



Constrained Multi-Objective Optimization Using Steady state Genetic Algorithms

Deepti Chafekar

Jiang Xuan

Khaled Rasheed

The University of Georgia
Haibo Zhao, Ryan Millar

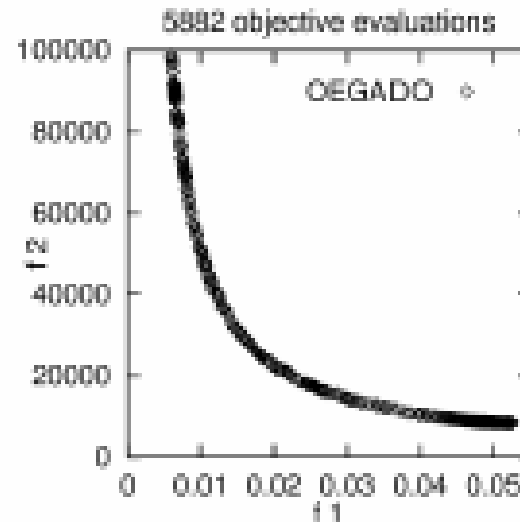


Pareto-Optimal Solutions

- For problems involving multi-objective optimization, there usually exists a set of non-dominated solutions (Pareto-optimal solutions)
- “A solution dominates another if it outperforms it in at least one objective and is not outperformed by it in any objective”

Pareto-Optimal Solutions

- The best Pareto front (the set of Pareto-optimal solutions) is one that is dense, accurate and evenly spread
- Good Example:





Steady State GAs

- For problems with a complex search space, many constraints, a very small feasible search space, and an expensive fitness function, steady state GAs perform better than Generational GAs
- The reason being that steady state GAs retain feasible points and have a higher selection pressure



Some Multi-Objective GA Implementations:

- Vector Evaluated Genetic Algorithm (VEGA)
- Non-Dominated Sorting Genetic Algorithm-II (NSGA-II)
- Strength Pareto Evolutionary Algorithm-II (SPEA-II)
- Pareto Envelope based Selection-II (PESA-II)



Two Methods

- Objective Exchange Genetic Algorithm for Design Optimization (OEGADO)
- Objective Switching Genetic Algorithm for Design Optimization (OSGADO)
- Both methods are multi-objective transformations of GADO (Genetic Algorithm for Design Optimization)



GADO: A Genetic Algorithm for Design Optimization

- Maintains a population of possible solutions
- Better individuals are created by crossover between two members of the population and a possible mutation of the new member created
- Fitness is based on the sum of a measure of merit calculated by a simulator or an analysis code and possibly a penalty function

The logo consists of a vertical black line on the left, a horizontal black line at the bottom, and three overlapping squares: a yellow one at the top left, a red one at the middle left, and a blue one at the bottom left. The word "GADO" is written in a blue, sans-serif font to the right of the vertical line.

GADO

- Diversity Maintenance Module
 - Monitors degree of diversity of population
 - If individuals become too similar, the module rebuilds the population with previously evaluated points to restore diversity
 - It also rejects any proposed points that are too similar to previously evaluated points



OEGADO

- Main Idea:
 - Run several single objective GAs concurrently
 - Each GA optimizes one objective
 - All GAs share the same representation and constraints, but have different populations
 - They exchange objectives after a certain number of iterations



OEGADO

- IOs are used to generate a population using approximations of the fitness function to replace a pure random population
- Least squares are used in each GA to form a reduced model of its own objectives
- The GAs exchange their reduced models with each other making each GA informed about the other GAs' objectives



The OEGADO Algorithm

1. Two GAs optimize separate objectives for the same number of iterations while forming reduced models of them
2. At intervals equal to twice the population size, the GAs exchange their reduced models
3. Initialization, mutation, and crossover are replaced by informed operators



The OEGADO Algorithm

4. The true fitness function is called
5. The individual is added to the population using the replacement strategy
6. Steps 2 through 5 are repeated until the maximum number of evaluations is exhausted



OEGADO

- A potential advantage to this method is speed.
- Each GA can run in parallel on different CPUs.



OSGADO Main Idea

- OSGADO

Objective Switching Genetic Algorithm for Design Optimization

- Main Idea

A single GA runs multiple objectives in a sequence switching objectives at certain intervals. Every objective is optimized for a certain number of evaluations, then a switch occurs and the next objective is optimized.



OSGADO Algorithm

- **Step 1**

GA is run initially with the first objective as the measure of merit for a certain number of evaluations. selection, crossover and mutation take place in the regular manner

- **Step 2**

After a certain number of evaluations, GA is run for the next objective. When the evaluation for the last objective are complete, GA switches back to the first objective

- **Step 3**

Repeat step 2 until the maximum number of evaluation is reached.



Experiments

- **Test Problems (4+2)**

4 problems are chosen from the benchmark domains commonly used in past multi-objective GA research;
2 problems are chosen from Engineering domain

- **Parameter Settings**

Trying to construct a fair competition environment;
Each optimization run was carried out with similar parameter settings for all methods

- **Results**

Run same problems using different approaches and compare the results

Experiments Test Problems

Problem	Variable bounds	Objectives functions $f(x)$ and Constraints $C(x)$
BNH	$x_1 \in [0,5]$ $x_2 \in [0,3]$	$f_1(x) = 4x_1^2 + 4x_2^2$ $f_2(x) = (x_1 - 5)^2 + (x_2 - 5)^2$ $C_1(x) \equiv (x_1 - 5)^2 + x_2^2 \leq 25$ $C_2(x) \equiv (x_1 - 8)^2 + (x_2 + 3)^2 \geq 7.7$
SRN	$x_1 \in [-20,20]$ $x_2 \in [-20,20]$	$f_1(x) = 2 + (x_1 - 2)^2 + (x_2 - 2)^2$ $f_2(x) = 9x_1 - (x_2 - 1)^2$ $C_1(x) \equiv x_1^2 + x_2^2 \leq 225$ $C_2(x) \equiv x_1 - 3x_2 + 10 \leq 0$
TNK	$x_1 \in [0, \pi]$ $x_2 \in [0, \pi]$	$f_1(x) = x_1$ $f_2(x) = x_2$ $C_1(x) \equiv x_1^2 + x_2^2 - 1 - 0.1 \cos(16 \arctan \frac{x_1}{x_2}) \geq 0$ $C_2(x) \equiv (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.5$
OSY	$x_1 \in [0,10]$ $x_2 \in [0,10]$ $x_3 \in [1,5]$ $x_4 \in [0,6]$ $x_5 \in [1,5]$ $x_6 \in [0,10]$	$f_1(x) = -[25(x_1 - 2)^2 + (x_2 - 2)^2 + (x_3 - 1)^2 + (x_4 - 4)^2 + (x_5 - 1)^2]$ $f_2(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2$ $C_1(x) \equiv x_1 + x_2 - 2 \geq 0$ $C_2(x) \equiv 6 - x_1 - x_2 \geq 0$ $C_3(x) \equiv 2 - x_2 + x_1 \geq 0$ $C_4(x) \equiv 2 - x_1 + 3x_2 \geq 0$ $C_5(x) \equiv 4 - (x_3 - 3)^2 - x_4 \geq 0$ $C_6(x) \equiv (x_5 - 3)^2 + x_6 - 4 \geq 0$

Problem	Variable bounds	Objectives functions $f(x)$ and Constraints $C(x)$
Two-bar Truss Design	$x_1 \in [0,0.01]$ $x_2 \in [0,0.01]$ $x_3 \in [1,3]$	$f_1(x) = x_1 \sqrt{16 + x_3^2} + x_2 \sqrt{1 + x_3^2}$ $f_2(x) = \max(\sigma_1, \sigma_2)$ $C_1(x) \equiv \max(\sigma_1, \sigma_2) \leq 10^5$ $\sigma_1 = 20 \sqrt{16 + x_3^2} / x_1 x_3$ $\sigma_2 = 80 \sqrt{1 + x_3^2} / x_2 x_3$
Welded Beam Design	$h \in [0.125,5]$ $b \in [0.125,5]$ $l \in [0.1,10]$ $t \in [0.1,10]$	$f_1(x) = 1.10471 h^2 l + 0.04811 t b (14 + l)$ $f_2(x) = 2.1952 / t^3 b$ $C_1(x) \equiv 13600 - \tau(x) \geq 0$ $C_2(x) \equiv 30000 - \sigma(x) \geq 0$ $C_3(x) \equiv b - h \geq 0$ $C_4(x) \equiv P_c(x) - 6000 \geq 0$ $\tau = \sqrt{(\tau')^2 + (\tau'')^2} + l \tau' \tau'' / \sqrt{0.25(l^2 + (h+t)^2)}$ $\tau' = 6000 / \sqrt{2} h l$ $\tau'' = \frac{6000(14 + 0.5l) \sqrt{0.25(l^2 + (h+t)^2)}}{2\sqrt{2} h l (l^2 / 12 + 0.25(h+t)^2)}$ $\sigma = 504000 / t^2 b$ $P_c = 64746.022(1 - 0.0282346 t) t b^3$



Experiments Parameter Settings

- Let $ndim$ be equal to the number of dimensions of the problems
 - **Population Size**
10* $ndim$ for OEGADO and OSGADO
100 for NSGA-II as recommended
 - **Number of Objective Evaluations**
OEGADO and OSGADO: 2*500* $ndim$ + sum of feasible points found by each GA in OEGADO model
NSGA-II: 2*population size*number of generations
- Set the number of generations of NSGA-II to be 10* $ndim$

Experiments Results (1)

- BNH & SRN are fairly simple and all three methods perform equally well
- TNK

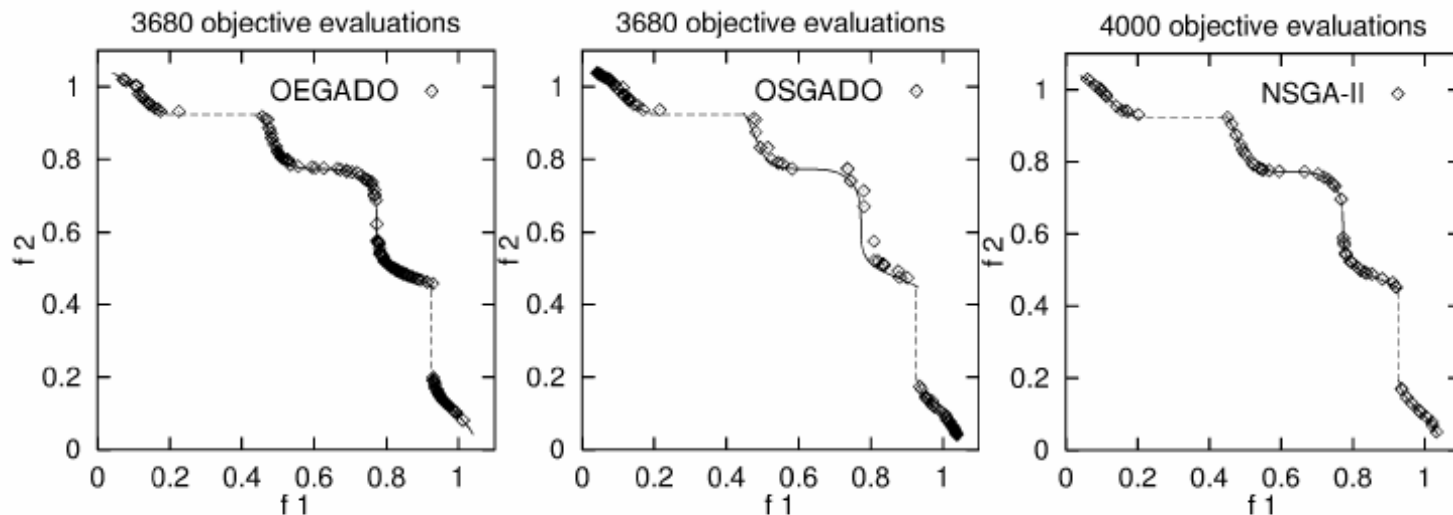


Fig. 1 Results for the benchmark problem TNK

Experiments Results (2)

- OSY

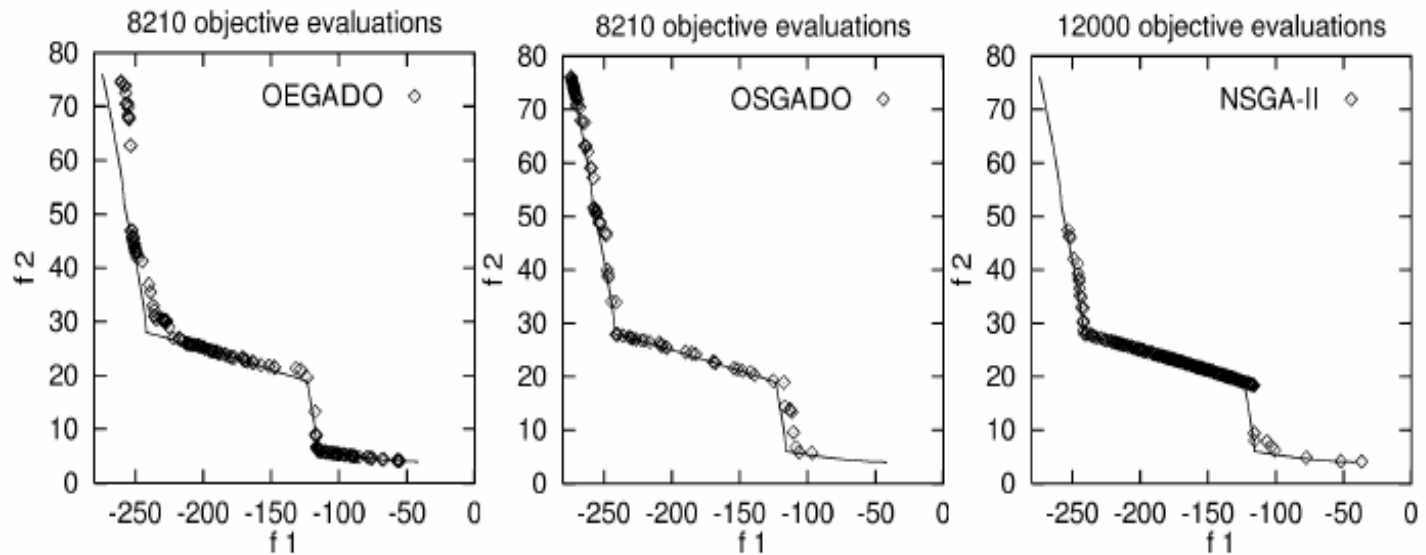


Fig. 2 Results for the benchmark problem OSY

Experiments Results (3)

- Two-bar Truss Design Problem

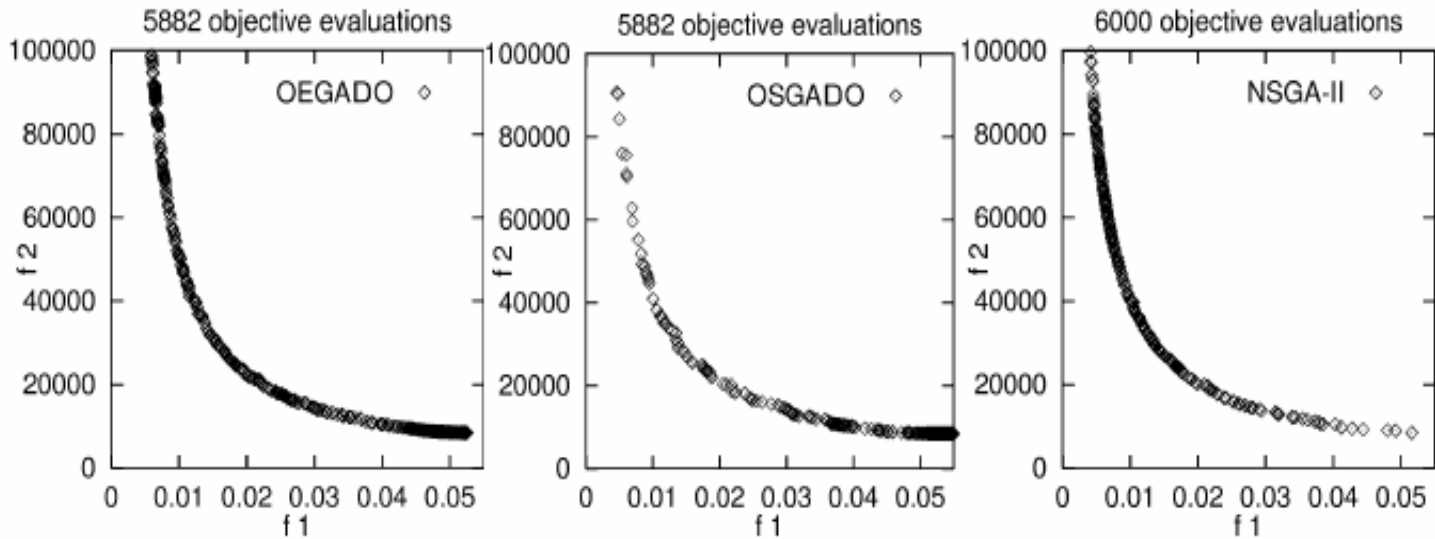


Fig. 3 Results for the Two-bar Truss design problem

Experiments Results (4)

- Welded Beam Design Problem

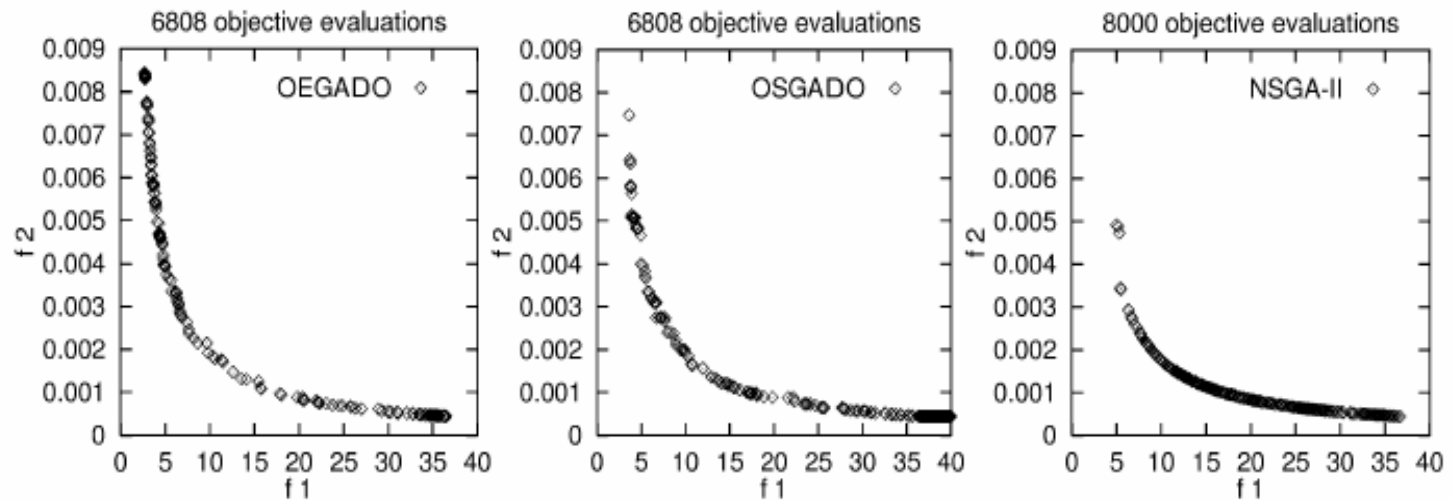


Fig. 4 Results for the Welded Beam design problem



In Our Opinion

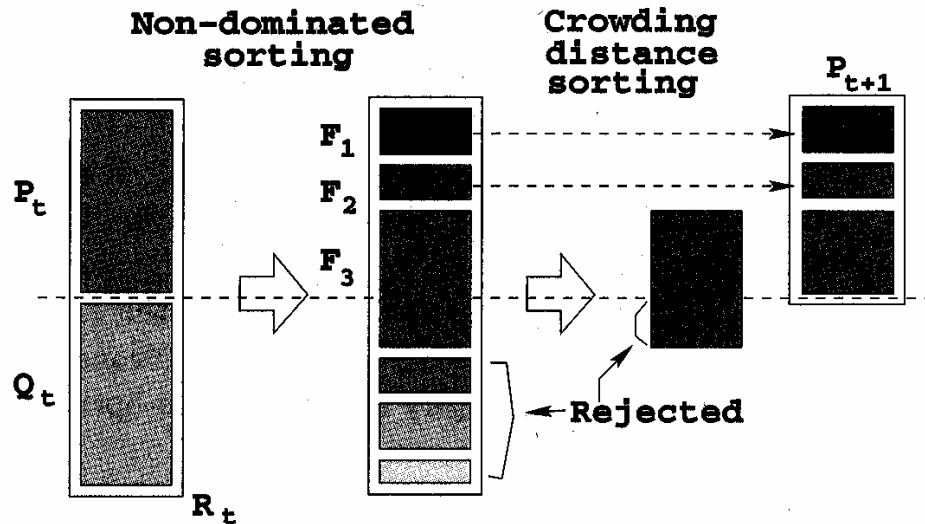
- The experiment results are very convincing, OEGADO and OSGADO beat NSGA-II successfully.
- But I guess we could not say “Objective Exchange” and “Objective Switching” definitely beat “Non-dominated Sorting” strategy based on these experiment results
- I think it will be interesting to apply “Non-dominated Sorting” strategy into GADO and run the experiments similarly



GAME OVER

P.S. Non-dominated Sorting

- Classifying the population into different fronts based on non dominated sorting ranks.





Details (1)

- In NSGA-II, the offspring population Q_t is first created by using the parent population P_t . However, instead of finding the non dominated front of Q_t only, first the two populations are combined together to form R_t of size $2N$. Then a non – dominated sorting is used to classify the entire population R_t .



Details (2)

- Once the non-dominated sorting is over, the new population is filled by solutions of different non-dominated fronts, one at a time. The filling starts with the best non-dominated front and continues with solutions of the second non-dominated front, followed by the third non-dominated front and so on. Since the overall population size of R_t is $2N$, not all fronts may be accommodated in N slots available in the new population. All fronts which could not be accommodated are simply deleted. When the last allowed front is being considered, there may exist more solutions in the last front than the remaining slots in the new population. Instead of arbitrarily discarding some members of the last front, it would be wise to use a niching strategy to choose the members of the last front, which reside in the least crowded region in that front.