Accumulative Sampling for Noisy Evolutionary Multi-Objective Optimization

Taejin Park
Pusan National University
Dept. of Computer Engineering
Jangjeon-Dong, San30
Gumjeong-Gu, Busan, Korea
+82-51-510-3531
parktj@pusan.ac.kr

Kwang Ryel Ryu
Pusan National University
Dept. of Computer Engineering
Jangjeon-Dong, San30
Gumjeong-Gu, Busan, Korea
+82-51-510-2453
krryu@pusan.ac.kr

ABSTRACT
Objective evaluation is subject to noise in many real-world problems. The noise can deteriorate the performance of multi-objective evolutionary algorithms, by misleading the population to a local optimum and reducing the convergence rate. This paper proposes three novel noise handling techniques: accumulative sampling, a new ranking method, and a different selection scheme for recombination. The accumulative sampling is basically a kind of dynamic resampling, but it does not explicitly decide the number of samples. Instead, it repeatedly takes additional samples of objectives for the solutions in the archive at every generation, and updates the estimated objectives using all the accumulated samples. The new ranking method combines probabilistic Pareto rank and crowding distance into a single aggregated value to promote the diversity in the archive. Finally, the fitness function and selection method used for recombination are made different from those for the archive to accelerate the convergence rate. Experiments on various benchmark problems have shown that the algorithm adopting all these features performs better than other MOEAs in various performance metrics.

Categories and Subject Descriptors
I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search

General Terms
Algorithms

Keywords
Multi-objective optimization, noisy optimization, evolutionary algorithm, dynamic resampling, probabilistic Pareto ranking

1. INTRODUCTION
Due to their population concept and inherent parallelism, multi-objective evolutionary algorithms (MOEAs) have been used as supporting tools for multi-criteria decision making in various fields, such as manufacturing, finance, logistics, material science and sustainable design. For the last two decades, Pareto-based ranking, diversity preservation, and elitism have been deeply studied, and evolutionary algorithms such as NSGA-II [5] and SPEA2 [18] adopting these schemes have been applied to various real-world multi-objective optimization problems (MOPs). Most of the researches on MOEAs deal with noiseless problems even though we often get uncertain or noisy objective functions for many real-world problems. Noise or uncertainty are caused by various reasons such as measurement error, unknown model parameter, and randomized simulation. The noise brings about various difficulties to MOEAs. Because of the noise, a superior solution may be evaluated worse than its true objectives and thus erroneously eliminated from the population. Similarly, an inferior solution may be evaluated better than its true objectives and is given the opportunity to reproduce. This can mislead the population to a local optimum, and reduce the convergence rate of the population.

For single objective problems, many works have been reported for evolutionary optimization in noisy environments. These works can be divided into four categories: *resampling*, *fitness approximation*, *population sizing*, and *modified selection*. Resampling methods take multiple samples of fitness and average them to reduce the effects of noise. In the simplest method, the number of samples is identical for all the solutions generated, and it is unchanged during the entire evolution. Since the sampling of fitness is usually time-consuming and costly, previous works [1] and [2] proposed methods of adapting the number of samples during the run for a better use of the limited computational time. Fitness approximation methods calculate the fitness of a solution by averaging its neighborhood's fitness, instead of drawing multiple samples of the fitness from the solution [13]. The population sizing method uses a larger population to reduce the influence of noise on an evolutionary process. Since an evolutionary algorithm (EA) samples promising regions of fitness landscape repeatedly, the population usually contains many similar solutions. Therefore, the average behavior of the population is not significantly affected by the noise of each individual solution if the population size is large enough [8][12]. Other works have suggested modified selection methods for noisy problems. For example, [11] proposed a noisy evolutionary strategy called thresholding method, which accepts an offspring only when its fitness is better than that of its parent by at least a predefined threshold.

Relatively a small number of works have been reported for MOPs in noisy environments where the noise blurs dominance relations between solutions or their Pareto ranks. [10] and [16] introduced probabilistic dominance and probabilistic Pareto ranking schemes in order to reduce the error of dominance relations caused by noise. These two works require prior knowledge about the noise model, such as the variance or confidence interval of the noise even though they are usually unknown. Therefore, some studies used resampling methods [4][6] to estimate the expected values and variances of objectives. In addition, [4] used fitness inheritance to reduce the computational burden of resampling, while [6] proposed stochastic and significant dominance schemes for better discrimination among candidate solutions. Another study modified the elite preservation scheme of MOEAs [3]. This work assigns a lifetime to a newly archived solution based on the number of solutions it dominates and reevaluates the expired individuals to decide whether they should be removed from or allowed to continually reside in the archive. This method can reduce the impact of inferior solutions which were erroneously believed to be superior.

In this paper, we propose three key features to be adopted to solve noisy MOPs efficiently. Firstly, we propose an accumulative sampling method. The accumulative sampling is basically a kind of dynamic resampling, which takes multiple samples for each objective of a solution and uses their average as its estimated objective. Although ordinary dynamic resampling methods decide the number of samples to be taken at the time the solution is first generated or evaluated, the accumulative sampling does not do so explicitly. The accumulative sampling takes only a minimum number of samples when a solution is generated, but then takes additional samples at every generation during its stay in the archive, and updates the estimated objectives using both newly and previously sampled values. Secondly, we propose a new ranking scheme which combines probabilistic Pareto rank and crowding distance into a single aggregated value. Probabilistic Pareto ranking can minimize the expected errors between the estimated ranks and real ranks. However, [6] and [15] pointed out that it can result in serious lack of diversity because each solution has a unique real-valued rank and thus tie-breaking by the crowding distance never occurs when selecting solutions for the archive and recombination. The combination of probabilistic Pareto rank and crowding distance helps to avoid the problem of lack of diversity. Finally, we propose to use different fitness functions and selection methods for the archive and recombination in order to accelerate the convergence of the algorithm.

Section 2 describes noisy multi-objective optimization problems. Section 3 gives the details of the proposed methods. Section 4 reports the experimental results comparing the performance of the proposed algorithm with those of other MOEAs. Finally, Section 5 gives some conclusions.

2. NOISY MULTI-OBJECTIVE PROBLEM

Without loss of generality, we can define multi-objective optimization as a minimization problem. Formally

$$\min f(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x})), \tag{1}$$

where **x** is a candidate solution and *m* is the number of objectives to be minimized. Solution **a** *dominates* solution **b** (**a** $\succ$ **b**), if every objective of **a** is less than or equal to that of **b** and at least one objective of **a** is less than that of **b**. This is formally defined as follows:

$$\begin{aligned}\mathbf{a} \succ \mathbf{b}, \text{ if } \forall i \in 1,2,\ldots,m : f_i(\mathbf{a}) \le f_i(\mathbf{b}) \wedge \\ \exists j \in 1,2,\ldots,m : f_j(\mathbf{a}) < f_j(\mathbf{b})\ \ .\end{aligned} \tag{2}$$

A solution is *Pareto optimal* if it is not dominated by any other possible solution. For a Pareto optimal solution, any of its objectives cannot be improved without deteriorating at least one other objective. The set of all possible Pareto optimal solutions is called *Pareto front*. Since an MOP typically has so many solutions in the Pareto front, an MOEA tries to find the best *subset* of the solutions which best represents the Pareto front.

A noisy MOP is defined as follows:

$$\min f(\mathbf{x}) = (E[f_1(\mathbf{x})], E[f_2(\mathbf{x})], \ldots, \mathrm{E}[f_m(\mathbf{x})]). \tag{3}$$

Assuming additive normal noise,

$$f_i(\mathbf{x}) = \hat{f}_i(\mathbf{x}) + N(0, \sigma_i^2), \tag{4}$$

where $\hat{f}_i(\mathbf{x})$ is the noiseless objective function. Since $E[f_i(\mathbf{x})] = \hat{f}_i(\mathbf{x})$, equation (3) implies that the goal of the optimization is to minimize the noiseless objective functions.

For noisy problems, the dominance relation between two solutions cannot be deterministically determined based on their estimated objective values. As a solution, previous works [10], [16], and [7] defined and used *probabilistic dominance* and *significant dominance* schemes. Assuming independence among all the objective functions, solution **a** *probabilistically dominates* solution **b** with a probability of

$$p(\mathbf{a} \succ \mathbf{b}) = \prod_{i=1}^{m} p(f_i(\mathbf{a}) < f_i(\mathbf{b})). \tag{5}$$

For two independent normal random variable *x* and *y*, the probability $p(x < y)$ can be calculated as follows:

$$p(x < y) = p(d > 0) = \Phi\left(\bar{d} / s_d\right) = \int_{-\infty}^{\bar{d}/\sigma_d} f(t)\,dt, \tag{6}$$

where $d = y - x$ is a normal random variable with the mean $\bar{d} = \bar{y} - \bar{x}$ and the variance $s_d^2 = s_x^2 + s_y^2$. Φ is the cumulative distribution function of the standard normal distribution *f*(*t*). However, the assumption of normal distribution is not valid when the sampling size is not large enough. Therefore, in this paper, we calculate the probability using the student *t*-distribution:

$$p(x < y) = p(d > 0) = F\left(\bar{d} / s_d, v\right) = \int_{-\infty}^{\bar{d}/s_d} f(t, v)\,dt, \tag{7}$$

where *F* is the cumulative distribution of the student-*t* distribution *f*(*t*, *v*) with *v* degree of freedom. If it is assumed that the two variables x and y have the same variance and their sample sizes are n_x and n_y, respectively, then s_d and v are calculated as

$$s_d^2 = \frac{(n_x - 1)s_x^2 + (n_y - 1)s_y^2}{n_x + n_y - 2}, \quad (8)$$

$$v = n_x + n_y - 2. \quad (9)$$

If their variances are different, s_d and v are calculated as

$$s_d^2 = \frac{s_x^2}{n_x} + \frac{s_y^2}{n_y}, \quad (10)$$

$$v = \frac{(s_x^2 / n_x + s_y^2 / n_y)}{(s_x^2 / n_x)^2 / (n_x - 1) + (s_y^2 / n_y)^2 / (n_y - 1)}. \quad (11)$$

To discriminate solutions in noisy environments, we can look for significant differences of objectives between two solutions. If every objective of solution **a** is significantly less than that of solution **b**, solution **a** *significantly dominates* solution **b**. A formal definition is given below.

Definition 1: Solution **a** *significantly dominates* solution **b** with a confidence level of approximately $(1 - m\alpha)$ if $p(f_i(\mathbf{a}) < f_i(\mathbf{b})) > (1 - \alpha)$ for each objective $i \in \{1, 2, \ldots m\}$.

3. PROPOSED METHODS

The MOEA proposed in this paper has three novel features to cope with the negative effect of noise: accumulative sampling, probabilistic Pareto rank combined with crowding distance, and a new parent selection scheme to accelerate the convergence.

3.1 Accumulative Sampling

Elitism is one of the most important features of recent MOEAs, such as NSGA-II and SPEA2, with Pareto-based ranking and diversity preservation. The MOEAs preserve some of the non-dominated solutions found throughout evolution in an additional space, named *archive*. While the solutions in the archive will eventually become the final output of the algorithm after its completion, they also guide the evolutionary process during the run of the algorithm by being selected as parents for recombination. Previous work [17] showed that the elitism helps to improve the convergence rate of MOEAs.

For noisy problems, however, the elitism can deteriorate the performance of the algorithms. Once archived, a solution is never reevaluated in ordinary MOEAs. Therefore, if the objectives of an inferior solution are erroneously estimated much better than its true values and selected for the archive, this solution can stay in the archive for many generations and be repeatedly selected as parents. This can mislead the evolution to a wrong direction. In addition, this inferior solution can dominate other superior solutions that should not actually be dominated and better than that solution, and thus it can prevent the superior solutions from being included in the archive. This makes it difficult for the archive to be updated with better solutions and thus decreases the convergence rate of the algorithm. Our preliminary experiments showed that the errors between the sampled and true objectives tend to become larger as the evolution progresses, making the problem worse.

A simple solution to this problem would be not to use elitism. There are two alternatives we can think of. One is not to use the archive like the original NSGA [14], and the other is to select parents from the population of the previous generation instead of the archive. However, these *non-elitistic* methods have to give up the advantages of the elitism. Note that if really good solutions are retained in the archive, they help to accelerate the convergence of the algorithm.

The *forgetting* method does not give up elitism but discards the sampled objectives of the archived solutions and takes new samples of the objectives at every generation. The repeated reevaluation can remove the inferior solutions erroneously included in the archive at the cost of increased number of evaluations at every generation. Previous work [3] used the concept of dominance-dependent lifetime to reduce the number of reevaluations. This method assigns lifetimes to solutions when they are generated, and reevaluates only those whose lifetimes expire. The length of the lifetime of a solution depends on the number of solutions in the archive that it dominates. The more solutions one dominates, the shorter life time it is assigned. These reevaluation-based methods effectively reduce the negative effect of the inferior outliers while taking advantage of the elitism.

What we propose in this paper is an *accumulative sampling* method for noisy MOPs. Similarly to the forgetting methods, this method repeatedly takes samples of the objectives for the solutions in the archive at every generation, but it does not discard the samples of the objectives taken in the previous generations. What it does after sampling is to recalculate the average over not only the objective samples newly taken but also the samples taken in the previous generations for each objective function and to use it as the estimated objective of the solution. By the accumulative sampling, the estimated objectives of a solution in the archive are getting more accurate as generation goes on until it is removed from the archive. Therefore, even if a solution is erroneously included in the archive by being overestimated, it will be removed from the archive in several generations after the estimated objectives become accurate enough. Moreover, if the objectives of a good solution in the archive are evaluated much worse than its true objective values at a generation, the sampled objectives in the prior generations can compensate the error and the solution can stay in the archive.

The accumulative sampling is a kind of dynamic resampling. As mentioned earlier, resampling methods take several samples of objectives and average them to reduce the effect of noise. In a static resampling method, the number of samples is identical for all the individuals and is unchanged during the whole run of the algorithm. For an effective utilization of the limited computational time given for optimization, dynamic resampling adapts the number of samples during the algorithm's run. While other dynamic resampling methods decide the number of samples when the solution is generated or evaluated, the accumulative sampling does not make such an explicit decision. Instead, the number of samples is implicitly decided in proportion to the length of stay of each solution in the archive.

3.2 Combined Ranking

Ordinary MOEAs select solutions for the archive and recombination based on the Pareto rank and crowding distance. Given two solutions, the one with a higher Pareto rank is considered better. If their Pareto ranks are the same, the solution with a larger crowding distance is preferred. The Pareto ranks are calculated based on the Pareto relations among the solutions. The non-dominated solutions are usually assigned the highest rank '1' and the dominated solutions are given lower ranks. For noisy problems, it is difficult to determine the true Pareto rank of a solution due to the uncertainty on the Pareto dominance relations. One of the following methods may be considered as a solution to this problem:

1. Simply ignore the existence of noise and consider the estimated objectives as the true objectives.
2. Calculate Pareto ranks based on significant dominance relations instead of deterministic dominance relations
3. Calculate probabilistic Pareto ranks based on probabilistic dominance relations

The first method considers the estimated objectives as the true objectives, and uses ordinary MOEAs without any change. In this case, the errors between real and estimated Pareto ranks can cause problems. A typical problem is that the selections are governed more by Pareto ranks than they are when there is no noise. When the population gets filled with only non-dominated solutions, as frequently observed in rather early generations of non-noisy MOEAs, the selection for the archive and parents is solely controlled by the crowding distance. However, in presence of noise, the selection is controlled more by the Pareto rank than the crowding distance because the population tends to consist of individuals of erroneous multiple ranks. This causes severe loss of diversity. The second method decides the dominance relation more conservatively. Therefore, most solutions become non-dominated even in early generations and the selections are more dependent on the crowding distances than the Pareto-ranks. While it promotes the diversity of the archive and population, it reduces the convergence rate of the algorithm. The third method can minimize the expected error between the estimated and real ranks. However, previous work [6] and [15] pointed out that the crowding distance is never used for selection because each solution has a unique real-valued rank. Therefore the algorithms using the probabilistic Pareto rank suffer from severe lack of diversity.

We adopt both the second and the third methods with some modifications. Unlike the usual Pareto ranking, the solutions are divided into only two ranks by the significant dominance relations. If a solution $\mathbf{x}_i$ is significantly non-dominated, it is assigned a *significant Pareto rank* of '1' (i.e. $rank_i = 1$). Otherwise, $rank_i = 2$. To avoid significant dominance by outliers we allow only the *reliable* solutions to significantly dominate others. The reliable solution is defined as follows:

Definition 2: Solution $\mathbf{x}$ is *reliable* if the number of samples used to estimate each of its objectives is larger than a certain threshold N_r.

For tie-breaking, we combine the probabilistic Pareto rank and the crowding distance into a single value as follows for use as the fitness $CR(\mathbf{x}_i)$ of a solution $\mathbf{x}_i$:

$$CR(\mathbf{x}_i) = \min(R_{PR}(\mathbf{x}_i), \alpha \cdot R_{CD}(\mathbf{x}_i)), \tag{12}$$

where $R_{PR}(\mathbf{x}_i)$ is the relative rank of solution $\mathbf{x}_i$ after sorting the composite population **CP** according to their probabilistic Pareto ranks. **CP** consists of all the solutions in the archive and the offspring population. $R_{CD}(\mathbf{x}_i)$ is the rank by crowding distance of solution $\mathbf{x}_i$ within **CP**. By crowding distance, the highest (or the minimum) rank '1' is given to the one with the largest distance. Parameter α reflects the importance of crowding distance relative to Pareto rank. Typically, $\alpha \geq 1$ because the Pareto rank is usually considered more important than the crowding distance. The probabilistic Pareto rank $PR(\mathbf{x}_i)$ of solution $\mathbf{x}_i$ is calculated as follows:

$$PR(\mathbf{x}_i) = \sum_{\mathbf{x}_j \in \mathbf{CP}} P(\mathbf{x}_j \succ \mathbf{x}_i). \tag{13}$$

It is a simplified version of the probabilistic Pareto rank of [10] and [4]. The crowding distances are calculated in the same way as that of NSGA-II with one difference. NSGA-II gives an infinite value to the extreme solutions. In addition to that, we give an infinite crowding distance to the *reliable extreme* solutions, whose at least one objective is the minimum of all the reliable solutions of the same rank.

For the archive selection, we use a strategy similar to that of NSGA-II, with the comparison operator ($\prec_n$) modified as follows:

$$\mathbf{x}_i \prec_n \mathbf{x}_j \quad \text{if } (rank_i < rank_j) \text{ or } ((rank_i = rank_j) \text{ and } (fitness_i < fitness_j))$$

where $fitness_i = CR(\mathbf{x}_i)$. After sorting the solutions in **CP** using $\prec_n$, the best N_{arc} solutions are selected for the archive, where N_{arc} denotes the archive size. The measure used for $fitness_i$ becomes different for parent selection as described below.

3.3 Parent Selection

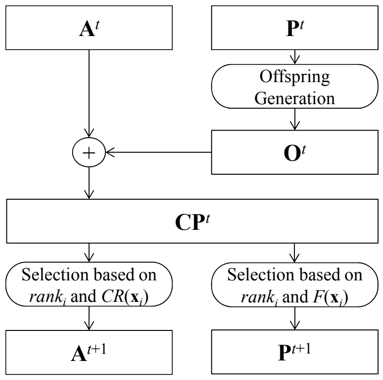
Compared to the scheme using only the probabilistic Pareto rank, the above combined ranking preserves more diversity at the cost of reducing the convergence rate. To accelerate the convergence without losing the advantage of preserving diversity, we use a strategy for the parent selection which is different from that of the archive selection. The following fitness function $F(\mathbf{x}_i)$ instead of $CR(\mathbf{x}_i)$ is used for parent selection:

$$F(\mathbf{x}_i) = \sqrt{n_i} \cdot \sum_{\mathbf{x}_j \in \mathbf{CP}} P(\mathbf{x}_j \succ \mathbf{x}_i) \cdot S(\mathbf{x}_j), \tag{14}$$

where n_i denotes the number of samples used to estimate each objective of $\mathbf{x}_i$, and

$$S(\mathbf{x}_j) = \sum_{\mathbf{x}_k \in \mathbf{CP}} P(\mathbf{x}_j \succ \mathbf{x}_k). \tag{15}$$

This fitness function is quite similar to that of SPEA2 [18] except that it considers the number of samples and is based on the probabilistic dominance. Note that it does not take crowding distance into account but prefers solutions which are generated more recently, i.e., those with smaller n_i. Another difference in our strategy is that, instead of being selected directly from the archive, parents are selected from the parent population whose size is the same as that of the archive. The members of the parent population is selected from **CP** using the same method as that for the archive except that $fitness_i = F(\mathbf{x}_i)$. Note that the members in the parent population do not act as elites. They are simply discarded after the offspring are generated. Figure 1 shows the procedure of generating offspring and selecting solutions for the new archive and parent population. In this figure, $\mathbf{A}^t$, $\mathbf{P}^t$, $\mathbf{O}^t$, and $\mathbf{CP}^t$ denote the archive, parent population, offspring population, and composite population at generation t, respectively.



A^t
P^t
Offspring Generation
O^t
+
CP^t
Selection based on rank_i and CR(x_i)
Selection based on rank_i and F(x_i)
A^{t+1}
P^{t+1}

Figure 1. Procedure for generating offspring and selecting members for the new archive and parent population

4. EXPERIMENTAL RESULTS

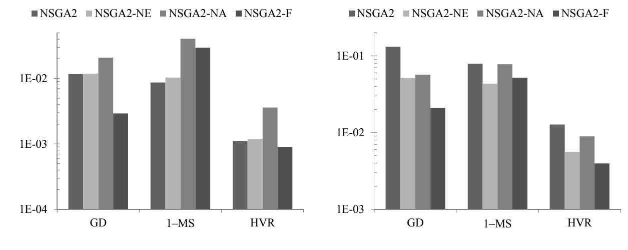
Five well-known benchmark problems of ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6 [17] are used in our experiments. Three levels of additive Gaussian noise are used: low ($\sigma_i = 0.01 \times (F_i^{max} - F_i^{min})$), medium ($\sigma_i = 0.05 \times (F_i^{max} - F_i^{min})$), and high ($\sigma_i = 0.1 \times (F_i^{max} - F_i^{min})$), where F_i^{max} and F_i^{min} are the maximum and minimum of the *i*th objective in the true Pareto front. In our report of results, ZDT*n*L, ZDT*n*M, and ZDT*n*H denote ZDT*n* with low, medium, and high noise level, respectively.

We compared the proposed algorithm (NMOE-AS) with NSGA-II (NSGA2) and its four variants: non-elitistic NSGA-II (NSGA2-NE), NSGA-II without archive (NSGA2-NA), NSGA-II with forgetting (NSGA2-F), and NSGA-II with accumulative sampling (NSGA2-AS). For all the algorithms, both the population size and archive size are set to 100. Simulated binary crossover (SBX) with $\eta_c = 15$, and a polynomial mutation with $\eta_m = 20$ are used. The crossover rate and mutation rate are set to 0.9 and $1/l$, respectively, where l is the length of a chromosome. The sampling size is set to 4 for NSGA2, NSGA2-NE, and NSGA2-NA, but 2 for both the population and archive of NSGA2-F and NMOE-AS. Note that the total number of samples for each objective taken in every generation is the same for all the algorithms.

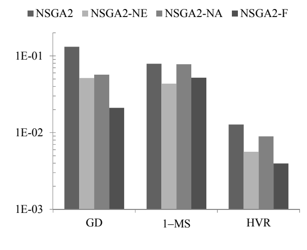
We used three performance metrics: ϒ metric or generational distance (GD) [5], maximum spread (MS) [9], and hyper volume ratio (HVR) [7]. GD and MS are indicators for convergence and diversity, respectively. HVR is used as an overall quality measure. 1,000 uniformly-spaced solutions are taken from the optimal Pareto front and used to calculate the metrics. For each experiment, 30 independent trials were executed and the results were averaged.

Figure 2 compares the results of NSGA2, NSGA2-NE, NSGA2-NA, and NSGA2-F on ZDT1, ZDT2, and ZDT3 with low and high noise levels. In this and the following results, the values of 1 – MS are shown instead of the values of MS. Therefore, smaller values are better for all the metrics. When the noise level is low, NSGA2 shows overall better results than NSGA2-NE and NSGA2-NA, which do not take elitism, though its performance in GD for ZDT3 is slightly worse than that of NSGA2-NE. However, when the noise level is high, NSGA2 is worse than not only NSGA2-NE but also NSGA2-NA, showing the worst performances in all the metrics. Especially, NSGA2 gives much worse results in GD than the other algorithms. This implies that the elitism helps NSGA2 to better solve the problems when the noise level is low, but it hinders rather than helps the convergence of NSGA2 when the noise level is high.

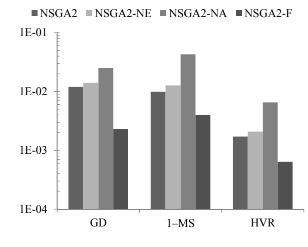
NSGA2-NA shows overall worse results than NSGA2-NE when the noise level is high except for in MS and HVR for ZDT2H, and the worst results when the noise level is low. The absence of archive really hurts the performance. For all the problems and



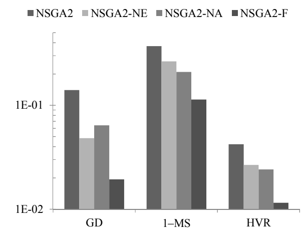
(a)



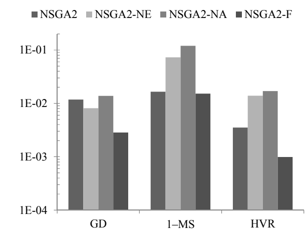
(b)



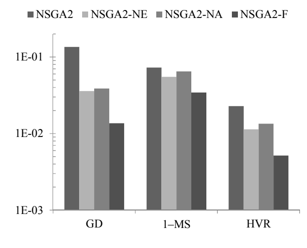
(c)



(d)

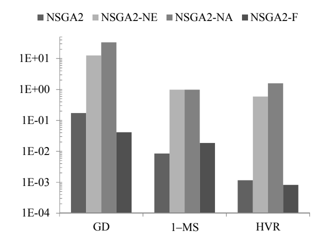


(e)

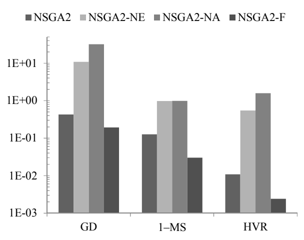


(f)

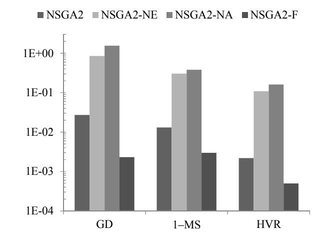
Figure 2. Experimental results of NSGA2, NSGA2-NE, NSGA2-NA, and NSGA2-F on (a) ZDT1L, (b) ZDT1H, (c) ZDT2L, (d) ZDT2H, (e) ZDT3L, and (f) ZDT3H



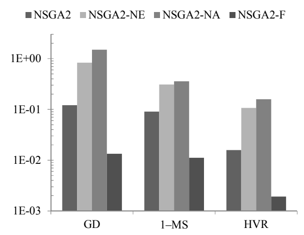
(a)



(b)

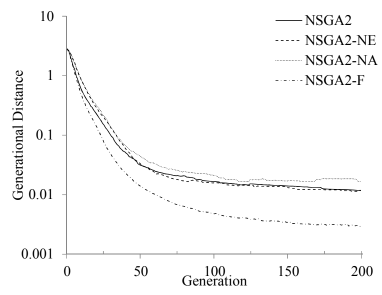


(c)

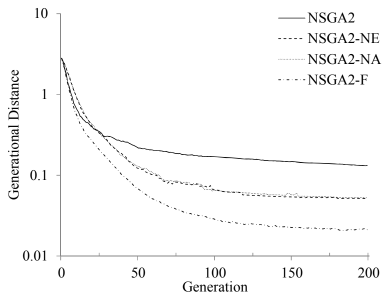


(d)

Figure 3. Experimental results of NSGA2, NSGA2-NE, NSGA2-NA, and NSGA2-F on (a) ZDT4L, (b) ZDT4H, (c) ZDT6L, and ZDT6H



(a)



(b)

Figure 4. Average GD curves of NSGA2, NSGA2-NE, NSGA2-NA, and NSGA2-F for (a) ZDT1L and (b) ZDT1H

noise levels, NSGA2-F gives the best results with only one exception in MS for ZDT1H, for which NSGA2-NE shows the best performance. By the forgetting method, NSGA2-F can take the advantage of the elitism even under the hindrance of noise.

Table 1. Experimental results of NSGA2-F, NSGA2-AS, and NMOE-AS on ZDT1, ZDT2, and ZDT3 with low, medium, and high noise levels

Problem	*Metric*	NSGA2-F	NSGA2-AS	NMOE-AS
ZDT1L	GD	2.92E-03	2.50E-03	**2.48E-03**
	1–MS	2.96E-02	3.54E-02	**1.94E-02**
	HVR	9.01E-04	9.76E-04	**5.33E-04**
ZDT1M	GD	1.17E-02	9.66E-03	**9.32E-03**
	1–MS	4.68E-02	4.45E-02	**2.45E-02**
	HVR	2.56E-03	2.29E-03	**1.66E-03**
ZDT1H	GD	2.11E-02	**1.76E-02**	1.91E-02
	1–MS	5.18E-02	5.45E-02	**3.70E-02**
	HVR	3.97E-03	3.52E-03	**3.02E-03**
ZDT2L	GD	2.29E-03	2.12E-03	**2.08E-03**
	1–MS	3.97E-03	5.13E-03	**2.77E-03**
	HVR	6.39E-04	7.82E-04	**4.82E-04**
ZDT2M	GD	1.04E-02	8.80E-03	**7.89E-03**
	1–MS	1.18E-02	1.04E-02	**8.50E-03**
	HVR	2.02E-03	1.74E-03	**1.49E-03**
ZDT2H	GD	1.94E-02	1.66E-02	**1.60E-02**
	1–MS	1.14E-01	9.15E-02	**1.46E-02**
	HVR	1.15E-02	1.04E-02	**2.67E-03**
ZDT3L	GD	2.85E-03	2.68E-03	**2.44E-03**
	1–MS	1.52E-02	1.44E-02	**6.08E-03**
	HVR	9.88E-04	1.52E-03	**6.29E-04**
ZDT3M	GD	8.47E-03	**7.27E-03**	7.89E-03
	1–MS	1.46E-02	1.69E-02	**1.07E-02**
	HVR	2.29E-03	2.23E-03	**1.92E-03**
ZDT3H	GD	1.36E-02	**1.23E-02**	1.30E-02
	1–MS	3.44E-02	2.95E-02	**1.84E-02**
	HVR	5.15E-03	4.51E-03	**3.52E-03**

Figure 3 shows the results of NSGA2, NSGA2-NE, NSGA2-NA, and NSGA2-F on ZDT4 and ZD6 with low and high noise levels. The results are quite different from those seen before. For these problems with both the noise levels, NSGA2 shows much better performance than NSGA2-NE and NSGA2-NA in all of GD, MS, and HVR. NSGA2-NE and NSGA2-NA show similar results even though the results of NSGA2-NE are slightly better than those of NSGA2-NA for all the problems. Both elitism and the archive seem to be essential to solve these problems efficiently. For all the problems, NSGA2-F is still the clear winner.

Figure 4 compares average GD curves of NSGA2, NSGA2-NE, NSGA2-NA, and NSGA2-F for ZDT1L and ZDT1H. With low noise level, NSGA2 performs a little better in GD than NSGA2-NE and NSGA2-NA up to about generation 50. After then, NSGA2 and NSGA2-NE show similar curves. With high noise level, NSGA2 still shows better GD before generation 25, but become much worse than NSGA2-NE and NSGA2-NA after then. NSGA2-NE and NSGA2-NA show quite similar GD curves with high noise level. NSGA2-F shows the fastest convergence and better GD than the other algorithms with both the noise levels all through the generations. With both the noise levels, NSGA2 and

Table 2. Experimental results of NSGA2-F, NSGA2-AS, and NMOE-AS on ZDT4 and ZDT6 with low, medium, and high noise levels

Problem	*Metric*	NSGA2-F	NSGA2-AS	NMOE-AS
ZDT4L	GD	4.14E-02	8.59E-02	**1.61E-02**
	1–MS	1.85E-02	1.46E-02	**1.20E-02**
	HVR	8.18E-04	8.58E-04	**5.92E-04**
ZDT4M	GD	9.46E-02	**9.22E-02**	1.01E-01
	1–MS	2.94E-02	1.82E-02	**1.10E-02**
	HVR	1.78E-03	1.35E-03	**1.20E-03**
ZDT4H	GD	1.92E-01	**9.50E-02**	2.43E-01
	MS	3.01E-02	3.16E-02	**1.73E-02**
	HVR	2.42E-03	2.58E-03	**2.04E-03**
ZDT6L	GD	**2.32E-03**	1.14E-02	6.48E-03
	1–MS	**2.97E-03**	3.76E-03	3.39E-03
	HVR	**4.97E-04**	6.30E-04	5.58E-04
ZDT6M	GD	**8.40E-03**	2.14E-02	1.65E-02
	1–MS	6.62E-03	9.51E-03	**6.46E-03**
	HVR	1.14E-03	1.60E-03	**1.12E-03**
ZDT6H	GD	**1.34E-02**	1.88E-02	3.07E-02
	1–MS	1.11E-02	1.39E-02	**9.63E-03**
	HVR	1.92E-03	2.39E-03	**1.70E-03**

NSGA2-F show similar convergence rate and GD curves during the first 10 generations, but the convergence rate of NSGA2 gradually becomes slower than that of NSGA2-F after then.

Table 1 and 2 compare NSGA2-F, NSGA2-AS, and NMOE-AS on all the problems with low, medium, and high noise levels. In the tables, the best result for each performance metric is indicated by boldface. For ZDT1–4, NMOE-AS gives the best results in MS and GD metrics. For ZDT1L, ZDT3M, ZDT3H, ZDT4M, and ZDT4H, NSGA2-AS shows the best results in GD metric. Exceptionally for ZDT6, NSGA2-F shows the best results in GD with all the noise levels and in MS and HVR with low noise level.

Figure 6 shows the archive update rate, which is measured by the proportion of the solutions in the archive that are replaced with newly generated ones at each generation, in a typical run on ZDT1H. The update rate of NSGA2 begins at a very low value and remains at near-zero values during the entire generations. The update rate of NSGA2-NE starts at a much higher value of about 0.7, but it gradually decreases and finally stays at near-zero values after about 130 generations. On the other hands, the update rates of NSGA2-F and NMOE-AS stay at relatively high values of around 0.2 all through the generations. They do not show any tendency of decrease in early generations.

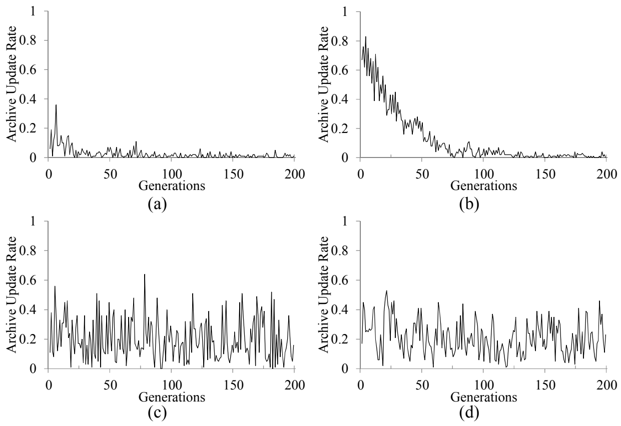


Figure 6. Archive update rate of (a) NSGA2, (b) NSGA2-NE, (c) NSGA2-F, and (d) NMOE-AS

Figure 5 compares the average error between the true and estimated objectives of the solutions in the archive in a typical run on ZDT1H. NSGA2 and NSGA2-NE show similar curves. They begin at around 0.045, which are lower values than those of NSGA2-F, NSGA2-AS, and NMOE-AS, but it gradually increases with generations. Especially, they soar up in very early generations and remain much higher than those of the other three algorithms after 25 generations. This explains why the archive update rates of these algorithms are so low and gradually decrease in early generations. The errors of NSGA2-F, NSGA2-AS, and NMOE-AS do not show any tendency of increase over all the generations. NMOE-AS shows the overall smallest error over the entire generations, and NSGA2-AS marks the second best.

Figure 7 shows the averaege sample size to estimate each objective of the solutions in the archive in a typical run on ZDT1H. Without the accumulative smapling, the average sample sizes of NSGA2 and NSGA-F stay at fixed values of 2 and 4, respectively. Both NSGA2-AS and NMOE-AS show a tendency of increase of their average sample sizes. The average sample size of NSGA2-AS stays relatively constant inbetween 5 and 8 after about 50 generations, but that of NMOE-AS shows a continual increase over the entire generations. Even though the sample size of NSGA2-AS and NMOE-AS for new offsping is set to 2 which is a smaller value than 4 of NSGA2, the average sample sizes of them are maintained larger than that of NSGA2 except for only the earliest few generations. Over all the genrations, the average sample size of NMOE-AS is larger than that of NSGA2-AS. This brings about the smallest error on the sampled objectives of the NMOE-AS and the smaller error of NSGA2-AS than that of NSGA2-F.

Figure 7 reveals an additional advange of the accmulative sampling. In most other dynamic resampling methods, the total

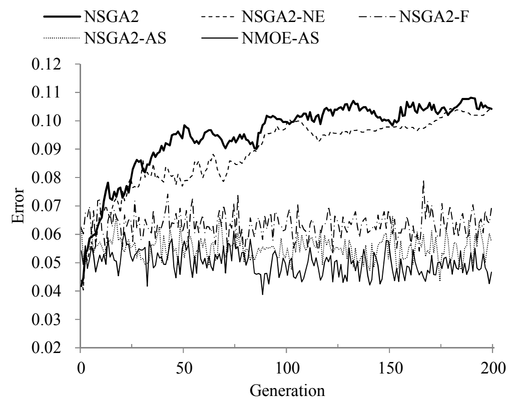


Figure 5. Average error between the true and sampled objectives of the solutions in the archive on ZDT1H

sample size of all the solutions is kept constant although some works proposed methods of controlling the total sample size for each generation. The accumulative sampling does not have an explicit mechanism to control the total sample size, but it is implicitly controlled over genrations as shown in the figure.

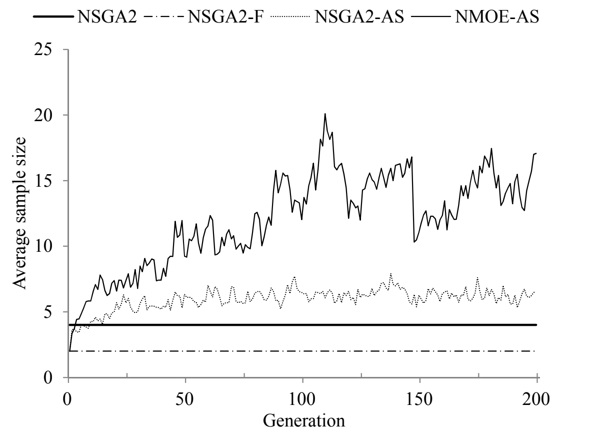


Figure 7. Average sample size to estimate each objective of the solutions in the archive on ZDT1H

5. CONCLUSIONS

We have seen that the elitism of NSGA-II decreases the convergence rate and the performance becomes worse than those of the non-elitistic algorithms for highly noisy problems. We have also seen that forgetting and accumulative sampling are both simple but powerful methods, which can exploit the advantage of elitism without serious reduction of the convergence rate. In this paper, we have proposed the accumulative sampling method as a better option to solve noisy MOPs. The accumulative sampling makes the estimated objectives of the solutions in the archive to become more accurate as generation goes on. We have also proposed a new ranking method which combines probabilistic Pareto rank and crowding distance into an aggregated value. The new method avoids the loss of diversity in the archive by making up the unique value problem of the probabilistic Pareto ranking. We have yet additionally proposed a parent selection strategy which is different from that of the archive selection in order to accelerate the convergence of the algorithm. Experiments using various benchmark problems have shown that the algorithm adopting all these features performs better than NSGA-II, non-elitistic NSGA-II, and the forgetting method in various performance metrics.

6. ACKNOWLEDGMENTS

This work has been co-supported by the research project (Project ID: 10036222) which is funded by Korea Ministry of Knowledge Economy, and the Grant of the Korean Ministry of Education, Science and Technology (The Regional Core Research Program/Institute of Logistics Information Technology).

7. REFERENCES

[1] Aizawa, A.N. and Wah, B.W. Dynamic control of genetic algorithms in a noisy environment. *Proceedings of the 5th International Conference on Genetic Algorithms*, (1993), 48–55.

[2] Aizawa, A.N. and Wah, B.W. Scheduling of Genetic Algorithms in a Noisy Environment. *Evolutionary Computation 2*, 2 (1994), 97–122.

[3] Buche, D., Stoll, P., Dornberger, R., and Koumoutsakos, P. Multiobjective evolutionary algorithm for the optimization of noisy combustion processes. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews) 32*, 4 (2002), 460–473.

[4] Bui, L.T., Abbass, H.A., and Essam, D. Fitness inheritance for noisy evolutionary multi-objective optimization. *Proceedings of the 2005 conference on Genetic and evolutionary computation - GECCO'05*, (2005), 779–785.

[5] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation 6*, 2 (2002), 182–197.

[6] Eskandari, H., Geiger, C.D., and Bird, R. Handling uncertainty in evolutionary multiobjective optimization: SPGA. *IEEE Congress on Evolutionary Computation, 2007. CEC 2007*, (2007), 4130–4137.

[7] Eskandari, H. and Geiger, C.D. Evolutionary multiobjective optimization in noisy problem environments. *Journal of Heuristics 15*, 6 (2009), 559–595.

[8] Fitzpatrick, J.M. and Grefenstette, J.J. Genetic algorithms in noisy environments. *Machine Learning 3*, 2-3 (1988), 101–120.

[9] Goh, C.K. and Tan, K.C. An Investigation on Noisy Environments in Evolutionary Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation 11*, 3 (2007), 354–381.

[10] Hughes, E. Evolutionary multi-objective ranking with uncertainty and noise. *Evolutionary Multi-Criterion Optimization*, Springer (2001), 329–343.

[11] Markon, S., Arnold, D.V., Back, T., Beielstein, T., and Beyer, H.-G. Thresholding-a selection operator for noisy ES. *Proceedings of the 2001 Congress on Evolutionary Computation*, (2001), 465–472.

[12] Rattray, M. and Shapiro, J. Noisy fitness evaluation in genetic algorithms and the dynamics of learning. *Foundations of genetic algorithms 4*, (1996), 117–139.

[13] Sano, Y. and Kita, H. Optimization of noisy fitness functions by means of genetic algorithms using history of search with test of estimation. *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02*, 1 (2002), 360–365.

[14] Srinivas, N. and Deb, K. Muiltiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation 2*, 3 (1994), 221–248.

[15] Syberfeldt, A., Ng, A., John, R.I., and Moore, P. Evolutionary optimisation of noisy multi-objective problems using confidence-based dynamic resampling. *European Journal of Operational Research 204*, 3 (2010), 533–544.

[16] Teich, J. Pareto-front exploration with uncertain objectives. *Evolutionary Multi-Criterion Optimization*, Springer (2001), 314–328.

[17] Zitzler, E., Deb, K., and Thiele, L. Comparison of multiobjective evolutionary algorithms: empirical results. *Evolutionary computation 8*, 2 (2000), 173–95.

[18] Zitzler, E., Laumanns, M., and Thiele, L. SPEA2: Improving the strength Pareto evolutionary algorithm. *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, International Center for Numerical Methods in Engineering (CIMNE) (2002), 95–100.