A new application: 
Going Through A Numerical Example

Note you have two types of variables: the x’s are decision variable values and the f’s are function values. It is easy to start confusing them when doing multi-objective analysis.
In Step 2. The above population (given in Table 18) of 6 is classified into the 3 fronts shown.
Algorithm parameters

- Step 1
  Set $\gamma_{\text{share}} = 0.15$, $\varepsilon = 0.22$

  $N = \text{population size} = 6$

  Hence, from Step 1 (for front $P_1$)

  $F_{\text{min}} = N + \varepsilon = 6 + 0.22 = 6.22$

  (this is a min-min problem, so we add $\varepsilon$)

  Step 2: Sort the population into fronts as shown in Figure 115.

  => to do min-min or max-min, we can swap the sign of one objective function to convert into a max-max or min-min,
A new application: Going Through A Numerical Example

Figure 115  The population is classified into three non-dominated fronts.

In Step 2. The above population (given in Table 18) of 6 is classified into the 3 fronts shown.
**Calculated distances and Crowding**

- Table 18 gives the values of the decision variables \( (x_1, x_2) \) and of the fitness associated with each decision variable combination for each of the two fitnesses. (The far left column is the number from Fig. 115 of each of the points associated with \( (x_1, x_2) \).

**Using equation 5.21 and Table 18 for distances of points on the Front 1:**

\[
\begin{align*}
    d_{13} &= 0.120 \\
    d_{15} &= 0.403 \\
    d_{35} &= 0.518
\end{align*}
\]

Note: \( d_{ij} = 1 \) for all \( j \)
Method to reduce fitness within a $P_i$ for Crowding
repeat slide

• Normalized Euclidian Distance between two points (indexed by $i$ and $j$) on the $m^{th}$ front:

$$d_{ij} = \sqrt{\sum_{k=1}^{P} \left( \frac{X_{k}^{(i)} - X_{k}^{(j)}}{X_{k}^{\max} - X_{k}^{\min}} \right)^2}$$  \hspace{1cm} (5.21)$$

$$s_{h}(d_{jk}) = \begin{cases} 1 - \left( \frac{d_{jk}}{s_{share}} \right)^2 & \text{if } d_{jk} \leq s_{share} \\ 0 & \text{otherwise} \end{cases}$$  \hspace{1cm} (4.59)$$
Niche Count

• The Niche Count for front $P_1$ is:

$$nc_1 = Sh(d_{11}) + Sh(d_{13}) + Sh(d_{15})$$

$$= 1 + 1.423 + 0$$

$$= 1.423$$

$$nc_3 = Sh(d_{21}) + Sh(d_{33}) + Sh(d_{56})$$

$$= 0.423 + 1 + 0$$

$$= 1.423$$

$$nc_5 = Sh(d_{51}) + Sh(d_{53}) + Sh(d_{55})$$

$$= 0 + 0 + 1$$

$$= 1.00$$

Why 0 and 1 here?
Overview of Fitness Calculation for NSGA algorithm

NSGA Fitness Assignment

Step 1 Choose sharing parameter $\sigma_{\text{share}}$ and a small positive number $\epsilon$ and initialize $F_{\text{min}} = N + \epsilon$. Set front counter $j = 1$. 

Step 2 Classify population $P$ according to non-domination: 
$(P_1, P_2, \ldots, P_\rho) = \text{Sort}(P, \preceq)$. 

Step 3 For each $q \in P_j$

Step 3a Assign fitness $F_j^{(q)} = F_{\text{min}} - \epsilon$. 

Step 3b Calculate niche count $n_{c_q}$ using equation (4.60) among solutions of $P_j$ only. 

Step 3c Calculate shared fitness $F_j^\prime(q) = F_j^{(q)}/n_{c_q}$. 

Step 4 $F_{\text{min}} = \min(F_j^\prime(q) : q \in P_j)$ and set $j = j + 1$. 

Step 5 If $j \leq \rho$, go to Step 3. Otherwise, the process is complete. 

This fitness assignment procedure can be embedded in a single-objective GA. The proportionate selection operator must be used. However, any crossover and mutation operator can be employed. In the following, we illustrate the above fitness assignment procedure on the Min-Ex problem.
A new application: Going Through A Numerical Example

Table 18  Fitness assignment under an NSGA-II

<table>
<thead>
<tr>
<th>Solution</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$f_1$</th>
<th>$f_2$</th>
<th>Front Id</th>
<th>Assigned fitness</th>
<th>Shared fitness</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.31</td>
<td>0.89</td>
<td>0.31</td>
<td>6.10</td>
<td>1</td>
<td>6.00</td>
<td>4.22</td>
</tr>
<tr>
<td>2</td>
<td>0.43</td>
<td>1.92</td>
<td>0.43</td>
<td>6.79</td>
<td>2</td>
<td>4.00</td>
<td>4.00</td>
</tr>
<tr>
<td>3</td>
<td>0.22</td>
<td>0.56</td>
<td>0.22</td>
<td>7.09</td>
<td>1</td>
<td>6.00</td>
<td>4.22</td>
</tr>
<tr>
<td>4</td>
<td>0.59</td>
<td>3.63</td>
<td>0.59</td>
<td>7.85</td>
<td>3</td>
<td>3.78</td>
<td>3.78</td>
</tr>
<tr>
<td>5</td>
<td>0.66</td>
<td>1.41</td>
<td>0.66</td>
<td>3.65</td>
<td>1</td>
<td>6.00</td>
<td>6.00</td>
</tr>
<tr>
<td>6</td>
<td>0.83</td>
<td>2.51</td>
<td>0.83</td>
<td>4.23</td>
<td>2</td>
<td>4.00</td>
<td>4.00</td>
</tr>
</tbody>
</table>

Assigned fitness is the fitness given to the whole front (level)
Shared fitness is the value for individual q after dividing assigned fitness by $nc_q$

**IMPORTANT**

These values are not related (directly) to $f_1$, $f_2$. It is a measure of how good the Pareto front is.
Step 3: Calculating Fitnesses

• Now the fitness for the point $q$ on front 1 is computed from:

$$F_i' (q) = \frac{F_i (q)}{nc}$$

Here $q \in P_i$, $F_i (q) = 0 = N + (3-3)$

• The computed values are on Fig 116
NSGA Fitness Values for all 6 points  
(values given in Table 18)

Figure 116  Shared fitness values of six solutions.
Fitness for fronts 2 and 3

• So the fitness for individuals \( q \) on front 2 is

\[
F'_{2}(q) = \left(\frac{F_{2}(q)}{n \cdot c_{q}}\right)^{2}
\]

where

\[
F_{2}(q) = \min F_{1}(q) - \varepsilon
\]

\[
= 4.22 - .22
\]

\[
= 4.00
\]

• Since there is no crowding on front 2, the value of fitness is 4.00 for all those points.
Computing a Solution

• The purpose of all the calculations reported in Table 18 is to convert your multi objective problem into a GA with a single objective.
• In this new GA problem the single objective for each decision vector \( q \) is the shared fitness value (computed in Step 3c).

Now the GA proceeds as normal with crossover and mutation.

The following 4 slides do not have elitism and the following 2 slides do not have mutation.
NSGA after 500 generations (without Mutation)

Figure 118  The NSGA distributes solutions near the Pareto-optimal region at the 500-th generation.
NSGA Solution after 50 generations (without mutation)

Figure 117 The NSGA distributes solutions near the Pareto-optimal region at the 50-th generation.
**Mutation**

- Fig 117 and 118 do not have mutation.
- Now consider addition of a mutation probability of 0.02 (=2%) applied to each bit in the string.
- The following figures Fig 119 and 120 show the results of the mutation.
- Has mutation improved the performance of the NSGA?
- What is the cause do you think?
- The next two slides show results if you do the GA with crossover and no mutation.
NSGA with Mutation 50 generations

Figure 119  Population at the 50-th generation with an NSGA having a mutation operator.
NGSA with Mutation 500 generations

Figure 120  Population at the 500-th generation with an NSGA having a mutation operator.
• Deb’s conclusion with regard to the previous two slides (119 and 120) are that for NSGAII, the GA should not include mutation unless it also includes elitism.

• This observation is apparently based on his experience and not just the two slides shown.