Metaheuristics for 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

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.

What is a multiobjective optimization problem?

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.

A Formal Definition

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:

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

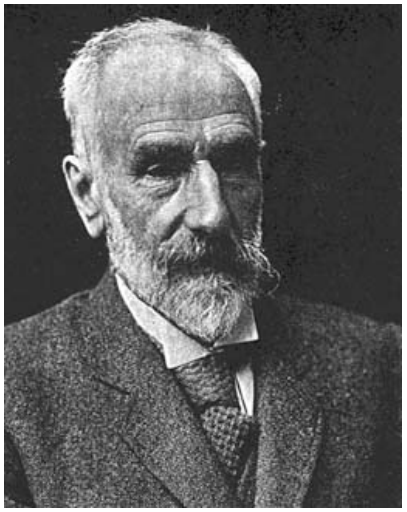
the p equality constraints

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

and will optimize the vector function

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

What is the notion of optimum in multiobjective optimization?



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.

What is the notion of optimum in multiobjective optimization?

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*.

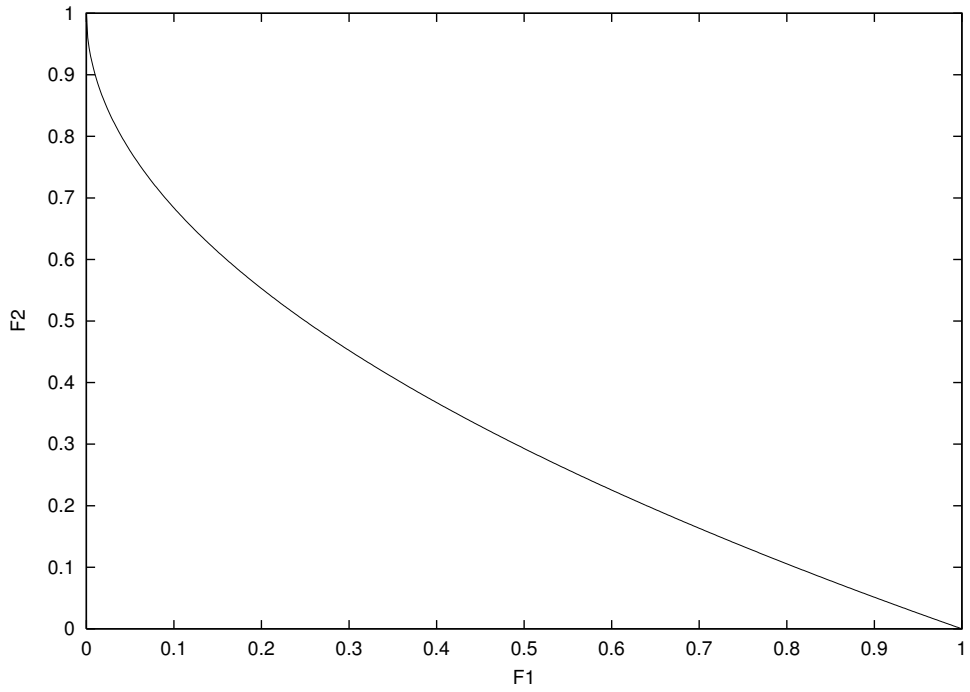
Definition of Pareto Optimality

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.

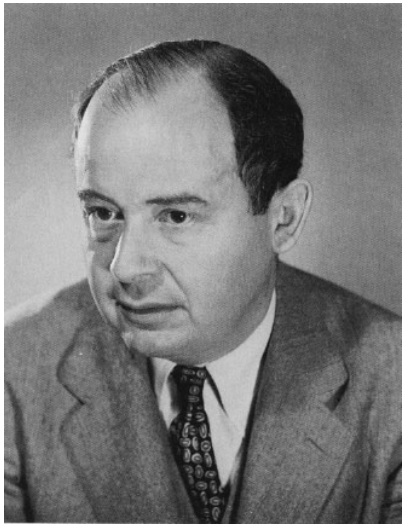
Definition of Pareto Optimality

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*.

Sample Pareto Front



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.

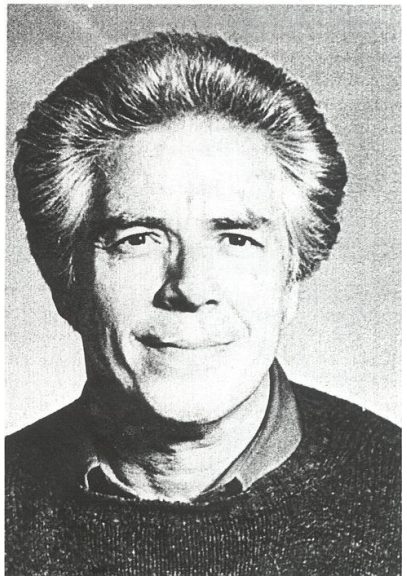
Some Historical Highlights

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

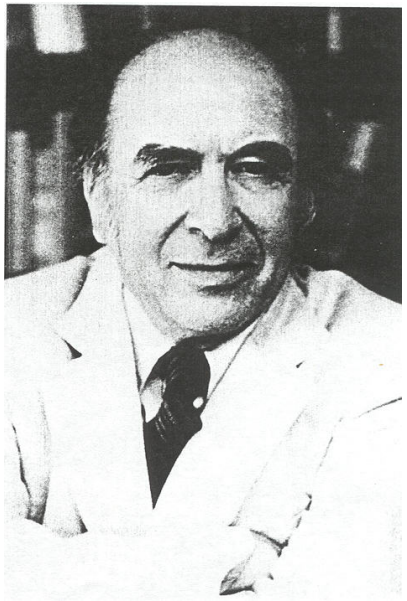
Cantor also introduced equivalence classes and stated the first sufficient conditions for the existence of a utility function.
Hausdorff also gave the first example of a complete ordering.

Some Historical Highlights



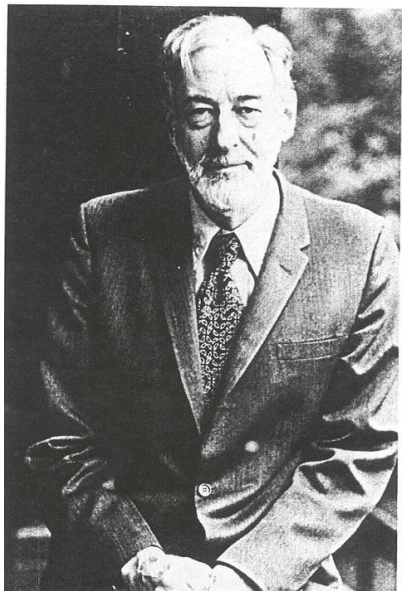
However, it was the concept of *vector maximum problem* introduced by Harold W. Kuhn and Albert W. Tucker (1951) which made multiobjective optimization a mathematical discipline on its own.

Some Historical Highlights



Nevertheless, 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

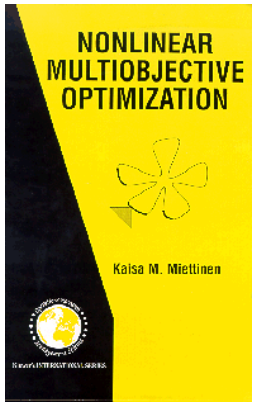


The application of multiobjective optimization to domains outside economics began with the work by Koopmans (1951) in production theory and with the work of Marglin (1967) in water resources planning.

Some Historical Highlights

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.

Current State of the Area



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 (e.g., they do not work when the Pareto front is concave or when the front is disconnected).

Why Heuristics?

Heuristics seem particularly suitable to solve multiobjective optimization problems, because they 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 this is a real concern for mathematical programming techniques. Additionally, many current heuristics (e.g., evolutionary algorithms, particle swarm optimization, etc.) are population-based, which means that we can aim to generate several elements of the Pareto optimal set in a single run.

Evolutionary Algorithms

We can consider, in general, two main types of multi-objective evolutionary algorithms (MOEAs):

1. Algorithms that do not incorporate the concept of Pareto dominance in their selection mechanism (e.g., approaches that use linear aggregating functions).
2. Algorithms that rank the population based on Pareto dominance. For example, MOGA, NSGA, NPGA, etc.

Evolutionary Algorithms

Historically, we can consider the existence of two main generations of MOEAs:

1. **First Generation**: Characterized by the use of Pareto ranking and niching (or fitness sharing). Relatively simple algorithms. Other (more rudimentary) approaches were also developed (e.g., linear aggregating functions). It is also worth mentioning VEGA, which is a population-based (not Pareto-based) approach.
2. **Second Generation**: The concept of elitism is introduced in two main forms: using $(\mu + \lambda)$ selection and using a secondary (external) population.

Representative MOEAs (First Generation)

- VEGA
- MOGA
- NSGA
- NPGA

Vector Evaluated Genetic Algorithm (VEGA)

- Proposed by Schaffer in the mid-1980s (1984,1985).
- It uses subpopulations that optimize each objective separately. The concept of Pareto optimum is not directly incorporated into the selection mechanism of the GA.

Vector Evaluated Genetic Algorithm (VEGA)

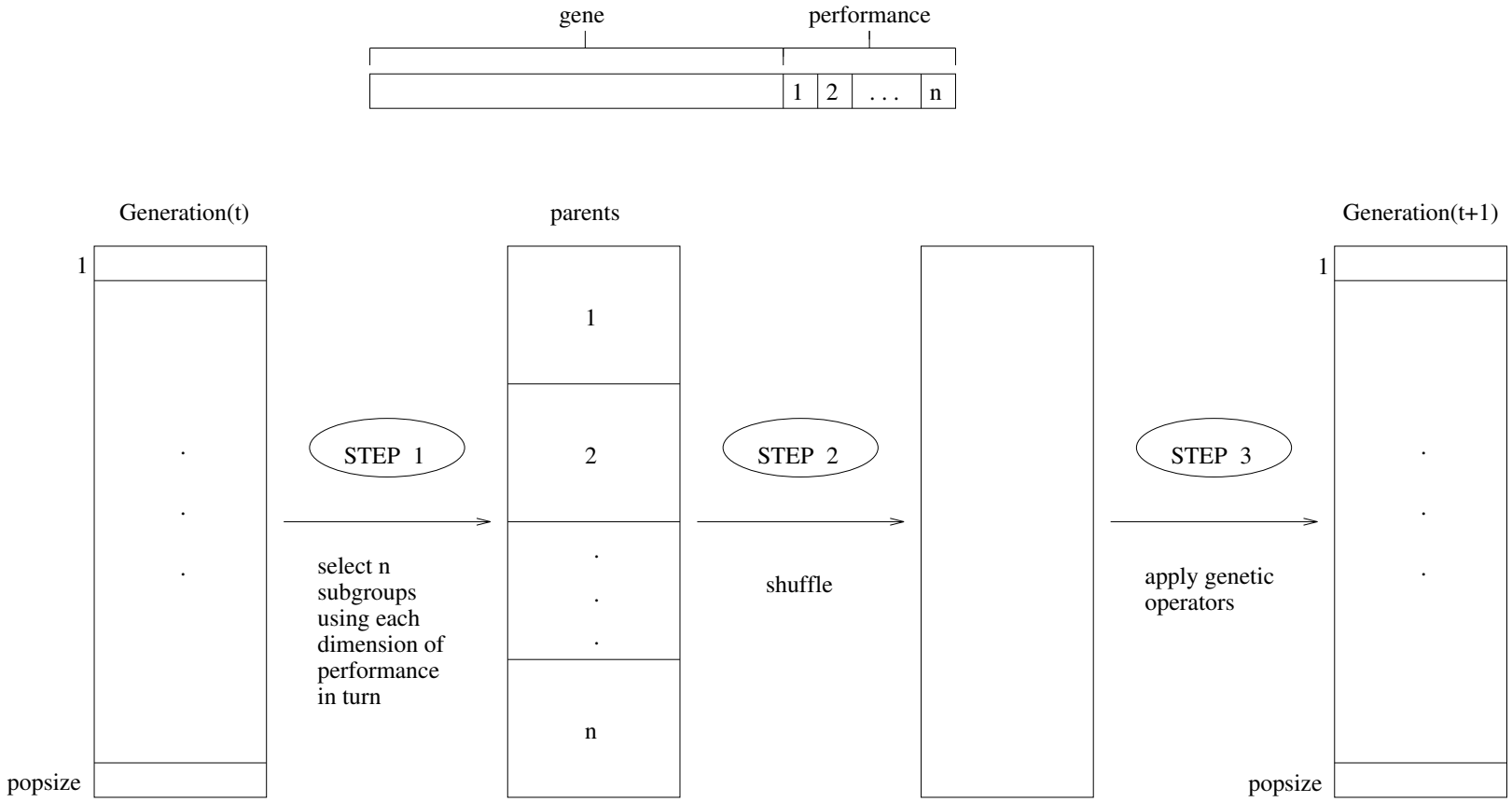


Figure 1: Schematic of VEGA selection.

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.

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.

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* is used in this case).

Representative MOEAs (Second Generation)

- SPEA and SPEA2
- NSGA-II
- PAES, PESA and PESA II
- MOMGA and MOMGA II
- The microGA for multiobjective optimization

The Strength Pareto Evolutionary Algorithm (SPEA)

SPEA was introduced by Zitzler & Thiele (1999).

It uses an external archive containing nondominated solutions previously found.

It computers a strength value similar to the ranking value used by MOGA.

A clustering technique called "average linkage method" is used to keep diversity.

The Strength Pareto Evolutionary Algorithm 2 (SPEA2)

A revised version of SPEA has been recently proposed: SPEA2 (Zitzler, 2001). SPEA2 has three main differences with respect to its predecessor: (1) it incorporates a fine-grained fitness assignment strategy which takes into account for each individual the number of individuals that dominate it and the number of individuals by which it is dominated; (2) it uses a nearest neighbor density estimation technique which guides the search more efficiently, and (3) it has an enhanced archive truncation method that guarantees the preservation of boundary solutions.

The Nondominated Sorting Genetic Algorithm II (NSGA-II)

Deb et al. (2000,2002) proposed a new version of the Nondominated Sorting Genetic Algorithm (NSGA), called NSGA-II, which is more efficient (computationally speaking), it uses elitism and a crowded comparison operator that keeps diversity without specifying any additional parameters.

The Pareto Archived Evolution Strategy (PAES)

PAES was introduced by Knowles & Corne (2000).

It uses a (1+1) evolution strategy together with an external archive that records all the nondominated vectors previously found.

It uses an adaptive grid to maintain diversity.

The Pareto Envelope-based Selection Algorithm (PESA)

PESA was proposed by Corne et al. (2000). This approach uses a small internal population and a larger external (or secondary) population. PESA uses the same hyper-grid division of phenotype (i.e., objective funcion) space adopted by PAES to maintain diversity. However, its selection mechanism is based on the crowding measure used by the hyper-grid previously mentioned. This same crowding measure is used to decide what solutions to introduce into the external population (i.e., the archive of nondominated vectors found along the evolutionary process).

The Pareto Envelope-based Selection Algorithm-II (PESA-II)

PESA-II (Corne et al., 2001) is a revised version of PESA in which region-based selection is adopted. In region-based selection, the unit of selection is a hyperbox rather than an individual. The procedure consists of selecting (using any of the traditional selection techniques) a hyperbox and then randomly select an individual within such hyperbox.

The Multi-Objective Messy Genetic Algorithm (MOMGA)

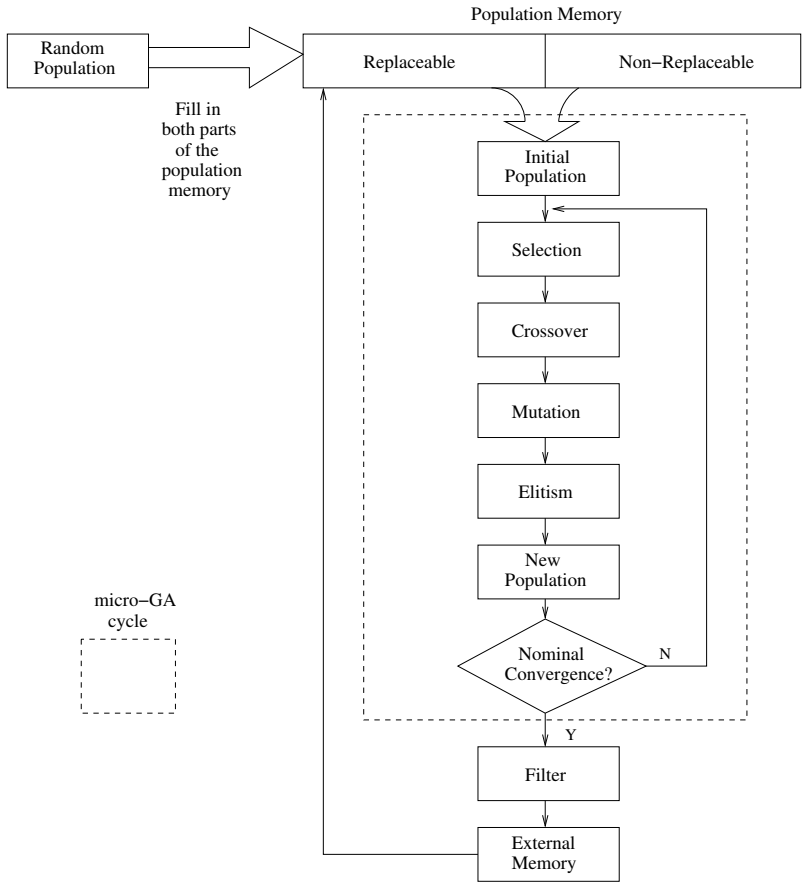
MOMGA was proposed by Van Veldhuizen and Lamont (2000). This is an attempt to extend the messy GA to solve multiobjective optimization problems.

MOMGA consists of three phases: (1) Initialization Phase, (2) Primordial Phase, and (3) Juxtapositional Phase. In the *Initialization Phase*, MOMGA produces all building blocks of a certain specified size through a deterministic process known as partially enumerative initialization. The *Primordial Phase* performs tournament selection on the population and reduces the population size if necessary. In the *Juxtapositional Phase*, the messy GA proceeds by building up the population through the use of the cut and splice recombination operator.

The Multi-Objective Messy Genetic Algorithm-II (MOMGA-II)

Zydallis et al. (2001) proposed MOMGA-II. In this case, the authors extended the fast-messy GA, which consists of three phases: (1) Initialization Phase, (2) Building Block Filtering, and (3) Juxtapositional Phase. Its main difference with respect to the original messy GA is in the two first phases. The *Initialization Phase* utilizes probabilistic complete initialization which creates a controlled number of building block clones of a specified size. The Building *Block Filtering Phase* reduces the number of building blocks through a filtering process and stores the best building blocks found. The *Juxtapositional Phase* is the same as in the MOMGA.

The Micro Genetic Algorithm for Multiobjective Optimization



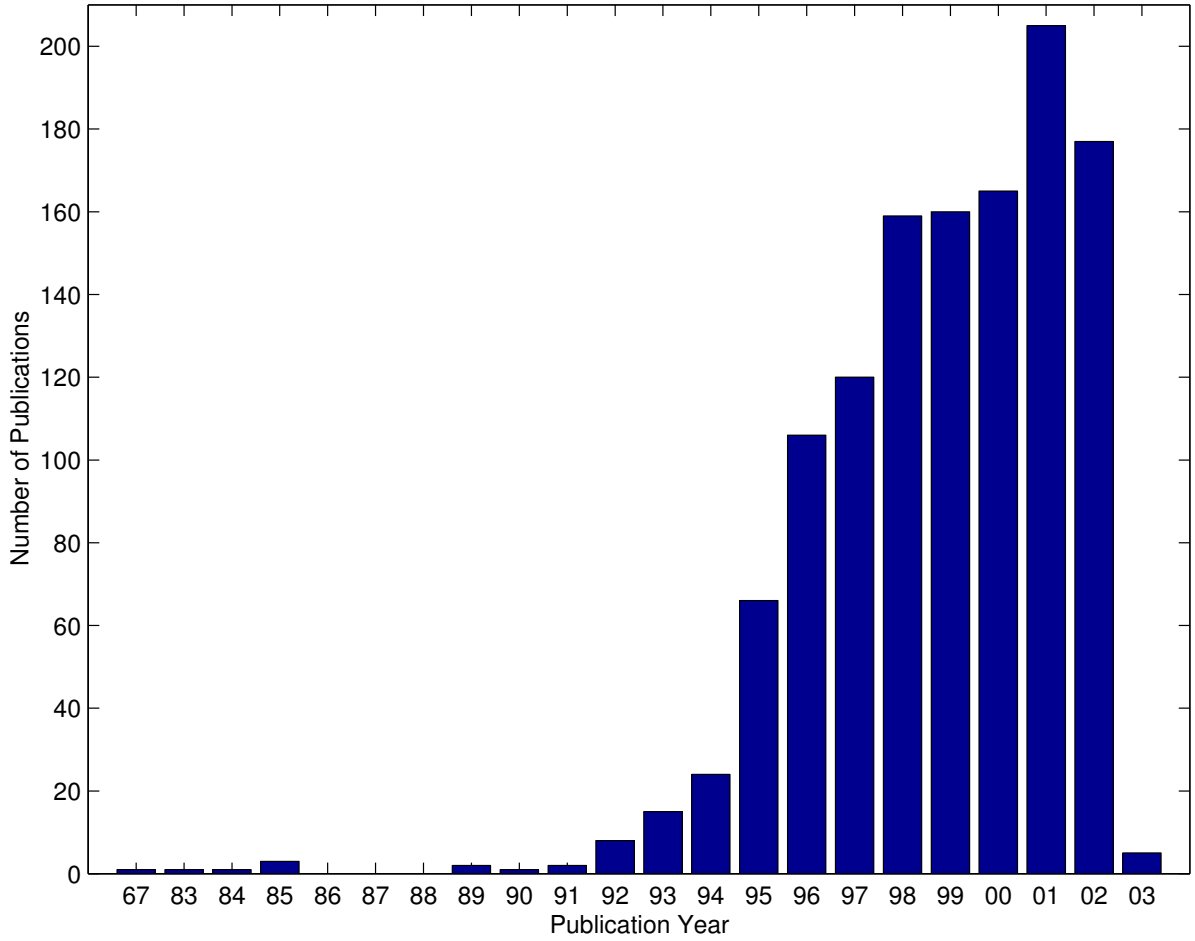
Current Trends in MOEAs

- After great success for almost 10 years, first generation MOEAs have finally started to become obsolete in the literature (NSGA, NPGA, MOGA and VEGA).
- From the late 1990s, second generation MOEAs are considered the state-of-the-art in evolutionary multiobjective optimization (e.g., SPEA, SPEA2, NSGA-II, MOMGA, MOMGA-II, PAES, PESA, PESA II, microGA, etc.).

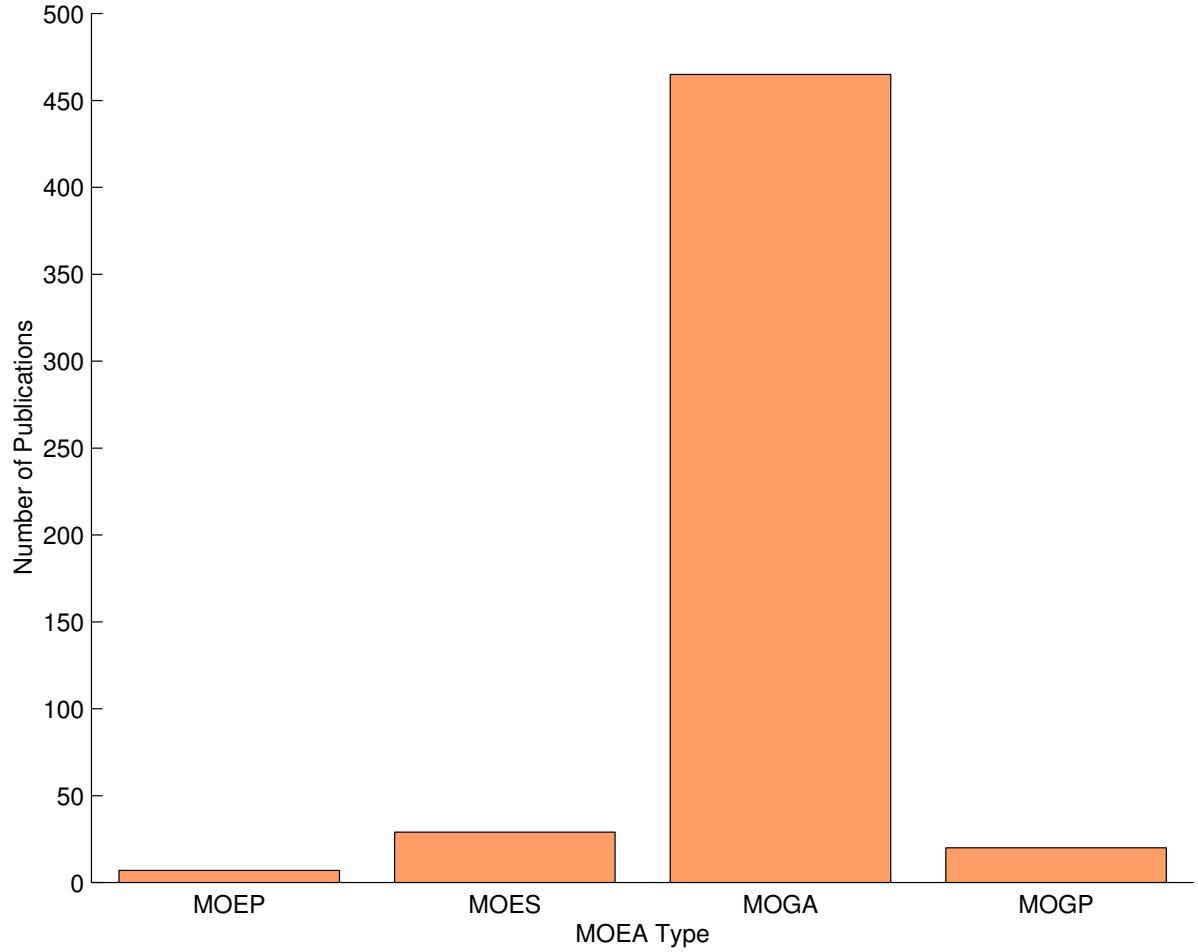
Current Trends in MOEAs

- Second generation MOEAs emphasize computational efficiency. One of the main goals is to find ways around the computational complexity of Pareto ranking ($O(kM^2)$, where k is the number of objective functions and M is the population size) and around the computational cost of niching ($O(M^2)$).
- Largely ignored by a significant number of researchers, non-Pareto MOEAs are still popular in Operations Research (e.g., in multiobjective combinatorial optimization), where they have been very successful.

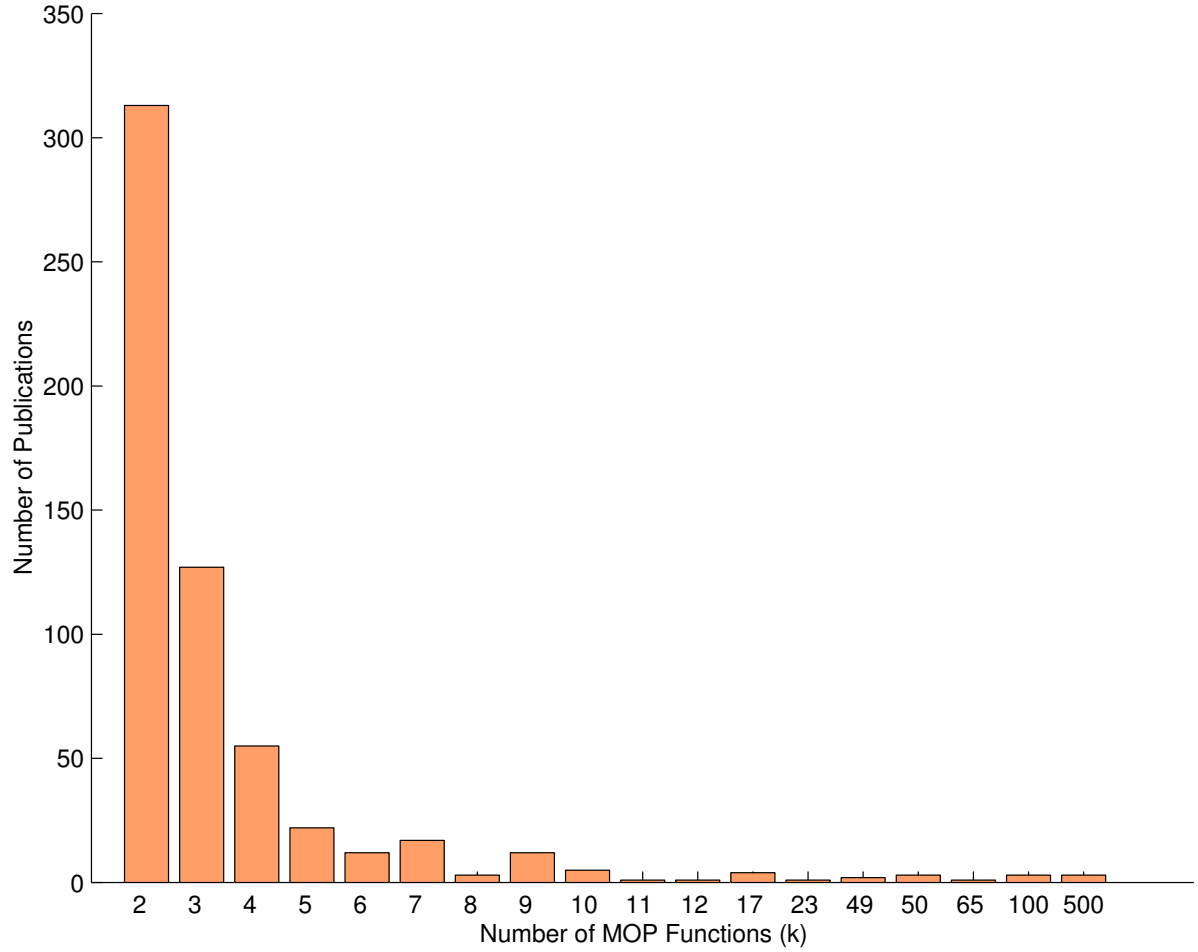
Current state of the literature (beginning of 2003)



Citations by type of MOEA (mid-2002)



Number of objective functions used (mid-2002)



Alternative Heuristics

- Simulated Annealing
- Tabu Search
- Ant System
- Particle Swarm Optimization
- Artificial Immune System
- Cultural Algorithms

Simulated Annealing

Based on an algorithm originally proposed by Metropolis et al. (1953) to simulate the evolution of a solid in a heat bath until it reaches its thermal equilibrium.

Kirkpatrick et al. (1983) and Černy (1985) independently pointed out the analogy between the "annealing" process proposed by Metropolis and combinatorial optimization and proposed the so-called "simulated annealing algorithm".

Simulated Annealing

1. Select an initial (feasible) solution s_0
2. Select an initial temperature $t_0 > 0$
3. Select a cooling schedule CS
4. Repeat

   Repeat

   Randomly select $s \in N(s_0)$ (N = neighborhood structure)

   $\delta = f(s) - f(s_0)$ (f = objective function)

   If $\delta < 0$ then $s_0 \leftarrow s$

   Else

   Generate random x (uniform distribution in the range $(0, 1)$)

   If $x < \exp(-\delta/t)$ then $s_0 \leftarrow s$

   Until max. number of iterations $ITER$ reached

   $t \leftarrow CS(t)$
5. Until stopping condition is met

Simulated Annealing

SA generates local movements in the neighborhood of the current state, and accepts a new state based on a function depending on the current "temperature" t. The two main parameters of the algorithm are $ITER$ (the number of iterations to apply the algorithm) and CS (the cooling schedule), since they have the most serious impact on the algorithm's performance.

Despite the fact that it was originally intended for combinatorial optimization, other variations of simulated annealing have been proposed to deal with continuous search spaces.

Simulated Annealing

The key in extending simulated annealing to handle multiple objectives lies in determining how to compute the probability of accepting an individual $\vec{x}'$ where $f(\vec{x}')$ is dominated with respect to $f(\vec{x})$.

Simulated Annealing

Some multiobjective versions of SA are the following:

- Serafini (1994): Uses a target-vector approach to solve a bi-objective optimization problem (several possible transition rules are proposed).
- Ulungu (1993): Uses an L_∞-Tchebycheff norm and a weighted sum for the acceptance probability.
- Czyzak & Jaszkiewicz (1997,1998): Population-based approach that also uses a weighted sum.
- Ruiz-Torres et al. (1997): Uses Pareto dominance as the selection criterion.
- Suppapitnarm et al. (1999,2000): Uses Pareto dominance plus a secondary population.

Some Applications of Multiobjective Simulated Annealing

- Design of a cellular manufacturing system (Czyzak, 1997).
- Nurse scheduling (Jaszkiewicz, 1997).
- Portfolio optimization (Chang, 1998).
- Aircrew rostering (Lučić & Teodorović, 1999).
- Ship design (Ray, 1995).
- Optimization of bicycle frames (Suppapitnarm, 1999).
- Parallel machine scheduling (Ruiz-Torres, 1997)
- Analog Filter Tuning (Thompson, 2001)

Tabu Search

In general terms, tabu search has the three following components (Glover & Laguna, 1997):

- A short-term memory to avoid cycling.
- An intermediate-term memory to intensify the search.
- A long-term memory to diversify the search.

Tabu Search

1. Select $x \in \mathcal{F}$ ($\mathcal{F}$ represents feasible solutions)
2. $x^* = x$ (x^* is the best solution found so far)
3. $c = 0$ (iteration counter)
4. $T = \emptyset$ (T set of "tabu" movements)
5. If $\mathcal{N}(x) - T = \emptyset$, goto step 4 ($\mathcal{N}(x)$ is the neighborhood function)
6. Otherwise, $c \leftarrow c + 1$

   Select $n_c \in \mathcal{N}(x) - T$ such that: $n_c(x) = opt(n(x) : n \in \mathcal{N}(x) - T)$

   $opt()$ is an evaluation function defined by the user
7. $x \leftarrow n_c(x)$

   If $f(x) < f(x^*)$ then $x^* \leftarrow x$
8. Check stopping conditions:

   Maximum number of iterations has been reached

   $\mathcal{N}(x) - T = \emptyset$ after reaching this

   step directly from step 2.
9. If stopping conditions are not met, update T

   and return to step 2

Tabu Search

The basic idea of tabu search is to create a subset T of $\mathcal{N}$, whose elements are called "tabu moves" (historical information of the search process is used to create T). Membership in T is conferred either by a historical list of moves previously detected as improductive, or by a set of tabu conditions (e.g., constraints that need to be satisfied). Therefore, the subset T constrains the search and keeps tabu search from becoming a simple hillclimber. At each step of the algorithm, a "best" movement (defined in terms of the evaluation function $opt()$) is chosen. Note that this approach is more aggressive than the gradual descent of simulated annealing.

Tabu Search

Tabu search tends to generate moves that are in the area surrounding a candidate solution. Therefore, the main problem when extending this technique to deal with multiple objectives is how to maintain diversity so that the entire Pareto front can be generated. The proper use of the historial information stored is another issue that deserves attention.

Tabu Search

Some multiobjective versions of tabu search are the following:

- Hansen (1997): MOTS*, which uses a λ-weighted Tchebycheff metric.
- Gandibleux et al. (1997): MOTS, which is based on the use of an utopian reference point.
- Hertz et al. (1994): Proposed 3 approaches (weighted sum of objectives, lexicographic ordering and the ε-constraint method).

Some Applications of Multiobjective Tabu Search

- Resource constrained project scheduling (Viana and Pinho de Sousa, 2000).
- Flowshop scheduling (Marett and Wright, 1996).
- Cell formation problems (Hertz et al., 1994).

Ant System

The Ant System (AS) is a meta-heuristic inspired by colonies of real ants, which deposit a chemical substance on the ground called *pheromone* (Dorigo, 1999). This substance influences the behavior of the ants: they tend to take those paths where there is a larger amount of pheromone. Pheromone trails can thus be seen as an indirect communication mechanism among ants. From a computer science perspective, the AS is a multi-agent system where low level interactions between single agents (i.e., artificial ants) result in a complex behavior of the entire ant colony.

Ant System

The AS was originally proposed for the traveling salesman problem (TSP), and most of the current applications of the algorithm require the problem to be reformulated as one in which the goal is to find the optimal path of a graph. A way to measure the distances between nodes is also required in order to apply the algorithm.

Ant-Q

Gambardella and Dorigo (1995) realized the AS can be interpreted as a particular kind of distributed learning technique and proposed a family of algorithms called Ant-Q. This family of algorithms is really a hybrid between Q-learning and the AS. The algorithm is basically a reinforcement learning approach with some aspects incrementing its exploratory capabilities.

Ant System and Ant-Q

Some multiobjective versions of AS and Ant-Q are the following:

- Mariano and Morales (1999): proposed Multi-Objective Ant-Q (MOAQ), which uses lexicographic ordering.
- Gambardella et al. (1999): proposed the use of two ant colonies (one for each objective), and applied lexicographic ordering.
- Iredi et al. (2001): proposed a multi colony approach to handle the two objectives of a single machine total tardiness problem.
- Gagné et al. (2001): proposed an approach in which the heuristic values used to decide the movements of an ant take into consideration several objectives.

Some Applications of Multiobjective Ant System or Ant-Q

- Optimization of a water distribution irrigation network (Mariano and Morales, 1999).
- Vehicle routing problems (Gambardella et al., 1999).
- Single machine total tardiness problem (Iredi et al., 2001).
- Industrial scheduling (Gravel et al. 2001).
- Reliability optimization (Shelokar et al., 2002).

Particle Swarm Optimization

Kennedy and Eberhart (1995) proposed an approach called "particle swarm optimization" (PSO) inspired by the choreography of a bird flock. The idea of this approach is to simulate the movements of a group (or population) of birds which aim to find food. The approach can be seen as a distributed behavioral algorithm that performs (in its more general version) multidimensional search. In the simulation, the behavior of each individual is affected by either the best local (i.e., within a certain neighborhood) or the best global individual.

Particle Swarm Optimization

It is worth mentioning that PSO is an unconstrained search technique. Therefore, it is also necessary to develop an additional mechanism to deal with constrained multiobjective optimization problems. The design of such a mechanism is also a matter of current research even in single-objective optimization (see for example (Ray, 2001)).

Particle Swarm Optimization

1. For i = 1 to M (M = population size)
   Initialize $P[i]$ randomly
   (P is the population of particles)
   Initialize $V[i] = 0$ (V = speed of each particle)
   Evaluate $P[i]$
   $GBEST$ = Best particle found in $P[i]$
2. End For
3. For i = 1 to M
   $PBESTS[i] = P[i]$
   (Initialize the "memory" of each particle)
4. End For

Particle Swarm Optimization

5. Repeat

   For i = 1 to M

   $V[i] = w \times V[i] + C_1 \times R_1 \times (PBESTS[i] - P[i])$

   $+C_2 \times R_2 \times (PBESTS[GBEST] - P[i])$

   (Calculate speed of each particle)

   (W = Inertia weight, C_1 & C_2 are positive constants)

   (R_1 & R_2 are random numbers in the range [0..1])

   $POP[i] = P[i] + V[i]$

   If a particle gets outside the pre-defined hypercube

   then it is reintegrated to its boundaries

   Evaluate $P[i]$

   If new position is better then $PBESTS[i] = P[i]$

   $GBEST$ = Best particle found in $P[i]$

   End For

6. Until stopping condition is reached

Particle Swarm Optimization

To extend PSO for multiobjective optimization, it is necessary to modify the guidance mechanism of the algorithm such that nondominated solutions are considered as leaders. Note however, that it's important to have a diversity maintenance mechanism. Also, an additional exploration mechanism (e.g., a mutation operator) may be necessary to generate all portions of the Pareto front (mainly in disconnected fronts).

Particle Swarm Optimization

Some multiobjective versions of particle swarm optimization are the following:

- Moore & Chapman (1999): Based on Pareto dominance. The authors emphasize the importance of performing both an individual and a group search (a cognitive component and a social component). No scheme to maintain diversity is adopted.
- Ray & Liew (2002): Uses Pareto dominance and combines concepts of evolutionary techniques with the particle swarm. The approach uses crowding to maintain diversity and a multilevel sieve to handle constraints.

Particle Swarm Optimization

- Parsopoulos & Vrahatis (2002): Uses an aggregating function (three types of approaches were implemented: a conventional linear aggregating function, a dynamic aggregating function and the bang-bang weighted aggregation approach (Jin, 2001) in which the weights are varied in such a way that concave portions of the Pareto front can be generated).
- Hu & Eberhart (2002): Only one objective is optimized at a time using a scheme similar to lexicographic ordering. Note that lexicographic ordering tends to be useful only when few objective functions are used (two or three), and it may be sensitive to the ordering of the objectives.

Particle Swarm Optimization

- Coello & Lechuga (2002): Uses Pareto dominance and a secondary population to retain the nondominated vectors found along the search process. The approach is very fast and has performed well compared to other techniques considered representative of the state-of-the-art in evolutionary multiobjective optimization.

Particle Swarm Optimization

- Fieldsend & Singh (2002): Also uses Pareto dominance and a secondary population. However, in this case, a data structure called "dominated trees" is used to handle an unconstrained archive, as to avoid the truncation traditionally adopted with MOEAs. A mutation operator (called "turbulence") is also adopted.

Some Applications of Multiobjective Particle Swarm Optimization

It has been used only in test functions taken from the evolutionary multiobjective optimization literature and from the engineering optimization literature.

Artificial Immune System

Computationally speaking, the immune system is a highly parallel intelligent system that is able to learn and retrieve previous knowledge (i.e., it has "memory") to solve recognition and classification tasks. Due to these interesting features, several researchers have developed computational models of the immune system and have used it for a variety of tasks.

Artificial Immune System

There are several computational models of the immune system, from which the main ones are the following:

- Immune network theory.
- Negative selection.
- Clonal selection theory

Artificial Immune System

The main issues to extend an artificial immune system to deal with multiple objectives are how to influence the propagation of antibodies (i.e., how to couple the Pareto selection mechanism) and how to maintain diversity. The use of a secondary population may also be useful, if possible in the model adopted.

Artificial Immune System (fitness scoring)

Repeat

1. Select an antigen $\mathcal{A}$ from PA
   (PA = Population of Antigens)
2. Take (randomly) R antibodies from PS
   (PS = Population of Antibodies)
3. For each antibody $r \in R$, match it against
   the selected antigen $\mathcal{A}$
   Compute its match score (e.g., using Hamming distance)
4. Find the antibody with the highest match score
   Break ties at random
5. Add match score of winning antibody to its fitness

Until maximum number of cycles is reached

Artificial Immune System

There have been very few attempts to extend an artificial immune system for multiobjective optimization:

- Yoo & Hajela (1999): Use of a linear aggregating function combined with the fitness scoring function previously indicated.
- Kurapati & Azarm (2000): Hybridization of an artificial immune system with a multiobjective evolutionary algorithm.
- Cui et al. (2001): Another hybrid approach that uses entropy to maintain diversity.

Artificial Immune System

- Anchor et al. (2002): Adopt both lexicographic ordering and Pareto-based selection in an evolutionary programming algorithm used to detect attacks with an artificial immune system for virus and computer intrusion detection.
- Coello and Cruz (2002): Extend a clonal selection algorithm to handle multiple objectives. A secondary population is adopted.

Some Applications of Multiobjective Artificial Immune Systems

- Structural optimization (Yoo & Hajela, 1999).
- Computer security (Anchor et al., 2002).
- Multidisciplinary design optimization (Kurapati & Azarm, 2000).

Cultural Algorithms

Some social researchers have suggested that culture might be symbolically encoded and transmitted within and between populations, as another inheritance mechanism. Using this idea, Reynolds (1994) developed a computational model in which cultural evolution is seen as an inheritance process that operates at two levels: the micro-evolutionary and the macro-evolutionary levels.

Cultural Algorithms

At the micro-evolutionary level, individuals are described in terms of "behavioral traits" (which can be socially acceptable or unacceptable). These behavioral traits are passed from generation to generation using several socially motivated operators. At the macro-evolutionary level, individuals are able to generate "mappa", or generalized descriptions of their experiences. Individual mappa can be merged and modified to form "group mappa" using a set of generic or problem specific operators. Both levels share a communication link.

Cultural Algorithms (pseudo-code)

1. $t = 0$ (t = iteration counter)
2. Initialize $POP(0)$ (POP = Population)
3. Initialize $BELF(0)$ ($BELF$ = Belief Network)
4. Initialize $CHAN(0)$ ($CHAN$ = Communication Channel)
5. Evaluate $POP(0)$
6. t=1

Repeat

    Communicate ($POP(0)$, $BELF(t)$)

    Adjust ($BELF(t)$)

    Communicate ($BELF(t)$, $POP(t)$)

    Modulate Fitness ($BELF(t)$, $POP(t)$)

    $t \leftarrow$ t + 1

    Select $POP(t)$ from $POP(t-1)$

    Evolve $POP(t)$

    Evaluate $POP(t)$

Until Stopping Condition is Reached

Cultural Algorithms

It is possible to extend a cultural algorithm to multiobjective optimization problems if nondominance is incorporated in the acceptance mechanism of the approach.

The approach could work in a similar way to some proposals to extend the ant system to handle multiple objectives. In this case, an individual's cultural component could lead it to a local nondominated solution, and the global mechanism of the approach (intended for sharing group's solving experiences and behaviors) could lead the population towards global nondominated solutions.

Cultural Algorithms

The same acceptance mechanism could incorporate additional criteria to encourage a smooth distribution of nondominated solutions (e.g., make unacceptable a nondominated solution generated in a region of the search space that is already too densely populated).

Cultural Algorithms

There are only 2 known efforts to use cultural algorithms to solve multiobjective optimization problems:

1. The shell available at:

   `http://zeus.cs.wayne.edu/~sms/caep/cultural.html`

   However, in this shell a simple linear combination of weights is used to aggregate multiple objectives into a single scalar value.

2. The algorithm (based on Pareto dominance and using a secondary population) presented at this conference (Coello & Landa, 2003).

Some Applications of Multiobjective Cultural Algorithms

Only test functions and some engineering optimization problems (Coello & Landa, 2003a).

The main motivation for using cultural algorithms in multiobjective optimization should be to reduce computational cost. The use of domain knowledge extracted during the evolutionary process should allow faster convergence rates, but no proposals in that direction are currently available.

Statistics of alternative heuristics (mid-2002)

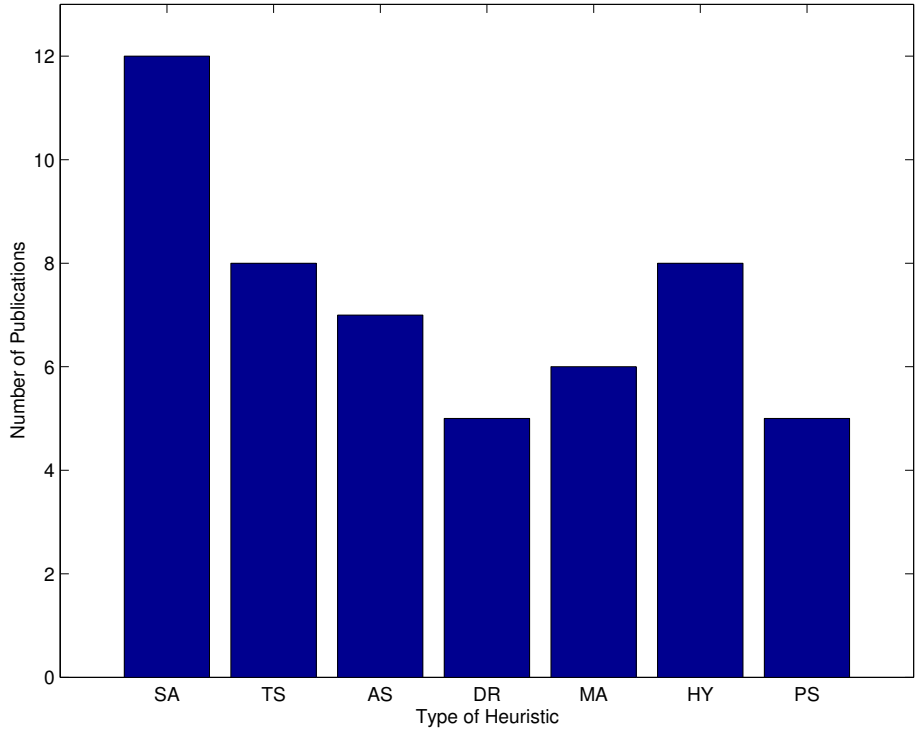


Figure 2: The following labels are used: SA = Simulated Annealing, TS = Tabu Search, AS = Ant System, DR = Distributed Reinforcement Learning, MA = Memetic Algorithm, HY = Hybrid techniques, PS = Particle Swarm Optimization.

Promising areas of future research

- Incorporation of user's preferences.
- Emphasis on efficiency (alternative data structures and clever algorithms that minimize Pareto dominance checkings).
- More test functions and metrics.

Promising areas of future research

- More theoretical studies (convergence, mathematical models, etc.).
- New approaches (hybrids with other heuristics) and extensions of alternative heuristics (e.g., scatter search, cultural algorithms, reinforcement learning, etc.).
- New applications.
- What to expect for the third generation?

Promising areas of future research

- Tackling dynamic (multiobjective) test functions, handling uncertainty and high epistasis.
- Answering fundamental questions such as: what makes difficult a multiobjective optimization problem for an EA? Can we really produce reliable metrics for multiobjective optimization? Can we design robust MOEAs? Is there a way around the dimensionality curse in multiobjective optimization? Can we benefit from coevolutionary schemes?

To know more about evolutionary multiobjective optimization

Please visit our EMOO repository located at:

http://delta.cs.cinvestav.mx/~ccoello/EMOO

with mirrors at:

http://www.jeo.org/emo

and:

http://www.lania.mx/~ccoello/EMOO

To know more about evolutionary multiobjective optimization

EMOO Web page

Complete List of References in alphabetical order

Books

Book Chapters

Journal Papers

Conference Papers

Technical Reports

Masters Theses

PhD Theses

Researchers

Software

Test Functions and Problems

Events

Miscellaneous

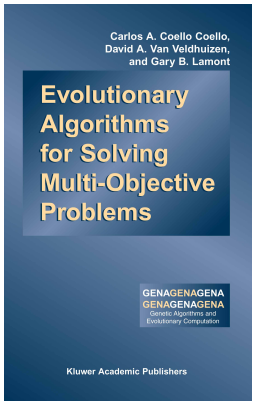
Other links of interest

To know more about evolutionary multiobjective optimization

The EMOO repository currently contains:

- Over 1230 bibliographic references including 50 PhD theses
- Contact info of 50 EMOO researchers
- Public domain implementations of SPEA, NSGA, NSGA-II, the microGA, MOPSO and PAES, among others.

To know more about evolutionary multiobjective optimization



You can consult the following book recently published:

Carlos A. Coello Coello, David A. Van Veldhuizen and Gary B. Lamont, **Evolutionary Algorithms for Solving Multi-Objective Problems**, Kluwer Academic Publishers, New York, May 2002, ISBN 0-3064-6762-3.