一、 背景介绍

由于 NSGA 算法有以下的不足：**1）非支配排序的计算复杂度很高**：计算的时间复杂度为 $O(MN^3)$,其中 M 是目标的数量，N 是种群的大小。**2）缺少精英策略**：当找到最优解，精英策略可以减少最优解在进化过程中的损失，因此精英策略可以提高遗传算法的性能。**3）需要指定共享半径**：确保种群多样性依赖于共享半径，所以最主要的问题就是需要人为设定共享半径，虽然现在有一些动态调整共享半径的方法，但是最理想的是不需要通过指定共享半径来确保种群多样性。所以为了解决 NSGA 算法存在的不足，我们提出了 NSGA 算法的改进版本：NSGA-Ⅱ。

本文的主要贡献是**提出了 NSGA-Ⅱ算法，解决了 NSGA 算法的不足之处，而且将 NSGA-Ⅱ算法应用在带约束的多目标优化问题上也表现出很好的性能。**

二、 算法流程

在介绍 NSGA-Ⅱ算法之前，先要了解 NSGA 算法，所以我先去看了提出 NSGA 算法的论文：**Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms**。下面先介绍 NSGA 算法：

2.1 NSGA 算法

NSGA 算法和 GA 算法只是在选择操作开始之前根据个体之间的支配关系对种群进行了非支配分层，它们的选择、交叉、变异算子是相同的，即 NSGA 算法在选择操作开始之前首先对种群进行非支配分层，分配虚拟适应度值，然后使用共享小生境算法重新计算虚拟适应度值。

（1） 非支配排序

对于最小化多个目标函数 $F(x)=(F_1(x),F_2(x),...,F_M(x))$，种群中的个体 x_i,x_j，如果对于所有的 $q=1,2,3,4,...,M$，都有 $F_q(x_i)\le F_q(x_j)$,则称 x_i 支配 x_j，即 x_i 优于 x_j。非支配排序算法流程如下：

算法 1　第一级非支配排序

```
for i in population size N
    for j in population size N and j ≠ i
        for q in object size M
            if Fq(xi) > Fq(xj) then xi is not nondominated individual //xi 不是非支配个体
            else then xi is nondominated individual    //xi 是非支配个体
```

通过算法 1 得到了种群第一级非支配层，然后忽略这些被标记为非支配的个体，即这些非支配个体不再进行下一轮的比较，接着重复算法 1 就能得到第二级非支配层，依此类推，直到整个种群被分层。

可以看出算法 1 的时间复杂度为 $O(MN^2)$，重复算法 1 直到整个种群被分层，那么最坏的情况即每一级只有一个非支配个体，总共有 N 级非支配层，总的时间复杂度就是 $O(MN^3)$。

（2） 共享小生境

假设第 p 级非支配层上有 n_p 个个体，每个个体的虚拟适应度为 f_p，令 i，j=1,2,3,…,n_p，实现步骤如下：

① 按照公式 1 计算属于同一级非支配层的两个个体 x_i 和 x_j 的距离 d_{ij}

$$d_{ij}=\sqrt{\sum_{m=1}^{M}\left(\frac{F_m(x_i)-F_m(x_j)}{F_m^{\max}-F_m^{\min}}\right)} \tag{1}$$

其 M 是目标数量，$F_m^{\max}$、$F_m^{\min}$ 分别是 F_m 目标函数的最大值和最小值。

② 按照公式 2 计算两个个体 x_i 和 x_j 的共享函数 $sh(d_{ij})$

$$sh(d_{ij})=\begin{cases}1-\left(\dfrac{d_{ij}}{\sigma_{share}}\right)^{\alpha}, & d_{ij}\le\sigma_{share}\\ 0, & d_{ij}>\sigma_{share}\end{cases} \tag{2}$$

其中 σ_{share} 是共享半径，α 是常数

③ 按照公式 3 计算个体 x_i 的小生境数量 c_i

$$c_i=\sum_{j=1}^{n_p}sh(d_{ij}),i=1,2,3,...,n_p \tag{3}$$

④ 最后按照公式 4 计算个体 x_i 的虚拟适应度 $f_p'(x_i)$

$$f_p'(x_i)=\frac{f_p(x_i)}{c_i} \tag{4}$$

共享小生境算法流程如下：

算法 2 共享小生境

```
for p in nondominated layer size
    for i in np
        for j in np
            calculate dij and sh(dij)    //按照公式 1 和 2 计算距离和共享函数
    calculate ci and fp(xi)    //按照公式 3 和 4 计算个体 i 的小生境数量和虚拟适应度
```

NSGA 的算法流程图如图 2-1 所示。

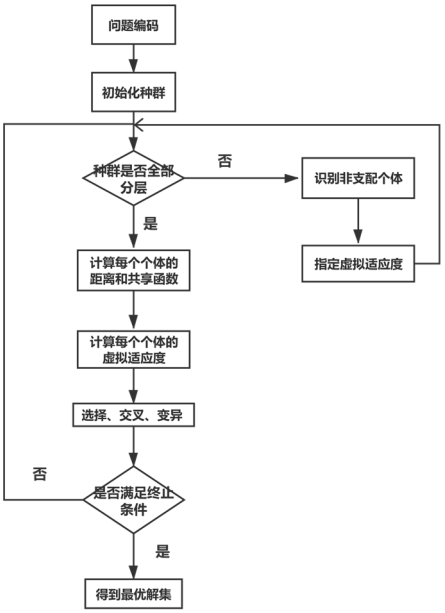

图 2-1 NSGA 流程图

2.2 NSGA-Ⅱ算法

NSGA 算法有三个主要的缺点：计算复杂度太高、缺少精英策略和需要人为指定共享半径，NSGA-Ⅱ算法使用三种策略分别解决了上述的三个缺点：通过使用快速非支配排序将计算复杂度降为 $O(MN^2)$，使用精英策略提高算法的性能，使用拥挤度比较算子替代共享小生境方法来保证种群的多样性，且不需要人为指定参数。

（1）快速非支配排序

对于每个个体 p 我们定义了两个属性：n_p 和 S_p ，n_p 表示支配个体 p 的数量，S_p 表示 p 支配的个体集合，第一级非支配层的计算我们仍然采用非支配排序，时间复杂度为 $O(MN^2)$，第二级非支配层我们的计算方法为：对第一级非支配层的每个个体 p，我们遍历 p 支配的个体集合 S_p 的每个个体 q，让 n_q 减一，之后判断 n_q 是否等于 0，如果等于，则把个体 q 加入到第二级非支配层。依此类推，直到将种群全部分层。快速非支配排序算法流程如下:

算法 3　快速非支配排序

```
for p in population size N    //初始化第一级非支配层
    Sp = null
    np = 0
    for q in population size N
        for m in object size M
            if (Fm (xp) > Fm(xq)) then p is dominated by q
            else then p is nondominated by q
        if (p is nondominated by q) then add q into Sp
        else (p is dominated by q) then np = np + 1
    if np = 0 then prank = 1                    //设置个体 p 的非支配层的级数
    add p into nondominated layer L1   //将个体 p 加入第一级非支配层
i = 1
while the polulation is not all stratified     //当种群还没有被全部分层
  while Li is not null
        Li+1 = null
        for p in Li
        for q in Sp
        nq = nq – 1
        if nq = 0 then
            qrank = i + 1     //设置个体 q 的非支配层的级数
            add q into Li+1   //将个体 q 加入下一级非支配层
  i = i + 1
```

可以看出计算第一级非支配层的时间复杂度为 $O(MN^2)$，计算第二级直到所有种群都分层的时间复杂度为 $O(N^2)$，所以快速非支配排序算法的时间复杂度为 $\max(O(N^2),O(MN^2))=O(MN^2)$。

(2) 拥挤度比较算子

拥挤度 i_d 表示个体周围的密度，可以理解为个体 i 周围只包含个体 i 不包含其他任何个体的最大的长方形。首先对于同一级非支配层 I 的每个个体 i，按照目标函数的值对每个个体排序，令边界的两个个体 I_1 和 I_L 拥挤度(L 表示 I 的大小)为无穷大，然后遍历其余的个体，按照公式(5)计算其余每个个体的拥挤度 i_d。

$$i_d = \sum_{m=1}^{M} \frac{F_m(i+1) - F_m(i-1)}{F_m^{\max} - F_m^{\min}} \tag{5}$$

计算个体拥挤度的算法流程如下：

算法 4　计算个体拥挤度

```
for m in object size M
    I = sort(I,m)
    I(1) = I(L) =  ∞
    for i = 2 to L-1
        calculate id  //按照公式 5 计算
```

经过前面非支配排序和拥挤度计算，现在每个个体 i 都有两个属性：排序 i_{rank} 和拥挤度大小 i_d，依据上面两个属性，我们可以定义拥挤度比较算子$<_n$：当 $i_{rank}<j_{rank}$ 或者 $i_{rank}=j_{rank}$ 且 $i_d>j_d$ 时，$i <_n j$，即个体 i 优于个体 j。

(3) 精英策略

采用精英策略可以防止父代中优秀个体的流失，通过将父代 P_t 和子代 Q_t 的个体混合得到种群 R_t(大小为 2N)，然后对种群 R_t 进行非支配排序，先选出第一级非支配层的个体 F_1 加入新的父代 P_{t+1},如果 P_{t+1} 的大小没有达到 N，则继续选择第二级非支配层的个体 F_2 加入 P_{t+1},依此类推，直到 F_1 到 F_n 的个体加起来大于 N，这时我们对 F_n 层的个体进行拥挤度排序，按照拥挤度排序选出其中的优秀个体加入 P_{t+1},直到 P_{t+1} 的大小达到 N。NSGA-Ⅱ算法流程如下：

算法 5　NSGA-Ⅱ

R_t = combine P_t with Q_t
F = fast nondominated sort(R_t)　　**//对种群 Rt 进行快速非支配排序**
P_{t+1} = null and i = 1
while $|P_{t+1}| + |F_i| \le N$
　　crowding distance calculate(F_i)　　**//计算 Fi 的个体拥挤度**
　　add F_i into P_{t+1}
　　i = i + 1
sort(F_i,$<_n$)　　**//对 Fi 的个体按照拥挤度降序排列**
add F_i 中前 N-$|P_{t+1}|$个个体　into P_{t+1}
Q_{t+1} = make new population(P_{t+1})　**//使用交叉、变异算子产生后代种群**
t = t + 1

NSGA-Ⅱ算法流程图如图 2-2 所示。

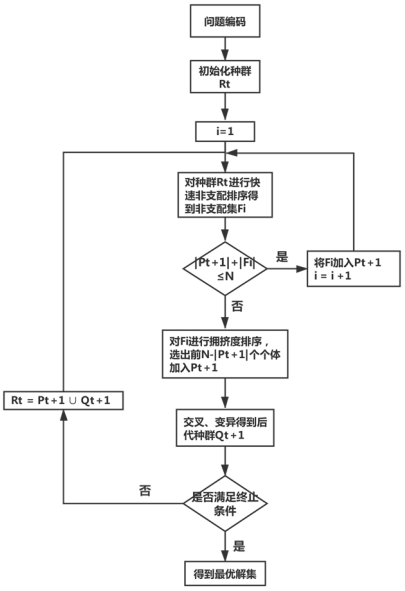

图 2-2 NSGA-Ⅱ算法流程图

考虑 NSGA-Ⅱ整个算法最坏情况的时间复杂度，由于快速非支配排序的时间复杂度为 $O(M(2N)^2)$，拥挤度计算的时间复杂度为 $O(M(2N)\log(2N))$，拥挤度排序的时间复杂度为 $O(2N\log(2N))$，所以 NSGA-Ⅱ的时间复杂度由快速非支配排序的时间复杂度 $O(MN^2)$所决定，而且快速非支配层的排序是逐级进行的，如果排在前面的非支配层的个体数量已经达到 N 了，那就不用对剩余的个体进行非支配层排序了。

NSGA-Ⅱ在几个复杂测试函数上能取得不错的效果主要有以下三个原因：1）**快速非支配排序**的时间复杂度较低。2）**精英策略**可以保护父代优秀个体，从而提高算法性能。3）用**拥挤度比较算子**代替共享小生境方法来保护解的多样性，从而避免人为指定参数。