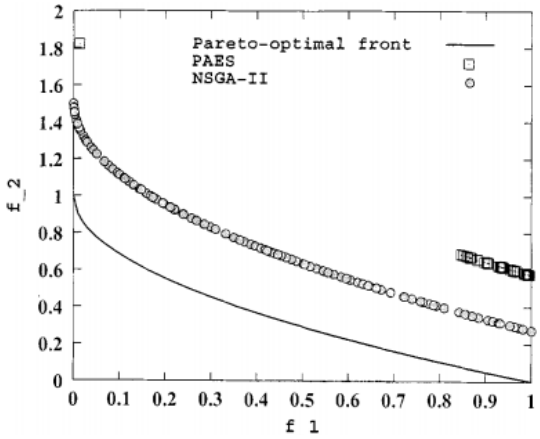

图 10 NSGA-II 相对 SPEA 在问题 ZDT4 上能找到收敛性，扩展性更好的解

问题 ZDT4 在搜索空间有 21^9 或者 7.94(10^{11})不同的局部 Pareto 占优前沿面，但是其中只有一个与全局 Pareto 占优前沿面相关。决策空间中两个连续局部 Pareto 占优集合中的解的欧氏距离设为 0.25。图 10 表明实数编码 NSGA-II 与 PAES 都陷入不同的局部 Pareto 占优集合中，但是 NSGA-II 的收敛性以及找到多样性解的性更更优。二进制编码 GAs 在收敛于全局 Pareto 占优前沿面上存在困难，这个问题在其它单目标的研究中[5]也被发现。

在一个相似十变量 Rastrigin 的函数中[这里的 $g(x)$ 函数]，研究明显表明对于单目标二进制编码 GAs（带有锦标赛选择，单点交叉和按位变异）至少需要种群大小为 500，模拟运行多于 50%才能找到全局最优解。由于我们已经设定种群大小为 100，所以通过多目标优化 GAs 找到全局 Pareto 占优解时不大可能的。但是 NSGA-II 能够找到具有很好传播性能的解即使在局部 Pareto 占优的情况下。由于 SPEA 在这个问题上收敛性很差（见表 2），我们就不再图中展示 SPEA 的结果。

最后，图 11 表明 SPEA 找到了一个在 ZDT6 问题上比其它算法收敛性都要好的非支配解。但是解的分布情况还是实数编码 NSGA-II 找到的最好。

D.　不同参数设置

本次研究中，我们并没有为 NSGA-II 找到更好地参数设置做出很多的尝试。但是在这个部分，我们进行附加实验从而展示了一些不同参数设置对 NSGA-II 性能的影响。

首先,我们保持其它参数不变，但是将迭代次数增加到 500 次（不是原来的 250 次）。表 4 中展示了问题 POL, KUR, ZDT3, ZDT4, 和 ZDT6 的收敛性以及多样性度量值。现在我们得到了十分收敛于真实 Pareto 占优前沿面一个很好的分布结果。表格说明在所有的这些困难问题中，实数编码 NSGA-II 能够收敛到非常接近真实 Pareto 前沿面，除了问题 ZDT6，它可能需要对 NSGA-II 进行较难的参数设置。特别地，ZDT3 和 ZDT4 的结果随迭代次数增加而提高。

问题 ZDT4 有一些局部 Pareto 前沿面并且每个都与一个特定值 $g(x)$ 相关。为了能够跳出局部最优需要对决策向量进行大的改变。除非变异或者交叉操作算子能够生成出在另一个更好区域的解，否则提高到真实 Pareto 前沿面的性能是不可能的。我们使用带有较小的分布索引 $\eta_m = 10$ 的 NSGA-II（实数编码）进行变异操作，，此操作有助于生成比以前分布更好地扩展解。其余参数的设置像以前一样确定。问题 ZDT4 上收敛度量标准 Υ 和多样性度量标准 Δ 经过 250 次迭代后的值如下：

$$\overline{\Upsilon} = 0.029\,544 \quad \sigma_{\Upsilon}^2 = 0.002\,145$$
$$\overline{\Delta} = 0.498\,409 \quad \sigma_{\Delta}^2 = 0.003\,852.$$

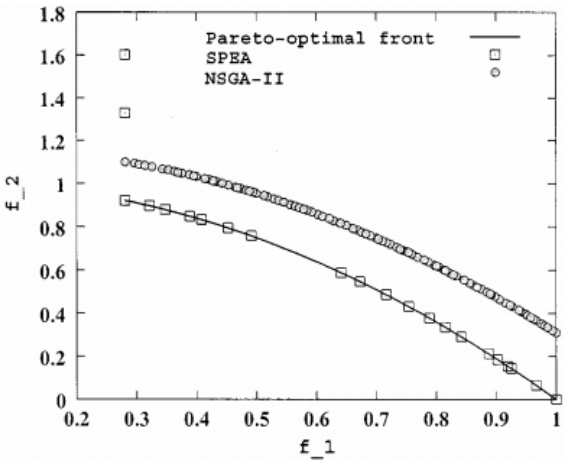

图 11 在问题 ZDT6 实数编码 NSGA-II 相比 SPEA 能找到扩展性更好的解，但是 SPEA 的收敛性更好

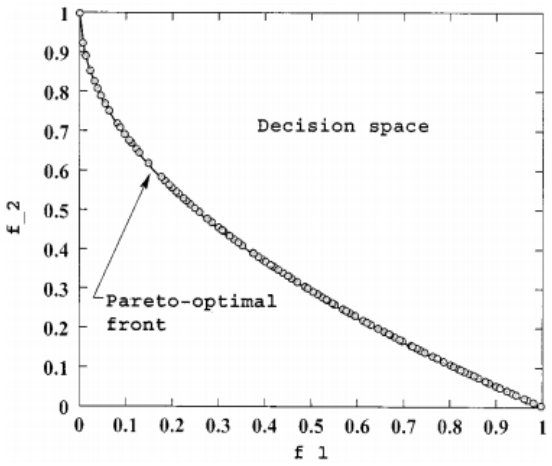

图 12 通过 NSGA-II 在问题 ZDT4 上获得的非支配解

表 4
迭代次数升至 500 次收敛性与多样性度量值的平均值和方差

Convergence metric, Υ					
	POL	KUR	ZDT3	ZDT4	ZDT6
Mean	0.015882	0.026544	0.018510	0.090692	0.276609
Variance	0.000001	0.000017	0.000227	0.053460	0.015843
Diversity metric, Δ					
	POL	KUR	ZDT3	ZDT4	ZDT6
Mean	0.467022	0.418889	0.688218	0.440022	0.655896
Variance	0.002186	0.000530	0.000610	0.026729	0.003302

这些问题相对 PEAS 和 SPEA 要好得多，如图 2 所示。为了说明收敛性和解的扩展性，我们在图 12 中绘制了 250 次迭代后的一组非支配解。图说明 NSGA-II 能够以 $g(x)=1.0$ 找到 Pareto 前沿面上的解。

V. 旋转问题

在早期研究[3]中讨论过，在决策变量中的相互作用会给任何多目标优化算法包括 EAs 带来更高层次的问题。在本节内容，我们生成这样的一个问题并且调查先前三个 MOEAs 在下面强性问题上的运行过程。

$$\begin{aligned} &\text{minimize} \quad f_1(\mathbf{y}) = y_1 \\ &\text{minimize} \quad f_2(\mathbf{y}) = g(\mathbf{y})\exp(-y_1/g(\mathbf{y})) \\ &\text{where} \quad g(\mathbf{y}) = 1 + 10(n-1) \\ &\qquad\qquad + \sum_{i=2}^{n} \left[y_i^2 - 10\cos(4\pi y_i)\right] \\ &\text{and} \quad \mathbf{y} = \mathcal{R}\mathbf{x} \\ &\qquad -0.3 \le x_i \le 0.3, \quad \text{for } i = 1, 2, \ldots, n. \end{aligned} \tag{2}$$

EA 以决策向量 x 运行，但是上述目标函数而是根据向量 y 定义， y 是通过固定旋转矩阵 $\Re$ 转变决策向量 x 而计算出来的。这样，目标函数是决策向量线性结合的函数。为了维持能在 Pareto 前沿面的解的扩展性以及收敛到任何特定的解，需要 EA 能够以一种特殊的方式进行决策变量更新。通过一个遗传搜索算子，例如这里使用的变量位置 SBX 操作，这对 EA 来说成为了一个艰难的任务。然而，在这里，我们只对三个精英机制 MOEAS 的整体行为评价感兴趣。

我们设置种群大小为 100,并且运行每个算法 500 次。对于 SBX，我们设置 $\eta_c = 10$， 为变异设 $\eta_m = 50$。为了限制 Pareto 占优解处在所描述的变量边界，我们通过对两个目标加入一个固定的较大的惩罚来防止解的 $|f_1| > 0.3$。图 13 展示了使用 NSGA-II,PAES,SPEA 经过 500 次迭代后得到的解。能够观察到，NSGA-II 所得到的解相比 PAES，SPEA 所得到的解更加接近与真实前沿面。需要向 Pareto 占优前沿面前进的相互关联参数的更新致使这类问题解决起来变得困难。NSGA-II 的通过实数编码的交叉和变异操作的精英保留操作能够找到一些接近 Pareto 占优前沿面的解[其中 $g(\mathrm{y})=1$ 使得 $f_2 = \exp(-f_1)$]。这个问题的例子说明单目标优化问题中已知困难之一（连接问题[11],[12]）同样会给多目标优化问题带来困难。然而需要更多的系统研究充分着力解决多目标优化中的连接问题。

VI. 约束处理

在过去，第一作者和他的学生们应用了一个针对单目标问题的惩罚无参数约束处理方法。这些研究[2],[6]已经表明如何将基于竞标赛选择的算法应用于种群方法的约束处理，这种方法优于一些其它已经存在的约束处理方法。一个简单的方法可以被引进到上述的 NSGA-II 中，进而解救约束多目标优化问题。

A. 推荐的约束处理方法（约束 NSGA-II）

这些约束处理方法使用二进制竞标赛选择，即选择出种群中的两个解并且挑选出较好的解。在约束存在的情况下，每一个解不是可行解就是非可行解。因此，这里可能最多有三种情况：1）两个解都是可行的；2）一个可行另一个不可行；3）两个解都是不可行的。多于单目标优化，我们对每种情况使用一个简单规则。

情况 1) 选择有更好目标函数值的解。

情况 2) 选择可行解。

情况 3) 选择整体上违反约束更小的解。

由于不存在约束和目标函数相比较的情况，所以没有必要拥有惩罚参数，惩罚参数能使得提出的约束处理方法更加有用且引人注目。

在多目标优化的环境中，后面两种情况可以这样使用，第一种情况就要像以前使用拥挤比较算子那样解决。为了保持 NSGA-II 算法步骤的模块性，我们修改了两个解 i 和 j 的支配定义。

定义 1：解 i 如果是约束意义上的支配解 j，当且仅当下面的条件是正确的。

1) 解 i 是可行的， j 不是.

2) 解 i 和 j 都是可行的，但是解 i 有更小的整体约束违背值。

3) 解 i 和 j 都是可行的并且解 i 支配解 j.

使用这样的约束支配准则的影响是任何可行解的非支配排名都会比非可行解的排名好。所有可行解依据它们的基于

表 5

研究中的约束测试问题

Problem	n	Variable bounds	Objective functions	Constraints
CONSTR	2	$x_1 \in [0.1, 1.0]$ $x_2 \in [0, 5]$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = (1 + x_2)/x_1$	$g_1(\mathbf{x}) = x_2 + 9x_1 \geq 6$ $g_2(\mathbf{x}) = -x_2 + 9x_1 \geq 1$
SRN	2	$x_i \in [-20, 20]$ $i = 1, 2$	$f_1(\mathbf{x}) = (x_1 - 2)^2 + (x_2 - 1)^2 + 2$ $f_2(\mathbf{x}) = 9x_1 - (x_2 - 1)^2$	$g_1(\mathbf{x}) = x_1^2 + x_2^2 \leq 225$ $g_2(\mathbf{x}) = x_1 - 3x_2 \leq -10$
TNK	2	$x_i \in [0, \pi]$ $i = 1, 2$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = x_2$	$g_1(\mathbf{x}) = -x_1^2 - x_2^2 + 1 + 0.1\cos(16\arctan(x_1/x_2)) \leq 0$ $g_2(\mathbf{x}) = (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.5$
WATER	3	$0.01 \leq x_1 \leq 0.45$ $0.01 \leq x_2 \leq 0.10$ $0.01 \leq x_3 \leq 0.10$	$f_1(\mathbf{x}) = 106780.37(x_2 + x_3) + 61704.67$ $f_2(\mathbf{x}) = 3000x_1$ $f_3(\mathbf{x}) = (305700)2289x_2/(0.06 \times 2289)^{0.65}$ $f_4(\mathbf{x}) = (250)2289\exp(-39.75x_2 + 9.9x_3 + 2.74)$ $f_5(\mathbf{x}) = 25(1.39/(x_1x_2) + 4940x_3 - 80)$	$g_1(\mathbf{x}) = 0.00139/(x_1x_2) + 4.94x_3 - 0.08 \leq 1$ $g_2(\mathbf{x}) = 0.000306/(x_1x_2) + 1.082x_3 - 0.0986 < 1$ $g_3(\mathbf{x}) = 12.307/(x_1x_2) + 49408.24x_3 + 4051.02 \leq 50000$ $g_4(\mathbf{x}) = 2.098/(x_1x_2) + 8046.33x_3 - 696.71 \leq 16000$ $g_5(\mathbf{x}) = 2.138/(x_1x_2) + 7883.39x_3 - 705.04 \leq 10000$ $g_6(\mathbf{x}) = 0.417(x_1x_2) + 1721.26x_3 - 136.54 \leq 2000$ $g_7(\mathbf{x}) = 0.164/(x_1x_2) + 631.13x_3 - 54.48 \leq 550$

所有目标函数都是求最小化

目标函数值的非支配等级进行排名。然而，在两个非可行解之中，拥有更小约束违背值的解会有更好地排名。此外，非支配准则的修改不会改变 NSGA-II 的计算复杂度。NSGA-II 的剩余步骤和先前描述的一样使用即可。

因此，一个非可行解违背约束少量的情况下可能与另外一个较大程度上违背不同约束的解分到同一支配等级。这样可能会造成算法在非可行解空间徘徊更多此迭代在通过约束界限到达可行解区域。此外，由于 Fonseca-Fleming 的方法需要以约束违背值进行支配检查，本文中提出的方法计算复杂度更低，且更简单。

B. Ray–Tai–Seow 的约束处理方法

Ray 等人[17]提出了一个更加详细的约束处理技术，其中所有约束的约束违背值并不是简单地相加。相反，同样需要进行约束违背的非支配检查。在这里我们给出了这个方法步骤概括。

三种不同非支配的种群排名第一次被运行。第一排名以 M 目标函数执行并且将排名结果存储在 N 维的向量 R_{obj} 中。第二次排名 R_{con} 仅仅以所有约束（它们的 J ）的约束违背值而执行并且没有使用任何目标函数信息。因此每个约束的约束违背被当做一个标准并且种群的非支配分类使用约束违背值执行。注意到对于一个可行解来说所有的约束违背都为 0,。因此所有的可行解在 R_{con} 中排名为 1。第三种排名是结合目标函数以及约束违背值而执行的[共 $(M + J)$ 个值]。这会产生排名 R_{com} 。即使目标函数值和约束违背值被一起使用，这个算法的好的一个方面是不需要任何惩罚参数。在支配检查中，标准单独进行比较的，因此排出了使用惩罚参数的需要。一旦排名结束，所有的在 R_{com} 中有最好的排名的可行解被选择作为新的种群。如果需要更多的种群，它们就会系统地在剩余解中产生。通过强调中 R_{obj} 排名在选择算子中以及中 R_{con} 排名在交叉算子中的重要性，研究者规定了一套系统的多目标 GA，它同样包括小生境保留算子。更多细节，读者可以参考[17]。即使研究者没有用他们的算法与其它算法进行比较，但是他们展示了这种约束处理方法在一些工程设计问题上的工作。然而，由于需要三种不同标准集合的非支配排序以及算法引进了很多不同操作，如何在更复杂的问题上执行，特别是有该方法有关的计算负担方面，它仍然需要进一步研究。

在接下来的章节，我们选择四个问题并比较了简单 NSGA-II 与 Ray–Tai–Seow 的方法。

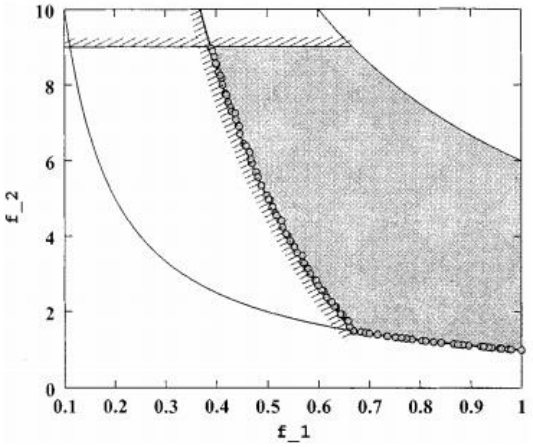

图 14 通过 NSGA-II 在约束问题 CONSTR 上获得的非支配解

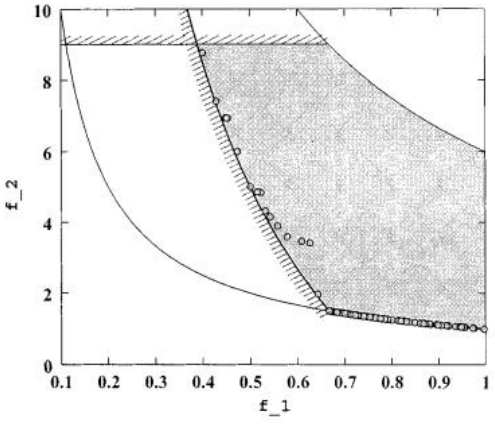

图 15 通过 Ray-Tai-Seow 在约束问题 CONSTR 上获得的非支配解

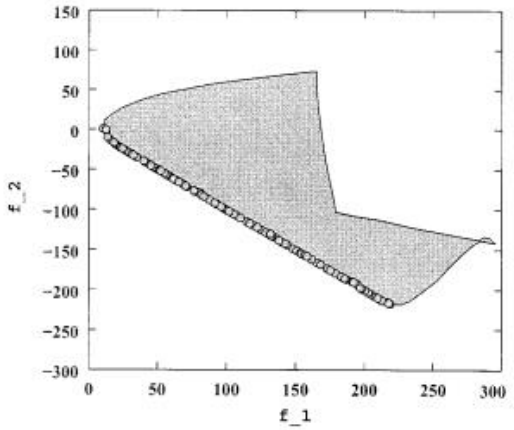

图 16 通过 NSGA-II 在约束问题 SRN 上获得的非支配解

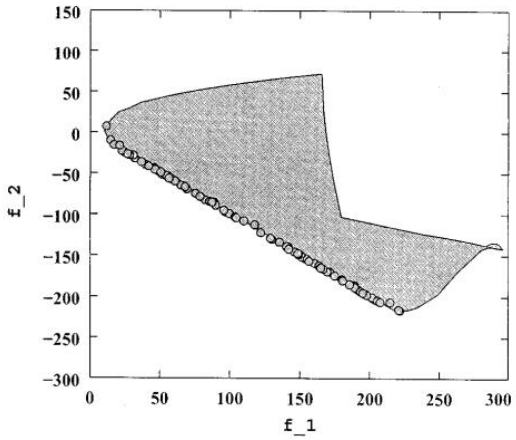

图 17 通过 Ray–Tai–Seow 在约束问题 SRN 上获得的非支配解

C. 仿真结果

我们选择四个约束测试问题（见表 5），它们在先前的研究中被使用过。在第一个问题中，非约束 Pareto 前沿面的一部分是非可行的。因此导致的约束 Pareto 占优区域是第一约束边界以及一些非约束 Pareto 占优区域的结合。第二个问题 SRN 在 NSGA[20]研究中被使用过。这里的约束 Pareto 占优集合是一些非约束 Pareto 占优集合的子集。第三个问题 TNK 首先被 Tanaka 等人[21]提出，它有非连续的 Pareto 占优区域，完全落在第一个约束边界中。在下节，我们会展示以上每个问题的约束占优区域。第四个问题 WATER 是一个五目标和七约束问题，在文献中[17]有尝试解决。因为有五个目标，很难讨论约束在非约束 Pareto 区域的影响。在下一节，我们展示了$\binom{5}{2}$或者十次成对的获得的非支解的图。在这里我们应用实数编码 NSGA-II。

在所有的问题中，我们设置种群大小 100，分别设置实数编码交叉和变异算子的分布索引值为 20 和 100。运行带约束处理技术的 NSGA-II（实数编码）以及 Ray–Tai–Seow 约束处理算法[17]最大 500 次迭代。我们选择相对较大的迭代次数来研究如果解的扩展性在大数目迭代下能够被维持。然而，在这种情况下，我们在可能早的 200 次迭代中就获得了合理扩展性好的解。交叉和变异概率设置如前。

图 14 中展示了在经过 500 次迭代后得到的 100 个非支配解的集合。图中说明 NSGA-II 能够在两个 Pareto 占优区域均匀地维持解。值得一提的是为了能够在约束边界维持解的扩展性，解必须以一种特别地方式依据约束函数进行修改。这对于每个搜索操作来说都成为了一个难点。图 15 展示了使用 Ray–Tai–Seow 算法经过 500 次迭代获得的解。明显可以看出，NSGA-II 执行结果比 Ray–Tai–Seow 方法好就收敛到 Pareto 前沿面以及维持种群非支配解多样性来说。

接下来，我们考虑测试问题 SRN。如图 16 展示了 NSGA-II 经过 500 次迭代后的非支配解。图中展示了 NSGA-II 如何在 Pareto 前沿面上引进随机解。Ray–Tai–Seow 方法同样能够在这个测试问题上逼近前沿面。

图 18 和 19 中展示了可行解空间以及通过 NSGA-II 和 Ray–Tai–Seow 算法的到的非支配解。这里，Pareto 占优区域

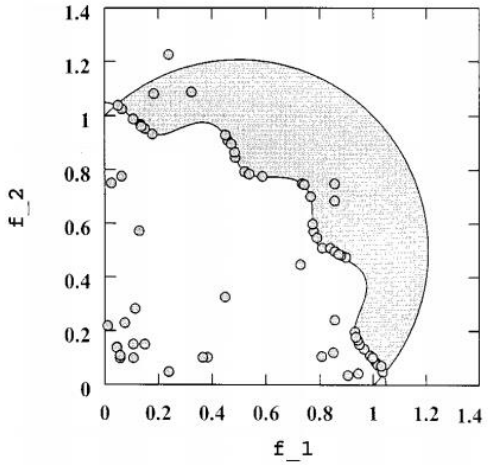

图 19 通过 Ray–Tai–Seow 在约束问题 TNK 上获得的非支配解

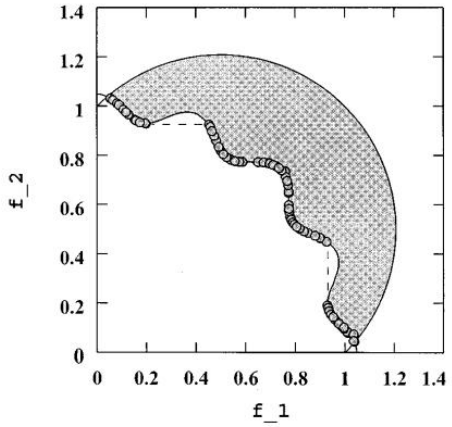

图 18 通过 NSGA-II 在约束问题 TNK 上获得的非支配解

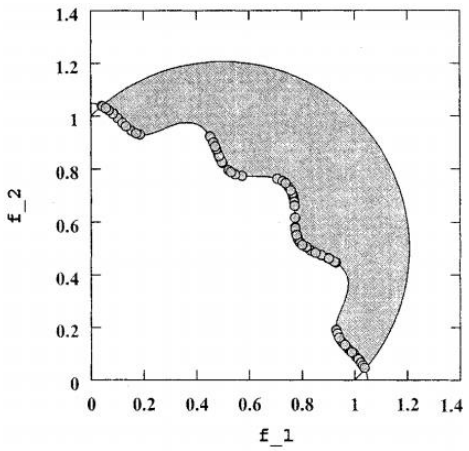

图 20 通过运用 Fonseca–Fleming 约束处理策略的 NSGA-II 在约束问题 TNK 上获得的非支配解

表 6
在已获得的非支配解中观察到的目标函数值的上下界限

Algorithm	f_1	f_2	f_3	f_4	f_5
NSGA-II	0.798 – 0.920	0.027 – 0.900	0.095 – 0.951	0.031 – 1.110	0.001 – 3.124
Ray-Tai-Seow	0.810 – 0.956	0.046 – 0.834	0.967 – 0.934	0.036 – 1.561	0.211 – 3.116

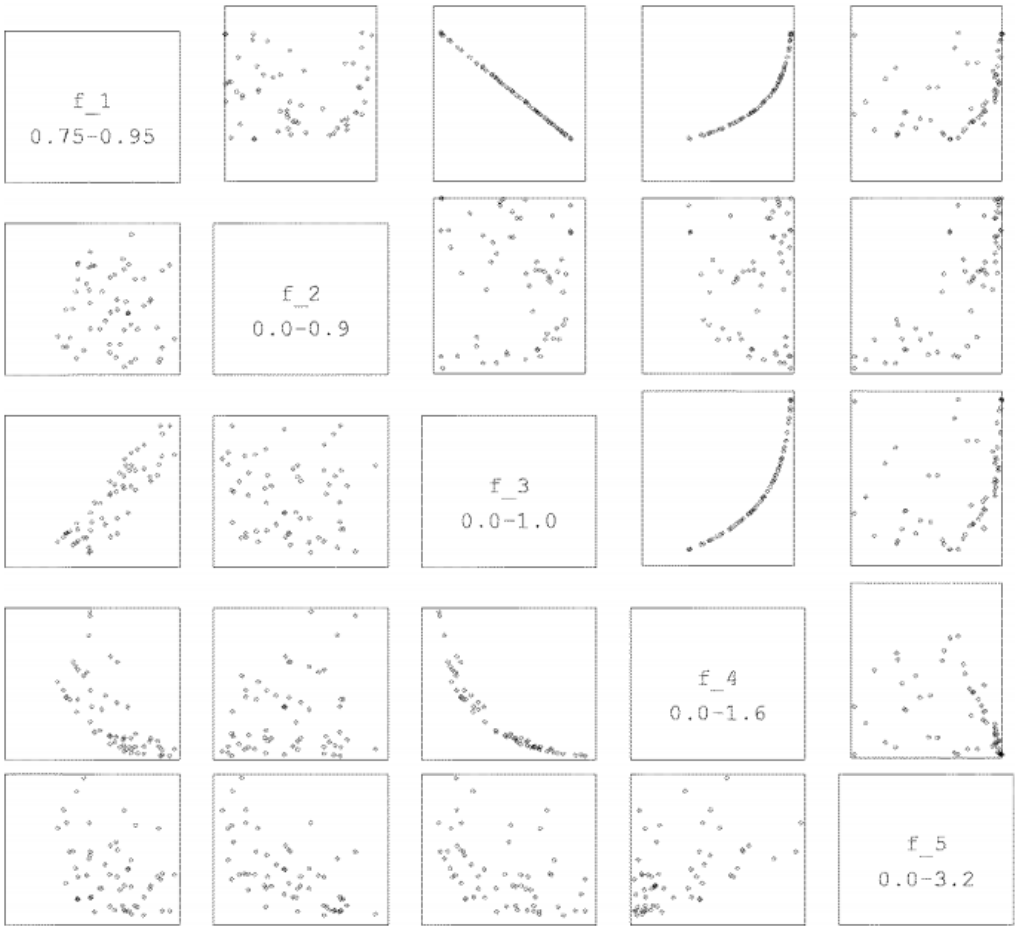

图 21 NSGA-II 的结果在图中上对角部分，Ray–Tai–Seow 的结果在低对角部分展示。对比(i,j)图（$i>j$ 代表 Ray–Tai–Seow 算法所得）与(j,i)图（NSGA-II）。每个坐标的标签和范围如对角盒子所示。

是不连续的并且 NSGA-II 很容易在在 Pareto 前沿面上找到分布广泛的扩展解。即使 Ray–Tai–Seow 算法在 Pareto 前言面找到了一些解，这里存在存在很多非可行解经过 500 次迭代后。为了能够说明 Fonseca–Fleming 约束处理策略的工作性能，我们将它和 NSGA-II 一起用于解决 TNK 上。图 20 展示了经过 500 次迭代后的 100 个种群成员，参数设置与图 18 一样。两个图都说明 NSGA-II 结合本文所提出的算法与 Fonseca–Fleming 约束处理策略都可以处理的很好。

Rayet 等人 [17]在他们的研究中使用过问题 WATER。他们通过一下方式对目标函数进行标准化：

$$f_1/8(10^4),\quad f_2/1500,\quad f_3/3(10^6),\quad f_4/6(10^6),\quad f_5/8000.$$

由于 WATER 问题中存在 5 个目标函数，我们观察已获得非非支配解的标准化目标函数值得取值方范围。表 5 展示了与 Ray–Tai–Seow 算法的比较。在大多数目标函数中，NSGA-II 相比 Ray–Tai–Seow 可以找到扩展性更好的解。为了展示这五个标准化目标函数中的成对相互作用，我们在图 21 中绘制了两个算法$\binom{5}{2}$或十次的相互作用。NSGA-II 的结果在图中上对角部分，Ray–Tai–Seow 的结果在低对角部分展示。图的坐标可以通过观察相关对角框以及它们的范围就可以得到。例如，处在第一行第三列图的纵坐标为 f_1，横坐标为 f_3。由于这个图属于对角区域的上侧，所以它是使用 NSGA-II 获得的。为了比较这个图与使用 Ray–Tai–Seow 方法得到的类似图，我们在第三行第一列找到这个图。针对这个图，纵轴坐标设为 f_3，横轴坐标设为 f_1。为了在两个图中得到更好地比较，我们按 Ray–Tai–Seow 原图的样子观察它，但是把 NSGA-II 的结果顺时针旋转 90 度观察。这会使得两种情况下的标签和坐标范围相同。

我们观察到 NSGA-II 的图相比 Ray–Tai–Seow 的图有更成形的模式。例如 NSGA-II 中图 f_1- f_3, f_1- f_4 以及 f_3- f_4 的相互作用是非常明显的。即使 Ray–Tai–Seow 算法所得解存在类似的模式，但是不能很充分收敛到真实前沿面。

VII. 结论

我们已经提出了基于非支配排序的一种快速和基于精英 MOEA 的算法。从文献中引进的九个不同难度的测试问题，与其它两个精英机制 MOEAs 算法相比，NSGA-II 能够维持扩展性更好的解并能更好的收敛于已知的非支配前沿面。然而在一个问题中，PAES 能够更好的收敛于真实 Pareto 前沿

面。PAES 通过在搜索空间中一个确定的预先指定数量的相等大小的格子控制解的拥挤程度来维持解的多样性。在那个问题中，这样一个与基于变异方法影响结合的确定性拥挤程度的算法，与 NSGA-II 和 SPEA 中一个动态且无参数的拥挤距离方法相比，更能收敛于真实 Pareto 前沿面是有好处的说法是受质疑的。然而 NSGA-II 使用的多样性保留机制是目前发现三种研究方法中最好的。

在一个有很强参数互相作用的问题中，NSGA-II 与其它两种方法相比能够更加逼近真实前沿面，但是重点是三种方法在解决所谓的高度强性问题上都面临着问题。即使这一直是单目标优化问题中的一个重要问题，本文展示了高强度问题同样会给 MOEAs 带来困难。更重要的是，该领域的研究者应该为多目标优化思考出这样的强行问题来测试一个新提出的算法。

我们同样为约束多目标优化提出了一个扩展带约束支配的简单定义。即使这个新的定义可以被用在其他任何 MOEAs 上，实数编码的 NSGA-II 通过这个定义证明相对其它近期提出的约束处理方法来说，能够更好解决四个不同问题。

拥有快速非支配排序程序，精英策略，无参数的方法以及一个简单高效的约束处理性质的方法，NSGA-II，在将来应该引发关注，并增加它的应用。

REFERENCES

[1] K. Deb, Multiobjective Optimization Using Evolutionary Algorithms. Chichester, U.K.: Wiley, 2001.

[2] ——,“An efficient constraint-handling method for genetic algorithms,”Comput. Methods Appl. Mech. Eng., vol. 186, no. 2–4, pp. 311–338,2000.

[3] ——,“Multiobjective genetic algorithms: Problem difficulties and construction of test functions,” inEvol. Comput., 1999, vol. 7, pp. 205–230.

[4] K. Deb and D. E. Goldberg, “An investigation of niche and species formation in genetic function optimization,” in Proceedings of the Third International Conference on Genetic Algorithms, J. D. Schaffer, Ed. San Mateo, CA: Morgan Kauffman, 1989, pp. 42–50.

[5] K. Deb and S. Agrawal, “Understanding interactions among genetic algorithm parameters,” in Foundations of Genetic Algorithms V,W. Banzhaf and C. Reeves, Eds. San Mateo, CA: Morgan Kauffman, 1998, pp. 265–286.

[6] K. Deb and R. B. Agrawal,“Simulated binary crossover for continuous search space,”in Complex Syst., Apr. 1995, vol. 9, pp. 115–148.

[7] C. M. Fonseca and P. J. Fleming, “Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization,” in Proceedings of the Fifth International Conference on Genetic Algorithms,S. Forrest, Ed. San Mateo, CA: Morgan Kauffman, 1993, pp. 416–423.

[8] ——, “On the performance assessment and comparison of stochastic multiobjective optimizers,” in Parallel Problem Solving from Nature IV, H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, Eds. Berlin, Germany: Springer-Verlag, 1996, pp. 584–593.

[9] ——, “Multiobjective optimization and multiple constraint handling with evolutionary algorithms—Part I: A unified formulation,”IEEE Trans. Syst., Man, Cybern. A, vol. 28, pp. 26–37, Jan. 1998.

[10] ——, “Multiobjective optimization and multiple constraint handling with evolutionary algorithms—Part II: Application example,”IEEE Trans. Syst., Man, Cybern. A, vol. 28, pp. 38–47, Jan. 1998.

[11] D. E. Goldberg, B. Korb, and K. Deb, “Messy genetic algorithms: Motivation, analysis, and first results,” in Complex Syst., Sept. 1989, vol. 3, pp. 93–530.

[12] G. Harik, “Learning gene linkage to efficiently solve problems of bounded difficulty using genetic algorithms,” llinois Genetic Algorithms Lab., Univ. Illinois at Urbana-Champaign, Urbana, IL, IlliGAL Rep. 97005, 1997.

[13] J. Horn, N. Nafploitis, and D. E. Goldberg, “A niched Pareto genetic algorithm for multiobjective optimization,” in Proceedings of the First IEEE Conference on Evolutionary Computation, Z. Michalewicz, Ed. Piscataway, NJ: IEEE Press, 1994, pp. 82–87.

[14] J. Knowles and D. Corne, “The Pareto archived evolution strategy: A new baseline algorithm for multiobjective optimization,” in Proceedings of the 1999 Congress on Evolutionary Computation. Piscataway, NJ: IEEE Press, 1999, pp. 98–105.

[15] F. Kursawe, “A variant of evolution strategies for vector optimization,” in Parallel Problem Solving from Nature, H.-P. Schwefel and R. Männer, Eds. Berlin, Germany: Springer-Verlag, 1990, pp. 193–197.

[16] C. Poloni, “Hybrid GA for multiobjective aerodynamic shape optimization,” in Genetic Algorithms in Engineering and Computer Science,G. Winter, J. Periaux, M. Galan, and P. Cuesta, Eds. New York: Wiley, 1997, pp. 397–414.

[17] T. Ray, K. Tai, and C. Seow, “An evolutionary algorithm for multiobjective optimization,”Eng. Optim., vol. 33, no. 3, pp. 399–424, 2001.

[18] G. Rudolph, “Evolutionary search under partially ordered sets,” Dept. Comput. Sci./LS11, Univ. Dortmund, Dortmund, Germany, Tech. Rep. CI-67/99, 1999.

[19] J. D. Schaffer, “Multiple objective optimization with vector evaluated genetic algorithms,” in Proceedings of the First International Conference on Genetic Algorithms, J. J. Grefensttete, Ed. Hillsdale, NJ: Lawrence Erlbaum, 1987, pp. 93–100.

[20] N. Srinivas and K. Deb, “Multiobjective function optimization using nondominated sorting genetic algorithms,”Evol. Comput., vol. 2, no. 3, pp. 221–248, Fall 1995.

[21] M. Tanaka, “GA-based decision support system for multicriteria optimization,” inProc. IEEE Int. Conf. Systems, Man and Cybernetics-2, 1995, pp. 1556–1561.

[22] D. Van Veldhuizen, “Multiobjective evolutionary algorithms: Classifications, analyzes, and new innovations,” Air Force Inst. Technol., Dayton, OH, Tech. Rep. AFIT/DS/ENG/99-01, 1999.

[23] D. Van Veldhuizen and G. Lamont, “Multiobjective evolutionary algorithm research: A history and analysis,” Air Force Inst. Technol., Dayton, OH, Tech. Rep. TR-98-03, 1998.

[24] E. Zitzler, “Evolutionary algorithms for multiobjective optimization: Methods and applications,” Doctoral dissertation ETH 13398, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, 1999.

[25] E. Zitzler, K. Deb, and L. Thiele, “Comparison of multiobjective evolutionary algorithms: Empirical results,”Evol. Comput., vol. 8, no. 2, pp. 173–195, Summer 2000.

[26] E. Zitzler and L. Thiele, “Multiobjective optimization using evolutionary algorithms—A comparative case study,” in Parallel Problem Solving From Nature, V, A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, Eds. Berlin, Germany: Springer-Verlag, 1998, pp.292–301.