Multi-Objective Optimization

A quick introduction

Giuseppe Narzisi

Courant Institute of Mathematical Sciences
New York University

24 January 2008

Outline

Motivation

- Most problems in nature have several (possibly conflicting) objectives to be satisfied.
- Many of these problems are frequently treated as single-objective optimization problems by transforming all but one objective into constraints.

Scope of Optimization

in practice

- Optimal design & manufacturing
- Inverse Problems: output known, find input.
- Parameter optimization for optimal performance
- System modeling
- Planning
- Optimal control
- Forecasting and prediction
- Data mining (classification, clustering, pattern recognition)
- Machine learning
- Bioinformatics

What is a MOOP?

In words

- Problem of finding a vector of decision variables which satisfies constraints and optimizes a vector function whose elements represent the objective functions.
- These functions form a mathematical description of performance criteria which are usually in conflict with each other.
- Hence, the term *optimize* means finding such a solution which would give the values of all the objective functions acceptable to the decision maker.

What is a MOOP?

Formal definition

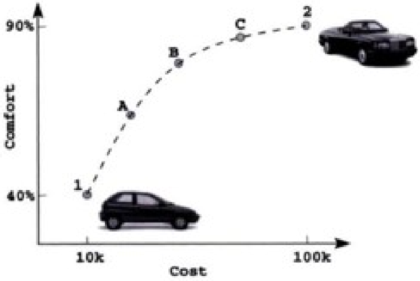
$$
\begin{array}{ll}
\mathit{min} & \mathcal{F}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_M(\mathbf{x})] \\
\mathit{subject\ to} & \mathcal{G}(\mathbf{x}) = [g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_J(\mathbf{x})] \geq 0 \\
 & \mathcal{H}(\mathbf{x}) = [h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_K(\mathbf{x})] = 0 \\
 & x_i^{(L)} \leq x_i \leq x_i^{(U)}, i = 1, \ldots, N
\end{array}
\tag{1}
$$

- $\mathbf{x} = (x_1, x_2, \ldots, x_N)^T$ is the vector of the N decision variables,
- M is the number of objectives f_i,
- J inequality and K equality constraints,
- $x_i^{(L)}$ and $x_i^{(U)}$ are respectively the lower and upper bound for each decision variables x_i.

Edgeworth-Pareto solution

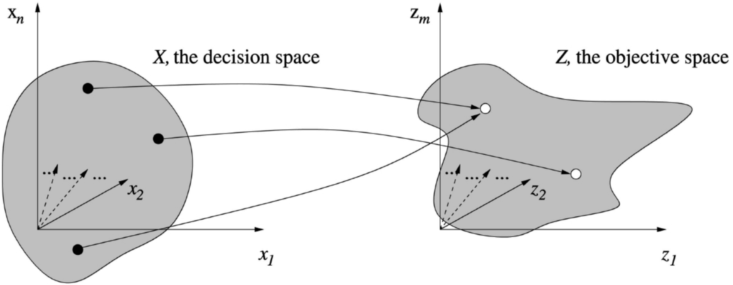
Notion of optimum for MOOPs

- Find good compromises (or *trade-offs*) instead of a single solution (global optimization).
- Originally proposed by Francis Ysidro Edgeworth in 1881; it was later generalized by Vilfredo Pareto (in 1896).



Decision space vs. Objective space

- In multi-objective optimization the objective function constitute a *multidimensional* space
- For each solution $\mathbf{x}$ in the *decision variable space* $\mathcal{X}$ there is a point in the *objective space* $\mathcal{Z}$ denoted by $f(\mathbf{x}) = \mathbf{z} = (z_1, z_2, \ldots, z_M)^T$.



Convex and Non-convex MOOP

Definition

A multi-objective optimization problem is convex if all objective functions are convex and the feasible region is convex.

Definition

A function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ *is a convex function if for any two pair of solutions* $\mathbf{x_1}, \mathbf{x_2} \in \mathbb{R}^n$*, the following condition is true:*

$$f(\lambda \mathbf{x_1} + (1-\lambda)\mathbf{x_2}) \leq \lambda f(\mathbf{x_1}) + (1-\lambda) f(\mathbf{x_2}), \tag{2}$$

for all $0 \leq \lambda \leq 1$

- Since a MOOP has two spaces, the convexity must be analyzed on both spaces.
- Moreover, although the search space can be non-convex, the Pareto optimal front can be convex.

Two approaches to Multi-Objective Optimization

- Although the solution of MOOP consists of a set of solutions, from a practical point, the user needs only one solution
- Question: *Which of this optimal solution must one choose?*
- Two possible approaches:
  1. PREFERENCE-BASED PROCEDURE:
     - Composite objective function as the weighted sum of the objectives
     - Only if a relative preference factor of the objectives is known in advance.
  2. IDEAL PROCEDURE:
     1. Find multiple trade-off solutions with a wide range of values for the objectives.
     2. Choose one of the obtained solution using higher-level information

Ideal Objective Vector

Definition

The m-th component of the ideal objective vector $\mathbf{z}^*$ *is the constrained minimum of the following problem:*

$$\begin{array}{ll} \textit{min} & f_m(\mathbf{x}) \\ \textit{subject to} & \mathbf{x} \in \mathcal{S} \end{array} \tag{3}$$

Thus, the ideal vector is defined as following:

$$\mathbf{z}^* = (f^*) = (f_1^*, f_2^*, \ldots, f_M^*)^T \tag{4}$$

where f_i^* is the function value associated with the minimum solution (x_i^*) for the *m*-th objective function.

Ideal Objective Vector

Considerations

- In general, the ideal objective vector corresponds to a non-existent solution.
- The only way an ideal objective vector corresponds to a feasible solution is when the minimum of all objective functions are identical.
- In this case the objectives are not conflicting.
- The ideal objective vector denotes the array with the lower bound of all objective functions.

Utopian Objective Vector

Some algorithms may require a solution which is strictly better than any other solution in the search space. For this purpose we define:

Definition

A Utopian objective vector $\mathbf{z}^{**}$ *is a vector whose components are slightly smaller than that of the ideal objective vector:* $z_i^{**} = z_i^* - \epsilon_i$ *with* $\epsilon_i > 0$ *for all* $i \in 1, 2, \ldots, M$

Nadir Objective Vector

Definition

The m-th component of the nadir objective vector $\mathbf{z}^{\mathbf{nad}}$ *is the constrained maximum of the following problem:*

$$\begin{array}{ll} \mathit{max} & f_m(\mathbf{x}) \\ \mathit{subject\ to} & \mathbf{x} \in \mathcal{P} \end{array} \tag{5}$$

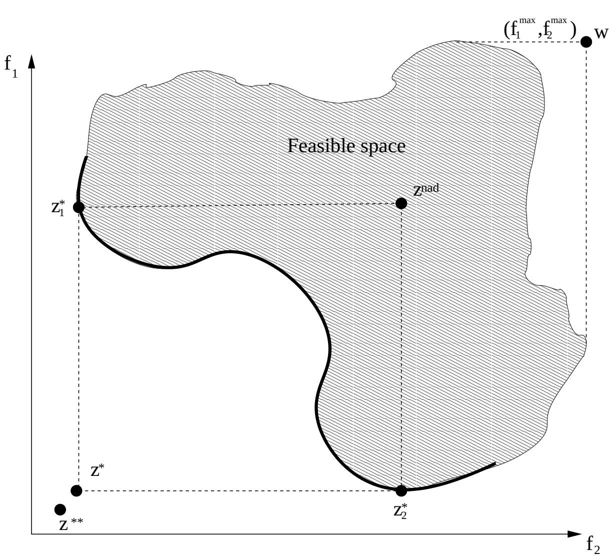
where $\mathcal{P}$ is the Pareto-optimal set.
Unlike the ideal objective vector, $\mathbf{z}^{\mathbf{nad}}$ represents the upper bound of each objective function in the Pareto-optimal set (not in the entire search space).

Nadir & Ideal objective vectors

- The nadir objective vector may represent an existing or non-existing solution (depending on the convexity and continuity of the Pareto-optimal set).
- In order to normalize the each objective, the knowledge of the nadir and ideal vectors can be used as follows:

$$f_i^{norm} = \frac{f_i - z_i^*}{z_i^{nad} - z_i^*} \tag{6}$$

Ideal, Utopian and Nadir objective vectors



Dominance Relation

Definition

A solution $\mathbf{x_1}$ is said to dominate another solution $\mathbf{x_2}$, and we write $\mathbf{x_1} \preceq \mathbf{x_2}$, if both the following conditions are true:

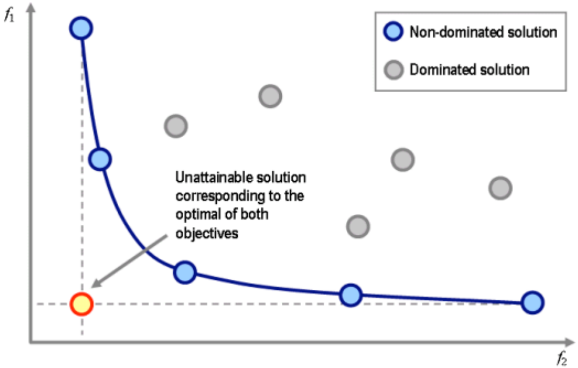
1. *The solution $\mathbf{x_1}$ is no worse than $\mathbf{x_2}$ in all objectives.*
2. *The solution $\mathbf{x_1}$ is strictly better than $\mathbf{x_2}$ in at least one objective.*

$$\mathbf{x_1} \preceq \mathbf{x_2} \textit{ iff } \begin{cases} f_i(\mathbf{x_1}) \leq f_i(\mathbf{x_2}) & \forall i \in 1, \ldots, M \\ \exists j \in 1, \ldots, M & f_j(\mathbf{x_1}) < f_j(\mathbf{x_2}) \end{cases} \tag{7}$$

if any of the above conditions is violated the solution $\mathbf{x_1}$ does not dominate the solution $\mathbf{x_2}$.

Dominance Relation

Example



Dominance Relation

Properties

- REFLEXIVE
  - The dominance relation is *not reflexive* since any solution $\mathbf{x}$ does not dominate itself (by definition of dominance).
- SYMMETRIC
  - The dominance relation is *not symmetric* because $\mathbf{x} \preceq \mathbf{y}$ does not imply $\mathbf{y} \preceq \mathbf{x}$. In fact the opposite is true, if $\mathbf{x} \preceq \mathbf{y}$ then $\mathbf{y} \npreceq \mathbf{x}$
- ANTISYMMETRIC
  - Since the dominance relation is not symmetric it cannot be antisymmetric as well.
- TRANSITIVE
  - The dominance relation is transitive. If $\mathbf{x} \preceq \mathbf{y}$ and $\mathbf{y} \preceq \mathbf{z}$ than $\mathbf{x} \preceq \mathbf{z}$.

Dominance Relation

Transitivity

$$\mathbf{x} \preceq \mathbf{y} \; iff \begin{cases} f_i(\mathbf{x}) \leq f_i(\mathbf{y}) & \forall i \in 1, \ldots, M \\ \exists j \in 1, \ldots, M & f_j(\mathbf{x}) < f_j(\mathbf{y}) \end{cases} \tag{8}$$

$$\mathbf{y} \preceq \mathbf{z} \; iff \begin{cases} f_i(\mathbf{y}) \leq f_i(\mathbf{z}) & \forall i \in 1, \ldots, M \\ \exists k \in 1, \ldots, M & f_k(\mathbf{y}) < f_k(\mathbf{z}) \end{cases} \tag{9}$$

$\mathbf{x} \preceq \mathbf{z}$

- $f_i(\mathbf{x}) \leq f_i(\mathbf{y}) \leq f_i(\mathbf{z}) \quad \forall i \in 1, \ldots, M$
- $\exists l \in 1, \ldots, M \quad f_l(\mathbf{x}) < f_l(\mathbf{z})$
  - if $k = j$ then $f_j(\mathbf{x}) < f_j(\mathbf{y}) < f_j(\mathbf{z})$
  - if $k \neq j$ then
    - if we chose j: $f_j(\mathbf{x}) < f_j(\mathbf{y}) \leq f_j(\mathbf{z})$
    - if we chose k: $f_k(\mathbf{x}) \leq f_k(\mathbf{y}) < f_k(\mathbf{z})$

Dominance Relation

More properties

- Another interesting property is that if a solution **x** does not dominate a solution **y**, this does not imply that **y** dominates **x** (for example they can be both non-dominated).
- The dominance relation qualifies as an ordering relation, because it is at least transitive.
- Since the dominance relation is not reflexive, it is a *strict partial order*.

In general, if a relation is reflexive, antisymmetric and transitive, it is called a *partial order* relation. However, the dominance relation is not reflexive and not antisymmetric, so the dominance relation is not a partial order relation in the the general sense.

Non-dominated set

- For a given set of solution we can perform all possible pairwise comparisons and find which solution dominates which and which solutions are not dominated with respect to each other.
- This set has the property of dominating all the solutions that do not belong to the set.

Definition (Non-dominated set)

Among a set of solutions $\mathcal{P}$, the non-dominated set of solutions $\mathcal{P}'$ are those that are not dominated by any member of the set $\mathcal{P}$.

Globally and Locally Pareto-optimal sets

Like there are global and local optimal solutions in the case of single-objective optimization we can define global and local Pareto-optimal sets.

When the set $\mathcal{P}$ is the entire search space ($\mathcal{P} = \mathcal{S}$) then the resulting non-dominated set $\mathcal{P}'$ is called *Pareto-Optimal set*.

Definition (Globally Pareto-Optimal set)

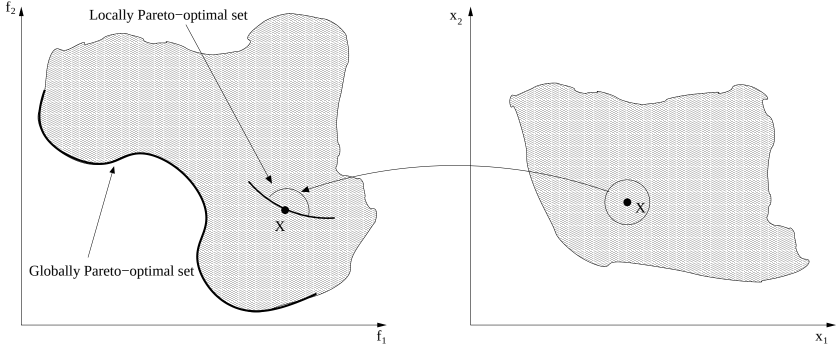
The non-dominated set of the entire feasible search space $\mathcal{S}$ is the globally Pareto-optimal set.

Definition (Locally Pareto-Optimal set)

If for every member $\mathbf{x}$ *in a set* $\mathcal{P}$*, there exists no solutions* $\mathbf{y}$ *in the neighborhood of* $\mathbf{x}$*,* $||\mathbf{y} - \mathbf{x}|| \leq \varepsilon$*, dominating any member of the set* $\mathcal{P}$*, then* $\mathcal{P}$ *constitutes a locally Pareto-optimal set.*

Globally and Locally Pareto-optimal sets

picture



Strong dominance & Weak Pareto-Optimality

The previous definition of dominance is usually referred as the *weak* dominance relation. The *strong* version is defined as follow:

Definition (Strong dominance)

A solution $\mathbf{x}$ *strongly dominates a solution* $\mathbf{y}$, $\mathbf{x} \prec \mathbf{y}$, *if solution* $\mathbf{x}$ *is strictly better than solution* $\mathbf{y}$ *in all* M *objectives.*

Clearly, if $\mathbf{x} \prec \mathbf{y}$ than $\mathbf{x} \preceq \mathbf{y}$ but the vice versa does not hold.

Definition (Weakly non-dominated set)

Among a set of solutions $\mathcal{P}$ *the weakly non-dominated set of solutions* $\mathcal{P}'$ *are those that are not strongly dominated by any other member of the set* $\mathcal{P}$.

We have that $|\mathcal{P}'| \geq |\mathcal{P}|$.

Motivation

- *How do we find the non-dominated set in a given population of solutions?*
- Since the non-dominated set may be required to be identified in each iteration of a multi-objective optimization algorithm, we are interested in computationally *efficient* procedures.
- We will discuss here three procedures, starting from one that is naive and slow to one that is efficient and fast.

Approach 1

Naive and slow

Non-dominated set (V. 1)

```
i := 1
P' := ∅
while (i ≤ |P|)
   j := 1
   while (j ≤ |P|)
     if (j ≠ i)
       if (x_j ⪯ x_i) continue
       else j := j + 1
   end_while
   if (j = |P|) P' := P' ∪ {i}
   i := i + 1
end_while
```

- Let us define the complexity here as the total number of function evaluations.
- the inner while loop requires $O(N)$ comparisons for domination and each comparison needs M function value comparisons. So we have a total complexity of $O(MN)$.
- the outside loop requires $O(N)$ comparisons as well, so we have a total complexity of $O(MN^2)$.

Approach 2

Continuously updated

Non-dominated set (V. 2)

```
P' := {1}
i := 2
while (i ≤ |P|)
   j := 1
   while (j ≤ |P'|)
      if (x_i ≼ x_j)
         P' := P' \ {j}
      else if (x_j ≼ x_i)
         continue
      j := j + 1
   end_while
   if (j = |P'|)
      P' := P' ∪ {i}
   i := i + 1
end_while
```

- $\mathbf{x_2}$ compared with $\mathbf{x_1}$
  $\mathbf{x_3}$ compared at most with $\mathbf{x_1}, \mathbf{x_2}$
  and so on...
- This requires in the worst case:
  $1 + 2 + \cdots + (N - 1) = N(N - 1)/2$
  domination checks.
- Although the total complexity is still $O(MN^2)$, this method requires typically half of that required by approach 1.

Approach 3

Kung et al.'s efficient method (1975)

```
Front(P)
Sort_1(P)
if (|P| = 1)
    then return(P)
else
    T := Front([P_1, P_{⌊|P|/2⌋}])
    B := Front([P_{⌊|P|/2⌋+1}, P_{|P|}])
    i := 1, M := ∅
    while (i ≤ |B|)
      j := 1
        while (j ≤ |T|)
          if (x_j ⊀ x_i)
            then j := j + 1
          else continue
        end_while
      if (j = |T|) M := M ∪ {i}
      i := i + 1
    end_while
return(T ∪ M)
```

- It can be shown that the complexity of this approach is $O(N(logN)^{M-2})$ for $M \geq 4$ and $O(NlogN)$ for $M = 2, 3$, where $N = |\mathcal{P}|$.
- The details of the complexity calculation can be found in: Kung et al. [2].

Non-dominated sorting of a Population

Non-Dominated Sorting

for $j = 1, 2, \ldots$
    $\mathcal{P}_j = \oslash$
end_for
$j := 1$
while $(\mathcal{P} \neq \oslash)$
    $\mathcal{P}' := \mathit{NDS}(\mathcal{P})$
    $\mathcal{P}_j := \mathcal{P}'$
    $\mathcal{P} := \mathcal{P} \setminus \mathcal{P}'$
    $j := j + 1$
end_while
return$(\mathcal{P}_i, i = 1, 2, \ldots, j)$

- Some algorithms require to the population to be classified into several levels of non-domination.
- Complexity? The sum of the individual complexities involved in identifying each non-dominated set.
- $|\mathcal{P}|$ decreases after each non-dominated set computation
- For practical purpose the complexity is governed by the procedure for identifying the first non-dominated set.

An $O(MN^2)$ non-dominated sorting procedure

Non-Dominated Sorting

for $i = 1, 2, \ldots, |\mathcal{P}|$
    $n_i := 0$
    $\mathcal{S}_i := \oslash$
    for $j = 1, 2, \ldots, |\mathcal{P}|$
      if $(x_i \preceq x_j) \& (j \neq i)$
        $\mathcal{S}_i := \mathcal{S}_i \cup \{j\}$
      else if $(x_j \preceq x_i)$
        $n_i := n_i + 1$
    end_for
    if $(n_i = 0)$
      $\mathcal{P}_1 := \mathcal{P}_1 \cup \{i\}$
end_for
$k := 1$
while $(\mathcal{P}_k \neq \oslash)$
    $Q := \oslash$
    for each $i \in \mathcal{P}_k$
      for each $j \in \mathcal{S}_i$
        $n_j := n_j - 1$
        if $(n_j = 0)$ $Q := Q \cup \{j\}$
      end_for
    end_for
    $k := k + 1$
    $\mathcal{P}_k := Q$
end_while

- For each solution we compute two entities:
  - *domination count* n_i: the number of solutions that dominate x_i.
  - $\mathcal{S}_i$: the set of solutions which solution i dominates.
- Complexity is $O(MN^2)$:
  - $O(MN^2)$ for computing n_i and $\mathcal{S}_i$
  - $O(N^2)$ for computing all the fronts $\mathcal{P}_k, k = 1, 2, \ldots$ (non-domination levels).
    Note that each solution will be visited at most $N-1$ times before its domination count become zero and will never be visited again. There are at most $N-1$ such solutions.
- Even if the time complexity is reduced to $O(MN^2)$ the storage has been increased to $O(N^2)$.

Karush-Kuhn-Tucker Theorem for MOOPs

necessary conditions

Theorem

A necessary condition for a solution $\mathbf{x}^*$ *to be Pareto-optimal is that there exist vectors* $\lambda > 0$ *and* $\mathbf{u} \geq 0$ *where (*$\lambda \in \mathbb{R}^M$*,* $\mathbf{u} \in \mathbb{R}^J$ *and* $\lambda, \mathbf{u} \neq \mathbf{0}$*) such that the following conditions are true:*

$$\begin{gathered} \sum_{m=1}^{M} \lambda_m \bigtriangledown f_m(\mathbf{x}^*) - \sum_{j=1}^{J} u_j \bigtriangledown g_j(\mathbf{x}^*) = 0, \textit{and} \\ u_j g_j(\mathbf{x}^*) = 0, \forall j = 1, 2, \ldots, j \end{gathered} \tag{10}$$

The above theorem does not guarantee the existence of a Pareto-optimal solution. This means that any solution that satisfies equation (10) is not necessarily a Pareto-optimal solution.

Optimality conditions

special cases

- For an unconstrained MOOP the above theorem reduces to:

$$\sum_{m=1}^{M} \lambda_m \bigtriangledown f_m(\mathbf{x}^*) = 0 \tag{11}$$

- For a problem with $n = M$ (equal num. of decision variables and objectives) we have:

$$\sum_{m=1}^{M} \bigtriangledown f_m(\mathbf{x}^*) = 0 \tag{12}$$

  The determinant of the partial derivative matrix must be zero for Pareto-optimal solutions.

- For a two-variable, two objective MOOP the above condition reduces to:

$$\frac{\partial f_1}{\partial x_1}\frac{\partial f_2}{\partial x_2} = \frac{\partial f_1}{\partial x_2}\frac{\partial f_2}{\partial x_1} \tag{13}$$

Karush-Kuhn-Tucker Theorem for MOOPs

sufficient conditions

The following theorem offers sufficient conditions for a solution to be Pareto-optimal for convex function.

Theorem

Let the objective functions be convex and the constraint functions non-convex. Let the objective and constraint functions be differentiable at solution $\mathbf{x}^*$. A sufficient condition for $\mathbf{x}^*$ to be Pareto-optimal is that there exist vectors $\lambda > 0$ and $\mathbf{u} \geq 0$ where ($\lambda \in \mathbb{R}^M$ and $\mathbf{u} \in \mathbb{R}^J$) such that the following conditions are true:

$$\begin{aligned}&\textstyle\sum_{m=1}^{M} \lambda_m \bigtriangledown f_m(\mathbf{x}^*) - \sum_{j=1}^{J} u_j \bigtriangledown g_j(\mathbf{x}^*) = 0, \textit{and} \\ &u_j g_j(\mathbf{x}^*) = 0, \forall j = 1, 2, \ldots, j\end{aligned} \tag{14}$$

For Further Reading

Kalyanmoy Deb
Multi-Objective Optimization Using Evolutionary Algorithms.
John Wiley & Sons, Inc, New York, NY, USA, 2001.

Kung, H. T., Luccio, F., and Preparata, F. P. 1975.
On Finding the Maxima of a Set of Vectors.
J. ACM 22, 4 (Oct. 1975), 469-476.