A Short Tutorial on Evolutionary Multiobjective Optimization

Carlos A. Coello Coello

CINVESTAV-IPN
Depto. de Ingeniería Eléctrica
Sección de Computación
Av. Instituto Politécnico Nacional No. 2508
Col. San Pedro Zacatenco
México, D. F. 07300, MEXICO
ccoello@cs.cinvestav.mx

Why Multiobjective Optimization?

Most optimization problems naturally have several objectives to be achieved (normally conflicting with each other), but in order to simplify their solution, they are treated as if they had only one (the remaining objectives are normally handled as constraints).

Basic Concepts

The **Multiobjective Optimization Problem** (MOP) (also called multicriteria optimization, multiperformance or vector optimization problem) can be defined (in words) as the problem of finding (Osyczka, 1985):

> 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.

Basic Concepts

The general Multiobjective Optimization Problem (MOP) can be formally defined as:

Find the vector $\vec{x}^* = [x_1^*, x_2^*, \ldots, x_n^*]^T$ which will satisfy the m inequality constraints:

$$g_i(\vec{x}) \geq 0 \quad i = 1, 2, \ldots, m \tag{1}$$

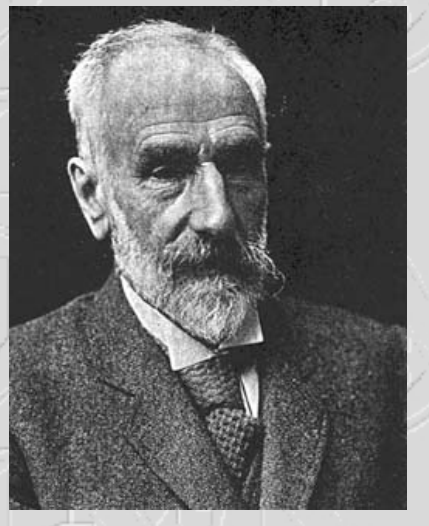
the p equality constraints

$$h_i(\vec{x}) = 0 \quad i = 1, 2, \ldots, p \tag{2}$$

and will optimize the vector function

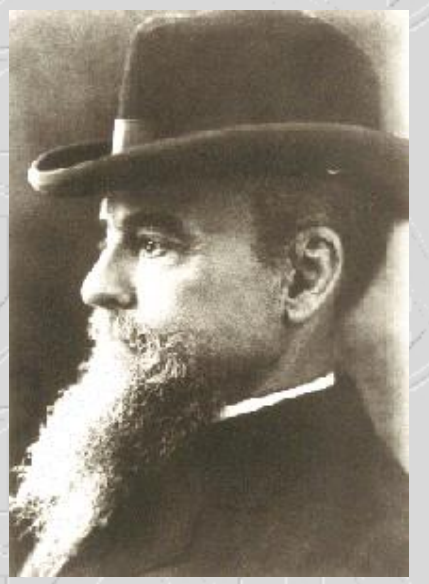
$$\vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_k(\vec{x})]^T \tag{3}$$

Basic Concepts



Having several objective functions, the notion of "optimum" changes, because in MOPs, we are really trying to find good compromises (or "trade-offs") rather than a single solution as in global optimization. The notion of "optimum" that is most commonly adopted is that originally proposed by Francis Ysidro Edgeworth in 1881.

Basic Concepts



This notion was later generalized by Vilfredo Pareto (in 1896). Although some authors call *Edgeworth-Pareto optimum* to this notion, we will use the most commonly accepted term: *Pareto optimum*.

Basic Concepts

We say that a vector of decision variables $\vec{x}^* \in \mathcal{F}$ is *Pareto optimal* if there does not exist another $\vec{x} \in \mathcal{F}$ such that $f_i(\vec{x}) \leq f_i(\vec{x}^*)$ for all $i = 1, \ldots, k$ and $f_j(\vec{x}) < f_j(\vec{x}^*)$ for at least one j.

Basic Concepts

In words, this definition says that $\vec{x}^*$ is Pareto optimal if there exists no feasible vector of decision variables $\vec{x} \in \mathcal{F}$ which would decrease some criterion without causing a simultaneous increase in at least one other criterion. Unfortunately, this concept almost always gives not a single solution, but rather a set of solutions called the *Pareto optimal set*. The vectors $\vec{x}^*$ correspoding to the solutions included in the Pareto optimal set are called *nondominated*. The plot of the objective functions whose nondominated vectors are in the Pareto optimal set is called the *Pareto front*.

An Example

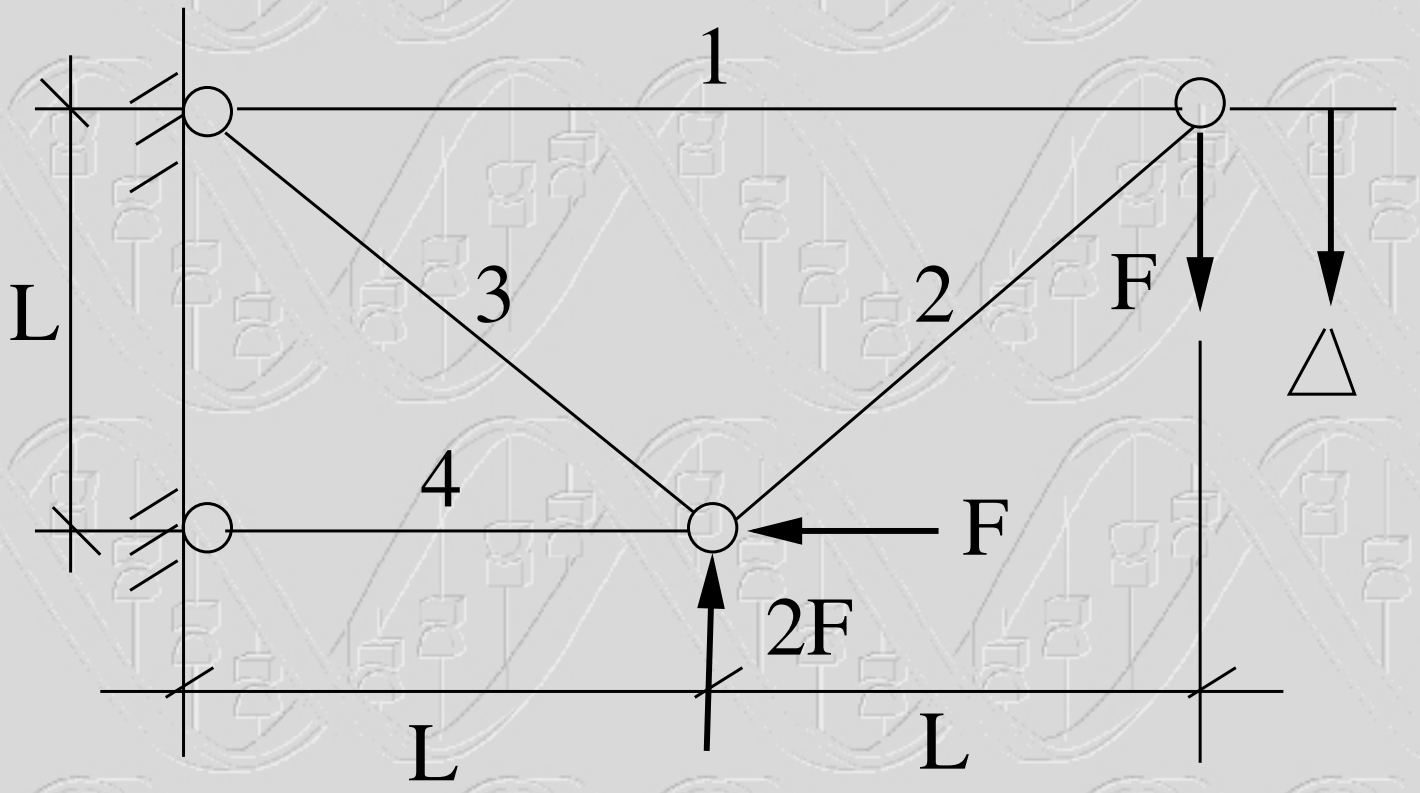


Figura 1: A four-bar plane truss.

Example

$$\text{Minimize} \begin{cases} f_1(\vec{x}) = L\left(2x_1 + \sqrt{2}x_2 + \sqrt{x_3} + x_4\right) \\ f_2(\vec{x}) = \frac{FL}{E}\left(\frac{2}{x_1} + \frac{2\sqrt{2}}{x_2} - \frac{2\sqrt{2}}{x_3} + \frac{2}{x_4}\right) \end{cases} \tag{4}$$

such that:

$$\begin{aligned} (F/\sigma) &\leq x_1 \leq 3(F/\sigma) \\ \sqrt{2}(F/\sigma) &\leq x_2 \leq 3(F/\sigma) \\ \sqrt{2}(F/\sigma) &\leq x_3 \leq 3(F/\sigma) \\ (F/\sigma) &\leq x_4 \leq 3(F/\sigma) \end{aligned} \tag{5}$$

where $F = 10$ kN, $E = 2 \times 10^5$ kN/cm^2, $L = 200$ cm, $\sigma = 10$ kN/cm^2.

Example

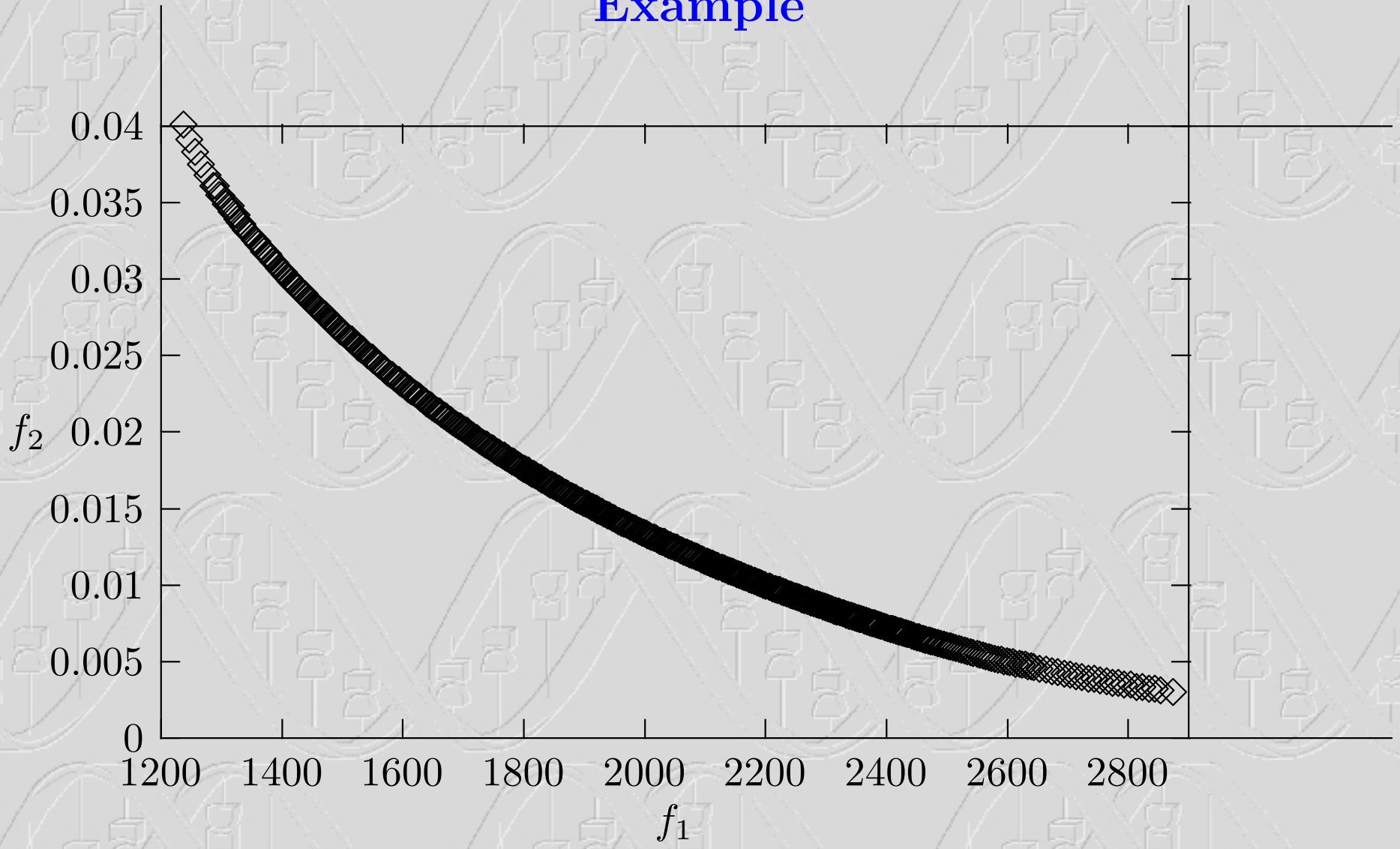
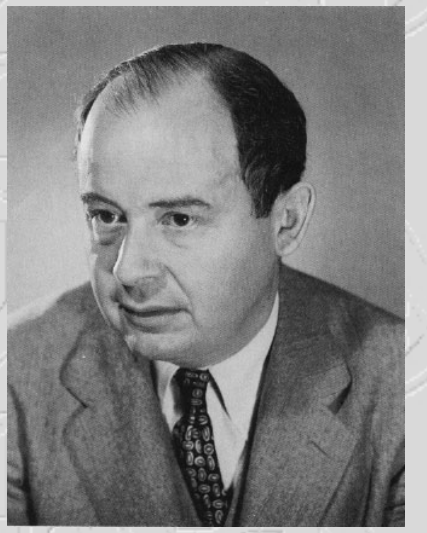


Figura 2: True Pareto front of the four-bar plane truss problem.

Some Historical Highlights



As early as 1944, John von Neumann and Oskar Morgenstern mentioned that an optimization problem in the context of a social exchange economy was "a peculiar and disconcerting mixture of several conflicting problems" that was "nowhere dealt with in classical mathematics".

Some Historical Highlights

In 1951 Tjalling C. Koopmans edited a book called *Activity Analysis of Production and Allocation*, where the concept of "efficient" vector was first used in a significant way.

Some Historical Highlights

The origins of the mathematical foundations of multiobjective optimization can be traced back to the period that goes from 1895 to 1906. During that period, Georg Cantor and Felix Hausdorff laid the foundations of infinite dimensional ordered spaces.

Some Historical Highlights

Cantor also introduced equivalence classes and stated the first sufficient conditions for the existence of a utility function. Hausdorff also gave the first example of a complete ordering. However, it was the concept of *vector maximum problem* introduced by Harold W. Kuhn and Albert W. Tucker (1951) which made multiobjective optimization a mathematical discipline on its own.

Some Historical Highlights

However, multiobjective optimization theory remained relatively undeveloped during the 1950s. It was until the 1960s that the foundations of multiobjective optimization were consolidated and taken seriously by pure mathematicians when Leonid Hurwicz generalized the results of Kuhn & Tucker to topological vector spaces.

Some Historical Highlights

The application of multiobjective optimization to domains outside economics began with the work by Koopmans (1951) in production theory and with the work of Marglin (1967) in water resources planning. The first engineering application reported in the literature was a paper by Zadeh in the early 1960s. However, the use of multiobjective optimization became generalized until the 1970s.

Some Historical Remarks

Currently, there are over 30 mathematical programming techniques for multiobjective optimization. However, these techniques tend to generate elements of the Pareto optimal set one at a time. Additionally, most of them are very sensitive to the shape of the Pareto front (i.e., they do not work when the Pareto front is concave).

Why Evolutionary Algorithms?

The potential of evolutionary algorithms in multiobjective optimization was hinted by Rosenberg in the 1960s, but the first actual implementation was produced in the mid-1980s (Schaffer, 1984). During ten years, the field remain practically inactive, but it started growing in the mid-1990s, in which several techniques and applications were developed.

Why Evolutionary Algorithms?

Evolutionary algorithms seem particularly suitable to solve multiobjective optimization problems, because they deal simultaneously with a set of possible solutions (the so-called population). This allows us to find several members of the Pareto optimal set in a single run of the algorithm, instead of having to perform a series of separate runs as in the case of the traditional mathematical programming techniques. Additionally, evolutionary algorithms are less susceptible to the shape or continuity of the Pareto front (e.g., they can easily deal with discontinuous or concave Pareto fronts), whereas these two issues are a real concern for mathematical programming techniques.

Classifying Techniques

We will use the following simple classification of Evolutionary Multi-Objective Optimization (EMOO) approaches:

- Non-Pareto Techniques
- Pareto Techniques
- Recent Approaches

Classifying Techniques

Non-Pareto Techniques include the following:

- Aggregating approaches
- VEGA
- Lexicographic ordering
- The ε-constraint Method
- Target-vector approaches

Classifying Techniques

Pareto-based Techniques include the following:

- Pure Pareto ranking
- MOGA
- NSGA
- NPGA

Classifying Techniques

Finally, we will also briefly review two recent approaches

- PAES
- SPEA

Non-Pareto Techniques

- Approaches that do not incorporate directly the concept of Pareto optimum.
- Incapable of producing certain portions of the Pareto front.
- Efficient and easy to implement, but appropriate to handle only a few objectives.

Aggregation Functions

- These techniques are called "aggregating functions" because they combine (or "aggregate") all the objectives into a single one. We can use addition, multiplication or any other combination of arithmetical operations.
- Oldest mathematical programming method, since aggregating functions can be derived from the Kuhn-Tucker conditions for nondominated solutions.

Aggregation Functions

An example of this approach is a sum of weights of the form:

$$\min \sum_{i=1}^{k} w_i f_i(\vec{x}) \tag{6}$$

where $w_i \geq 0$ are the weighting coefficients representing the relative importance of the k objective functions of our problem. It is usually assumed that

$$\sum_{i=1}^{k} w_i = 1 \tag{7}$$

Advantages and Disadvantages

- Easy to implement
- Efficient
- Will not work when the Pareto front is concave, regardless of the weights used (Das, 1997).

Sample Applications

- Truck packing problems (Grignon, 1996).
- Real-time scheduling (Montana, 1998).
- Structural synthesis of cell-based VLSI circuits (Arslan, 1996).
- Design of optical filters for lamps (Eklund, 1999).

Vector Evaluated Genetic Algorithm (VEGA)

- Proposed by Schaffer in the mid-1980s (1984,1985).
- Only the selection mechanism of the GA is modified so that at each generation a number of sub-populations was generated by performing proportional selection according to each objective function in turn. Thus, for a problem with k objectives and a population size of M, k sub-populations of size M/k each would be generated. These sub-populations would be shuffled together to obtain a new population of size M, on which the GA would apply the crossover and mutation operators in the usual way.

VEGA

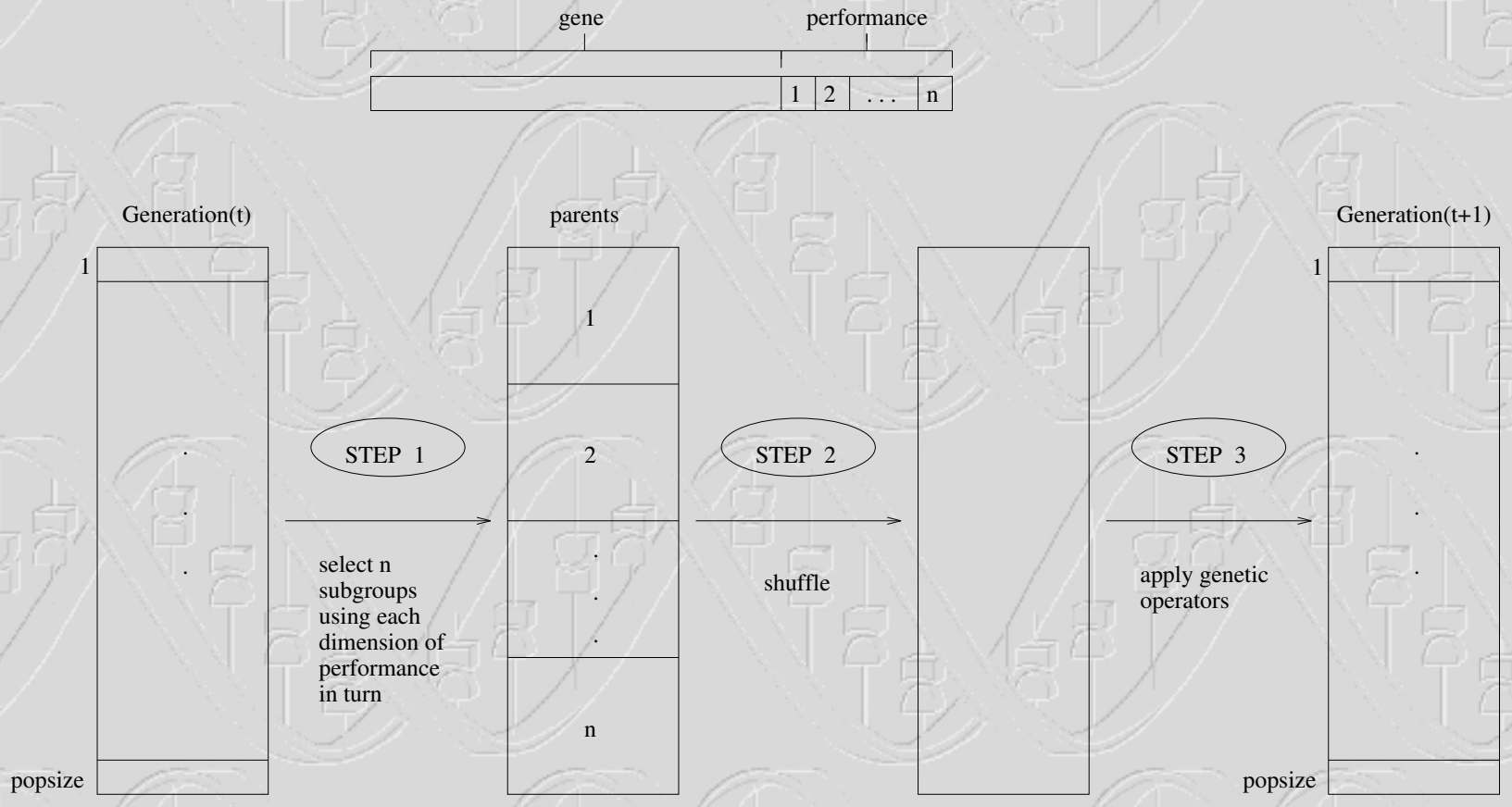


Figura 3: Schematic of VEGA selection.

Advantages and Disadvantages

- Efficient and easy to implement.
- If proportional selection is used, then the shuffling and merging of all the sub-populations corresponds to averaging the fitness components associated with each of the objectives. In other words, under these conditions, VEGA behaves as an aggregating approach and therefore, it is subject to the same problems of such techniques.

Sample Applications

- Optimal location of a network of groundwater monitoring wells (Cieniawski, 1995).
- Combinational circuit design at the gate-level (Coello, 2000).
- Design multiplierless IIR filters (Wilson, 1993).
- Groundwater pollution containment (Ritzel, 1994).

Lexicographic Ordering

In this method, the user is asked to rank the objectives in order of importance. The optimum solution is then obtained by minimizing the objective functions, starting with the most important one and proceeding according to the assigned order of importance of the objectives.

It is also possible to select randomly a single objective to optimize at each run of a GA.

Advantages and Disadvantages

- Efficient and easy to implement.
- Requires a pre-defined ordering of objectives and its performance will be affected by it.
- Selecting randomly an objective is equivalent to a weighted combination of objectives, in which each weight is defined in terms of the probability that each objective has of being selected. However, if tournament selection is used, the technique does not behave like VEGA, because tournament selection does not require scaling of the objectives (because of its pairwise comparisons). Therefore, the approach may work properly with concave Pareto fronts.
- Inappropriate when there is a large amount of objectives.

Sample Applications

- Symbolic layout compaction (Fourman, 1985).
- Tuning of a fuzzy controller for the guidance of an autonomous vehicle in an elliptic road (Gacôgne, 1997).

The ε-Constraint Method

This method is based on minimization of one (the most preferred or primary) objective function, and considering the other objectives as constraints bound by some allowable levels ε_i. Hence, a single objective minimization is carried out for the most relevant objective function subject to additional constraints on the other objective functions. The levels ε_i are then altered to generate the entire Pareto optimal set.

Advantages and Disadvantages

- Easy to implement.
- Potentially high computational cost (many runs may be required).

Sample Applications

- Preliminary design of a marine vehicle (Lee, 1997).
- Groundwater pollution containment problems (Ranjithan, 1992).
- Fault tolerant system design (Schott, 1995).

Target-Vector Approaches

- Definition of a set of goals (or targets) that we wish to achieve for each objective function. EA minimizes differences between the current solution and these goals.
- Can also be considered aggregating approaches, but in this case, concave portions of the Pareto front could be obtained.
- Examples: Goal Programming, Goal Attainment, min-max method.

Advantages and Disadvantages

- Efficient and easy to implement.
- Definition of goals may be difficult in some cases and may imply an extra computational cost.
- Some of them (e.g., goal attainment) may introduce a misleading selection pressure under certain circumstances.
- Goals must lie in the feasible region so that the solutions generated are members of the Pareto optimal set.

Sample Applications

- Design of multiplierless IIR filters (Wilson, 1993).
- Structural optimization (Sandgren, 1994; Hajela, 1992).
- Optimization of the counterweight balancing of a robot arm (Coello, 1998).

Pareto-based Techniques

- Suggested by Goldberg (1989) to solve the problems with Schaffer's VEGA.
- Use of nondominated ranking and selection to move the population towards the Pareto front.
- Requires a ranking procedure and a technique to maintain diversity in the population (otherwise, the GA will tend to converge to a single solution, because of the stochastic noise involved in the process).

Fitness Sharing

Goldberg & Richardson (1987) proposed the use of an approach in which the population was divided in different subpopulations according to the similarity of the individuals in two possible solution spaces: the decoded parameter space (phenotype) and the gene space (genotype). They defined a sharing function $\phi(d_{ij})$ as follows:

$$\phi(d_{ij}) = \begin{cases} 1 - \left(\frac{d_{ij}}{\sigma_{sh}}\right)^{\alpha}, & d_{ij} < \sigma_{share} \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

where normally $\alpha = 1$, d_{ij} is a metric indicative of the distance between designs i and j, and σ_{share} is the sharing parameter which controls the extent of sharing allowed.

Fitness Sharing

The fitness of a design i is then modified as:

$$f_{s_i} = \frac{f_i}{\sum_{j=1}^{M} \phi(d_{ij})} \tag{9}$$

where M is the number of designs located in vicinity of the i-th design.

Deb and Goldberg (1989) proposed a way of estimating the parameter σ_{share} in both phenotypical and genotypical space.

Fitness Sharing

In phenotypical sharing, the distance between 2 individuals is measured in decoded parameter space, and can be calculated with a simple Euclidean distance in an p-dimensional space, where p refers to the number of variables encoded in the GA; the value of d_{ij} can then be calculated as:

$$d_{ij} = \sqrt{\sum_{k=1}^{p} (x_{k,i} - x_{k,j})^2} \tag{10}$$

where $x_{1,i}, x_{2,i}, \ldots, x_{p,i}$ and $x_{1,j}, x_{2,j}, \ldots, x_{p,j}$ are the variables decoded from the EA.

Fitness Sharing

In genotypical sharing, d_{ij} is defined as the Hamming distance between the strings and σ_{share} is the maximum number of different bits allowed between the strings to form separate niches in the population.

The experiments performed by Deb and Goldberg (1989) indicated that phenotypic sharing was better than genotypic sharing.

Other authors have also proposed their own methodology to compute σ_{share} (see for example: (Fonseca & Fleming, 1993)).

Pure Pareto Ranking

Although several variations of Goldberg's proposal have been proposed in the literature (see the following subsections), several authors have used what we call "pure Pareto ranking". The idea in this case is to follow Goldberg's proposal as stated in his book (1989).

Advantages and Disadvantages

- Relatively easy to implement.
- Problem to scale the approach, because checking for nondominance is $O(kM^2)$, where k is the amount of objectives and M is the population size.
- Fitness sharing is $O(M^2)$.
- The approach is less susceptible to the shape or continuity of the Pareto front.

Sample Applications

- Optimal location of a network of groundwater monitoring wells (Cieniawski, 1995).
- Pump scheduling (Schwab, 1996;Savic, 1997).
- Feasibility of full stern submarines (Thomas, 1998).
- Optimal planning of an electrical power distribution system (Ramírez, 1999).

Multi-Objective Genetic Algorithm (MOGA)

- Proposed by Fonseca and Fleming (1993).
- The approach consists of a scheme in which the rank of a certain individual corresponds to the number of individuals in the current population by which it is dominated.
- It uses fitness sharing and mating restrictions.

Advantages and Disadvantages

- Efficient and relatively easy to implement.
- Its performance depends on the appropriate selection of the sharing factor.
- MOGA has been very popular and tends to perform well when compared to other EMOO approaches.

Some Applications

- Fault diagnosis (Marcu, 1997).
- Control system design (Chipperfield 1995; Whidborne, 1995; Duarte, 2000).
- Wing planform design (Obayashi, 1998).
- Design of multilayer microwave absorbers (Weile, 1996).

Nondominated Sorting Genetic Algorithm (NSGA)

- Proposed by Srinivas and Deb (1994).
- It is based on several layers of classifications of the individuals. Nondominated individuals get a certain dummy fitness value and then are removed from the population. The process is repeated until the entire population has been classified.
- To maintain the diversity of the population, classified individuals are shared (in decision variable space) with their dummy fitness values.

NSGA

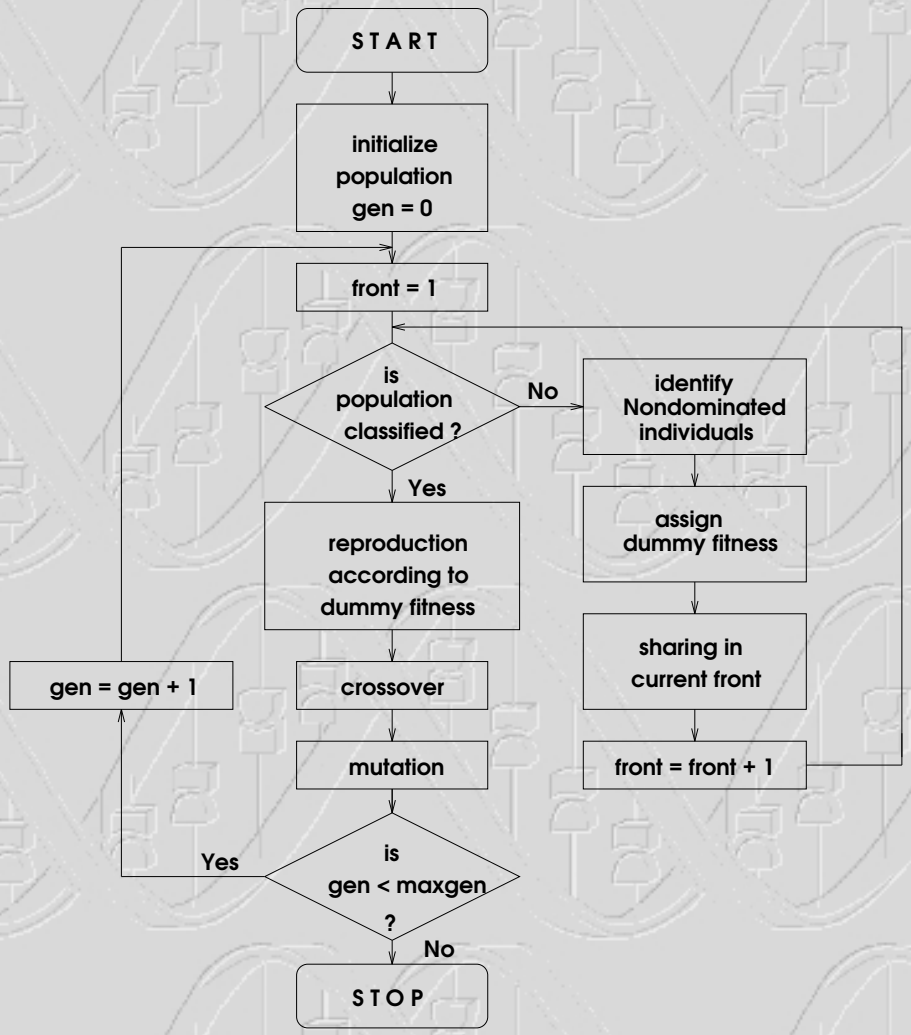


Figura 4: Flowchart of the Nondominated Sorting Genetic Algorithm (NSGA).

Advantages and Disadvantages

- Relatively easy to implement.
- Seems to be very sensitive to the value of the sharing factor.
- Has been recently improved (NSGA II) with elitism and a crowded comparison operator that keeps diversity without specifying any additional parameters.

Sample Applications

- Airfoil shape optimization (Mäniken, 1998).
- Scheduling (Bagchi, 1999).
- Minimum spanning tree (Zhou, 1999).
- Computational fluid dynamics (Marco, 1999).

Niched-Pareto Genetic Algorithm (NPGA)

- Proposed by Horn et al. (1993,1994).
- It uses a tournament selection scheme based on Pareto dominance. Two individuals randomly chosen are compared against a subset from the entire population (typically, around 10 % of the population). When both competitors are either dominated or nondominated (i.e., when there is a tie), the result of the tournament is decided through fitness sharing in the objective domain (a technique called *equivalent class sharing* was used in this case).

NPGA

The pseudocode for Pareto domination tournaments assuming that all of the objectives are to be maximized is presented in the next slide. S is an array of the N individuals in the current population, $random\_pop\_index$ is an array holding the N indices of S, in a random order, and t_{dom} is the size of the comparison set.

NPGA

```
function selection /* Returns an individual from the current population S */
begin
    shuffle(random_pop_index); /* Re-randomize random index array */
    candidate_1 = random_pop_index[1];
    candidate_2 = random_pop_index[2];
    candidate_1_dominated = false;
    candidate_2_dominated = false;
    for comparison_set_index = 3 to t_dom + 3 do
    /* Select t_dom individuals randomly from S */
        begin
            comparison_individual = random_pop_index[comparison_set_index];
            if S[comparison_individual] dominates S[candidate_1]
                then candidate_1_dominated = true;
            if S[comparison_individual] dominates S[candidate_2]
                then candidate_2_dominated = true;
        end /* end for loop */
    if ( candidate_1_dominated AND ¬ candidate_2_dominated )
        then return candidate_2;
    else if ( ¬ candidate_1_dominated AND candidate_2_dominated )
        then return candidate_1;
    else
        do sharing;
end
```

Advantages and Disadvantages

- Easy to implement.
- Efficient because does not apply Pareto ranking to the entire population.
- It seems to have a good overall performance.
- Besides requiring a sharing factor, it requires another parameter (tournament size).

Sample Applications

- Automatic derivation of qualitative descriptions of complex objects (Ruspini, 1999).
- Feature selection (Emmanouilidis, 2000).
- Optimal well placement for groundwater containment monitoring (Horn, 1994).
- Investigation of feasibility of full stern submarines (Thomas, 1998).

Recent approaches

The *Pareto Archived Evolution Strategy* (PAES) was introduced by Knowles & Corne (2000).

This approach is very simple: it uses a (1+1) evolution strategy (i.e., a single parent that generates a single offspring) together with a historical archive that records all the nondominated solutions previously found (such archive is used as a comparison set in a way analogous to the tournament competitors in the NPGA).

Recent approaches

PAES also uses a novel approach to keep diversity, which consists of a crowding procedure that divides objective space in a recursive manner. Each solution is placed in a certain grid location based on the values of its objectives (which are used as its "coordinates" or "geographical location").

A map of such grid is maintained, indicating the amount of solutions that reside in each grid location. Since the procedure is adaptive, no extra parameters are required (except for the number of divisions of the objective space). Furthermore, the procedure has a lower computational complexity than traditional niching methods.

PAES has been used to solve the off-line routing problem (1999) and the adaptive distributed database management problem (2000).

Recent approaches

The *Strength Pareto Evolutionary Algorithm* (SPEA) was introduced by Zitzler & Thiele [Zitzler 99c].

This approach was conceived as a way of integrating different EMOO techniques. SPEA uses an archive containing nondominated solutions previously found (the so-called external nondominated set). At each generation, nondominated individuals are copied to the external nondominated set. For each individual in this external set, a strength value is computed.

Recent approaches

This "strength" is similar to the ranking value of MOGA, since it is proportional to the number of solutions to which a certain individual dominates. The fitness of each member of the current population is computed according to the strengths of all external nondominated solutions that dominate it.

Additionally, a clustering technique called "average linkage method" (Morse, 1980) is used to keep diversity.

SPEA has been used to explore trade-offs of software implementations for DSP algorithms (1999) and to solve 0/1 knapsack problems (1999a).

Theory

The most important theoretical work related to EMOO has concentrated on two main issues:

- Studies of convergence towards the Pareto optimum set (Rudolph, 1998, 2000; Hanne 2000,2000a; Veldhuizen, 1998).
- Ways to compute appropriate sharing factors (or niche sizes) (Horn, 1997, Fonseca, 1993).

Theory

Much more work is needed. For example:

- To study the structure of fitness landscapes (Kaufmann, 1989) in multiobjective optimization problems.
- Detailed studies of the different aspects involved in the parallelization of EMOO techniques (e.g., load balancing, impact on Pareto convergence, etc.).

Test Functions

- Good benchmarks were disregarded for many years.
- Recently, there have been several proposals to design test functions suitable to evaluate EMOO approaches.
- Constrained test functions are of particular interest.
- Multiobjective combinatorial optimization problems have also been proposed.

Metrics

Three are normally the issues to take into consideration to design a good metric in this domain (Zitzler, 2000):

1. Minimize the distance of the Pareto front produced by our algorithm with respect to the true Pareto front (assuming we know its location).
2. Maximize the spread of solutions found, so that we can have a distribution of vectors as smooth and uniform as possible.
3. Maximize the amount of elements of the Pareto optimal set found.

Metrics

Enumeration: Enumerate the entire intrinsic search space explored by an EA and then compare the true Pareto front obtained against those fronts produced by any EMOO approach.

Obviously, this has some serious scalability problems.

Metrics

Spread: Use of a statistical metric such as the chi-square distribution to measure "spread" along the Pareto front.

This metric assumes that we know the true Pareto front of the problem.

Metrics

Attainment Surfaces: Draw a boundary in objective space that separates those points which are dominated from those which are not (this boundary is called "attainment surface").

Perform several runs and apply standard non-parametric statistical procedures to evaluate the quality of the nondominated vectors found.

It is unclear how can we really assess how much better is a certain approach with respect to others.

Metrics

Generational Distance: Estimates how far is our current Pareto front from the true Pareto front of a problem using the Euclidean distance (measured in objective space) between each vector and the nearest member of the true Pareto front.

The problem with this metric is that only distance to the true Pareto front is considered and not uniform spread along the Pareto front.

Metrics

Coverage: Measure the size of the objective value space area which is covered by a set of nondominated solutions.

It combines the three issues previously mentioned (distance, spread and amount of elements of the Pareto optimal set found) into a single value. Therefore, sets differing in more than one criterion cannot be distinguished.

Promising areas of future research

- Incorporation of preferences
- Emphasis on efficiency
- More test functions and metrics

Promising areas of future research

- More theoretical studies
- New approaches (hybrids with other heuristics)
- New applications