Evolutionary Algorithms: Multi-Objective Optimization

Jiří Kubalík
Department of Cybernetics, CTU Prague

http://cw.felk.cvut.cz/doku.php/courses/a4m33bia/start
Multi-Objective Optimization

Many real-world problems involve multiple objectives

- **Conflicting objectives**
  
  - A solution that is extreme with respect to one objective requires a compromise in other objectives.
  
  - A sacrifice in one objective is related to the gain in other objective(s).

Motivation example: Buying a car

- two extreme hypothetical cars 1 and 2,
- cars with a trade-off between cost and comfort – A, B, and C.

©Kalyanmoy Deb: Multi-Objective Optimization using Evolutionary Algorithms.
Multi-Objective Optimization

Many real-world problems involve multiple objectives

- **Conflicting objectives**
  - A solution that is extreme with respect to one objective requires a compromise in other objectives.
  - A sacrifice in one objective is related to the gain in other objective(s).

Motivation example: Buying a car

- two extreme hypothetical cars 1 and 2,
- cars with a trade-off between cost and comfort – A, B, and C.

- **Which solution** out of all of the trade-off solutions is the best with respect to all objectives?

Without any further information those trade-offs are indistinguishable.

⇒ a number of optimal solutions is sought in multiobjective optimization!
Multi-Objective Optimization: Definition

General form of multi-objective optimization problem

Minimize/maximize $f_m(x)$, $m = 1, 2, ..., M$
subject to $g_j(x) \geq 0$, $j = 1, 2, ..., J$
$h_k(x) = 0$, $k = 1, 2, ..., K$
$x_i^{(L)} \leq x_i \leq x_i^{(U)}$, $i = 1, 2, ..., n$.

- $x$ is a vector of $n$ decision variables: $x = (x_1, x_2, ..., x_n)^T$;

- **Decision space** is constituted by variable bounds that restrict each variable $x_i$ to take a value within a lower $x_i^{(L)}$ and an upper $x_i^{(U)}$ bound;

- Inequality and equality constraints

- A solution $x$ that satisfies all constraints and variable bounds is a **feasible solution**, otherwise it is called a **infeasible solution**;

- **Feasible space** is a set of all feasible solutions;

- Objective functions $f(x) = (f_1(x), f_2(x), ..., f_M(x))^T$ constitute a multi-dimensional **objective space**.
For each solution $x$ in the decision space, there exists a point in the objective space

$$f(x) = z = (z_1, z_2, ..., z_M)^T$$
Motivation Example: Cantilever Design Problem

**Task** is to design a beam, defined by two decision variables
- diameter $d$,
- length $l$.

that can carry an end load $P$ and is optimal with respect to the following **objectives**
- $f_1$ – minimization of weight,
- $f_2$ – minimization of deflection.

Obviously, conflicting objectives!

subject to the following **constraints**
- the developed maximum stress $\sigma_{max}$ is less than the allowable strength $S_y$,
- the end deflection $\delta$ is smaller than a specified limit $\delta_{max}$. 

©Kalyanmoy Deb: Multi-Objective Optimization using Evolutionary Algorithms.
Cantilever Design Problem: Decision and Objective Space

©Kalyanmoy Deb: Multi-Objective Optimization using Evolutionary Algorithms.
Non-Conflicting Objectives

There exist multiple Pareto-optimal solutions in a problem only if the objectives are conflicting to each other.

- If this does not hold then the cardinality of the Pareto-optimal set is one.
  - This means that the optimum solution corresponding to any objective is the same.

Example: Cantilever beam design problem

- $f_1$ – minimizing the end deflection $\delta$,
- $f_2$ – minimizing the maximum developed stress in the beam $\sigma_{max}$.
Dominance and Pareto-Optimal Solutions

©Kalyanmoy Deb: Multi-Objective Optimization using Evolutionary Algorithms.
**Dominance and Pareto-Optimal Solutions**

**Domination:** A solution \( x^{(1)} \) is said to dominate the other solution \( x^{(2)} \), \( x^{(1)} \preceq x^{(2)} \), if \( x^{(1)} \) is no worse than \( x^{(2)} \) in all objectives and \( x^{(1)} \) is strictly better than \( x^{(2)} \) in at least one objective.

Solutions A, B, C, D are non-dominated solutions (Pareto-optimal solutions)

Solution E is dominated by C and B (E is non-optimal).
Properties of Dominance-Based Multi-Objective Optimization

Non-dominated set – Among a set of solutions $P$, the non-dominated set of solutions $P'$ are those that are not dominated by any member of the set $P$.

The non-dominated set of the entire feasible search space is the globally Pareto-optimal set.

©Kalyanmoy Deb: Multi-Objective Optimization using Evolutionary Algorithms.
Properties of Dominance-Based Multi-Objective Optimization

If for every member $x$ in a set $P$ there exists no solution $y$ in the neighborhood of $x$ such that $\|y - x\| \leq \epsilon$ (where $\epsilon$ is a small positive number) dominating any member of the set $P$, then solutions belonging to the set $P$ constitute a **locally Pareto-optimal set**.
Goals of Dominance-Based Multi-Objective Optimization

Every finite set of solutions \( P \) can be divided into two non-overlapping sets
- non-dominated set \( P_1 \) – contains all solutions that do not dominate each other, and
- dominated set \( P_2 \) – at least one solution in \( P_1 \) dominates any solution in \( P_2 \).

In the absence of other factors (e.g. preference for certain objectives, or for a particular region of the tradeoff surface) there are two goals of the multi-objective optimization
- **Quality** – To find a set of solutions as close as possible to the Pareto-optimal front.
- **Spread** – To find a set of solutions as diverse as possible.
Differences with Single-Objective Optimization

Two (orthogonal) goals instead of one

- progressing towards the Pareto-optimal front,
- maintaining a diverse set of solutions in the non-dominated set.

Dealing with two search spaces

- objective vs. decision space,
- in which space the diversity must be achieved?

No artificial fix-ups

- weighted sum approach – multiple objectives are weighted and summed together to create a composite objective function.
  Its performance depends on the chosen weights.
- \( \varepsilon \)-constraint method – chooses one of the objective functions and treats of the objectives as constraints by limiting each of them within certain predefined limits.
  Also depends on the chosen constraint limits.
Difficulties with Classical Optimization Algorithms

- The convergence to an optimal solution depends on the chosen initial solution.
- Most algorithms tend to get stuck to a suboptimal solution.
- An algorithm efficient in solving one optimization problem may not be efficient in solving a different optimization problem.
- Algorithms are not efficient in handling problems having a discrete search space.
- Algorithms cannot be efficiently used on a parallel machine.
Multi-Objective Evolutionary Algorithms

- **Pareto Archived Evolution Strategy (PAES)**

- **Multiple Objective Genetic Algorithm (MOGA)**

- **Niched-Pareto Genetic Algorithm (NPGA)**

- **SPEA2**
- **NSGA**

- **NSGA-II**

- . . .
Non-Dominated Sorting Genetic Algorithm (NSGA)

Common features with the standard GA
- variation operators – crossover and mutation,
- selection method – Stochastic Reminder Roulette-Wheel,
- standard generational evolutionary model.

What distinguishes NSGA from the SGA
- fitness assignment scheme which prefers non-dominated solutions, and
- fitness sharing strategy which preserves diversity among solutions of each non-dominated front.

Algorithm NSGA
1. Initialize population of solutions
2. Repeat
   - Calculate objective values and assign fitness values
   - Generate new population
   - Until stopping condition is fulfilled
Fitness Sharing

**Diversity preservation method** originally proposed for solving multi-modal optimization problems so that GA is able to sample each optimum with the same number of solutions.

**Idea** – diversity in the population is preserved by degrading the fitness of similar solutions

**Algorithm** for calculating the shared fitness function value of i-th individual in population of size \( N \)

1. calculate *sharing function* value with all solutions in the population according to

   \[
   Sh(d_{ij}) = \begin{cases} 
   1 - \left( \frac{d_{ij}}{\sigma_{share}} \right)^\alpha, & \text{if } d_{ij} \leq \sigma_{share} \\
   0, & \text{otherwise}.
   \end{cases}
   \]

2. calculate niche count \( nc_i \) as follows

   \[
   nc_i = \sum_{j=1}^{N} Sh(d_{ij})
   \]

3. calculate *shared fitness* as

   \[
   f_i' = \frac{f_i}{nc_i}
   \]

**Remark:** If \( d = 0 \) then \( Sh(d) = 1 \) meaning that two solutions are identical. If \( d \geq \sigma_{share} \) then \( Sh(d) = 0 \) meaning that two solutions do not have any sharing effect on each other.
Fitness Sharing: Example

Bimodal function - six solutions and corresponding shared fitness functions

- $\sigma_{\text{share}} = 0.5$, $\alpha = 1$.

<table>
<thead>
<tr>
<th>Sol. $i$</th>
<th>String</th>
<th>Decoded value</th>
<th>$x^{(i)}$</th>
<th>$f_i$</th>
<th>$nc_i$</th>
<th>$f'_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>110100</td>
<td>52</td>
<td>1.651</td>
<td>0.890</td>
<td>2.856</td>
<td>0.312</td>
</tr>
<tr>
<td>2</td>
<td>101100</td>
<td>44</td>
<td>1.397</td>
<td>0.948</td>
<td>3.160</td>
<td>0.300</td>
</tr>
<tr>
<td>3</td>
<td>011101</td>
<td>29</td>
<td>0.921</td>
<td>0.246</td>
<td>1.048</td>
<td>0.235</td>
</tr>
<tr>
<td>4</td>
<td>001011</td>
<td>11</td>
<td>0.349</td>
<td>0.890</td>
<td>1.000</td>
<td>0.890</td>
</tr>
<tr>
<td>5</td>
<td>110000</td>
<td>48</td>
<td>1.524</td>
<td>0.997</td>
<td>3.364</td>
<td>0.296</td>
</tr>
<tr>
<td>6</td>
<td>101110</td>
<td>46</td>
<td>1.460</td>
<td>0.992</td>
<td>3.364</td>
<td>0.295</td>
</tr>
</tbody>
</table>

Let’s take the first solution

- $d_{11} = 0.0$, $d_{12} = 0.254$, $d_{13} = 0.731$, $d_{14} = 1.302$, $d_{15} = 0.127$, $d_{16} = 0.191$
- $Sh(d_{11}) = 1$, $Sh(d_{12}) = 0.492$, $Sh(d_{13}) = 0$, $Sh(d_{14}) = 0$,
  $Sh(d_{15}) = 0.746$, $Sh(d_{16}) = 0.618$.
- $nc_1 = 1 + 0.492 + 0 + 0 + 0.746 + 0.618 = 2.856$
- $f'(1) = f(1)/nc_1 = 0.890/2.856 = 0.312$
**NSGA: Fitness Assignment**

**Input:** Set $P$ of solutions with assigned objective values.

**Output:** Set of solutions with assigned fitness values (the bigger the better).

1. Choose sharing parameter $\sigma_{share}$, small positive number $\epsilon$,
   initialize $F_{max} = \text{PopSize}$ and front counter $front = 1$
2. Find set $P' \subset P$ of non-dominated solutions
3. For each $q \in P'$
   - assign fitness $f(q) = f_{max}$,
   - calculate sharing function with all solutions in $P'$ niche count $nc_q$ among solutions of $P'$ only,
     the normalized Euclidean distance $d_{ij}$ is calculated
   - calculate shared fitness $f'(q) = f(q)/nc_q$.
4. $f_{max} = \min(f'(q) : q \in P') - \epsilon$
   $P = P \setminus P'$
   $front = front + 1$
5. If not all solutions are assessed go to step 2, otherwise stop.
NSGA: Fitness Assignment cont.

Example:
- First, 10 solutions are classified into different non-dominated fronts.
- Then, the fitness values are calculated according to the fitness sharing method.
  - The sharing function method is used front-wise.
  - Within a front, less dense solutions have better fitness values.

©Kalyanmoy Deb: Multi-Objective Optimization using Evolutionary Algorithms.
NSGA: Conclusions

Computational complexity
■ Governed by the non-dominated sorting procedure and the sharing function implementation.
  – non-dominated sorting – complexity of $O(MN^3)$.
  – sharing function – requires every solution in a front to be compared with every other solution in the same front, total of $\sum_{j=1}^{\rho} |P_j|^2$, where $\rho$ is a number of fronts. Each distance computation requires evaluation of $n$ differences between parameter values. In the worst case, when $\rho = 1$, the overall complexity is of $O(nN^2)$.

Advantages
■ Assignment of fitness according to non-dominated sets – makes the algorithm converge towards the Pareto-optimal region.

■ Sharing allows phenotypically diverse solutions to emerge.

Disadvantages
■ sensitive to the sharing method parameter $\sigma_{share}$.
  – some guidelines for setting the parameter based on the expected number of optima $q$.
    
    $$
    \sigma_{share} = \frac{0.5}{\sqrt{q}}
    $$
  – or dynamic update procedure of $\sigma_{share}$.
NSGA-II

Fast non-dominated sorting approach
- Computational complexity of $O(MN^2)$.

Diversity preservation
- the sharing function method is replaced with a crowded comparison approach,
- parameterless approach.

Elitist evolutionary model
NSGA-II: Diversity preservation

Density estimation

![Graph showing density estimation](image)

©Kalyanmoy Deb: Multi-Objective Optimization using Evolutionary Algorithms.

Crowded comparison operator

Every solution in the population has two attributes
1. non-domination rank ($i^{\text{rank}}$), and
2. crowding distance ($i^{\text{distance}}$).

A partial order $\prec_n$ is defined as:

$$i \prec_n j \text{ if } (i^{\text{rank}} < j^{\text{rank}}) \text{ or } ((i^{\text{rank}} = j^{\text{rank}}) \text{ and } (i^{\text{distance}} > j^{\text{distance}}))$$
1. Current population $P_t$ is sorted based on the non-domination
   Each solution is assigned a fitness equal to its non-domination level (1 is the best).

2. The usual binary tournament selection, recombination, and mutation are used to create a child population $Q_t$ of size $N$.

3. Combined population $R_t = P_t \cup Q_t$ is formed.
   Elitism is ensured.

4. Population $P_{t+1}$ is formed according to the following schema
Simulation Results: NSGA vs. NSGA-II

Comparison of NSGA and NSGA-II on bi-objective 0/1 Knapsack Problem with 750 items. NSGA-II outperforms NSGA with respect to both performance measures.
NSGA-II: Simulation Results on Different Types of Problems

Problem with continuous Pareto-optimal front

Problem with discontinuous Pareto-optimal front

©Kalyanmoy Deb et al.: A Fast and Elitist Multi-Objective Genetic Algorithm: NSGA-II.
NSGA-II: Constraint Handling Approach

Binary tournament selection with modified domination concept is used to choose the better solution out of the two solutions $i$ and $j$, randomly picked up from the population.

In the presence of constraints each solution in the population can be either feasible or infeasible, so that there are the following three possible situations:

- both solutions are feasible,
- one is feasible and other is not,
- both are infeasible.

Constrained-domination: A solution $i$ is said to constrained-dominate a solution $j$, if any of the following conditions is true

1. Solution $i$ is feasible and solution $j$ is not.
2. Solutions $i$ and $j$ are both infeasible, but solution $i$ has a smaller overall constraint violation.
3. Solutions $i$ and $j$ are feasible, and solution $i$ dominates solution $j$. 
NSGA-II: Simulation Results cont.

Comparison of NSGA-II and Ray-Kang-Chye’s Constraint handling approach

Comparison of NSGA-II and Ray-Kang-Chye's Constraint handling approach
Strength Pareto Evolutionary Algorithm 2 (SPEA2)

SPEA2 uses two sets of solutions

- **regular population** of newly generated solutions, and
- **archive**, which contains a representation of the nondominated front among all solutions considered so far.

**The archive size is fixed**, i.e., whenever the number of nondominated individuals is less than the predefined archive size, the archive is filled up by dominated individuals.

A **truncation method** is invoked when the nondominated front exceeds the archive limit.

A member of the archive is only removed if 1) a solution has been found that dominates it or 2) the maximum archive size is exceeded and the portion of the front where the archive member is located is overcrowded.

Using the archive makes it possible not to lose certain portions of the current nondominated front due to random effects.

All individuals in the archive participate in selection.

SPEA2

- uses the concept of **Pareto dominance** in order to assign scalar fitness values to individuals;
- uses a **fine-grained fitness** assignment strategy which **incorporates density information** in order to distinguish between solutions that are indifferent, i.e., do not dominate each other.
SPEA2: Algorithm

Input: $N$ is the population size, $\overline{N}$ is the archive size.

1. **Initialization**: Generate an initial population $P_0$ and create the empty archive $\overline{P}_0 = \emptyset$. Set $t = 0$.

2. **Fitness assignment**: Calculate fitness of individuals in $P_t$ and $\overline{P}_t$.

3. **Environmental selection**: Copy all individuals in $P_t$ and $\overline{P}_t$ to $\overline{P}_{t+1}$.
   - If size of $\overline{P}_{t+1}$ exceeds $\overline{N}$ then reduce $\overline{P}_{t+1}$ using the truncation operator.
   - If size of $\overline{P}_{t+1}$ is less than $\overline{N}$ then fill $\overline{P}_{t+1}$ with dominated solutions in $P_t$ and $\overline{P}_t$.

4. **Termination**: If $t > T$ then return nondominated solutions in $\overline{P}_{t+1}$. Stop.

5. **Mating selection**: Perform binary tournament selection with replacement on $\overline{P}_{t+1}$ in order to fill the mating pool.

6. **Variation**: Apply recombination and mutation operators to the mating pool and fill $P_{t+1}$ with the generated solutions.
   - $t = t + 1$
   - Go to Step 2.
**SPEA2: Fitness Assignment**

**Fitness assignment** (fitness is to minimized) – for each individual both dominating and dominated solutions are taken into account.

- Each individual $i$ in the archive $\overline{P}_t$ and the population $P_t$ is assigned a strength value $S(i)$, representing the number of solutions it dominates.

- The raw fitness $R(i)$ of an individual $i$ is calculated as

$$R(i) = \sum_{j \in P_t + \overline{P}_t, j \succ i} S(j)$$

that is $R(i)$ is determined by the strengths of its dominators in both archive and population.

$R(i) = 0$ corresponds to a nondominated solution.

Since the **raw fitness assignment** is based on the concept of Pareto dominance, it may fail when most individuals do not dominate each other.
**SPEA2: Density Estimation**

**Density information** is incorporated to discriminate between individuals having identical raw fitness values.

The density at any point is estimated as a (decreasing) function of the distance to the $k$-th nearest data point – calculated as the inverse of the distance to the $k$-th nearest neighbor.

- $k$ equal to the square root of the sample size is used: $k = \sqrt{N + N}$.
- Density $D(i)$ is calculated as
  \[ D(i) = \frac{1}{\sigma_i^k + 2} \]
  where $\sigma_i^k$ is the distance to the $k$-th nearest neighbor and it is made sure that $D(i) < 1$.

**Final fitness** is given as

\[ F(i) = R(i) + D(i) \]
SPEA2: Environmental Selection

If after copying all nondominated individuals from archive and population to the archive of the next generation

- the archive is too small (i.e. \(|P_{t+1} < N|\)), the best \(N - |P_{t+1}|\) dominated solutions (w.r.t. fitness) in the previous archive and population are copied to the new archive;
- the archive is too large (i.e. \(|P_{t+1} > N|\)), individuals from \(P_{t+1}\) are iteratively removed until \(|P_{t+1}| = N|\).

At each iteration, the individual which has the minimum distance to another individual is chosen (a tie is broken by considering the second smallest distances and so forth).
MOEA Performance Measures

The result of a MOEA run is not a single scalar value, but a collection of vectors forming a non-dominated set.

1. Comparing two MOEA algorithms requires comparing the non-dominated sets they produce. However, there is no straightforward way to compare different non-dominated sets.

**Three goals that can be identified and measured:**

1. The distance of the resulting non-dominated set to the Pareto-optimal front should be minimized.

2. A good (in most cases uniform) distribution of the solutions found is desirable.

3. The extent of the obtained non-dominated front should be maximized, i.e., for each objective, a wide range of values should be present.
Properties of metrics for comparing non-dominated sets

1. **Pareto compatibility** – a comparison metric $R$ is compatible with an outperformance relation if for each pair of non-dominated sets $A$ nad $B$, such that $A \leq B$, $R$ will evaluate $A$ as being better than $B$.

   Outperformance relation $\leq$ – a non-dominated set $A$ completely outperforms set $B$ if each point in $B$ is dominated by a point in $A$.

2. **Direct comparative metric** – compares $A$ and $B$ directly using a scalar measure $R(A, B)$ to describe how much better $A$ is than $B$.

3. **Reference metric** – use a reference set; it scores both sets against this reference set and compares the results.

4. **Independent metric** – measures some property of each set that is not dependent on any other, or any reference set.

5. **Transitive metric** – induces a complete ordering of all possible non-dominated sets. It ensures that if $A$ beats $B$, and $B$ beats $C$ then it is always true that $A$ beats $C$.

6. **Cardinal metrics** – counts the number of vectors in sets.
**S Metric**

**Size of the space covered** $S(X)$ – it calculates the *hypervolume* of the multi-dimensional region enclosed by a set $A$ and a *reference point* (usually so-called *Utopian* point). The hypervolume expresses the size of the region $A$ dominates.

So, the bigger the value of this measure the better the quality of $A$ is, and vice versa.

Pros:
- Compatible with the outperformance relations.
- Independent.
- Differentiates between different degrees of complete outperformance of two sets.
- Scaling independent.
- Intuitive meaning/interpretation.

Cons:
- Requires defining some upper boundary of the region. This choice does affect the ordering of non-dominated sets.
- It has a large computational overhead, $O(n^{k+1})$, where $n$ is the number of nondominated solutions and $k$ is the number of objectives, rendering it unusable for many objectives or large sets.
- It multiplies apples by oranges, that is, different objectives together.
**Coverage of two sets** $C(X, Y)$ – given two sets of non-dominated solutions $X$ and $Y$ found by the compared algorithms, the measure $C(X, Y)$ returns a ratio of a number of solutions of $Y$ that are dominated by or equal to any solution of $X$ to the whole set $Y$.

- It returns values from the interval $[0, 1]$.
- The value $C(X, Y) = 1$ means that all solutions in $Y$ are covered by solutions of the set $X$. And vice versa, the value $C(X, Y) = 0$ means that none of the solutions in $Y$ are covered by the set $X$.
- Always both orderings have to be considered, since $C(X, Y)$ is not necessarily equal to $1 - C(Y, X)$.

Properties:
- It has low computational overhead.
- If two sets are of different cardinality and/or the distributions of the sets are non-uniform, then it gives unreliable results.

\[
C(A, B) = 0.75, \quad C(B, A) = 0.25
\]
**C Metric cnd.**

Properties:
- Any pair of C metric scores for a pair of sets A and B in which neither $C(A, B) = 1$ nor $C(B, A) = 1$, indicates that the two sets are incomparable according to the weak outperformance relation.
- It is cycleinducing – if three sets are compared using $C$, they may not be ordered.

Example:
- $C(A, B) = 0$, $C(B, A) = 3/4$
- $C(B, C) = 0$, $C(C, B) = 1/2$
- $C(A, C) = 1/2$, $C(C, A) = 0$

$C$ considers $B$ better than $A$, $A$ better than $C$, but $C$ better than $B$. 

Reading

- Kalyanmoy Deb: Multi-objective optimization using evolutionary algorithms
  http://books.google.com/books?id=OSTn4GSy2uQC&printsec=frontcover&dq=deb&hl=cs&cd=1

  http://sci2s.ugr.es/docencia/doctobio/2002-6-2-DEB-NSGA-II.pdf

  http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.112.5073&rep=rep1&type=pdf


  http://www.lania.mx/ccoello/knowles02a.ps.gz