A study on population size and selection lapse in many-objective optimization

Heman Aguirre, Arnaud Liefooghe, Sébastien Verel, Kiyoshi Tanaka

2013 IEEE Congress on Evolutionary Computation
June 20-23, Cancún, México

A Study on Population Size and Selection Lapse in Many-objective Optimization

Hernán Aguirre* Arnaud Liefooghe† Sébastien Verel‡ Kiyoshi Tanaka*

* Faculty of Engineering Shinshu University
4-17-1 Wakasato, Nagano, 380-8553 Japan
Email:ahernan@shinshu-u.ac.jp, ktanaka@shinshu-u.ac.jp

† Université Lille 1 LIFL, UMR CNRS 8022, France
Inria Lille-Nord Europe, France
Email: arnaud.liefooghe@lifl.fr

‡ Université Nice Sophia-Antipolis, Nice France
Inria Lille-Nord Europe, France
Email:verel@i3s.unice.fr

***Abstract*—In this work we study the effects of population size on selection and performance scalability of two dominance-based algorithms applied to many-objective optimization. Our aim is to understand the relationship between the size of the Pareto optimal set, a characteristic of the many-objective problem at hand, the population size and the ability of the algorithm to retain Pareto optimal solutions in its population and find new ones. This work clarifies important issues of the dynamics of evolutionary algorithms on many-objective landscapes, particularly related to survival selection. It shows that optimal solutions are dropped from the population in favor of suboptimal solutions that appear non-dominated when survival selection is applied. It also shows that this selection lapse, the dropping of optimal solution, affects the discovery of new optimal solutions and is correlated to population size and the distribution of solutions that survival selection renders. Selection makes less mistakes with larger populations and when the distribution of solutions is better controlled. The results of this study will be helpful to properly set population size and have a clearer idea about the performance expectation of the algorithm.**

I. Introduction

Conventional evolutionary multi-objective (EMO) algorithms [1] are known to scale up poorly to high dimensional objective spaces [2], particularly dominance-based algorithms. This lack of scalability has been attributed mainly to some characteristics of the landscapes and inappropriate operators for selection and variation. In many-objective landscapes the number of solutions in the Pareto optimal set increases exponentially [3] with the number of objectives. Assuming that the size of the decision space remains unchanged, an increase in the number of objectives also implies that these large number of Pareto optimal solutions become spread over broader regions of decision space [4]. Analysis of many-objective landscapes shows that this is the case for the Pareto optimal set and also for Pareto suboptimal sets [3]. These characteristics of many-objective landscapes are directly correlated to the effectiveness of the operators of selection and variation and affect the dynamics of the optimizer. For example, the large number of non-dominated solutions causes dominance-based selection to become random, and their spread on decision space causes mating to pair distant solutions making recombination too disruptive. In addition to the operators for selection and variation, the population size greatly influences the dynamics of the algorithm. However, its effects on large dimensional objectives spaces are not well understood.

In this work we study the effects of population size on performance scalability of two dominance-based algorithms applied to many-objective optimization, a conventional evolutionary multi-objective (EMO) algorithm and an evolutionary many-objective (EMyO) algorithm, using relatively small population sizes and also using large population sizes set as a fraction of the true Pareto optimal set. We are particularly interested in understanding the relationship between the size of the Pareto optimal set, a characteristic of the many-objective problem at hand, the population size the algorithm uses and the ability of the algorithm to retain true Pareto optimal solutions in its population and find new ones. We analyze the dynamics of the algorithms by observing at each generation the number of true Pareto optimal solutions that the algorithms find. We use MNK-landscapes with $3-6$ objectives and 20 bits, for which it is possible to know by enumeration all true Pareto optimal solutions of the landscapes. This work clarifies important issues of the dynamics of evolutionary algorithms on many-objective landscapes, particularly related to survival selection. The results of this study will also be helpful to properly set population size and have a clearer idea about the performance expectation of the algorithm.

II. Methodology

In our study we use four MNK-landscapes [3] randomly generated with $m = 3, 4, 5, 6$ objectives, $n = 20$ bits, and $k = 1$ epistatic bit. For each landscape we enumerate all its solutions and classify them to non-dominated fronts. The exact number of true Pareto optimal solutions POS^T found by enumeration are $|POS^T| = 152, 1554, 6265$, and 16845

for $m = 3$, 4, 5, and 6 objectives, respectively. Similarly, the exact number of non-dominated fronts of the landscapes are 258, 76, 29, and 22, respectively.

We run the algorithms for a fixed number of generations, collecting all generated solutions. Once evolution is over, we compare the set of POS^T with the sets of unique non-dominated solutions obtained at each generation after survival selection to determine which are true Pareto optimal solutions, count the number of true Pareto optimal solutions found at each generation, and their accumulated number found during evolution. That is, in our analysis we assume that the aim of the algorithms is to find all true Pareto optimal solutions. In this work, we relate performance scalability to the capacity of the algorithm to achieve its aim either when we increase the number of objectives for a given population size or when we increase the population size for a given number of objectives.

The motivation to use landscapes with $n = 20$ bits and $k = 1$ epistatic bit is that in small landscapes with minimum non-linearity it should be relatively simple for the algorithm to hit the optimal set. Experiments in these small and simple problems allow us to focus our study on the ability of the algorithm to retain optimal solutions in its population, which is directly correlated to the effectiveness of survival selection and the maintenance of selection pressure to find new optimal solutions. Also, it is feasible to enumerate these small landscapes to know exactly all true Pareto optimal solutions, which are important in our analysis.

In this work we use NSGA-II as the evolutionary multi-objective optimizer and the Adaptive ε-Sampling and ε-Hood algorithm [5] as the evolutionary many-objective optimizer. NSGA-II is a well known algorithm and due to space limitations it is not described here. The interested reader is referred to [6] for details about NSGA-II. In the following we describe the recently proposed many-objective optimizer used in our study.

III. The AεSεH Algorithm

A. Concept

Adaptive ε-Sampling and ε-Hood (AεSεH) [5] is an elitist evolutionary many-objective algorithm that applies ε-dominance principles both for survival selection and for clustering and mating solutions located close by in objective space.

In multi-objective optimization, dominance is used for survival selection and to rank solutions in the elite surviving population, so that mating selection can give more reproductive opportunities to dominant individuals. However, this is impractical in many-objective optimization because of the large number of non-dominated solutions. In AεSεH, dominance is used during the survival selection step of the algorithm to eliminate inferior dominated solutions, but it has no role in ranking the surviving population. For the majority of generations, survival selection is achieved by *ε-sampling*, a ε-dominance based procedure, which samples randomly from the large set of non-dominated solutions and eliminates solutions ε-dominated by the samples. The aim is to get a set of surviving solutions spaced according to the distribution implicit in the mapping function $\boldsymbol{f}(\boldsymbol{x}) \mapsto^{\epsilon} \boldsymbol{f}'(\boldsymbol{x})$ used for ε-dominance. Only during the few initial generations, where the number of non-dominated solutions is smaller than the size of the surviving population, ε-sampling plays no role during survival.

After survival selection, in AεSεH there is not an explicit ranking that could be used to bias mating. Rather, the algorithm uses a procedure called *ε-hood creation* to cluster solutions in objective space. This method is also based on ε-dominance. Here, a randomly sampled solution from the surviving population and its ε-dominated solutions determine the neighborhood, so that recombination can take place between individuals located close by in objective space. The motivation to restrict mating is to enhance the effectiveness of recombination in many-objective problems, where the difference in variable space between individuals in the population is expected to be larger than in multi-objective problems and therefore more disruptive for recombination.

The main steps of the AεSεH are as follows.

Step 1 Set $\varepsilon_s \leftarrow 0$, used in *$\varepsilon$-sampling truncation*, and its initial step of adaptation Δ_s. Set $\varepsilon_h \leftarrow 0$, used in *$\varepsilon$-hood creation*, its initial step of adaptation Δ_h and the reference number of neighborhoods N_H^{Ref} to adapt ε_h. Set population $\mathcal{P} \leftarrow \emptyset$ and create randomly the initial population $\mathcal{Q}$. Set population size $P_{size} \leftarrow |\mathcal{Q}|$.

Step 2 Evaluate the offspring population $\mathcal{Q}$.

Step 3 Calculate non-dominated sorting on the population that results from joining the current population $\mathcal{P}$ and its offspring $\mathcal{Q}$ to obtain the non-dominated fronts $\mathcal{F} = \{\mathcal{F}_i\}, i = 1, 2, \cdots, N_F$.

Step 4 Truncate the sorted non-dominated fronts $\mathcal{F}$ to obtain the surviving population $\mathcal{P}$ of size P_{size} using the *ε-sampling truncation* procedure set with parameter ε_s.

Step 5 Adapt ε_s and its step of adaptation Δ_s using as reference the population size P_{size} and the number of sampled solutions N_S returned by *ε-sampling* called from *ε-sampling truncation*.

Step 6 Create neighborhoods from the surviving population $\mathcal{P}$ with the procedure *ε-hood creation*, set with parameter ε_h.

Step 7 Adapt ε_h and its step of adaptation Δ_h using as reference the number of created neighborhoods N_H and the number N_H^{Ref} specified by the user.

Step 8 Create a pool of mates $\mathcal{P}'$ with procedure *ε-hood mating* that pairs solutions within the neighborhoods.

Step 9 Recombine and mutate the mated individuals in $\mathcal{P}'$ to create the offspring population $\mathcal{Q}$.

Step 10 Repeat from **Step 2** if the termination criterion has not been met.

The next sections include additional details about the main procedures of the algorithm.

B. ε-Sampling Truncation (Step 4)

Survival selection is implemented by the *ε-sampling truncation* procedure. This procedure receives the sets of solutions $\mathcal{F}$ created by non-dominated sorting and selects exactly P_{size} surviving solutions from them.

In case the number of non-dominated solutions $|\mathcal{F}_1| > P_{size}$, it calls *$\varepsilon$-sampling* with parameter ε_s to get from $\mathcal{F}_1$

its extreme solutions $\mathcal{E}$, a subset of randomly sampled solutions $\mathcal{S}$ and their ε_s-dominated solutions $\mathcal{D}^{\varepsilon_s}$. The surviving population $\mathcal{P}$ always includes extreme solutions $\mathcal{E}$ and it is complemented with solutions from $\mathcal{S}$ and possibly from $\mathcal{D}^{\varepsilon_s}$. If $\mathcal{S}$ overfills $\mathcal{P}$, solutions in $\mathcal{S}$ are randomly eliminated as survivors. Otherwise, if after adding $\mathcal{S}$ to $\mathcal{P}$ there is still room for some solutions, the required number are randomly chosen from $\mathcal{D}^{\varepsilon_s}$.

Otherwise ($|\mathcal{F}_1| < P_{size}$), the sets of solutions $\mathcal{F}_i$ are copied iteratively until $\mathcal{P}$ is filled. If set $\mathcal{F}_i$, $i > 1$, overfills $\mathcal{P}$, the required number of solutions are chosen randomly from it.

C. ε-Hood Creation and ε-Hood Mating (Step 6 and Step 8)

Neighborhoods are created from the surviving population by the *ε-hood creation* procedure, which is also based on ε-dominance. This procedure randomly selects an individual from the surviving population and applies ε-dominance with parameter ε_h. A neighborhood is formed by the selected solution and its ε_h-dominated solutions. Neighborhood creation is repeated until all solutions in the surviving population have been assigned to a neighborhood.

Mating for recombination is implemented by the procedure *ε-hood mating*. Neighborhoods are considered to be elements of a list. Neighborhoods are assigned to the list in the same order they were created. To select two mates, first a neighborhood from the list is specified deterministically in a round-robin schedule. Then, two individuals are select randomly within the specified neighborhood, so that an individual will recombine with other individual that is located close by in objective space. Due to the round-robin schedule, the next two mates will be selected from the next neighborhood in the list. When the end of the neighborhoods list is reached, mating continues with the first neighborhood in the list. Thus, all individuals have the same probability of being selected within a specified neighborhood, but due to the round-robin scheduling individuals belonging to neighborhoods with fewer members have more recombination opportunities that those belonging to neighborhoods with more members. Once the pool of all mates $\mathcal{P}^{'}$ has been established, they are recombined and mutated according to the order they were selected during mating.

D. Adaptation (Step 5 and Step 7)

The number of sampled solutions N_S by ε-sampling depends on the value set to ε_s (≥ 0). Larger values of ϵ_s imply that sampled solutions ε_s-dominate larger areas, increasing the likelihood of having more ε_s-dominated solutions excluded from the sample. The algorithm adapts ε_s at each generation so that N_S is close to the population size P_{size}. The closer N_S is to P_{size}, the larger the number of surviving solutions that will be spaced according to the distribution implicit in the mapped function used for ε-dominance.

Similarly, the number of created neighborhoods N_H depends on the value set to ε_h (≥ 0). Larger values of ε_h imply that sampled solutions ε_h-dominate larger areas, increasing the likelihood of having more ε_h-dominated solutions that form its neighborhood, and therefore fewer neighborhoods are created. The algorithm adapts ε_h at each generation so that N_H is close to a user specified number N_H^{Ref}.

The adaptation rule, similar for both processes, is as follows. If $N > Ref$ it increases the step of adaptation $\Delta \leftarrow \min(\Delta \times 2, \Delta_{max})$ and $\epsilon \leftarrow \epsilon + \Delta$. Otherwise, if $N < Ref$ it decreases $\Delta \leftarrow \max(\Delta \times 0.5, \Delta_{min})$ and $\epsilon \leftarrow \max(\epsilon - \Delta, 0.0)$. In this work we set initial values $\epsilon_0 = 0.0$ and $\Delta_0 = 0.005$. Also, $\Delta_{max} = 0.05$ and $\Delta_{min} = 0.0001$.

In the case of adapting the parameter ε_s used for truncation, the above rule is called with $\varepsilon = \varepsilon_s$, $\Delta = \Delta_s$, $N = N_S$, and $Ref = P_{size}$. On the other hand, in the case of the parameter ε_h used for neighborhood creation, the above rule is called with $\varepsilon = \varepsilon_h$, $\Delta = \Delta_h$, $N = N_H$, and $Ref = N_H^{Ref}$.

IV. Experimental Results and Discussion

A. Operators of Variation and Parameters

In both algorithms, NSGA-II and AεSεH, we use two point crossover with rate $pc = 1.0$, and bit flip mutation with rate $pm = 1/n$. Also, for AεSεH we set the reference neighborhood size $H_{size_{'}}^{Ref}$ to 20 individuals. The mapping function $\boldsymbol{f}(\boldsymbol{x}) \mapsto^{\epsilon} \boldsymbol{f}(\boldsymbol{x})$ used for ε-dominance in ε-sampling truncation and ε-hood creation is additive, as follows

$$f_i^{'} = f_i + \varepsilon, \qquad i = 1, 2, \cdots, m \tag{1}$$

B. NSGA-II

In this section we study the effects of population size on NSGA-II. Let us denote by $\mathcal{F}_1$ the set of non-dominated solutions in the combined population $\mathcal{P} \cup \mathcal{Q}$ of parents and offspring before survival selection, by $F_1 \subseteq \mathcal{F}_1$ the set of non-dominated solutions in population $\mathcal{P}$ after survival selection, and by F_1^T the set of solutions in F_1 that are true Pareto optimal solutions.

Fig.1 shows the number of solutions in F_1 and F_1^T over the generations for $m = 3$, 4, and 5 objectives, running the algorithm for 100 generations with three different population sizes $|\mathcal{P}| = 50$, 100 and 200. First we analyze results for $m = 3$ objectives. When we set population size to $|\mathcal{P}| = 50$ or 100, a value smaller than the number of true Pareto optimal solutions $|POS^T| = 152$, it can be seen in Fig.1 (a.1) and (a.2) that after few generations all solutions in the population are non-dominated, $|F_1| = |\mathcal{P}|$. However, not all solutions in F_1 are true Pareto optimal solutions, i.e. $|F_1^T| < |F_1| = |\mathcal{P}|$. Also, it is important to note that $|F_1^T|$ fluctuates up and down after an initial increase. On the other hand, when we set the population size to a value larger than the number of true Pareto optimal solutions, $|\mathcal{P}| = 200 > |POS^T| = 152$, it can be seen in Fig.1 (a.3) that the instantaneous non-dominated set is a subset of the population, $\mathcal{F}_1 = F_1 \subset P$. Also, note that from generation 35 onwards, all non-dominated solutions in the population are also true Pareto optimal, $F_1 = F_1^T$. In this case, the algorithm finds and keeps in $\mathcal{P}$ almost all true Pareto optimal solutions, 147 out of 152, during the latest stage of the search.

It is known that the number of true Pareto optimal solutions $|POS^T|$ increases considerably with the number of objectives. However, this is often ignored and the algorithm is set with a very small population size compared to $|POS^T|$. To study these cases, Fig.1 (b.1)-(b.3) and (c.1)-(c.3) show results for $m = 4$ and $m = 5$ objectives setting population size to the

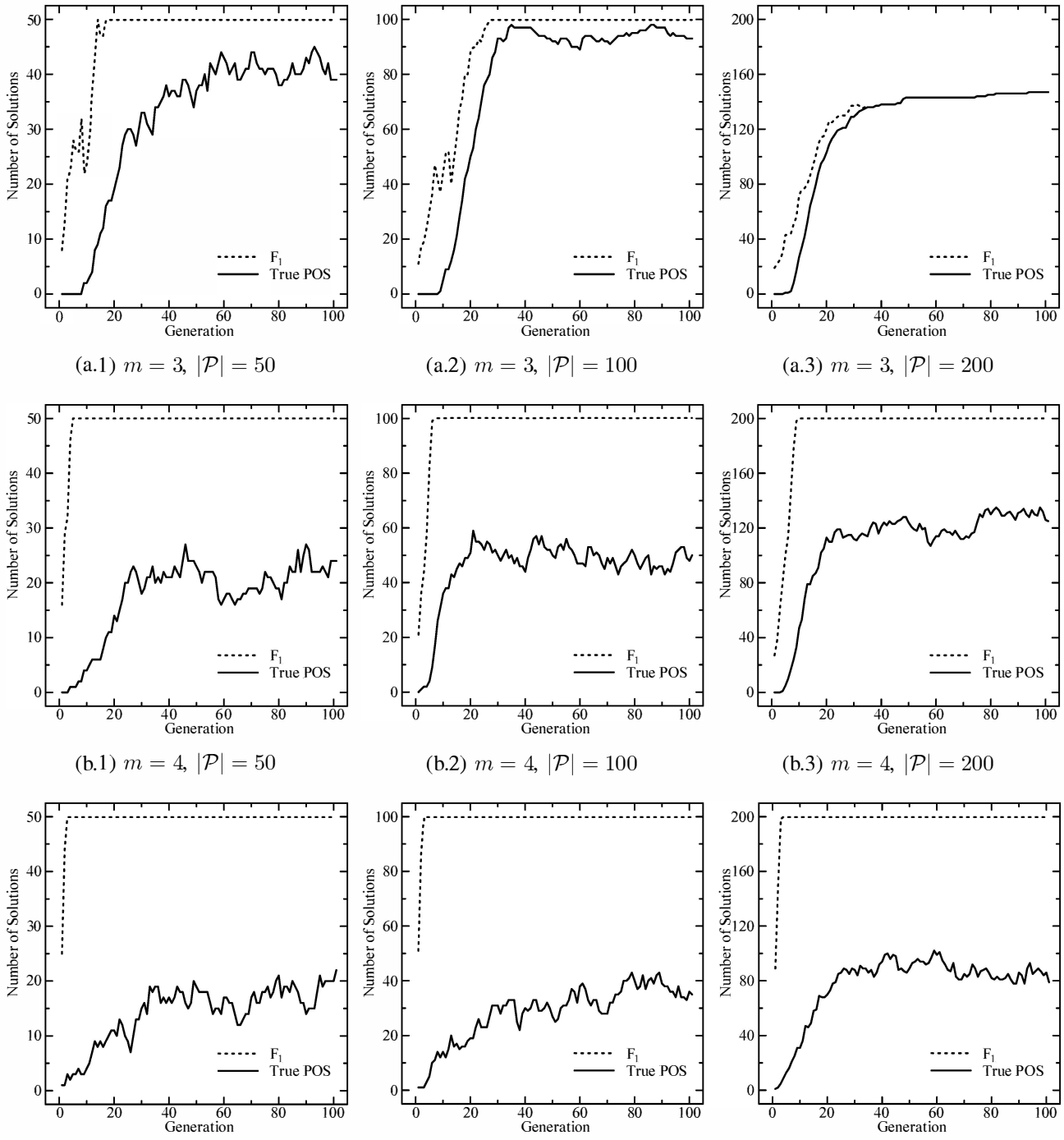
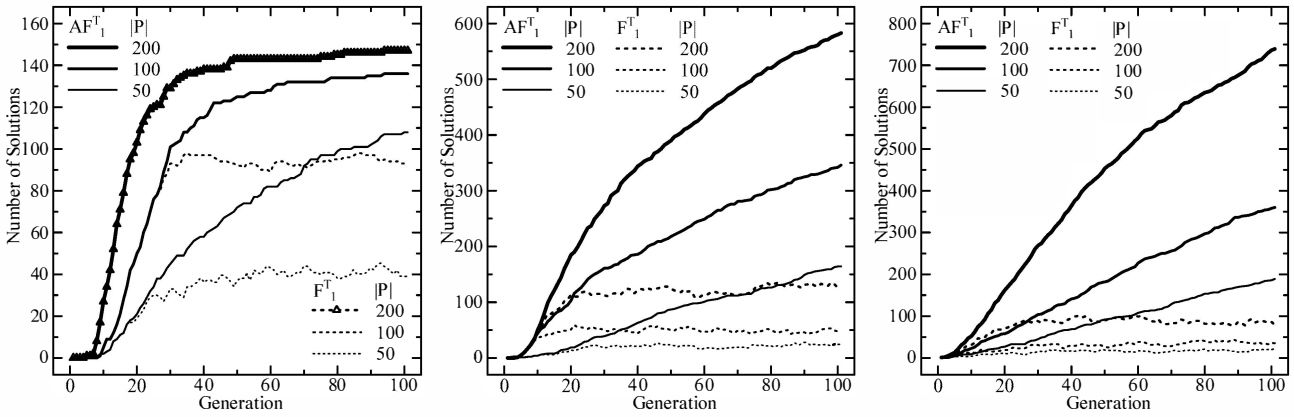


Fig. 1. Results by NSGA-II. Number of non-dominated F_1 and actual number of true Pareto optimal solutions F_1^T in the population over the generations. $|POS^T| = 152$, 1554 and 6265 for $m = 3$, 4 and 5 objectives, respectively.

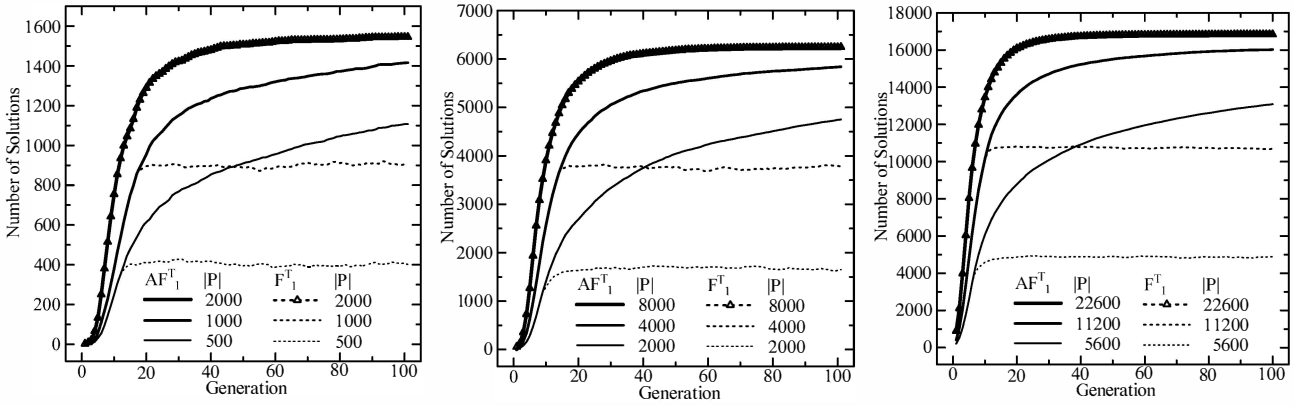
same values used for $m = 3$ objectives, which are very small compared to $|POS^T|$. That is, $|\mathcal{P}| \leq 200 < |POS^T| = 1554$ and $|\mathcal{P}| \leq 200 < |POS^T| = 6265$, respectively. Note that these settings of population size magnify the difficulties observed for $m = 3$ with $|\mathcal{P}| = 50$ or $|\mathcal{P}| = 100$. That is, fewer solutions are true Pareto optimal, although the set of non-dominated solutions of the population quickly contains mutually non-dominated solutions only. Also, larger fluctuations are observed in the number of true Pareto optimal solutions F_1^T.

In general, if $|\mathcal{P}|$ is set to a value smaller than $|POS^T|$, the algorithm cannot keep all true Pareto optimal solutions in the population. However, we would expect an ideal algorithm to keep as many true Pareto optimal solutions as the size of its population, $|F_1^T| = |F_1| = |\mathcal{P}| < |POS^T|$. This is not



(a) $m = 3$, $|POS^T| = 152$ (b) $m = 4$, $|POS^T| = 1554$ (c) $m = 5$, $|POS^T| = 6265$

Fig. 2. Results by NSGA-II increasing the number of objectives keeping the population size small compared to the number of true Pareto Optimal Solutions of the landscape POS^T. Number of true Pareto optimal solutions found at each generation F_1^T and their accumulated number over the generations AF_1^T. Population sizes $|\mathcal{P}| = 50$, 100, and 200 for $m = 3$, 4, and 5 objectives.



(a) $m = 4$, $|POS^T| = 1554$ (b) $m = 5$, $|POS^T| = 6265$ (c) $m = 6$, $|POS^T| = 16845$

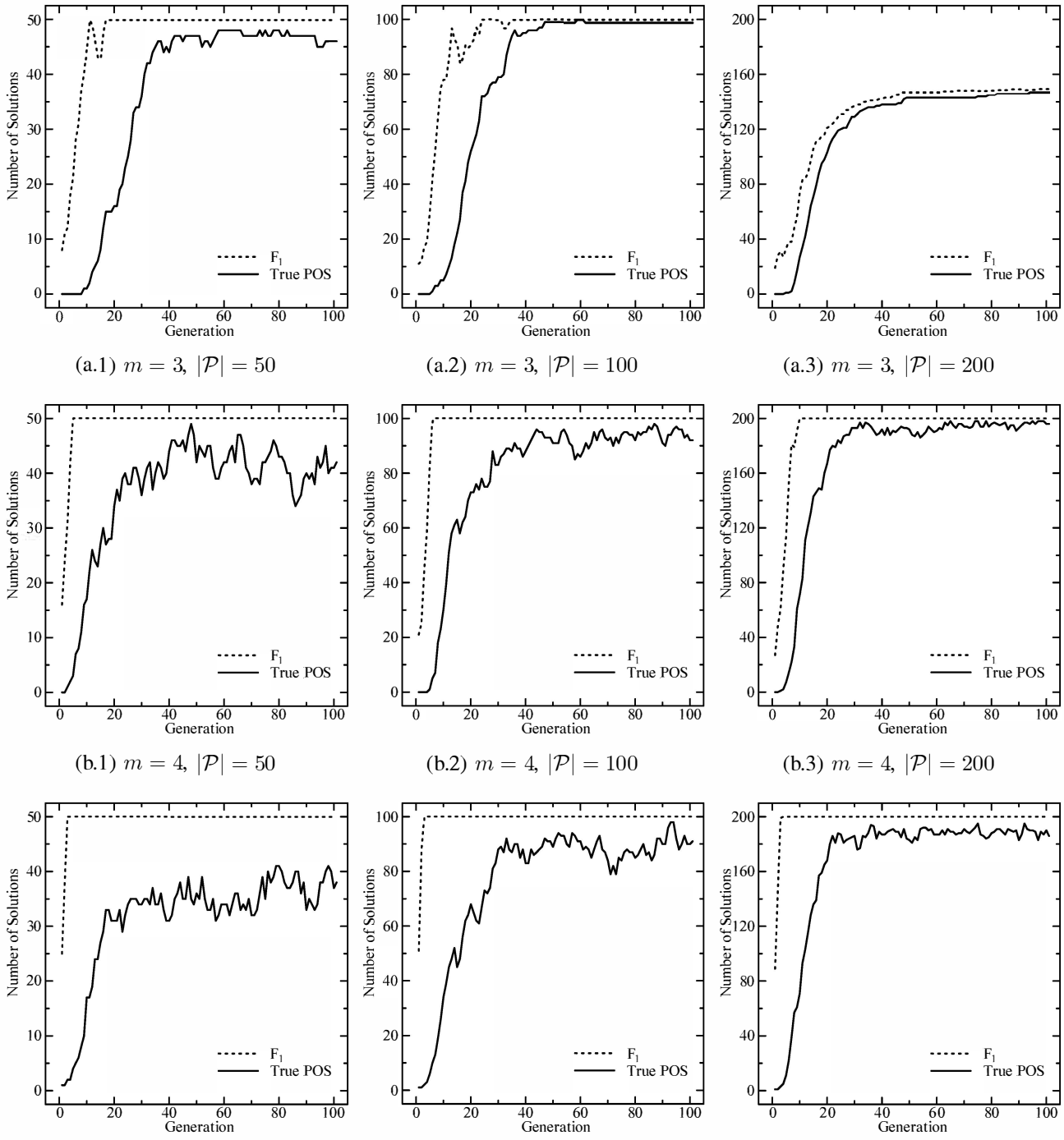
Fig. 3. Results by NSGA-II increasing the number of objectives setting the population size as a fraction of the number of true Pareto Optimal Solutions of the landscape POS^T. Number of true Pareto optimal solutions found at each generation F_1^T and their accumulated number over the generations AF_1^T. Population sizes $1/3$, $2/3$ and $4/3$ of POS^T for $m = 4$, 5, and 6 objectives.

what we observe. When population sizes much smaller than the number of true Pareto optimal solutions are used, it is clear from our results that the capacity of the algorithm to keep true Pareto optimal solutions reduces considerably with the number of objectives, although the size of search space remains fixed and there are many more true Pareto optimal solutions in landscapes with a larger number of objectives.

To explain this behavior, Fig.2 shows the instantaneous number of true Pareto optimal solutions in the population $|F_1^T|$ and its accumulated number $|AF_1^T|$ over the generations for population sizes $|\mathcal{P}| = 50$, 100, and 200. Note that a large number of true Pareto optimal solutions are found by the algorithm. However, not all these solutions remain in the population (except in the case $m = 3$ $|\mathcal{P}| = 200$). Some of these solutions are lost from one generation to the next one during the survival selection step of the algorithm. At this step, the algorithm joins the population $\mathcal{P}$ with the offspring population $\mathcal{Q}$ and ranks individuals with respect to dominance-depth. The best rank is given to true Pareto optimal solutions and also to some others that are not true optimal but appear non-dominated in the combined population. As indicated above, we call the set of best ranked non-dominated solutions obtained from $\mathcal{P} \cup \mathcal{Q}$ as $\mathcal{F}_1$. If this set $\mathcal{F}_1$ is larger than the population $\mathcal{P}$, a sample of them $\mathcal{P} = F^1 \subseteq \mathcal{F}_1$ is chosen based on crowding distance during the survival step. At this point, some true Pareto optimal solutions are dropped in favor of less crowded non-optimal solutions. Summarizing, $\mathcal{P} = F^1 \subset \mathcal{F}_1$ and therefore $F_1^T \subset F_1$ is more likely to occur for population sizes smaller than the number of true Pareto optimal solutions $|POS^T|$.

For a given population size less true Pareto optimal solutions are observed in the instantaneous population $\mathcal{P}$ when we increase the number of objectives. However, the accumulated number of true Pareto optimal solutions is bigger in problems with a larger number of objectives. For example, see Fig.1 (a.2), (b.2), (c.2) and Fig.2 (a)-(c) for $\mathcal{P} = 100$. This suggests that the algorithm finds a larger number of different true Pareto optimal solutions from generation to generation in problems with more objectives, although its ability to retain non-dominated solutions in the instantaneous population reduces.

Fig.2 (a) and Fig.3 (a)-(c) show results for $m = 3$, 4, 5 and 6 objectives using population sizes that correspond approximately to $1/3$, $2/3$ and $4/3$ of the set POS^T, re-



(a.1) $m = 3$, $|\mathcal{P}| = 50$ (a.2) $m = 3$, $|\mathcal{P}| = 100$ (a.3) $m = 3$, $|\mathcal{P}| = 200$

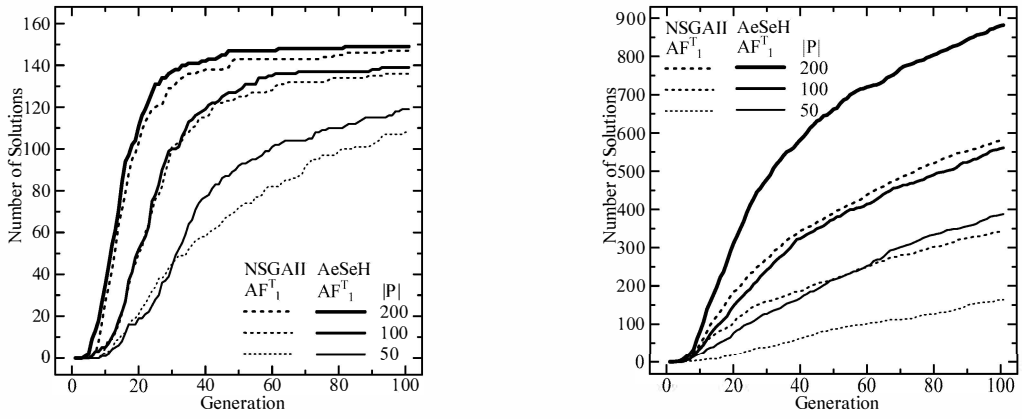
(b.1) $m = 4$, $|\mathcal{P}| = 50$ (b.2) $m = 4$, $|\mathcal{P}| = 100$ (b.3) $m = 4$, $|\mathcal{P}| = 200$

(c.1) $m = 5$, $|\mathcal{P}| = 50$ (c.2) $m = 5$, $|\mathcal{P}| = 100$ (c.3) $m = 5$, $|\mathcal{P}| = 200$

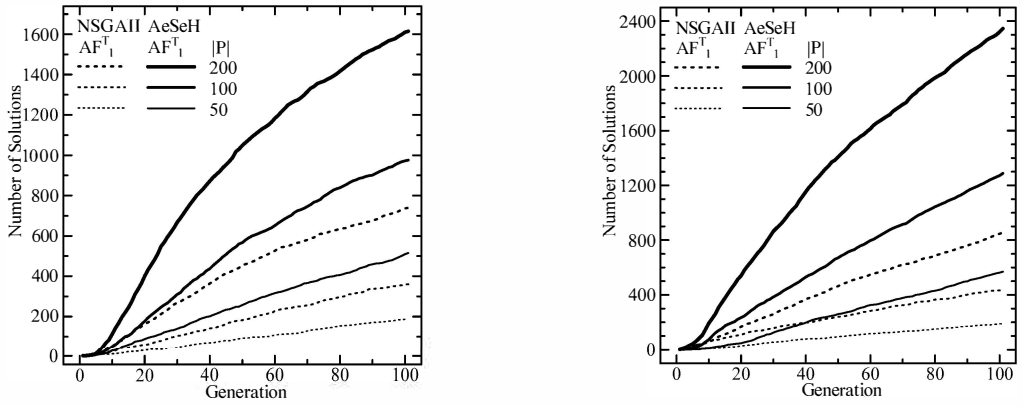
Fig. 4. Results by AεSεH. Number of non-dominated F_1 and actual number of true Pareto optimal solutions F_1^T in the population over the generations. $|POS^T| = 152$, 1554 and 6265 for $m = 3$, 4 and 5 objectives, respectively.

spectively. From these figures note that increasing population size from $1/3$ to $4/3$ of POS^T translates into a striking performance scalability of the algorithm, measured in terms of the number of true Pareto optimal solutions found and kept in the population. For population size $4/3$ of POS^T the number of $AF_1^T = F_1^T \subset F_1$ and the algorithm can actually find and keep in the population 147 out of 152, 1545 out of 1554, 6248 out of 6265, and 16842 out of 16845 true Pareto optimal solutions for 3, 4, 5 and 6 objectives, respectively.

These results show that the effectiveness of the algorithm in many-objective landscapes depends strongly on the size of the population. However, it should be noted that larger populations demand more computational time and memory. Also, a relatively larger number of solutions need to be

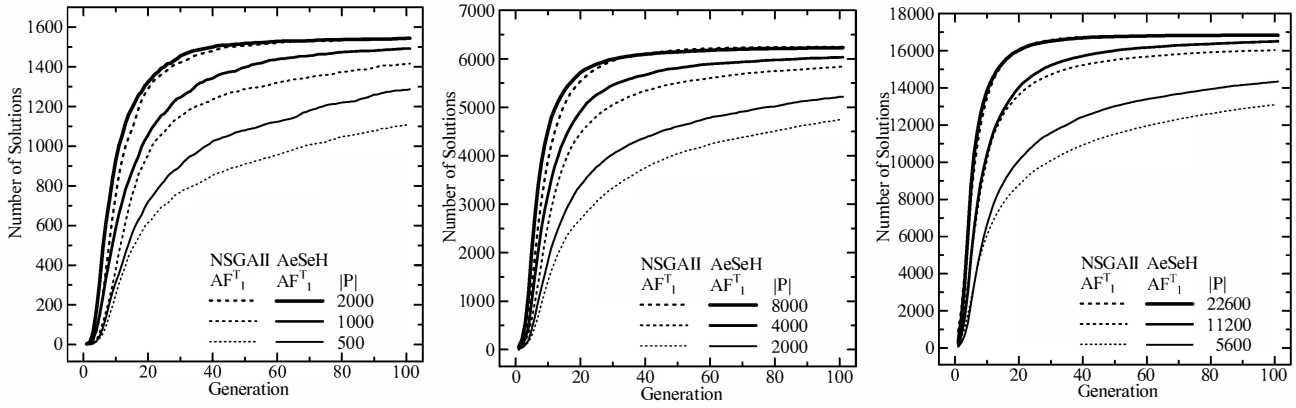


(a) $m = 3$, $|POS^T| = 152$ (b) $m = 4$, $|POS^T| = 1554$



(c) $m = 5$, $|POS^T| = 6265$ (d) $m = 6$, $|POS^T| = 16845$

Fig. 5. Results by AεSεH increasing the number of objectives keeping the population size small compared to the number of true Pareto Optimal Solutions of the landscape POS^T. Number of true Pareto optimal solutions found at each generation F_1^T and their accumulated number over the generations AF_1^T. Population sizes $|\mathcal{P}| = 50$, 100, and 200 for $m = 3$, 4, and 5 objectives. Results by NSGA-II are also included for comparison.



(a) $m = 4$, $|POS^T| = 1554$ (b) $m = 5$, $|POS^T| = 6265$ (c) $m = 6$, $|POS^T| = 16845$

Fig. 6. Results by AεSεH increasing the number of objectives setting the population size as a fraction of the number of true Pareto Optimal Solutions of the landscape POS^T. Number of true Pareto optimal solutions found at each generation F_1^T and their accumulated number over the generations AF_1^T. Population sizes $1/3$, $2/3$ and $4/3$ of POS^T for $m = 4$, 5, and 6 objectives. Results by NSGA-II are also included for comparison.

evaluated for the same number of generations. For example, after 100 generations, using a population size $4/3$ of POS^T, the conventional EMO algorithm used in this study evaluates approximately a number of solutions equivalent to 2%, 19%, 76% and 215% of the size of the search space for $m = 3$, 4, 5, and 6 objectives, respectively. Thus, although a sufficiently large population allows us to find almost all true Pareto optimal solutions, the effectiveness of large populations should be evaluated against a reasonable budget of fitness evaluations.

C. AεSεH

In the following we study the effects of population size on the AϵSϵH algorithm. Fig.4 shows the number of solutions in F_1 and F_1^T over the generations for $m = 3$, 4, and 5 objectives, running AϵSϵH for 100 generations with three different population sizes $|\mathcal{P}| = 50$, 100 and 200. Comparing with Fig.1, it can be seen that AεSεH finds and keeps more true Pareto optimal solutions in the population than NSGA-II. This difference becomes more remarkable for 4 or more objectives in all population sizes. For example, in $m = 4$ and 5 objectives AεSεH keeps in the population approximately twice as many true Pareto solutions than NSGA-II. This shows the advantages of using ε-sampling during the truncation of the population in the many-objective optimizer. However, similar to NSGA-II, note that for the same population size less true Pareto solutions are present in the instantaneous population $\mathcal{P}$ for landscapes with a larger number of objectives.

Fig.5 shows the accumulated number of true Pareto optimal solutions found by AεSεH together with those found by NSGA-II for comparison. Results are shown for $m = 3$, 4, 5, and 6 objectives using population size $|\mathcal{P}| = 50$, 100 and 200. Note that a larger number of true Pareto optimal solutions are accumulated by AεSεH than by NSGA-II. That is, in addition to instantaneously keeping in the population a larger number of true Pareto optimal solutions, the many-objective optimizer is able to find more new solutions from generation to generation, which translates into a larger number of accumulated true Pareto optimal solutions. Note that the difference in the number of accumulated solutions becomes larger with the number of objectives. Also, it is worth noting that for 4 or more objectives the number of accumulated true Pareto optimal solutions found by AεSεH with a population of 50 is larger than the number of accumulated solutions by NSGA-II with a population of 100. Likewise, AεSεH with a population of 100 performs similar or better than NSGA-II with a population of 200.

Fig.5 (a) and Fig.6 (a)-(c) show results by AεSεH and NSGA-II for $m = 3$, 4, 5 and 6 objectives using population sizes that correspond approximately to $1/3$, $2/3$ and $4/3$ of the set of true Pareto optimal solutions POS^T, respectively. From these figures note that when a population larger than $|POS^T|$ is used both algorithms perform similarly in terms of the number of true Pareto optimal solutions found. However, for smaller populations, AεSεH is more effective and efficient, especially for 4 or more objectives. For example, using a population size equivalent to $1/3$ of $|POS^T|$, NSGA-II finds more than 70% of the true Pareto optimal solutions after 100 generations. AεSεH with the same population size can find a similar number of solutions expending only half the number of generations and fitness evaluations. As mentioned above and shown in Fig.5, AεSεH's efficiency becomes more evident when very small population sizes are used.

V. Conclusions

In this work we analyzed the effects of population size on selection and performance scalability of a conventional evolutionary multi-objective algorithm and an evolutionary many-objective algorithm for many-objective optimization, correlating population size to the size of the optimal set. We showed that optimal solutions are dropped from the population when survival selection is applied, especially in small populations. Since both algorithms used in our study are dominance based, this suggests that rather than the inability to distinguish among non-dominated solutions, an important issue for scalability of the algorithm is the distribution of solutions that survival selection renders. The many-objective optimizer uses the ε-sampling procedure, which induces a uniform distribution, and keeps more optimal solutions in the population than the multi-objective optimizer that uses crowding distance with no clear indication of what distribution it produces. Larger populations can cover a broader region in objective space, reduce the effect of dropping optimal solutions, and increase the effectiveness of the optimizers in many-objective problems. In fact, the performance of a conventional evolutionary multi-objective optimizer can scale up fairly well to high dimensional objective spaces if the size of the population is sufficiently large compared to the size of the true Pareto optimal set. If the aim of the optimizer is to find all optimal solutions, our study suggests that population sizes similar or larger than the size of the optimal set might be required. However, this population sizes become impractical for many-objective problems due to the exponentially large number of solutions in the optimal set. If the aim of the algorithm is to find a significant number of optimal solutions under a reasonable budget of fitness evaluations, then more efficient algorithms like the many-objective optimizer used in this study become important. We also showed that the rate of discovering new optimal solutions is higher in problems with more objectives, although the ability to keep optimal solutions in the population reduces with the number of objectives.

As future work, we would like to analyze population size and its relationship to scalability using larger landscapes and other problems. Also, we would like to analyze in a similar manner other multi and many-objective optimizers that use different strategies for survival selection. Particularly, it will be interesting to study the behavior of indicator based algorithms [7].

References

[1] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*, John Wiley & Sons, Chichester, West Sussex, England, 2001.

[2] H. Ishibuchi, N. Tsukamoto, and Y. Nojima, "Evolutionary Many-Objective Optimization: A Short Review", *In Proc. IEEE Congress on Evolutionary Computation (CEC 2008)*, IEEE Press, pp.2424-2431, 2008.

[3] H. Aguirre and K. Tanaka, "Insights on Properties of Multi-objective MNK-Landscapes", *Proc. 2004 IEEE Congress on Evolutionary Computation*, IEEE Service Center, pp.196–203, 2004.

[4] H. Sato, H. Aguirre and K. Tanaka, "Genetic Diversity and Effective Crossover in Evolutionary Many-objective Optimization", *Proc. Learning and Intelligent Optimization Conference (LION 5)*, Lecture Notes in Computer Science (Springer), vol.6683, pp.91-105, Jan. 2011.

[5] H. Aguirre, A. Oyama, and K. Tanaka, "Adaptive ε-Sampling and ε-Hood for Evolutionary Many-Objective Optimization", Evolutionary Multi-criterion Optimization Conference, Lecture Notes in Computer Science, Springer, vol. 7811, 2013, to appear.

[6] K. Deb, S. Agrawal, A. Pratap and T. Meyarivan, "A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II", *KanGAL report 200001*, 2000.

[7] E. Zitzler, S. Kunzli, "Indicator-based Selection in Multiobjective Search", Proc. 8th Int'l Conference on Parallel Problem Solving from Nature - PPSN VIII, Lecture Notes in Computer Science, Springer, vol. 3242, pp. 832?842, 2004.