How Well Do Multi-objective Evolutionary Algorithms Scale to Large Problems

Kata Praditwong and Xin Yao

Abstract— **In spite of large amount of research work in multi-objective evolutionary algorithms, most have evaluated their algorithms on problems with only two to four objectives. Little has been done to understand the performance of the multi-objective evolutionary algorithms on problems with a larger number of objectives. It is unclear whether the conclusions drawn from the experiments on problems with a small number of objectives could be generalised to those with a large number of objectives. In fact, some of our preliminary work [1] has indicated that such generalisation may not be possible. This paper first presents a comprehensive set of experimental studies, which show that the performance of multi-objective evolutionary algorithms, such as NSGA-II and SPEA2, deteriorates substantially as the number of objectives increases. NSGA-II, for example, did not even converge for problems with six or more objectives. This paper analyses why this happens and proposes several new methods to improve the convergence of NSGA-II for problems with a large number of objectives. The proposed methods categorise members of an archive into small groups (non-dominated solutions with or without domination), using dominance relationship between the new and existing members in the archive. New removal strategies are introduced. Our experimental results show that the proposed methods clearly outperform NSGA-II in terms of convergence.**

I. Introduction

Multi-objective problems involve two sets of variables. The first set is a set of decision variables (input of a particular problem) and the second set is a set of objective functions or objective values (output of the problem). Thus, the solutions will be measured in a space of objective functions. One factor that makes multi-objective optimisation very difficult is the number of objectives to optimise. In this work, we consider a large number of objectives optimised as a large problem.

Thus, Deb [2] suggests that the number of objectives makes an impact on the difficulty of the optimisation problems. The dimensions of the objective space vary according to the number of objectives. Moreover, Farina and Amato [3] explained the rapidly growth of a non-dominated space when a number of objectives increases. The factor makes difficulty to an optimiser to find non-dominated solutions close to the Pareto front.

Although many researchers have invented multi-objective evolutionary algorithms (MOEAs), problem generators, and performance measurements, the number of publications related to a large number of objectives is very small. Almost all simulated results focus on two or three objectives [4]. Behaviour of MOEAs solving high-dimensional problems is still under investigation with convergence properties.

Khare et al. [1] have shown that the three algorithms (NSGA-II [5], PESA [6] and SPEA2 [7]) solved problems (DTLZ1, DTLZ2, DTLZ3 and DTLZ6 in [8]) up to 8 objectives. An important comparison in many objectives from [1] is that the scalability of objectives in each algorithm is different. PESA has powerful scalability while NSGA-II has a lack of scalability in terms of convergence. Generally, NSGA-II has performed well in two or three objectives [9] and [10].

Purshouse and Fleming [9] mentioned ineffectiveness of NSGA-II in large numbers of objectives. The results from this study showed that NSGA-II performs very well in small numbers of objectives but its performance is very poor when the number objectives is large. Moreover, the large population size cannot help NSGA-II to escape from poor performance on large numbers of objective problems.

Hughes [10] compared efficiency of two approaches, Pareto based and non-Pareto based ranking. Multiple Single Objective Pareto Sampling (MSOPS) [11] has been used, which is a hybrid method of the aggregation method with evolutionary algorithm. Repeated Single Objective (RSO) [10] is a single objective EA based on the weighted min-max approach. In large number objectives, MSOPS and RSO were better than NSGA-II but in small number of objectives NSGA-II was better than MSOPS and RSO.

Kukkonen and Deb [12] proposed truncating non-dominated solutions in large numbers of objectives problems to improve efficiency of distribution of candidate solution. Farina and Amato [3], [13], and [14] introduced the concepts of fuzzy dominance and fuzzy optimality.

Köppen et al. [15] and [16] implemented fuzzification of the Pareto dominance relation based on the fuzzy ranking scheme [15]. The novel MOEA, Fuzzy-Dominance-Driven GA (FDD-GA) [16] has been evaluated on the Pareto-Box problem up to 15 objectives. FDD-GA outperformed NSGA-II in large numbers of objectives but NSGA-II was superior in a few objectives.

Köppen and Yoshida [17] modified NSGA-II using the new measurement schemes instead of the crowding distance. The new schemes identify that a solution is nearly-dominated by another solution. There are three criteria:

- the number of better objectives;
- the magnitude of all better objectives; or
- a multi-objective approach based on the above criteria.

The comparison was provided up to 15 objectives. The novel NSGA-II using the multi-objective approach was superior to

Kata Praditwong and Xin Yao are with The Centre of Excellence for Research in Computational Intelligence and Applications (CERCIA), School of Computer Science,The University of Birmingham, Edgbaston, Birmingham, B15 2TT, UK, (email:kxp,xin@cs.bham.ac.uk).

1-4244-1340-0/07$25.00

the other two approaches but the multi-objective approach uses more computational effort.

Wagner et al. [4] compared three approaches in MOEAs on problems with many objectives. The first approach is the Pareto-based algorithms such as NSGA-II [5], SPEA2 [7] and ϵ-MOEA [18]. The second approach is the aggregation based methods, e.g. MSOPS [11] and RSO [11]. The last approach is indicator-based MOEA such as indicator-based EA (IBEA) [19] and S-metric Selection-Evolutionary Multi-objective Optimisation Algorithm (SMS-EMOA) [20] and [21]. ϵ-MOEA in Pareto-based approach was found to generate good solution near to the Pareto front but the quality of solutions is depended on the ϵ value. SMS-EMOA was the best algorithm in terms of hypervolume.

Di Pierro et al [22] suggested the preference order-based approach to deal with a problem in a large number of objectives. When all solutions are non-dominated, this method decides which solution is chosen into the next generation. The approach decreases the dimension of the objective space and finds the non-dominated solutions in the reduced space, called 'efficiency of order k', or 'k-optimality'. The non-dominated solution in the largest reduced space is classified as a well solution.

This paper is organised as follows: Section 2 explains why NSGA-II performed poorly for many objectives. Section 3 proposes the improved algorithm to solve the problems from the previous section. Prior to the conclusions of this paper, Section 4 gives results.

II. Why NSGA-II Performed Poorly for Many Objectives

Deb et al. [5] proposed Non-dominated Sorting Genetic Algorithm-II (NSGA-II) as the improved version of NSGA. The process of NSGA-II can be illustrated as following. It starts with initialising population P_t with population size N, and sets t to zero (t is the generation counter). The next step is to evaluate all objectives in each individual. The individual in the population is classified as small groups, called a front ($\mathcal{F}_i$, i = 1, 2, ...). Each front has a unique identifier called a rank. Assigning a rank to each individual is called as the non-dominated sorting. The best rank is one. In each front, each individual is assigned the crowding distance by the crowding sort. The detail of calculating the crowding distance should be explained later. NSGA-II uses crowded tournament selection to choose parent and apply mutation and crossover to generate the offspring, Q_t, with N individuals. The crowded tournament selection should be described later. Each offspring evaluates its own objectives. In each generation, the parent P_t and the offspring Q_t are combined as R_t. Each individual is assigned a rank and a crowded distance by the non-dominated sorting and the crowding sort respectively. The new population P_{t+1} collects members of each front from the first front until it fulls. If the new population has available space less than the remainder member in the front, the members with the longest crowding distance are included. The offspring is generated with the same process as described before. When the last generation finishes, the result is the individuals in P_t.

The crowding sort calculates and assigns a crowding distance to each individual in all fronts. The crowding distance of individual j is the summation of the distances of neighbouring of j in all objectives. All distance values are assigned to zero. For each objective, the members in $\mathcal{F}_i$ are sorted in worse order of the m^{th} objective. The distances of the both ends $\mathcal{F}_i(1)$ and $\mathcal{F}_i(l)$ are infinity. For individual k which is not the both ends, the difference of the objective values of the neighbours ($\mathcal{F}_i(k-1)$ and $\mathcal{F}_i(k+1)$) is added to the distance.

The crowded tournament selection starts with randomly choosing two individuals, i and j. The selection uses a rank as the first priority. It means that the selection always chooses an individual in the better non-dominated front. When two individuals are in the same front, the selection changes to use the crowding distance as a criterion. The individual with the longer distance is chosen as a parent.

The experiment in this section focuses on comparing the number of objectives which are optimised by NSGA-II, to the number of objectives of the test problem. The four scalable problems are DTLZ1, DTLZ2, DTLZ3 and DTLZ6 [8] which are minimisation problems. The number of objective is increased from 4 to 8. The observation is values of each objective in the initial and the final generations. This is difference from other experiment which reports results in terms of a performance metric of the obtained set. The other point is measuring performance between the beginning and the end of searching process.

The experiment in this section conducted NSGA-II with the parameter setting from [1]. The population sizes of 4, 6 and 8 objectives were 100, 250 and 600 respectively. In 4 objectives of DTLZ1 and DTLZ2, there were 300 generations and in that of DTLZ3 and DTLZ6, 500 generations. Furthermore, the number of generations in 6 and 8 objectives was doubled.

The two operators used to create the new offspring were the simulated binary crossover [23] and the polynomial mutation [24]. Selecting the two operators is based on the studies of real-valued genetic algorithm in [24]. Other parameters for NSGA-II are used in this work as the same in [1]. The crossover and mutation probabilites equal 0.7 and 1.0/n respectively, which n is the number of decision variables. The distribution indices for crossover and for mutation are 15 and 20 repectively.

The experiment differs from the work from Khare et al [1]. Firstly, this experiment requires the value of each objective at the beginning and the end rather than the performance metrics of the obtained solutions. Secondly, all test problems have been repeated independently 30 times. In previous work [1], the data in six and eight objectives were averaged only 10 runs. Finally, the statistical data of the comparison have been provided.

The experiment collects the average values of each objective value of solutions in the initial population and the

TABLE I

NSGA-II OPTIMISE PROBLEMS WITH FOUR OBJECTIVES. A TWO-TAILED PAIRED T-TEST WITH 29 DEGREES OF FREEDOM

	Obj 1	Obj 2	Obj 3	Obj 4
DTLZ1				
Start	29.597	31.456	58.057	81.696
Final	0.709	0.882	0.948	0.854
T-Test	**17.467**	**16.538**	**25.232**	**23.105**
DTLZ2				
Start	0.517	0.500	0.733	0.855
Final	0.402	0.407	0.403	0.424
T-Test	**8.455**	**5.441**	**19.978**	**23.426**
DTLZ3				
Start	260.762	249.680	394.222	405.940
Final	8.705	9.532	23.822	31.301
T-Test	**26.277**	**23.077**	**37.462**	**25.963**
DTLZ6				
Start	3.420	3.334	4.698	5.252
Final	1.963	1.890	2.806	2.402
T-Test	**20.815**	**17.875**	**27.327**	**48.570**

TABLE II

NSGA-II OPTIMISE PROBLEMS WITH SIX OBJECTIVES. A TWO-TAILED PAIRED T-TEST WITH 29 DEGREES OF FREEDOM

	Obj 1	Obj 2	Obj 3	Obj 4	Obj 5	Obj 6
DTLZ1						
Start	6.960	7.377	14.151	30.297	66.878	81.954
Final	56.998	62.711	60.945	50.623	23.924	9.116
T-Test	**-9.213**	**-10.241**	**-9.419**	**-3.377**	**7.770**	**16.88841**
DTLZ2						
Start	0.218	0.211	0.350	0.538	0.746	0.833
Final	0.839	0.813	0.794	0.744	0.657	0.474
T-Test	**-38.276**	**-36.501**	**-34.969**	**-12.315**	**4.395**	**17.422**
DTLZ3						
Start	105.543	112.313	169.746	279.882	415.871	393.463
Final	352.681	350.911	349.280	331.228	299.592	138.178
T-Test	**-31.360**	**-29.589**	**-24.030**	**-4.101**	**8.846**	**21.252**
DTLZ6						
Start	1.673	1.684	2.491	3.515	4.675	4.930
Final	2.877	2.873	2.990	3.037	3.300	2.390
T-Test	**-27.083**	**-24.919**	**-12.967**	**8.755**	**30.328**	**33.406**

TABLE III

NSGA-II OPTIMISE PROBLEMS WITH EIGHT OBJECTIVES. A TWO-TAILED PAIRED T-TEST WITH 29 DEGREES OF FREEDOM

	Obj 1	Obj 2	Obj 3	Obj 4	Obj 5	Obj 6	Obj 7	Obj 8
DTLZ1								
Start	1.608	1.755	3.312	7.117	16.025	35.038	65.037	73.920
Final	72.363	71.853	74.430	72.971	72.894	70.230	23.625	10.218
T-Test	**-52.385**	**-58.816**	**-55.251**	**-49.179**	**-33.897**	**-12.516**	**8.136**	**11.995**
DTLZ2								
Start	0.097	0.094	0.151	0.242	0.384	0.574	0.749	0.758
Final	0.880	0.879	0.860	0.873	0.826	0.795	0.740	0.634
T-Test	**-110.538**	**-78.751**	**-67.554**	**-60.537**	**-30.764**	**-17.827**	0.390	**4.8848**
DTLZ3								
Start	42.694	43.760	70.038	113.023	188.532	293.582	390.151	379.992
Final	463.165	449.655	447.713	441.469	407.489	396.428	344.299	194.994
T-Test	**-87.972**	**-74.029**	**-68.255**	**-48.804**	**-28.410**	**-12.127**	**3.320**	**17.999**
DTLZ6								
Start	0.757	0.758	1.130	1.707	2.565	3.657	4.716	4.767
Final	2.665	2.648	2.690	2.688	2.778	2.847	3.049	2.451
T-Test	**-122.371**	**-106.469**	**-75.095**	**-44.950**	**-9.400**	**34.919**	**52.160**	**60.519**

obtained solutions in the final generation. From the tables I to II, the data represents an average of 30 runs for each objective. The bold-face font presents t-test values significant values at 95% confidential level. This experiment uses a paired t-test to compare the value of each objective in the beginning and the final generations. The degrees of freedom of a paired t-test is $n-1$, which n is the number of repeating (in this work $n = 30$, 29 degrees of freedom).

The table I shows the average values of each objective in test problems with 4 objectives. All objective values in the final generation were lower than the values in the start generation. Also, statistical data confirmed that the differences of values in the beginning and the end were significant. That means NSGA-II optimised all objectives in small numbers of objectives.

The table II showed the convergence performance with test problem having 6 objectives. From the table II, the problems were classified into two groups according to the behaviour of NSGA-II solving on them. The first group consisted of three problems, DTLZ1, DTLZ2 and DTLZ3. NSGA-II optimised only the two objectives (objectives 5 and 6) in the first problem group. The second group contained only DTLZ6 that NSGA-II had a good optimisation in the three objectives (objectives 4, 5, and 6). All comparisons were significantly detected by the statistical test.

Also, NSGA-II showed similar behaviour in the 8 objective test problems and the detail showed in the table II. The two objectives (objective 7 and 8) in DTLZ1, DTLZ2 and DTLZ3 were optimised and the three objectives (objective 6, 7, and 8) in DTLZ6 were minimised. All difference values between the beginning and the final generations were statistical significant.

Generally, multi-objective optimisation will deal with conflicting objectives [25]. When one objective is minimising, an other will be increasing. Ideally, an algorithm having a convergence ability optimises all objectives. However, NSGA-II has a limitation in terms of the scalability of objectives. When a problem has 4 conflicting objectives, NSGA-II can optimise all of them. When the numbers of objective are 6 and 8, the algorithm can optimise only less than 4 objectives. It concludes that when a number of objective is increased, NSGA-II optimise a some of them for these testing problems. It means that NSGA-II finds solutions far away from the real Pareto front.

In NSGA-II evolving process, the convergence pressure is used in only the non-dominated sorting because this procedure uses the domination relation in the comparison. After separating a population into small groups, this algorithm assumes that all members in each group are equal in terms of domination relation. Also, the crowding sort function uses only the diversity estimator to decide which member will be deleted. NSGA-II restricts to use the dominated relation to indicate which new solution is non-dominated. The requirement is that no existing member can dominate the new one. However, the other knowledge is that the new solution compares with existing members by dominating them. There are two possible cases. The first case is that new member can dominate any existing member. The second case is that members cannot dominate each other. The quality

of both cases in terms of convergence should be different. NSGA-II concerns only the knowledge to decide the non-dominated solution and omits the other information. The non-dominated solution that can dominate existing members has a useful convergence (as shown in the next section).

III. IMPROVED MOEAS

The main concepts of the proposed methods are separating non-dominated solutions and handling them in different ways. The new method distinguishes some non-dominated solutions from others by comparing a new solution with the existing members. Moreover, the truncation avoids removing the better members of the archive. Thus, the proposed methods differ from NSGA-II in terms of separating the new members into two types and restricting the members that are removed. This research concentrates on the alternative way to preserve the convergence. To make unbiased comparison, the density estimation of the proposed methods is the crowding distance, the same with NSGA-II.

Normally, a population is used to evolve the candidate solutions. However, the proposed methods use the external population, called an archive, to store the non-dominated solutions from beginning of searching process. In these proposed methods, the non-dominated solutions are divided into two types by comparing with the members of the archive. The first type is a non-dominated solution which cannot dominate the archive members called *the non-dominated solution without domination*. The other type is *the solution with domination* which dominates an archive member. The member with domination is essential because it will give a useful convergence. Thus, it can be a member in an archive for the next generation. The members that will be truncated can only be new members without domination and existing members. The number of proposed methods is four from the number of combination of choosing deleted member from a set of members without domination or existing members.

Algorithm 1 Collecting Non-dominated Solutions for Four Algorithms

```
1: for i = 1 to sizeof(archive) do
2:    member(i).Dflag = 2
3: end for
4: for i = 1 to popsize do
5:    if ind(i) is non-dominated solution then
6:       if no member can dominates ind(i) then
7:          Set ind(i).Dflag = 0
8:          if ind(i) dominates a member then
9:             Set ind(i).Dflag = 1
10:            Delete dominated member
11:         end if
12:         Add ind(i) to archive
13:      end if
14:   end if
15: end for
```

This is the design of the four proposed algorithms. Collecting non-dominated solutions as shown in the algorithm 1 is the common part of the four algorithms. The target of this module is to collect the non-dominated solution from the population and deleting the dominated member from the archive. Firstly, the variable Dflag of all members is assigned to two. This means that all members are in the archive before the collection procedure. Next, this procedure gets a solution from the population one by one. The solution compares with the remainder in the population using dominated relationship. If it is a dominated solution, it is discarded. Otherwise, it compares with all members in the archive. If no member in the archive can dominate the new solutions, the Dflag of the new solution is assigned to zero. This means that the new solution becomes a new member in the archive. Next, if the new member can dominate a member in the archive, the dominated member of the archive is deleted and Dflag of the new member is set to one. This means that it is the new member with domination. Finally, the new member is added into the archive. For the removal procedure, each method has its own strategy.

Algorithm 2 Method A

```
1: if sizeof(archive) > capacity then
2:    for i=1 to sizeof(archive) do
3:       if member(i).Dflag == 0 then
4:          Put member(i) into trunc_pop
5:          Delete member(i) from archive
6:       end if
7:    end for
8:    Perform the Crowding-sort on trunc_pop
9:    repeat
10:      Delete member with the shortest distance from
         trunc_pop
11:   until sizeof(archive)+sizeof(trunc_pop) == capacity
12:   Combine trunc_pop to archive
13: end if
```

The removal strategy of method A as explained in the algorithm 2 is executed when the number of members of the archive is more than capacity of the archive. The removal process finds the members with a Dflag variable which contains zero value. This means that the member without domination moves the archive into the temporal population, called trunc_pop, to which crowding distance sorting is applied. The crowding distance sorting calculates the distance for all members of the temporal population. The member with the shortest distance is removed until the total size of the archive and the temporal population equals the capacity. Finally, the remaining member of the temporal population is added into the archive.

The removal strategy of method B as shown in the algorithm 3 is similar to the process of the method A. The difference is separating members from the archive to the temporal population in line 3 in the algorithm 3. In that line, the temporal population consists of two types, the members without domination (the value of Dflag is zero) and the existing members (value of Dflag is two). The main point

Algorithm 3 Method B

```
1: if sizeof(archive) > capacity then
2:   for i=1 to sizeof(archive) do
3:     if (member(i).Dflag == 0) OR (member(i).Dflag == 2) then
4:       Put member(i) into trunc_pop
5:       Delete member(i) from archive
6:     end if
7:   end for
8:   Perform the Crowding-sort trunc_pop
9:   repeat
10:    Delete member with the shortest distance from trunc_pop
11:  until sizeof(archive)+sizeof(trunc_pop) == capacity
12:  Combine trunc_pop with archive
13: end if
```

is to apply crowding distance calculation into different sets. The remainder parts of the method B use the same process in the method A.

Algorithm 4 Method C

```
1: if sizeof(archive) > capacity then
2:   Perform the Crowding-sort on archive
3:   for i=1 to sizeof(archive) do
4:     if member(i).Dflag == 0 then
5:       Put member(i) into trunc_pop
6:       Delete member(i) from archive
7:     end if
8:   end for
9:   repeat
10:    Delete member with the shortest distance from trunc_pop
11:  until sizeof(archive)+sizeof(trunc_pop) == capacity
12:  Combine trunc_pop with archive
13: end if
```

The method C modifies the order of processes in the removal strategy. In the algorithm 4, the removal strategy starts with assigning the crowding distance for all members in the archive. The next step is separating members of the archive to the temporal population. Only members which their Dflag value is zero are moved into the temporal population. The distance of members of temporal population is the distance which is assigned before separating. The distance in the method C is calculated in the members of archive before separating but the distance in the method A is calculated among the members in the temporal population. This is the main difference between the two methods. The deletion uses the shortest crowding distance as criteria. Finally, they combine the archive and the temporal population.

The removal strategy of the method D duplicates the process of the method C and modifies a separating part. The strategy begins with calculating crowding distance for all members of the archive. The modified part is that the temporal population consists of members without domination and existing members. The deleting part and the combining part are the same as in method C. The detail is shown in the algorithm 5.

Algorithm 5 Method D

```
1: if sizeof(archive) > capacity then
2:   Perform the Crowding-sort on archive
3:   for i=1 to sizeof(archive) do
4:     if (member(i).Dflag == 0) OR (member(i).Dflag == 2) then
5:       Put member(i) into trunc_pop
6:       Delete member(i) from archive
7:     end if
8:   end for
9:   repeat
10:    Delete member with the shortest distance from trunc_pop
11:  until sizeof(archive)+sizeof(trunc_pop) == capacity
12:  Combine trunc_pop with archive
13: end if
```

TABLE IV
SOLUTIONS AND THEIRS OBJECTIVE VALUES

Solution	f_1	f_2	Status
J	1.0	6.0	a member in archive
K	3.2	3.7	a member in archive
L	3.5	3.1	a member in archive
M	5.0	2.0	a member in archive
N	3.1	3.8	a member in archive
O	3.4	3.0	a member in archive
U	2.0	6.0	a new solution
V	4.0	2.5	a new solution
W	2.7	4.2	a new solution
X	3.0	3.3	a new solution
Y	3.3	2.7	a new solution
Z	6.0	1.0	a new solution

A. Example

This is an example of collecting and removing members in the archive. The detail is in the table IV. This problem is a bi-objective minimisation. The capacity of the archive, or population size in NSGA-II, is six.

1) NSGA-II: The NSGA-II combines both the existing members and the new solutions. Then, it uses the non-dominated sorting to classify the combined population into fronts. The non-dominated solutions belong to the first front. The members in the first front are J, M, V, W, X, Y, and Z. The number of member in the first front overflows, thus deleting the excessive members is depended upon the crowding distance. The crowding distances of all members are shown in the table V. Finally the solution with the shortest crowding distance is deleted. In this example, the victim is the solution Y.

2) The Four Methods: The four methods use the same collection procedure. In the example, the new solution X dominates K and N, also the solution Y dominates L and O. Thus, the solutions X and Y are the new members with

TABLE V
SOLUTIONS AND THEIRS CROWDING DISTANCE VALUES FOR NSGA-II

Solution	Crowding distance
J	∞
M	3.5
V	2.4
W	4.7
X	2.1
Y	1.8
Z	∞

domination. The solutions V, W and Z are the new member without domination. The new solution U is discarded because it is dominated by the existing member J. The existing members in the archive before the collection process are J and M. Now, the archive has seven candidate members, but its capacity is six. Thus, one member is deleted. Each method has its own finding the victim. The deleting processes are explained in the following:

Method A: The method A starts with keeping the new member with domination (X and Y) and existing members (J and M) in the archive before deleting the further members. The remain members (W, V, and Z) are moved to the temporal set for truncation. The crowding sort function is applied to the members (W, V and Z). The crowding distances of the solutions V, W, and Z are 6.5, infinity and infinity, respectively. The solution V is deleted because it has the shortest distance. The archive collects solutions W and Z.

Method B: The method B collects only the new members with domination (X and Y) into the archive and applies the crowding sort function to the remainder. The temporal set consists of five members (J, M, V, W, and Z). The distances of J, M, V, W, and Z are infinity, 3.5, 4.5, 6.5 and infinity, respectively. The solution M, the member with the shortest distance, is removed. The other members(J, W, V, and Z) are added into the archive.

Method C: Method C calculates the crowding distance to all members. The results are in the table V. Next, the archive gets the new members with domination (X and Y) and the existing members (J and M). The remaining members (V, W, and Z) should be truncated. The distances of solutions V, W and Z are 2.4, 4.7 and infinity. The solution V is deleted because of the shortest distance and the solutions W and Z are combined with the archive.

Method D: The method D assigns the crowding distance to all members as shown in table V. Only the new members with domination (X and Y) are in the archive. The truncation set consists of solutions J, M, V, W, and Z. The solution V with the shortest distance is deleted. The archive collects solutions J, M, W, and Z.

IV. EXPERIMENT AND RESULT

The aim of the experiment is to compare convergence efficiency of the archives of the proposed algorithms with NSGA-II. Firstly, the experiments executed NSGA-II which produced the offspring of every generation as output files (including the initial population and the population in the final generation in the files). The number of test problems (DTLZ1, DTLZ2, DTLZ3 and DTLZ6) was six objectives.

The population size is 250 individuals (the same numbers from the previous experiment) and the generation number is 200. From our experiment, the convergence metric was improved during 200 generations and it was changed a little bit after 200 generations. Other parameters used the same values from the previous experiment. To control other factors, NSGA-II performed and saved the offspring of each generation as files and the proposed ideas used offspring from the files to make their own archives. In other words, all methods are not complete algorithms to solve the problem. They are only archiving algorithms without a generator. Finally, five final archives and the starting archive were measured with the convergence metric and the diversity metrics. The experiment was repeated 30 times for each problem.

The measurement consisted of a metric of convergence, metrics of diversity. The convergence metric is proposed by Deb and Jain [26]. This metric computes the average of the smallest normalised Euclidean distance of all points in the obtained Pareto front to the reference set. In these problems, the Pareto fronts are known.

The concept of diversity metrics [26] is calculating distribution of projection of obtained solutions on an objective axis. The objective axis is divided into small areas according to a number of solutions. The diversity measurement is successful if all small areas have one or more representative points. The number of areas with representative point indicates a quality of diversity metric.

The authors [1] have divided diversity metrics into two cases according to the reference set. Diversity metric1 assumes that the algorithm is able to search to the real Pareto front. The project plan of diversity metric1 is based on the real Pareto front. For diversity metric2, the reference front is an obtained front instead of the real Pareto front.

Table VI displays the average values of the performance metrics of all runs and the table VII shows the statistical test values of differences between the initial populations and the obtained solutions from each algorithm. The values with the bold front face are significantly different at 95% confidential level. A statistical test in this experiment used the paired t-test because the four proposed methods received the input from NSGA-II. Thus, all of them are not independent. From these tables, the performance metrics at starting point were used as the criteria. The algorithms were divided into two groups. The first group contained methods A and C which had the convergence metric and the diversity metric1 better than the metrics of solutions in starting point. In DTLZ2, the statistical tests of the convergence metric were not detected as shown in the table VII. Thus, the methods A and C maintained the convergence in the same level with the starting archive. However, they were poor in the diversity metric2.

The second group had NSGA-II, the methods B and D. This group had the better diversity metric2, but they were poor in the convergence metric and the diversity metric1. The

TABLE VI

AVERAGE VALUES OF PERFORMANCE METRICS IN SIX-OBJECTIVE PROBLEMS

Metric	Problem	Start	NSGA-II	A	B	C	D
Convergence	DTLZ1	234.542	319.594	209.148	338.322	208.243	337.537
Metric	DTLZ2	0.737	1.611	0.753	1.649	0.752	1.653
	DTLZ3	976.288	1139.736	881.633	1196.717	896.215	1197.381
	DTLZ6	9.585	10.048	9.501	10.063	9.534	10.064
Diversity	DTLZ1	0.233	0.202	0.260	0.221	0.267	0.218
Metric1	DTLZ2	0.514	0.491	0.496	0.522	0.512	0.523
	DTLZ3	0.092	0.127	0.169	0.130	0.181	0.131
	DTLZ6	0.134	0.073	0.140	0.077	0.139	0.078
Diversity	DTLZ1	0.372	0.422	0.427	0.440	0.431	0.438
Metric2	DTLZ2	0.691	0.734	0.680	0.762	0.695	0.763
	DTLZ3	0.522	0.551	0.427	0.565	0.452	0.568
	DTLZ6	0.628	0.685	0.612	0.706	0.618	0.712

TABLE VII

STATISTICAL TEST OF PERFORMANCE METRICS IN COMPARISON OF ALL ALGORITHMS AND STARTING POINT. A TWO-TAILED PAIRED T-TEST WITH 29 DEGREES OF FREEDOM, N: NSGA-II

Metric	Problem	Start-N	Start-A	Start-B	Start-C	Start-D
Convergence	DTLZ1	**-17.026**	**6.225**	**-17.764**	**6.877**	**-17.321**
Metric	DTLZ2	**-55.589**	-1.208	**-61.157**	-1.135	**-63.445**
(Minimisation)	DTLZ3	**-19.518**	**8.756**	**-27.121**	**7.759**	**-27.070**
	DTLZ6	**-30.330**	**9.240**	**-34.854**	**7.070**	**-35.281**
Diversity	DTLZ1	**7.365**	**-4.886**	**2.101**	**-6.485**	**2.972**
Metric1	DTLZ2	**6.215**	**4.222**	**-2.177**	0.331	**-2.478**
(Maximisation)	DTLZ3	**-7.298**	**-17.526**	**-8.297**	**-20.140**	**-8.951**
	DTLZ6	**15.490**	-1.689	**15.919**	-1.367	**14.999**
Diversity	DTLZ1	**-14.391**	**-14.584**	**-14.733**	**-13.308**	**-15.060**
Metric2	DTLZ2	**-11.181**	**2.346**	**-18.534**	-0.989	**-19.654**
(Maximisation)	DTLZ3	**-7.292**	**21.804**	**-11.432**	**18.985**	**-11.341**
	DTLZ6	**-25.148**	**6.006**	**-43.113**	**4.042**	**-40.264**

comparison in the second group was significant differences as shown in the statistical data in the table VII.

The statistical comparison of the proposed methods and NSGA-II is shown in table VIII. It is certain that the methods A and C were better than NSGA-II in the convergence metric and the diversity metric1 with a statistical significance. The diversity metric2 of obtained sets of NSGA-II was more desirable than the sets of solutions from the methods A and C. The methods B and D were better than NSGA-II in both the diversity metrics, but they were low quality in the convergence metric.

Moreover, table VIII displays the comparison that made more substantial evidence to compare the two best algorithms (methods A and C). According to the convergence metric, both algorithms were comparable because each algorithm had better result for 2 problems. The statistical values showed that the two algorithms had the same convergence in DTLZ1 and DTLZ2. Thus, the method A was slightly better than the method C. However, the method C was better than the method A in terms of diversity metrics for the four problems.

The experiment shows the evidence that the non-dominated solutions were not equal in the convergence to the Pareto front. The proposed methods attempted to classify the solutions according to quality of the convergence and to maintain them in different ways. The MOEAs have different ways to choose a set of solutions. The archive preserved

TABLE VIII

STATISTICAL TEST OF PERFORMANCE METRICS IN COMPARISON OF THE PROPOSED METHODS WITH NSGA-II(N) AND COMPARISON OF METHODS A AND C. A TWO-TAILED PAIRED T-TEST WITH 29 DEGREES OF FREEDOM

Metric	Problem	N-A	N-B	N-C	N-D	A-C
Convergence	DTLZ1	**15.820**	**-5.876**	**16.126**	**-5.265**	0.557
Metric	DTLZ2	**36.543**	**-12.126**	**37.766**	**-12.728**	0.263
(Minimisation)	DTLZ3	**20.454**	**-19.941**	**19.420**	**-18.683**	**-3.467**
	DTLZ6	**44.303**	**-3.659**	**35.742**	**-3.609**	**-5.274**
Diversity	DTLZ1	**-13.872**	**-6.788**	**-13.623**	**-5.270**	**-2.138**
Metric1	DTLZ2	-1.186	**-16.884**	**-5.011**	**-15.379**	**-11.117**
(Maximisation)	DTLZ3	**-10.606**	**-2.080**	**-13.053**	-1.960	**-5.519**
	DTLZ6	**-18.919**	**-2.088**	**-18.721**	**-2.755**	0.922
Diversity	DTLZ1	-1.371	**-6.223**	-1.670	**-3.614**	-0.975
Metric2	DTLZ2	**11.875**	**-13.758**	**7.825**	**-13.331**	**-8.362**
(Maximisation)	DTLZ3	**35.549**	**-7.710**	**34.020**	**-6.961**	**-9.813**
	DTLZ6	**30.089**	**-14.696**	**29.927**	**-19.317**	**-5.429**

the non-dominated solutions with more convergence such as the methods A and C. This fact made that their own convergences were improved. However, the archive of the methods B and D collect the wrong types. Thus, they found the obtained solutions away from the real Pareto front. In the convergence, the main factor preventing a better convergence was the existing members because they have information accumulated from the beginning of the process.

The diversity metrics closely relate to the truncation method when an archive overflows. A main aim of truncation design is deleting excessive members from the archive. However, the remainder solutions after the truncation operator can represent the non-dominated front as close to the Pareto front and cover the entire front with a well spread. Thus, a latent issue for truncation is to select a representative subset of members in the archive. In the other words, it tries to delete other points near the representative. From the experiment, the best concept of maintaining diversity is to keep well non-dominated solutions (non-dominated solutions with domination and existing members) before truncation (as the method C does). The diversity should be calculated in the whole set of non-dominated solutions. For example, the method A applied crowding-sort to the subset of non-dominated solutions that are collected in this generation. The representative from the truncation of the method A is not useful because some members are discarded. The method C estimated the crowding distance with the entire set of non-dominated solutions. Thus, the diversity values of the method C was better than the diversity of the method A. More investigation is needed to improve the diversity.

V. CONCLUSIONS

Many MOEAs break down in dealing with problems with a large number of objectives. This paper shows how and why this happens using NSGA-II. Based on our experimental study and insight as to why NSGA-II failed, several new improvements were proposed. The new ideas are focused on maintenance of non-dominated solutions in archive. NSGA-II was interesting in how to find non-dominated solution

from population. However, non-dominated solution can be classified into solutions with domination or without domination. Our proposed methods use classification of non-dominated solutions to maintain the archive. The proposed new methods perform significantly better than NSGA-II on problems with many objectives. The new methods classify the non-dominated solutions in the archive into several groups and maintain them in different ways. Non-dominated solutions with domination can become members of the archive immediately. Also, an archive keeps non-dominated solutions without domination and a removal strategy applies if an archive overflows.

However, this paper is only the first step in the investigation of MOEAs on problems with a large number of objectives. The concept of classifying non-dominated solutions in this paper is new. It needs to be evaluated on more test problems. It also needs to be incorporated with other MOEAs other than NSGA-II. This is a simple way to fix NSGA-II. Finding better solutions would be to different algorithms, e.g. Two-Archive algorithm [27] or ϵ-MOEA [18].

REFERENCES

[1] V. Khare, X. Yao, and K. Deb, "Performance Scaling of Multi-objective Evolutionary Algorithms," in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, ser. Lecture Notes in Computer Science, vol. 2632. Faro, Portugal: Springer, April 2003, pp. 376–390.

[2] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. Chichester, UK: John Wiley & Sons, 2001.

[3] M. Farina and P. Amato, "On the Optimal Solution Definition for Many-criteria Optimization Problems," in *Proceedings of the NAFIPS-FLINT International Conference'2002*. Piscataway, New Jersey: IEEE Service Center, June 2002, pp. 233–238.

[4] T. Wagner, N. Beume, and B. Naujoks, "Pareto-, Aggregation-, and Indicator-Based Methods in Many-Objective Optimization," in *Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007*. Matshushima, Japan: Springer. Lecture Notes in Computer Science Vol. 4403, March 2007, pp. 742–756.

[5] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multi-Objective Genetic Algorithm-NSGA-II," *IEEE Transactions On Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[6] D. W. Corne, J. D. Knowles, and M. J. Oates, "The Pareto Envelope-based Selection Algorithm for Multiobjective Optimization," in *Proceedings of the Parallel Problem Solving from Nature VI Conference*, ser. Lecture Note in Computer Science, no. 1917. Paris, France: Springer, 2000, pp. 839–848.

[7] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the Strength Pareto Evolutionary Algorithm," in *EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, K. G. et al., Ed., Athens, Greece, 2002, pp. 95–100.

[8] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable Test Problems for Evolutionary Multi-Objective Optimization," Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Tech. Rep. TIK-Report No.112, July 2001.

[9] R. C. Purshouse and P. J. Fleming, "Evolutionary Multi-Objective Optimisation: An Exploratory Analysis," in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)*, vol. 3. Canberra, Australia: IEEE Press, December 2003, pp. 2066–2073.

[10] E. J. Hughes, "Evolutionary Many-Objective Optimisation: Many Once or One Many?" in *2005 IEEE Congress on Evolutionary Computation (CEC'2005)*, vol. 1. Edinburgh, Scotland: IEEE Service Center, September 2005, pp. 222–227.

[11] ——, "Multiple Single Objective Pareto Sampling," in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)*, vol. 4. Canberra, Australia: IEEE Press, December 2003, pp. 2678–2684.

[12] S. Kukkonen and K. Deb, "A Fast and Effective Method for Pruning of Non-dominated Solutions in Many-Objective Problems," in *Parallel Problem Solving from Nature - PPSN IX, 9th International Conference*, ser. Lecture Notes in Computer Science, no. 4193. Reykjavik, Iceland: Springer-Verlag, September 2006, pp. 553–562.

[13] M. Farina and P. Amato, "Fuzzy Optimality and Evolutionary Multiobjective Optimization," in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. Faro, Portugal: Springer. Lecture Notes in Computer Science. Volume 2632, April 2003, pp. 58–72.

[14] ——, "A fuzzy definition of "optimality" for many-criteria optimization problems," *IEEE Transactions on Systems, Man, and Cybernetics Part A—Systems and Humans*, vol. 34, no. 3, pp. 315–326, May 2004.

[15] M. Köppen and R. Vicente-Garcia, " A Fuzzy Scheme for the Ranking of Multivariate Data and its Application," in *Fuzzy Information, 2004. Processing NAFIPS '04. IEEE Annual Meeting of the*, June 2004, pp. 140–145.

[16] M. Köppen, R. Vicente-Garcia, and B. Nickolay, "Fuzzy-Pareto-Dominance and Its Application in Evolutionary Multi-objective Optimization," in *Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005*. Guanajuato, México: Springer. Lecture Notes in Computer Science Vol. 3410, March 2005, pp. 399–412.

[17] M. Köppen and K. Yoshida, "Substitute Distance Assignments in NSGA-II for Handling Many-Objective Optimization Problems," in *Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007*. Matshushima, Japan: Springer. Lecture Notes in Computer Science Vol. 4403, March 2007, pp. 727–741.

[18] K. Deb, M. Mohan, and S. Mishra, "Evaluating the ϵ-Domination Based Multi-Objective Evolutionary Algorithm for a Quick Computation of Pareto-Optimal Solutions," *Evolutionary Computation*, vol. 13, no. 4, pp. 501–525, Winter 2005.

[19] E. Zitzler and S. Künzli, "Indicator-based Selection in Multiobjective Search," in *Parallel Problem Solving from Nature - PPSN VIII*. Birmingham, UK: Springer-Verlag. Lecture Notes in Computer Science Vol. 3242, September 2004, pp. 832–842.

[20] M. Emmerich, N. Beume, and B. Naujoks, "An EMO Algorithm Using the Hypervolume Measure as Selection Criterion," in *Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005*. Guanajuato, México: Springer. Lecture Notes in Computer Science Vol. 3410, March 2005, pp. 62–76.

[21] B. Naujoks, N. Beume, and M. Emmerich, "Multi-objective Optimization using S-metric Selection: Application to three-dimensional Solution Spaces," in *2005 IEEE Congress on Evolutionary Computation (CEC'2005)*, vol. 2. Edinburgh, Scotland: IEEE Service Center, September 2005, pp. 1282–1289.

[22] F. di Pierro, S.-T. Khu, and D. A. Savi, "An Investigation on Preference Order Ranking Scheme for Multiobjective Evolutionary Optimization," *IEEE Transactions On Evolutionary Computation*, vol. 11, no. 1, pp. 17–45, 2007.

[23] K. Deb and A. Kumar, "Real-coded Genetic Algorithms with Simulated Binary Crossover: Studies on Multimodal and Multiobjective Problems," *Complex Systems*, vol. 9, pp. 431–454, 1995.

[24] K. Deb and M. Goyal, "A Combined Genetic Adaptive Search (GeneAS) for Engineering Design," *Computer Science and Informatics*, vol. 26, no. 4, pp. 30–45, 1996.

[25] R. C. Purshouse and P. J. Fleming, "Conflict, Harmony, and Independence: Relationships in Evolutionary Multi-criterion Optimisation," in *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*. Faro, Portugal: Springer. Lecture Notes in Computer Science. Volume 2632, April 2003, pp. 16–30.

[26] K. Deb and S. Jain, "Running Performance Metrics For Evolutionary Multi-Objective Optimization," Indian Institute of Technology, Kanpur, India, Tech. Rep. KanGAL Report Number 2002004, May 2002.

[27] K. Praditwong and X. Yao, "A New Multi-objective Evolutionary Optimisation Algorithm: The Two-Archive Algorithm," in *Proceedings of the 2006 International Conference on Computational Intelligence and Security*, Y.-M. Cheung, Y. Wang, and H. Liu, Eds., vol. 1, Guangzhou, China, 2006, pp. 286–291.