Accelerating Convergence Towards the Optimal Pareto Front

Mohsen Davarynejad*, Jafar Rezaei*, Jos Vrancken*, Jan van den Berg* and Carlos A. Coello Coello†
*Faculty of Technology, Policy and Management,
Delft University of Technology, the Netherlands
Email: {M.Davarynejad; J.Rezaei; J.L.M.Vrancken; J.vandenBerg}@tudelft.nl
†CINVESTAV-IPN, Departmento de Computación (Evolutionary Computation Group)
Av. IPN No. 2508, Col. San Pedro Zacatenco, México D.F. 07300, México
Email: ccoello@cs.cinvestav.mx

***Abstract*—Evolutionary algorithms have been very popular optimization methods for a wide variety of applications. However, in spite of their advantages, their computational cost is still a prohibitive factor in certain real-world applications involving expensive (computationally speaking) fitness function evaluations. In this paper, we depart from the observation that nature's survival of the fittest is not about exact measures of fitness; rather it is about rankings among competing peers. Thus, by exploiting this natural tolerance for imprecision, we propose here a new, fuzzy granules-based approach for reducing the number of necessary function calls involving time consuming real-world problems. Our proposed approach is compared with respect to the standard NSGA-II, using the *Set Coverage*, *Hypervolume* and *Generational Distance* performance measures. Our results indicate that our proposed approach is a very promising alternative for dealing with multi-objective optimization problems involving expensive fitness function evaluations.**

I. Introduction

Optimization using metaheuristics has become a very popular research topic in the last few years. Real-world problems, however, frequently have two or more (possibly conflicting) objectives that we aimed to optimize at the same time. Such problems are called *multi-objective* and have been intensively studied using metaheuristics (particularly, evolutionary algorithms) in the last few years [2].

As opposed to single-objective optimization problems in which we aim to find a single optimum solution, in multi-objective optimization problems (MOPs) the notion of *optimality* changes, since there is normally no single solution that is the best for all the criteria. The aim in this case is then to find a set of solutions for which no objective can be improved without worsening another. This set of solutions is known as the *Pareto optimal set* and their vectors are said to be non-dominated. When plotted in objective function space, these solutions are collectively known as the *Pareto front*.

A wide variety of multi-objective evolutionary algorithms (MOEAs) have been proposed since the inception of this field in the mid-1980s [7], [2]. However, MOEAs are known to be computationally expensive, since they normally require a high number of objective function evaluations in order to produce a reasonably good approximation of the Pareto front of the problem being solved. Nevertheless, relatively little research has been reported so far on the development of techniques that reduce the computational cost of MOEAs (see [25]). This paper seeks to contribute to this area by introducing a fuzzy granules-based approach for reducing the number of objective function evaluations required by a MOEA.

The remainder of this paper is organized as follows. Section II provides some basic multi-objective optimization concepts. The previous related work is discussed in Section III. Section IV presents the approach proposed in this paper. To illustrate the efficiency of the proposed method, the performance results on ZDT1-6 test problem is presented in Section V. The final section draws conclusions and considers implications for future research.

II. Basic Concepts

We are interested in solving problems of the type[1]:

$$\text{minimize } \vec{f}(\vec{x}) := [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_n(\vec{x})] \tag{1}$$

subject to:

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

$$h_j(\vec{x}) = 0 \quad j = 1, 2, \ldots, p \tag{3}$$

where $\vec{x} = [x_1, x_2, \ldots, x_m]^T$ is a vector of decision variables, $f_i : \mathbb{R}^m \rightarrow \mathbb{R}$, $i = 1, ..., n$ are the objective functions and $g_i, h_j : \mathbb{R}^m \rightarrow \mathbb{R}$, $i = 1, ..., q$, $j = 1, ..., p$ are the constraints of the problem.

To describe the concept of optimality, a few definitions are introduced.

Definition 1. Given two vectors $\vec{x}, \vec{y} \in \mathbb{R}^m$, $\vec{x}$ **dominates** $\vec{y}$ (denoted by $\vec{x} \prec \vec{y}$) if $f_i(\vec{x}) \leq f_i(\vec{y})$ for $i = 1, \ldots, n$, and that $\vec{x} \neq \vec{y}$.

Definition 2. A vector of decision variables $\vec{x} \in \mathcal{X} \subset \mathbb{R}^m$ is **nondominated** with respect to $\mathcal{X}$, if there does not exist another $\vec{x}' \in \mathcal{X}$ such that $\vec{x}' \prec \vec{x}$.

Definition 3. A vector of decision variables $\vec{x}^* \in \mathcal{F} \subset \mathbb{R}^m$ ($\mathcal{F}$ is the feasible region) is **Pareto-optimal** if it is nondominated

[1]Without loss of generality, we will assume only minimization problems.

with respect to $\mathcal{F}$.

Definition 4. The **Pareto Optimal Set** $\mathcal{P}^*$ is defined by:

$$\mathcal{P}^* = \{\vec{x} \in \mathcal{F} | \vec{x} \text{ is Pareto-optimal}\}$$

Definition 5. The **Pareto Front** $\mathcal{PF}^*$ is defined by:

$$\mathcal{PF}^* = \{\vec{f}(\vec{x}) \in \mathbb{R}^n | \vec{x} \in \mathcal{P}^*\}$$

The problem is to find the Pareto optimal set from the set $\mathcal{F}$ of all the decision variable vectors that satisfy (2) and (3). Note however that in practice, not all the Pareto optimal set is normally desirable (e.g., it may not be desirable to have different solutions that map to the same values in objective function space) or achievable.

III. Previous Related Work

Evolutionary algorithms usually require such a large number of function calls that this frequently makes them computationally prohibitive in some real-world applications. When dealing with MOPs, this issue becomes more critical, because more objectives are involved and this multiplies the computational cost, while also making the search more difficult. For dealing with expensive objective functions, it is relatively common to rely on approximate models that allow us to simplify the representation of real-world complex behaviors. [2] Approximation techniques may estimate each of the individuals' fitness value on the basis of previously observed objective function values of *neighboring* individuals. A wide range of approximation and meta-model techniques have been adopted in combination with evolutionary algorithms, including Kriging [24], artificial neural networks [26], and radial-basis-function networks [18]. Other authors have adopted fitness inheritance [21], cultural algorithms [15] and other fitness function approximation techniques [14] for the same purpose. Next, we will briefly review the most representative work on the use of mechanisms for handling expensive objective functions with MOEAs reported in specialized literature.

Fitness inheritance, a popular class of fitness approximation method, was originally introduced by Smith et al. [27] and is a very simple technique that works as follows: when assigning fitness to an individual, some times the objective function is evaluated as usual, but the rest of the time, the fitness assigned to the individual is the average (or a weighted average) of the fitness of its parents. This fitness assignment scheme operates based on the assumption of similarity between an offspring and its parents. Clearly, fitness inheritance cannot be applied all the time, since some true fitness function values are required in order to obtain enough information to guide the search. This approach uses a parameter called *inheritance proportion*, which regulates how many times the fitness has to be approximated. Very few authors have reported the use of fitness inheritance in MOPs. Ducheyne et al. [11] tested the performance of both average and weighted average fitness inheritance approaches and concluded that the usefulness of this technique was limited to cases in which the Pareto front is convex and continuous. Ducheyne et al. [10] also concluded that for non-convex Pareto fronts, fitness inheritance produces a slower convergence to the true Pareto front than when the approach is not adopted. Other authors, however, have successfully applied fitness inheritance to more complicated test problems having non-convex Pareto fronts (see [21]).

Another approach for dealing with expensive objective functions is based on learning and interpolation from representative small datasets of the *true* objective functions values in the desired design space which is known as *functional approximation* [14]. Function approximation methods provide a mapping between design space and objective functions space that may be multi-dimensional. The accuracy of these models depends greatly on the number of sample data points used and their location in the multi-dimensional space. Some examples of this sort of approach are the following: the response surface methodology that uses low-order polynomials and the least square estimations [17], [13], [12] and Gaussian processes (also known as Kriging) that build probability models by exploiting information recorded and use them to estimate the function values of new candidate solutions [3].

Artificial Neural networks (ANNs) can also be used for dealing with expensive objective functions. In fact, ANNs can be considered one of the best approaches to approximate a generic $\Re^m \Rightarrow \Re^n$ function[3], where m and n represent the number of decision variables and number of objectives, respectively. Although nonlinear interpolation can be used, it is shown that with a number of decision variables higher than 10, the interpolation problem becomes almost not tractable [20]. ANNs are successfully used for building approximate models in a number of complex multiplicative optimization problems. As an example, in [1], a generic supersonic aircraft configuration with two main goals (maximization of the total range of the aircraft and minimization of the ground sonic boom) and a number of buildability and mission constraints (such as structural integrity of the aircraft, take-off and landing field length) is optimized using ANNs to generate inexpensive surrogates. The approximation is used only where this is warranted. Using Latin Hypercube Sampling (LHS), 300 sample data were generate via CFD (Computational Fluid Dynamics) simulation are fitted using a single hidden layer perceptron with sigmoid activation functions to provide a general nonlinear higher fidelity model. In another study, Poloni et al. [20] used a combination of GAs and ANNs with a modified backpropagation algorithm, and a local search method to optimize the design of a sailing yacht fin keel which is a complex design problem in fluid dynamics. The ANN acted as a model for 3D Navier-Stokes simulation of the fin keel while cruising.

For more information on approaches for dealing with ex-

[2] This is based on the assumption that approximate models require small computational resources compared to the cost of complex simulations, which is normally the case when considering real-world problems.

[3] If they are provided with sufficient structural complexity and a rich training data set.

pensive objective functions in the context of multi-objective optimization, interested readers should refer to [25].

A. Final Remarks on Fitness Approximation

In most of the fitness approximation models currently available, the main problem is the lack of sufficient training data and hence the failure to reach a model with sufficient approximation accuracy. Since the evaluation of the original fitness function, in many practical problems, is obtained by some sort of analysis (i.e., fluid mechanics analysis, thermodynamic analysis) that is computationally expensive, the approximate model may be of low fidelity. Furthermore, if the training data does not cover the full domain range, large errors may occur due to extrapolation. Errors may also occur when the set of training points is not sufficiently dense and uniform. Here, we adopt the concept of information granulation as an attempt to address these difficulties.

IV. Adaptive Fuzzy Fitness Granulation (AFFG)

Granular computing is regarded as the processing of granules of information that are aggregated due to their indistinguishability, similarity, proximity or functionality in some context [30]. It is a vehicle for handling information, as well as a lack of it (uncertainty), at a level of coarseness that can solve problems appropriately and efficiently [5]. In problems with incomplete, uncertain or vague information, the *practical necessity*; and in problems with huge detailed information, the *simplicity* are the main reasons of popularity, respectively, of granular computing. It is widely used in many fields including interval analysis, Dempster-Shafer theory of belief functions, cluster analysis, optimization and problem solving [23], machine learning, bioinformatics, among other fields [19]. The concept of information granulation was proposed by Zadeh [31] (in the context of fuzzy set theory) as a technique by which a class of points (objects) is partitioned into granules. The fuzziness of granules and their attributes is characteristic of the ways by which human concepts and reasoning are formed, organized and manipulated. The concept of a granule is more general than that of a cluster, potentially giving rise to several conceptual structures in various fields of science as well as mathematics.

In the present paper, with the aim to reducing the computational cost of MOPs, the concept of information granulation and approximation in the context of rough set theory is studied to exploit the natural tolerance of EAs in fitness function computations. Nature's *survival of the fittest* is not about exact measures of fitness; rather it is about rankings among competing peers. By exploiting this natural tolerance for imprecision and aiming to exploit this uncertainty [16], optimization performance can be preserved by computing fitness only selectively and only to keep this ranking among individuals in a given population.

In the proposed algorithm, a pool of solutions with exact fitness values are maintained. Based on the maximum similarity of a new candidate solution to this pool, the fitness of individuals will be either approximated or calculated explicitly.

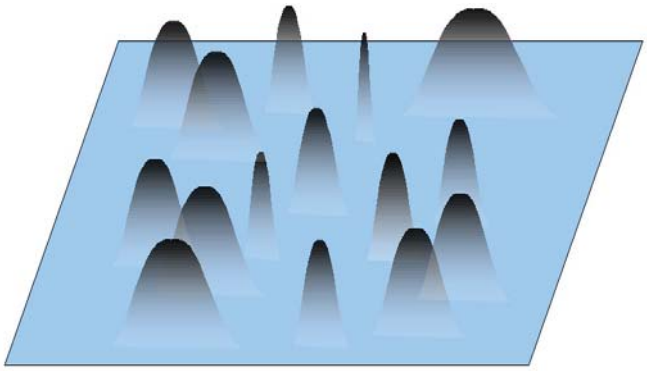

Fig. 1: A number of gaussian granules with different widths in a 2-D solution space. Once a new individual is sufficiently similar to a granule in the granule pool, then that granules' fitness is used instead as a crude estimate. Otherwise, that individual is added to the pool as a new fuzzy granule. Each granules' radius of influence is determined based on equation (7).

If a new individual is sufficiently similar to a known fuzzy granule, then that granules' fitness is used instead as a crude estimate. Otherwise, that individual is added to the pool as a new granule. In this fashion, regardless of the competitions' outcome, the fitness of the new individual is always a physically realizable one, even if it is a *crude* estimate and not an exact measurement. The pool size as well as each granules' radius of influence depends on the utility of each granule [6].Furthermore, to prevent the pool from growing too large, pool members are competing for survival and members with lower *life index* are gradually replaced by new granules. By splitting up the pool into two parts, the new granules are given a chance to survive a number of steps [4].

A. Algorithm's Structure

The preceding section provided a general overview of our approach. Going in more detail now, the algorithm's computation steps are as follows:

Step 1: Create a random parent population $P_1 = \{X_1^1,\ X_2^1,\ \ldots,\ X_j^1,\ \ldots,\ X_t^1\}$ of decision vectors, where, $X_j^i = \{x_{j,1}^i,\ x_{j,2}^i,\ \ldots\ , x_{j,r}^i,\ \ldots,\ x_{j,m}^i\}$ is the jth individual in the ith generation, $x_{j,r}^i$ the rth component of X_j^i, m the number of components of decision vector and t is the population size.

Step 2: Define a multi-set G of fuzzy granules $(C_k,\ \sigma_k,\ L_k)$ according to $G = \{(C_k,\ \sigma_k,\ L_k) | C_k \in \Re^m,\ \sigma_k \in \Re,\ L_k \in \Re,\ k = 1, \ldots, N_G\}$. G is initially empty. C_k is an m-dimensional vector of centers, σ_k is the width of membership functions (WMFs) of the kth fuzzy granule, and L_k is the granule's life index. A number of granules with different widths are shown in Figure 1.

Step 3:

- Choose the phenotype of chromosomes, X_j^i, as the center of granules, C_k.
- Rank P_1 and goto **step 8**.

Step 4: Define the membership $\mu_{k,r}$ of each $x^i_{j,r}$ to each granule member by a Gaussian similarity neighborhood function according to

$$\mu_{k,r}\left(x^i_{j,r}\right) = \exp\left(\frac{-\left(x^i_{j,r} - c_{k,r}\right)^2}{\left(\sigma_k\right)^2}\right),\ k = 1, 2, \ldots, N_G\ , \tag{4}$$

where N_G is the number of fuzzy granules.

Step 5: Compute the average similarity of the new decision vector $X^i_j = \{x^i_{j,1},\ x^i_{j,2},\ \ldots\ , x^i_{j,r},\ \ldots,\ x^i_{j,m}\}$ to each granule G_k using equation (5)

$$\overline{\mu}_{j,k} = \frac{\sum_{r=1}^{m} \mu_{k,r}\left(x^i_{j,r}\right)}{m} \tag{5}$$

Step 6: Either calculate the exact fitness value of X^i_j or estimate it by associating it to one of the granules in the pool in case there is a granule in the pool with similarity value higher than a predefined threshold, i.e.,

$$f\left(X^i_j\right) = \begin{cases} f\left(C_k\right) & \text{if } \max\limits_{k\in\{1,2,\ldots,N_G\}} \{\overline{\mu}_{j,k}\} > \theta^i\ , \\ f\left(X^i_j\right) & \text{otherwise.} \end{cases} \tag{6}$$

where $f(C_x)$ is the fitness function value of the fuzzy granule and $f(X^i_j)$ is the real fitness calculation of the individual.

Remark: θ^i is a predefined (time-varying) threshold that controls the minimum similarity a solution has to have with a pool member to be approximated. Here, θ^i is considered as a constant value for all simulations, and is set to 0.9. In general, as the population matures steadily, the algorithm needs to be more selective (to calculate the exact fitness more often), suggesting the need for a gradual increase of θ^i. Alternatively, if

$$\max_{k\in\{1,2,\ldots,N_G\}} \{\overline{\mu}_{j,k}\} < \theta^i$$

X^i_j is chosen as a newly created granule.

Step 7: If the population size is not completed, repeat **Steps 4 to 7**.

Step 8: When termination/evolution control criteria are not met:

- Create offspring population.
- Rank the granule pool.
- Assign σ_k based on equation (7).

$$\sigma_k = \sigma_{min} * ((1 - gr_\sigma) + gr_\sigma * \text{rank}(k)) \tag{7}$$

where rank(k) is the rank of the granule k among the granule set, and $\sigma_{min} \in \mathbb{R}_{>0}$ is a proportional constant that defines the minimum spread of granules. σ_{min} is a problem dependent design parameter.

Remark: σ_k, the distance measurement parameter that controls the degree of similarity between two individuals, controls the radius of influence of each granule. Instead of drawing the radius directly from the fitness (as in the single-objective optimization case [4]), as objectives are often non-commensurable and conflicting, dominance-based ranking is used. The spread of granules grow as their rank among granule members increases, with a rate of gr_σ. Here, gr_σ is set to 0.1 and $\sigma_{min} \in \{2^n | n \in \mathbb{Z}\}$.

- Goto **step 4**.

B. Controlling the granule pool length and protecting new pool members (innovation) through speciation

As the evolutionary procedures are applied, it is inevitable that new granules are generated and added to the pool. Depending on the complexity of the problem, the size of this pool can be excessive and become a computational burden itself. To prevent such unnecessary computational effort, a *life index* is introduced in order to appropriately decrease the size of the pool. In other words, it is better to remove granules that do not win new individuals, thereby producing a bias against individuals that have low fitness and were likely produced by a failed mutation attempt. L_k is initially set to 0 and subsequently updated as below,

$$L_k = \begin{cases} L_k + M & \text{if } k = K\ , \\ L_k & otherwise\ , \end{cases} \tag{8}$$

where M is the life reward of the granule and K is the index of the winning granule for each individual at generation i. Here, M is set at 1. At each table update, only the N_G granules with the highest L_k index are kept, and the others are discarded. In [5], an example has been provided that illustrates the competitive granule pool update law. Adding a new granule to the granule pool and assigning a life index to it, is a simple way of controlling the size of the granule pool, since the granules with the lowest life index will be removed from the pool. However, it may happen that the new granule is removed, even though it was just inserted into the pool. In order to prevent this, the pool is split into two parts with sizes εN_G and $(1-\varepsilon)N_G$. The first part is a FIFO (First In, First Out) queue and new granules are added to this part. If it grows above εN_G, then the top of the queue is moved to the other part. Removal from the pool takes place only in the $(1-\varepsilon)N_G$ part. In this way, new granules have a good chance to survive a number of steps. In all of the simulations that are conducted here, ε is set to 0.1.

V. Numerical results

In order to validate our proposed approach, we adopted the Zitzler-Deb-Thiele (ZDT) test problems [33] and compared our results with respect to those obtained with the standard NSGA-II [8]. The following parameters were adopted for our experiments:

- Population size = 50.
- Crossover rate = 0.9 (SBX).
- Binary tournament selection.
- Mutation rate of $1/m$, m = number of decision variables.
- Distribution indices for crossover η_c and mutation η_m: $\eta_c = 20$ and $\eta_m = 20$.

Problem	σ_{min}	N_G	Reference point
ZDT1	2^{-4}	100	$[1.1, 3.5]$
ZDT2	2^{-5}	100	$[1.1, 5.0]$
ZDT3	2^{-5}	100	$[1.1, 6.0]$
ZDT4	2^{-6}	100	$[1.1, 140]$
ZDT6	2^{-5}	100	$[1.1, 9.0]$

TABLE I: AFFG-NSGA-II utilized parameter values and reference points used for calculating I_H.

Problem	AFFG-NSGA-II mean, σ	NSGA-II mean, σ
ZDT1	$0.010165, 0.005744$	$0.102095, 0.029859$
ZDT2	$0.018143, 0.008509$	$0.716683, 0.365823$
ZDT3	$0.098656, 0.022421$	$0.236176, 0.048486$
ZDT4	$11.160124, 4.239201$	$20.191547, 11.658247$
ZDT6	$0.768217, 0.143028$	$1.328310, 0.224595$

TABLE II: Mean and standard deviation of the GD performance measure.

For assessing our results we adopted three performance measures: (1) Generational Distance (GD) [28], which measures how far the given solutions are, on average, from the true Pareto front, (2) the Hypervolume indicator I_H (also known as Lebesgue measure or S-metric) [32], which measures the volume of the dominated portion of the objective space which is enclosed by the reference set and (3) Set Coverage (SC) [33], which measures the percentage of solutions from one algorithm that are covered by the solutions of the other. To measure the Hypervolume, a single reference point, $R = r \in \Re^m$ was considered in all cases. This point corresponds to the worst value in each dimension of the fronts. The reference values we used here are given in Table I.

The performance measures to assess the results are presented in Tables II, III and IV. The measures are evaluated by conducting 30 independent runs per test problem per algorithm. Each run is restricted to 1,000 fitness function evaluations. Each table displays the average and standard deviation of each of the performance measures.

Figures 2 to 11 present results of 30 independent runs of the standard NSGA-II and the AFFG-NSGA-II, respectively, adopting the test problems ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6 (ZDT5 is a binary problem and was, therefore, omitted here), with a budget of only 1,000 fitness function evaluations. Each color corresponds to a single run.

The results clearly show that the proposed AFFG approach outperforms the standard NSGA-II. According to the

Problem	AFFG-NSGA-II mean, σ	NSGA-II mean, σ
ZDT1	$3.408204, 0.052768$	$2.689226, 0.164173$
ZDT2	$4.524421, 0.110119$	$2.227951, 0.350130$
ZDT3	$6.106243, 0.198963$	$4.516725, 0.267211$
ZDT4	$108.878924, 10.460062$	$100.619288, 9.466605$
ZDT6	$3.229885, 0.896935$	$1.178803, 0.176150$

TABLE III: Mean and standard deviation of the I_H performance measure.

Problem	AFFG-NSGA-II mean, σ	NSGA-II-AFFG mean, σ
ZDT1	$1.000000, 0.000000$	$0.000000, 0.000000$
ZDT2	$1.000000, 0.000000$	$0.000000, 0.000000$
ZDT3	$0.995745, 0.023307$	$0.003401, 0.018630$
ZDT4	$0.613805, 0.455574$	$0.324147, 0.427815$
ZDT6	$0.891819, 0.134759$	$0.033209, 0.081798$

TABLE IV: Mean and standard deviation of the SC performance measure.

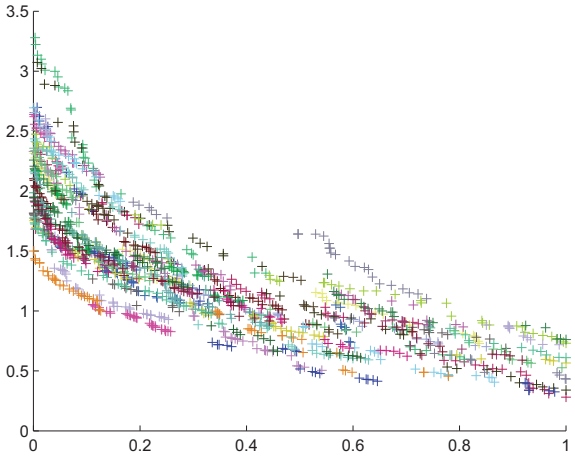

Fig. 2: 30 independent runs of the NSGA-II for the ZDT1 test problem using 1,000 real fitness function evaluations.

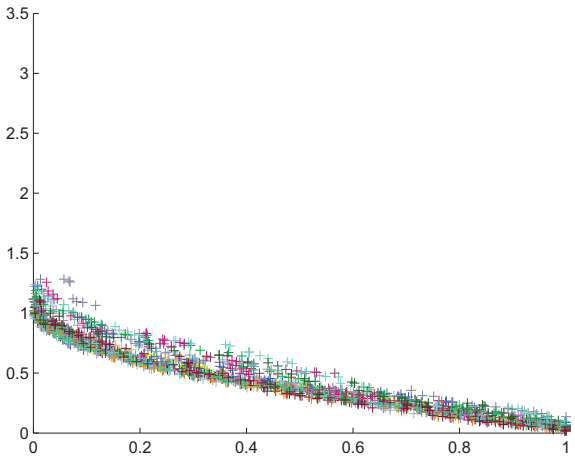

Fig. 3: 30 independent runs of the AFFG-NSGA-II over the ZDT1 problem using 1,000 real fitness function evaluations.

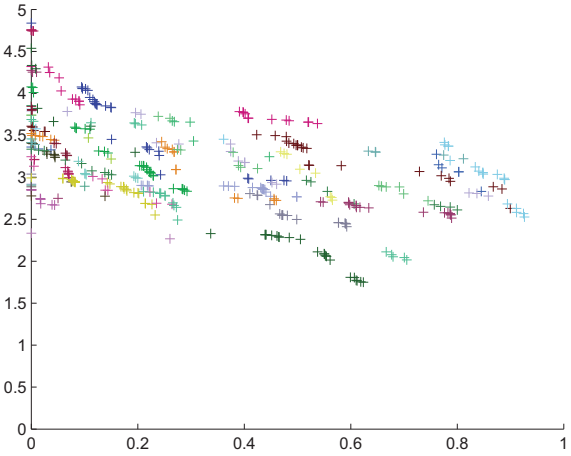

Fig. 4: 30 distinct runs of the NSGA-II over the ZDT2 problem using 1,000 real fitness function evaluations.

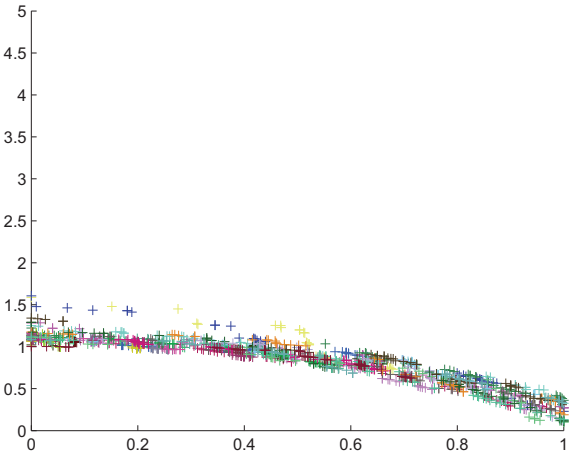

Fig. 5: 30 distinct runs of the AFFG-NSGA-II over the ZDT2 problem using 1,000 real fitness function evaluations.

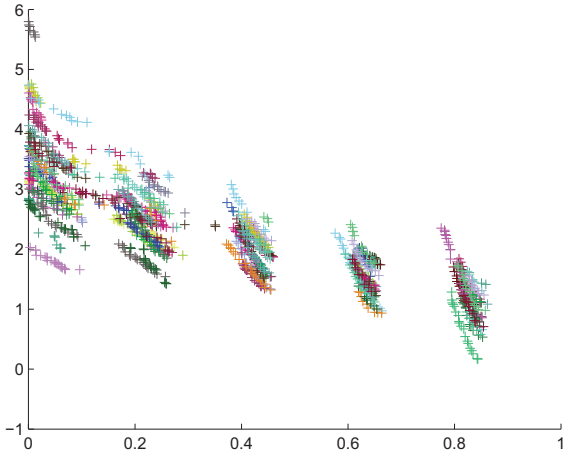

Fig. 6: 30 independent runs of the NSGA-II over the ZDT3 problem using 1,000 real fitness function evaluations.

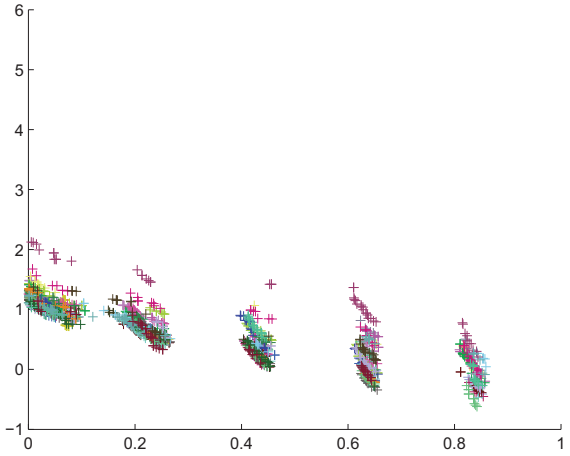

Fig. 7: 30 distinct runs of the AFFG-NSGA-II over the ZDT3 problem using 1,000 real fitness function evaluations.

Wilcoxon rank-sum test, the results of our proposed approach are better with a significance level of 5%. To further investigate the convergence speed of the proposed approach, in Figure 12, the changes in hypervolume metric is plotted against the number of fitness function evaluations, for the ZDT1 problem.

VI. Conclusions and Future Work

By combining the concepts of survival of the fittest and fuzzy granulation, which enables a faster convergence without degrading the estimated set of solutions, this paper presents an approach to speed up convergence towards the Pareto optimal front of multi-objective optimization problems. With the proposed approach, we can exploit the information obtained from our previous objective function evaluations. Our results indicate that the proposed approach is very promising, since it can achieve a faster convergence than the standard NSGA-II in the test problems adopted. However, a more thorough validation is still required (adopting other problems such as the DTLZ test problems [9]). It is also desirable to perform comparisons with respect to other fitness approximation methods such as curve fitting, fitness inheritance and artificial neural networks. As part of our future work, we are interested in studying the effect of the number of granules on the convergence rate. Additionally, in order to further test the robustness of our proposed approach, we want to study its sensitivity to its parameters and its scalability when increasing the number of decision variables and objectives. Adaptively changing θ^i and being more selective as the population matures (to calculate

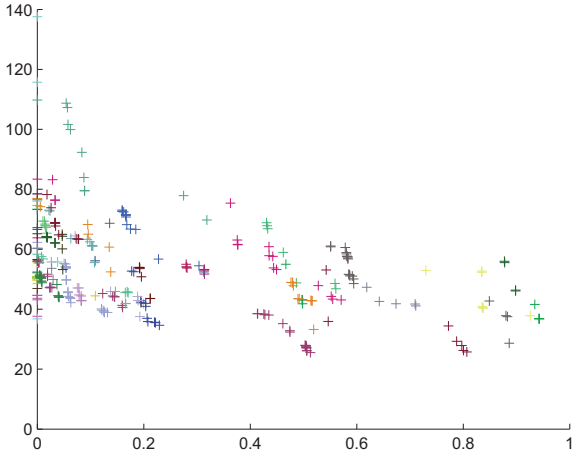

Fig. 8: 30 independent runs of the NSGA-II over the ZDT4 problem using 1,000 real fitness function evaluations.

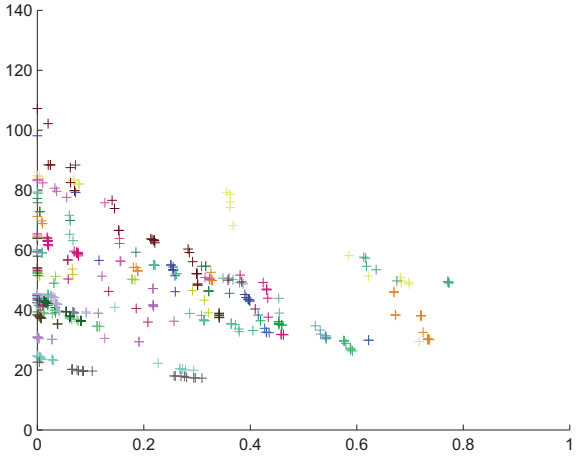

Fig. 9: 30 distinct runs of the AFFG-NSGA-II over the ZDT4 problem using 1,000 real fitness function evaluations.

the exact fitness more often), is indeed part of our ongoing research. Finally, we wish to apply our proposed approach to real-world problems in the field of multi-objective network design problems [29] and supplier selections [22].

ACKNOWLEDGMENT

This research received funding from the European Community's Seventh Framework Programme within the "Control for Coordination of Distributed Systems" (Con4Coord - FP7/2007-2013 under grant agreement no. INFSO-ICT-223844), the Next Generation Infrastructures Research Program of Delft University of Technology and the Mexican CONACyT Project No. 103570.

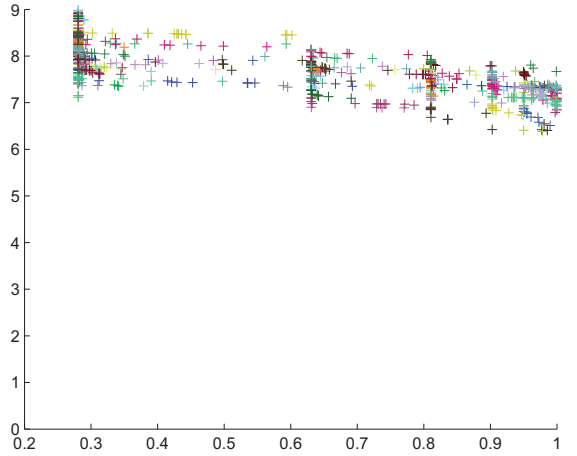

Fig. 10: 30 independent runs of the NSGA-II over the ZDT6 problem using 1,000 real fitness function evaluations.

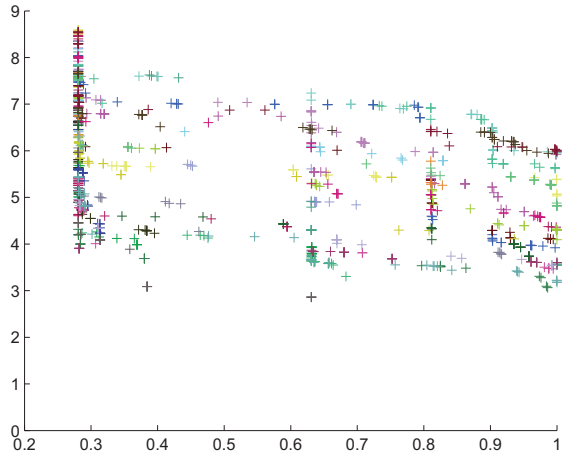

Fig. 11: 30 distinct runs of the AFFG-NSGA-II over the ZDT6 problem using 1,000 real fitness function evaluations.

REFERENCES

[1] J.J. Alonso, P. LeGresley, and V. Pereyra. Aircraft design optimization. *Mathematics and Computers in Simulation*, 79(6):1948–1958, 2009.

[2] C.A. Coello Coello, G.B. Lamont, and D.A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, New York, second edition, September 2007. ISBN 978-0-387-33254-3.

[3] S. D'Angelo and E.A. Minisci. Multi-objective evolutionary optimization of subsonic airfoils by kriging approximation and evolutionary control. In *2005 IEEE Congress on Evolutionary Computation (CEC'2005)*, volume 2, pages 1262–1267, Edinburg, Scotland, September 2005.

[4] M. Davarynejad, C.W. Ahn, J.L.M. Vrancken, J. van den Berg, and C.A. Coello Coello. Evolutionary hidden information detection by granulation-based fitness approximation. *Applied Soft Computing*, 10(3):719–729, 2010.

[5] M. Davarynejad, M.-R. Akbarzadeh-T, and N. Pariz. A novel general framework for evolutionary optimization: Adaptive fuzzy fitness granulation. In *IEEE Congress on Evolutionary Computation*, pages 951–956. IEEE, 2007.

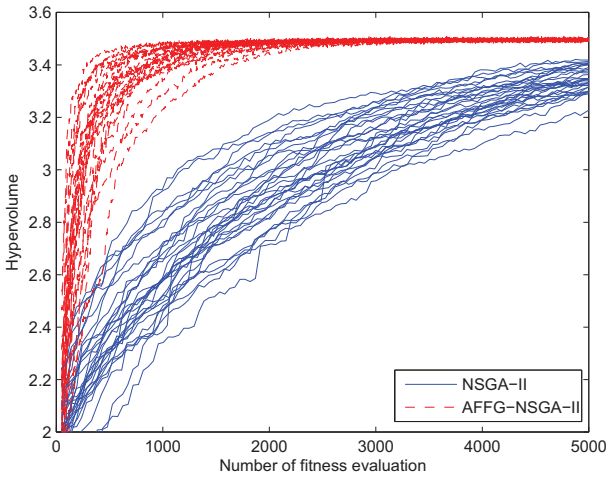

Fig. 12: Convergence of the hypevolume metric for the ZDT1 problem (30 distinct runs).

[6] M. Davarynejad, J. Vrancken, J. van den Berg, and C.A. Coello Coello. A Fitness Granulation Approach for Large-Scale Structural Design Optimization. In Raymond Chiong, Thomas Weise, and Zbigniew Michalewicz, editors, *Variants of Evolutionary Algorithms for Real-World Applications*. Springer-Verlag, Berlin, 2011.

[7] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, UK, 2001. ISBN 0-471-87339-X.

[8] K. Deb, A. Pratap, Sameer Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

[9] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable Test Problems for Evolutionary Multiobjective Optimization. In Ajith Abraham, Lakhmi Jain, and Robert Goldberg, editors, *Evolutionary Multiobjective Optimization. Theoretical Advances and Applications*, pages 105–145. Springer, USA, 2005.

[10] E.I. Ducheyne, B. De Baets, and R.R. De Wulf. Fitness inheritance in multiple objective evolutionary algorithms: A test bench and real-world evaluation. *Applied Soft Computing*, 8(1):337–349, 2008.

[11] E.I. Ducheyne, B. De Baets, and R. De Wulf. Is Fitness Inheritance Useful for Real-World Applications? In Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele, editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 31–42, Faro, Portugal, April 2003. Springer. Lecture Notes in Computer Science. Volume 2632.

[12] T. Goel, R. Haftka, W. Shyy, N. Queipo, R. Vaidyanathan, and K. Tucker. Response surface approximation of pareto optimal front in multi-objective optimization. *Computer Methods in Applied Mechanics and Engineering*, 196(4-6):879–893, 1 January 2007.

[13] T. Goel, R. Vaidyanathan, R. Haftka, W. Shyy, N. Queipo, and K. Tucker. Response surface approximation of pareto optimal front in multiobjective optimization. Technical Report 2004-4501, AIAA, 2004.

[14] Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing*, 9(1):3–12, 2005.

[15] R. Landa Becerra and C.A. Coello Coello. Solving Hard Multiobjective Optimization Problems Using ε-Constraint with Cultured Differential Evolution. In Thomas Philip Runarsson, Hans-Georg Beyer, Edmund Burke, Juan J. Merelo-Guervós, L. Darrell Whitley, and Xin Yao, editors, *Parallel Problem Solving from Nature - PPSN IX, 9th International Conference*, pages 543–552. Springer. Lecture Notes in Computer Science Vol. 4193, Reykjavik, Iceland, September 2006.

[16] D. Lim, Y. Jin, Y.S. Ong, and B. Sendhoff. Generalizing surrogate-assisted evolutionary computation. *IEEE Transactions on Evolutionary Computation*, 14(3):329–355, 2010.

[17] J.I. Madsen, W. Shyy, and R.T. Haftka. Response surface techniques for diffuser shape optimization. *AIAA journal*, 38(9):1512–1518, 2000.

[18] H. Nakayama, M. Arakawa, and K. Washino. Optimization for black-box objective functions. In Panos M. Pardalos, Ider Tseveendorj, and Rentsen Enkhbat, editors, *Optimization and Optimal Control*, pages 185–210. World Scientific, Singapore, 2003.

[19] W. Pedrycz, A. Skowron, and V. Kreinovich. *Handbook of granular computing*. Wiley-Interscience New York, NY, USA, 2008.

[20] C. Poloni, A. Giurgevich, L. Onesti, and V. Pediroda. Hybridization of a multi-objective genetic algorithm, a neural network and a classical optimizer for a complex design problem in fluid dynamics. *Computer Methods in Applied Mechanics and Engineering*, 186(2-4):403–420, 2000.

[21] M. Reyes Sierra and C.A. Coello Coello. A Study of Fitness Inheritance and Approximation Techniques for Multi-Objective Particle Swarm Optimization. In *2005 IEEE Congress on Evolutionary Computation (CEC'2005)*, volume 1, pages 65–72, Edinburgh, Scotland, September 2005. IEEE Service Center.

[22] J. Rezaei and M. Davoodi. Multi-objective models for lot-sizing with supplier selection. *International Journal of Production Economics*, 130(1):77–86, 2011.

[23] A. Rowhanimanesh and M.-R. Akbarzadeh-T. Perception-based evolutionary optimization: Outline of a novel approach to optimization and problem solving. In *IEEE International Conference on Systems Man and Cybernetics*, pages 4270–4275. IEEE Press, 2010.

[24] J. Sacks, W.J. Welch, T.J. Mitchell, and H.P. Wynn. Design and analysis of computer experiments. *Statistical science*, 4(4):409–423, 1989.

[25] L.V. Santana-Quintero, A. Arias Montaño, and C.A. Coello Coello. A Review of Techniques for Handling Expensive Functions in Evolutionary Multi-Objective Optimization. In Yoel Tenne and Chi-Keong Goh, editors, *Computational Intelligence in Expensive Optimization Problems*, pages 29–59. Springer, Berlin, Germany, 2010. ISBN 978-3-642-10700-9.

[26] M. Smith. *Neural Networks for Statistical Modeling*. von Nostrand, Reinhold, New York, USA, 1993.

[27] R.E. Smith, B. A. Dike, and S. A. Stegmann. Fitness inheritance in genetic algorithms. In *SAC '95: Proceedings of the 1995 ACM symposium on Applied computing*, pages 345–350, New York, NY, USA, 1995. ACM Press.

[28] D.A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, May 1999.

[29] L.J.J. Wismans, E.C. van Berkum, and M.C.J. Bliemer. Comparison of multi-objective evolutionary algorithms for optimization of externalities using dynamic traffic management measures. In *TRB conference*.

[30] Y.Y. Yao. Information granulation and rough set approximation. *International Journal of Intelligent Systems*, 16(1):87–104, 2001.

[31] L. A. Zadeh. Fuzzy sets and information granularity. In *Advances in Fuzzy Set Theory and Applications*, pages 3–18. North Holland, New York, USA, 1979.

[32] E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, November 1999.

[33] E. Zitzler, K. Deb, and L. Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195, Summer 2000.