2024 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE) | 979-8-3503-7162-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/CCECE59415.2024.10667327

Exploring Long-term Memory in Evolutionary Multi-objective Algorithms: A Case Study with NSGA-III

Masoud Kermani Poor*, Shahryar Rahnamayan†, Azam Asilian Bidgoli‡, Mehran Ebrahimi*
* Faculty of Science, University of Ontario Institute of Technology, Oshawa, ON, Canada
Email: masoud.kermanipoor@ontariotechu.net, mehran.ebrahimi@ontariotechu.ca
† Engineering Department, Brock University, St. Catharines, ON, Canada
Email: srahnamayan@brocku.ca
‡ Faculty of Science, Wilfrid Laurier University, Waterloo, ON, Canada
Email: aasilianbidgoli@wlu.ca

***Abstract*—In the field of many-objective optimization, obtaining a dense solution set is a challenging task, mostly due to having hyper-surface nature of Pareto-front; which cannot be covered by commonly utilized population sizes. This is particularly vital in scenarios where innovization and informed decision-making are crucial. The challenge stems from the constraints imposed by population size limitations in evolutionary algorithms, which impede the efficient exploration of multiple solutions. A contributing factor to this issue is the lack of long-term memory in the well-known evolutionary algorithms to retain these solutions. On the contrary, the effective training of machine learning-assisted optimization or innovization relies on a substantial amount of data, which can be provided by preserving these valuable solutions. Moreover, long-term memory can play a significant role in expensive many-objective optimization, where the repetition of the optimization process is both costly and time-consuming, similar to training deep neural networks. The study focuses on NSGA-III equipped with long-term memory and assessing its performance across 16 benchmark problems, encompassing DTLZ1 to DTLZ7 and WFG1 to WFG9, considering scenarios with 3, 5, and 10 objectives. This paper explores the benefits of incorporating long-term memory in terms of the ultimate optimization outcomes, including the number of non-dominated solutions, knee points, and Inverted Generational Distance (IGD).**

***Index Terms*—Many-objective optimization, Archive, Long-term memory, NSGA-III, Knee points, Pareto-front coverage, IGD**

I. Introduction

In order to effectively handle the complexity of real-world problems, many-objective optimizers (MaOO) and multi-objective optimizers (MOO) have become essential tools [1]. The adaptability of these algorithms has inspired great research interest, many published papers examining their wide applications across various fields [2]. Although the final solution sets generated by MOO and MaOO have been directly applied in domains like finance, engineering, and decision-making [3], [4], there is a vast opportunity to obtain priceless insights from the disposed non-dominated solutions produced by these algorithms. The collection of all generated non-dominated solutions during the optimization process could be a knowledge source that improves our understanding of issue environments and, as a result, encourages the development of better solutions. However, due to the limitation on population size of such algorithms, a subset of these solutions is survived, and the rest are eliminated.

In the context of utilization of a large set of non-dominated solutions, one of the innovative concepts by Deb is ”innovization,” which emphasizes the value of non-dominated solutions for obtaining deep insights [5]. Other researchers’ inquiries have been inspired by Deb’s work, and as a result, tactics that leverage the information embedded in non-dominated solution sets have been developed [6]. Bandaru’s work is a noteworthy example, since it demonstrates the adaptability of knowledge integration through the automation of the innovating process in genetic algorithms [7]. These initiatives underline the growing understanding of the untapped potential inherent within the non-dominated solutions created by MOO and MaOO algorithms. Utilizing any innovization algorithm during or after optimization can benefit from having a comprehensive non-dominated set of solutions.

Additionally, efforts have been focused on integrating machine learning approaches to improve the efficiency of MOO and MaOO algorithms. Bidgoli et al. used a multi-layer perceptron and several kinds of sampling strategies in a notable work to increase the number of non-dominated solution to tackle sparsity of Pareto-front in many-objective optimization [8]. The incorporation of machine learning creates opportunities for more resolution of optimization problems without increasing the size of population. Evolutionary algorithms, additionally, have embraced machine learning approaches such as Gaussian models to improve their performance measurements, with gains reported in performance metrics like Inverted Generational Distance (IGD) [9].

A common thread across this research landscape is the seamless integration of data mining and machine learning approaches to enhance the efficacy of MOO and MaOO algorithms. Expanding upon this framework, our research presents a new avenue of investigation by examining the integration of a long-term archive memory into evolutionary algorithms. This

method can continuously feed rich datasets to data mining and machine learning techniques in an effort to assess any improvements it might make to the innovization or machine learning assisted optimization methods, especially when they are tackling expensive optimization processes.

This study, investigate the concept of utilizing an archive memory to remain all non-dominated solutions during the optimization process. Meanwhile, there are other studies that utilize the archive to enhance the performance of their algorithm. A limited-size archive memory was established in earlier research to facilitate knowledge-based reinitialization processes [10]. Through the use of reinitialization during the optimization process, this intentional inclusion attempted to maintain population diversity. In order to improve the selection of peers for population crossover, a limited archive memory was introduced in another study [11]. In this approach, peers were deliberately chosen from the archive memory, offering a broad array of possibilities that led to enhanced performance when compared to the NSGA-II algorithm.

To the best of our knowledge, no thorough investigation has been conducted on the possible advantages of integrating an infinite long-term archive memory, despite the fact that previous research has looked at the usage of archive memory to improve the performance of multi-objective algorithms that are already well-established. By exploring the unexplored possibilities of a long-term archival memory in the framework of population-based algorithms, this work seeks to close this gap. Our research is focused on determining how it might affect the cover of the Pareto-front and enhance the MOO evaluation measures for an algorithm.

II. Background Review

A. Many-objective optimization

The process of simultaneously optimizing multiple conflicting objectives—which are frequently encountered in various real-world problems—is known as multi-objective optimization. The process entails identifying a set of solutions, referred to as the Pareto-front set, where achieving one goal would cause at least one other goal to deteriorate [12]. Because it offers a variety of options that allow decision-makers to weigh trade-offs between competing objectives and make well-informed decisions based on their preferences, multi-objective optimization has grown in popularity [13]. Equation (1) provides a definition for a multi-objective optimization problem (MOP):

$$\begin{aligned} &minimize\ F(X) = (f_1(x), f_2(x), ..., f_m(x))^T \\ &subject\ to\ x \in \Omega \end{aligned} \tag{1}$$

When vector $x = (x_1, x_2, \ldots, x_D)$ is defined in a nonempty decision space denoted by Ω, the objective function $F : \Omega \rightarrow$ $\wedge$ vector is composed of $M(M \geq 2)$ objectives, and the objective space $\wedge$ typically corresponds to $S \subset \mathbb{R}^M$.

The number of objectives being optimized is what separates many-objective optimization from multi-objective optimization. Many-objective optimization involves a larger number of objectives, usually four or more, whereas multi-objective optimization deals with small to moderate numbers of objectives $(2 \leq M \leq 3)$ [14]. Due to the growing complexity of the solution space and the difficulties in visualizing and interpreting the Pareto-optimal front, many-objective optimization poses additional challenges [15]. Because of this, specific methods and algorithms have been created to deal with the particular difficulties associated with many-objective optimization issues.

B. Non-dominated sorting (NDS)

An essential concept for solving multi-objective optimization issues is the non-dominated sorting algorithm, which is especially useful when considering evolutionary algorithms. When choosing evolutionary algorithms to solve MOO, MaOO problems, it is frequently utilized [16]. In Pareto-based multi-objective evolutionary algorithms, the non-dominated sorting algorithm is an essential component [17]. Finding effective solutions in many-objective evolutionary algorithms requires it [18].The main foundation concept behind the non-dominated sorting, the dominated concept which can be define as follows. If $x = (x_1, x_2, \ldots, x_D)$ and $x' = (x'_1, x'_2, \ldots, x'_D)$ are two vectors in decision space (Ω) in a minimization problem, then x dominates $\hat{x}(x > \hat{x})$ if and only if

$$\begin{aligned} &\forall\ i \in \{1, 2, ..., M\}, f_i(x) \leq f_i(x') \wedge \\ &\exists\ j \in \{1, 2, ..., M\} : f_i(x) < f_j(x') \end{aligned} \tag{2}$$

Based on the dominant concept the non-dominated solutions of population can be found, these solutions usually called Pareto-front or first front. By removing the non-dominated solution from the population and finding the non-dominated solutions from the changed population we can reach to the second front. By repeating this process, the population can be divided to different groups which can dominate each other.

III. Investigating on Utilization of Long-term Memory

Evolutionary algorithms are commonly utilized in real-world optimization and search problems. They are composed of several key operators that allow for the iterative improvement of potential solutions. The algorithm generates the first population of candidate solutions during the initialization phase. This group of individuals is where the evolutionary process begins. The crossover operator then steps in when two members of the original population are chosen to undergo genetic recombination. Through this procedure, new offsprings are created, inheriting characteristics from both parents. The mutation operator, which gives the offspring little, random alterations, adds even more genetic variation as a background genetic operation. These altered candidate solutions form a heterogeneous pool that is assessed to determine which candidates show the greatest promise.

After the evaluation, the survival operator is essential in figuring out who will make up the following generation. It evaluates the fitness of the parent population as well as the recently produced offsprings, choosing individuals according

to their qualities. The NDS process involves sorting the entire population of solutions into different levels or fronts. The first front consists of solutions that are not dominated by any other solution in the population. These are the best solutions available. The second front consists of solutions that are only dominated by those in the first front, and so on. This sorting allows for the identification of a hierarchy of solutions from most to least optimal, based on their performance across multiple objectives. Due to the limited population size in evolutionary algorithm, a set of high-ranked solutions in the size of the population should be selected to proceed. The selection for the next generation's population starts by choosing individuals from the fronts, starting from the first front and moving towards higher ones. The selection from each front is based on both the rank and the crowding distance (with larger distances being preferred). The best fit candidate solutions—those that best satisfy the optimization criteria—are advanced to the following iteration due to this selection procedure. Evolutionary algorithms gradually refine the solution space by using the initialization, crossover, mutation, and survival operators cyclically. This leads to the convergence of the algorithms towards optimal or nearly optimal solutions for the given issue.

In cases where the first front (the Pareto front) is overcrowded, containing many non-dominated solutions, the algorithm selects a subset of individuals by assigning them to the reference points, while the rest are discarded. However, these candidate solutions might represent critical points on the Pareto front that are essential not only for covering the entire front but also for being utilized in decision-making procedures. To address this, our proposed long-term memory preserve those non-dominated candidate solutions in an archive, as depicted in Algorithm 1. Therefore, in addition to the evolving original population, a separate set of disposed data is stored in long-term memory. In order to prevent the explosion of these points in the archive and properly managing memory utilization, a mechanism is introduced to periodically remove dominated solutions from the archive after specified iterations, the $clean\_archive\_interval$ parameter. By proactively deleting data, this approach maximizes the utilization of memory resources, freeing up memory for storing important information relevant to future generations and enabling the algorithm's ongoing evolution and improvement over time, as shown in Algorithm 2.

When evolutionary algorithms incorporate archive memory, it increases their ability to hold more non-dominated solutions without requiring extra fitness function calls. Storing all non-dominated solutions requires some memory space and the process of periodically removing dominated solutions from the archive consumes computational resources. However, these expenses are deemed insignificant in comparison to the advantages of obtaining a larger collection of non-dominated solutions and memory and computational power availability of nowadays computers.

IV. Experimental Results and Analysis

We performed a comparative analysis using the NSGA-III algorithm [19] as a case study on two benchmark function sets: WFG and DTLZ [20], [21] to determine the efficacy of incorporating archive memory in storing all explored non-dominated solutions. Table I provides more information about these function sets as representative indicators of having archive memory effectiveness. Using parameters from [19] listed in Table II, the NSGA-III algorithm was evaluated both with and without archive memory. The goal was to investigate how non-dominated solutions that could go as missed in the absence of archive memory are captured by it (it would be much important when multi-objective problem is expensive too).

Algorithm 1 An evolutionary algorithm with archive memory

Input max_eval : the maximum number of evaluation call
Output $result$: result of evolutionary algorithm,
$archive$: the non-dominated solution in memory

$P \leftarrow$ initialize the population
$archive \leftarrow P$
$n\_gen \leftarrow 0$
evaluate population, and update n_eval (number of evaluation)
while $n\_eval < max\_eval$ **do**
    select parents for recombination
    perform crossover and mutation
    evaluate new offsprings, and update n_eval
    $archive \leftarrow update\_archive(offsprings, n\_gen)$
    $P \leftarrow selection(P \cup offsprings)$
    $n\_gen \leftarrow n\_gen + 1$
end while
$result \leftarrow$ retrieve result from P
apply non-dominated sorting over archive memory and select first front
return $result$, $archive$

Algorithm 2 Update archive

Input $offsprings$: new offsprings to update archive memory
n_gen : the generation number
$clean\_archive\_interval$: the interval to cleanup the archive memory

Output $archive$ the cleaned archived memory

$archive \leftarrow archive \cup offsprings$
if $n\_gen \ mod \ clean\_archive\_interval$ **then**
    $archive \leftarrow nondominated\_solutions(archive)$
end if
return $archive$

We ran the experiment 31 times to determine the worth of keeping an archive dataset because the NSGA-III algorithm is stochastic. Table III, Table IV, Table V display the averages

for the number of non-dominated solutions and knee-points [22] obtained from the 31 runs. To have a better comparison, a column is added to shows the ratio between the results of NSGA-III with archive memory and without archive memory.

Long-term archive memory was added, it showed that it could hold a large number of non-dominated solutions that were found during the optimization process. In three, five, and ten-dimensional objective spaces, the average ratio of non-dominated solutions for NSGA-III with archive memory to original NSGA-III was around 120, 115, and 295 times, respectively. Although, this ratio is directly related to the objective function landscape, this is why DTLZ5 and WFG3 have a high ratio compared to other functions in all dimensions. Having a long-term archive memory helps the NSGA-III algorithm to remember more non-dominated solutions especially when the objective space has is high dimension. This improvement makes more data accessible, which helps machine learning techniques perform better if they use the non-dominated solutions for their training. More non-dominated solutions can also be helpful to decision-makers looking for information in particular fields. This characteristic is very important in expensive MOO/MaOO.

Knee points serve as a valuable source of information for decision-makers, enabling them to make informed choices based on the desired trade-offs between conflicting objectives. Understanding the trade-off characteristics of knee points assists in selecting solutions that align with specific preferences or constraints. For three, five, and ten-dimensional objective spaces, experiments show the average ratio of found knee points between NSGA-III with archive memory and NSGA-III of around 106, 257, and 415 times, respectively. Functions DTLZ5 and WFG3 exhibit the same behavior in terms of ratio of knee points, and they can obtain the highest ratio in all dimensions.

A comparison of the non-dominated of NSGA-III and of NSGA-III with archive memory solutions for the DTLZ benchmark functions and WFG benchmark functions is shown in Fig. 1 The results capture interesting behaviors of NSGA-III by adding archive memory. For example, it may lose extreme points of non-dominated solutions in its selection operator. Particularly noticeable examples of this behavior are DTLZ1 and DTL3. Ibrahim [23] pointed out that in NSGA-III, the association of points to reference points prior to non-dominated sorting may cause some generations to omit extreme non-dominated solutions.

To illustrate the diversity of the captured non-dominated solutions by archive memory, the NSGA-III selection which relies on the reference direction was applied with different number of reference points (91, 231, 496, 861) to it and compared with the result of the NSGA-III without archive memory Table 2. Based on the results of this selection, the archive memory is able to capture a good diversity of non-dominated solutions which indicate that the archive memory does not have a bios therefore it could be used as a data source for any machine learning approaches to improve the MOO/MaOO performance. Additionally, this drives dataset

TABLE I: Benchmark functions properties, M is the dimention of the objective space

Test function	Properties	Dimension
WFG1	convex, mixed	$(M-1)\times 2+10$
WFG2	convex, disconnected	$(M-1)\times 2+10$
WFG3	linear, degenerate	$(M-1)\times 2+10$
WFG4	concave	$(M-1)\times 2+10$
WFG5	concave	$(M-1)\times 2+10$
WFG6	concave	$(M-1)\times 2+10$
WFG7	concave	$(M-1)\times 2+10$
WFG8	concave	$(M-1)\times 2+10$
DTLZ1	linear	$M+4$
DTLZ2	concave	$M+9$
DTLZ3	concave	$M+9$
DTLZ4	concave	$M+9$
DTLZ5	degenerate	$M+9$
DTLZ6	degenerate	$M+9$
DTLZ7	disconnected	$M+19$

allows the decision makers to have a region of interest (ROI) which might be impossible to satisfy their needs.

The ultimate result of this approach is anticipated to be at least as good as the base evolutionary algorithm since the long-term archive memory solutions include the base evolutionary algorithm's findings. The average inverse generational distance (IGD) of experiments is calculated and shown in Table VI, which can approve that the result of having an archive memory does not reduce the performance of the main algorithm. Since the true Pareto-front of DTLZ5, DTLZ6, DTLZ7 are not available in many-objective spaces, we could not compute the IGD metric for these functions.

Following the completion of these experiments, the advantages of utilizing long-term memory to retain all non-dominated solutions have become distinctly evident. One of the issues which can be solve by having an archive memory is in interactive multi-objective optimization, when an MOO/MaOO works with a decision maker to guide the population to most interested area [24]. By having access to more non-dominated solution the algorithm can replace individuals who are not near to the most interested area with who are near to that. Another potential usage of this proposed method is to provide more non-dominated solutions to decision maker who has a most interested area which compare to the result of algorithm might be impossible to provide a good solution to them. Optimizing industrial problems with high computing costs is one of the challenges of MOO/MaOO. Limited population size causes algorithms to lose important Pareto-front data. Although capturing these fronts would require a large computational investment by increasing the population size, it becomes possible with archive memory. This data can subsequently be used by others to optimize their issues, so avoiding the cost of repeating optimizations.

V. Conclusion Remarks

In conclusion, the significance of using archive memory in the optimization process has been brought to light by our comparative study of the NSGA-III algorithm on the WFG and DTLZ benchmark function sets. While our proposed method has no direct impact on NSGA-III performance, the study

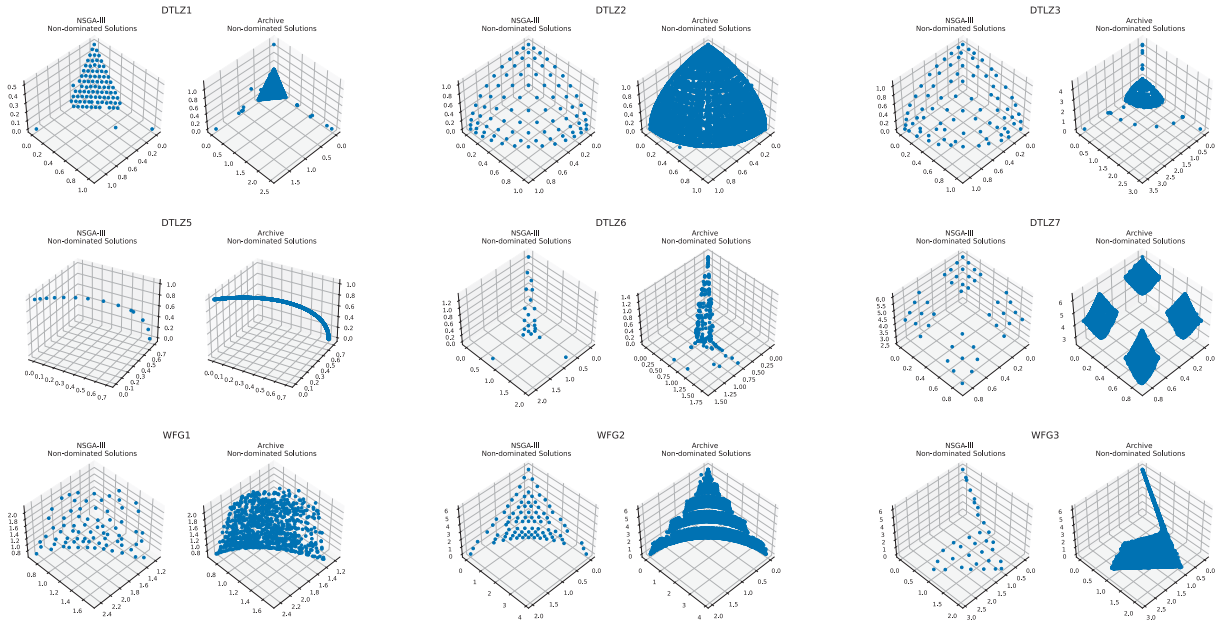

Fig. 1: Comparison between non-dominated solution of NSGA-III and archive-based NSGA-III in three objective dimension for benchmark set DTLZ and WFG

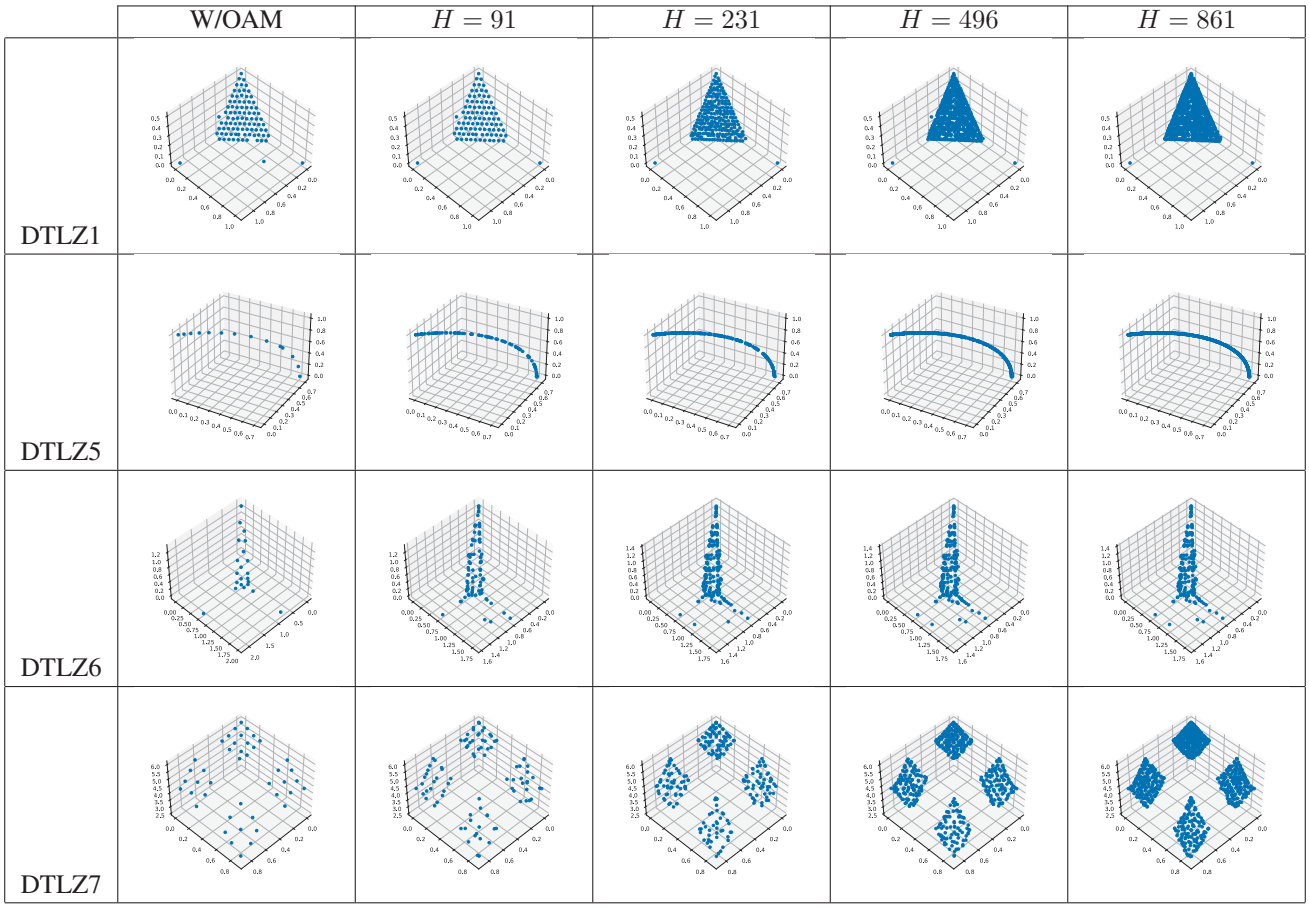

Fig. 2: Comparing the solution set of NSGA-III without archive memroy (W/OAM) with NSGA-III with selection by different number of reference points (H)

TABLE II: Parameter settings of NSGA-III algorithm for different objective space dimensions (M)

$M = 3$	Population size	92
	Number of fitness calls	$3000 \times D$
	Number of reference points	91
$M = 5$	Population size	212
	Number of fitness calls	$3000 \times D$
	Number of reference points	210
$M = 10$	Population size	276
	Number of fitness calls	$3000 \times D$
	Number of reference points	275

TABLE III: Comparison between the average number of non-dominated solutions and the average number of knee points between NSGA-III without archive memory (W/OAM) and NSGA-III with archive memory (W/AM), when $M = 3$. Numbers in ratio column is represented as times.

	Number of non-dominated solution			Number of knee points		
	W/OAM	W/AM	Ratio	W/OAM	W/AM	Ratio
DTLZ1	87	2333	**26.8**	0	2	
DTLZ2	91	11505	**126.4**	3	143	**47.7**
DTLZ3	74	1105	**14.9**	2	10	**5.0**
DTLZ4	82	10162	**123.9**	3	113	**37.7**
DTLZ5	14	4304	**307.4**	1	19	**19.0**
DTLZ6	23	356	**15.5**	1	11	**11.0**
DTLZ7	44	13182	**299.6**	2	723	**361.5**
WFG1	71	1119	**15.8**	4	49	**12.3**
WFG2	90	5180	**57.6**	6	188	**31.3**
WFG3	33	10585	**320.8**	1	205	**205.0**
WFG4	91	8554	**94.0**	1	72	**72.0**
WFG5	90	13474	**149.7**	1	302	**302.0**
WFG6	91	7593	**83.4**	1	108	**108.0**
WFG7	91	10424	**114.5**	1	123	**123.0**
WFG8	76	3749	**49.3**	3	109	**36.3**
WFG9	89	10371	**116.5**	2	430	**215.0**
	Average		**119.7**	Average		**105.8**

TABLE IV: Comparison between the average number of non-dominated solutions and the average number of knee points between NSGA-III without archive memory (W/OAM) and NSGA-III with archive memory (W/AM), when $M = 5$. Numbers in ratio column is represented as times.

	Number of non-dominated solution			Number of knee points		
	W/OAM	W/AM	Ratio	W/OAM	W/AM	Ratio
DTLZ1	179	2613	**14.6**	1	85	**85**
DTLZ2	210	17651	**84.1**	5	392	**78.4**
DTLZ3	123	1233	**10.0**	5	31	**6.2**
DTLZ4	210	18556	**88.4**	5	190	**38**
DTLZ5	45	14110	**313.6**	2	226	**113**
DTLZ6	190	5281	**27.8**	5	94	**18.8**
DTLZ7	104	16195	**155.7**	1	723	**723**
WFG1	153	5024	**32.8**	7	231	**33**
WFG2	210	19800	**94.3**	12	864	**72**
WFG3	90	29791	**331.0**	3	1210	**403.33**
WFG4	210	32352	**154.1**	3	1350	**450**
WFG5	210	28875	**137.5**	1	1155	**1155**
WFG6	210	20588	**98.0**	2	812	**406**
WFG7	210	23386	**111.4**	4	914	**228.5**
WFG8	194	9805	**50.5**	5	382	**76.4**
WFG9	210	27790	**132.3**	5	1165	**233**
	Average		**114.7**	Average		**257.4**

TABLE V: Comparison between the average number of non-dominated solutions and the average number of knee points between NSGA-III without archive memory (W/OAM) and NSGA-III with archive memory (W/AM), when $M = 10$. Numbers in ratio column is represented as times.

	Number of non-dominated solution			Number of knee points		
	W/OAM	W/AM	Ratio	W/OAM	W/AM	Ratio
DTLZ1	263	11238	**42.7**	**1**	63	**63.0**
DTLZ2	275	34750	**126.4**	**10**	1110	**111.0**
DTLZ3	211	5297	**25.1**	**9**	175	**19.4**
DTLZ4	275	46289	**168.3**	**10**	724	**72.4**
DTLZ5	70	22729	**324.7**	**2**	748	**374.0**
DTLZ6	268	32546	**121.4**	**6**	655	**109.2**
DTLZ7	193	39728	**205.8**	**1**	2239	**2239.0**
WFG1	65	10082	**155.1**	**3**	475	**158.3**
WFG2	265	61831	**233.3**	**12**	2657	**221.4**
WFG3	42	71999	**1714.3**	**2**	2399	**1199.5**
WFG4	260	82025	**315.5**	**11**	3314	**301.3**
WFG5	275	75471	**274.4**	**6**	3378	**563.0**
WFG6	275	64164	**233.3**	**8**	2718	**339.8**
WFG7	248	77102	**310.9**	**9**	3206	**356.2**
WFG8	250	46894	**187.6**	9	1961	**217.9**
WFG9	255	73321	**287.5**	10	2968	**296.8**
	Average		**295.4**	Average		**415.1**

TABLE VI: Comparisoin between the average IGD between NSGA-III without archive memory (W/OAM) and NSGA-III with archive memory (W/AM).

	M=3		M=5		M=10	
	W/OAM	W/AM	W/OAM	W/AM	W/OAM	W/AM
DTLZ1	3.44E-02	**2.20E-02**	9.47E-02	**6.63E-02**	7.70E-02	**6.13E-02**
DTLZ2	5.12E-02	**6.10E-03**	1.29E-02	**1.04E-02**	5.88E-02	**4.87E-02**
DTLZ3	4.43E-01	**3.90E-01**	2.70E+00	**2.59E+00**	2.15E+00	**2.07E+00**
DTLZ4	1.38E-01	**9.70E-02**	1.32E-02	**1.21E-02**	6.03E-02	**5.63E-02**
WFG1	1.15E+00	**1.13E+00**	1.79E+00	**1.76E+00**	2.73E+00	**2.64E+00**
WFG2	3.49E-01	**2.65E-01**	5.51E-01	**3.47E-01**	1.43E+00	**8.62E-01**
WFG3	1.49E-01	**3.26E-02**	3.80E-01	**1.34E-01**	1.47E+00	**1.71E-01**
WFG4	2.17E-01	**3.78E-02**	9.33E-01	**3.08E-01**	4.13E+00	**1.55E+00**
WFG5	2.30E-01	**8.33E-02**	7.97E-01	**3.36E-01**	4.55E+00	**1.90E+00**
WFG6	2.41E-01	**1.12E-01**	6.50E-01	**3.57E-01**	1.99E+00	**1.09E+00**
WFG7	2.14E-01	**3.43E-02**	6.57E-01	**3.00E-01**	2.81E+00	**1.65E+00**
WFG8	3.69E-01	**2.81E-01**	8.60E-01	**6.49E-01**	2.79E+00	**1.86E+00**
WFG9	2.55E-01	**1.24E-01**	6.79E-01	**6.22E-01**	2.73E+00	**2.46E+00**
Average	2.61E-01	**1.75E-01**	7.78E-01	**5.77E-01**	2.08E+00	**1.26E+00**

demonstrates the potential usage of long-term archive memory in capturing all non-dominated solutions during optimization process, which can be missed if we stay with population size restriction. The experiment demonstrated a significant increase in the number of knee points and non-dominated solutions in various dimensional spaces. Our suggested solution introduces archive memory without additional fitness calls, making it a computationally efficient option for optimization problems in the current computing landscape where memory restrictions are less of an issue. Have access to more non-dominated solutions and more knee points open doors for usage and enhance previous works. This study will play a pivotal role in decision-making, innovization, and machine learning-assisted optimization approaches. Going ahead, more studies require to investigate the comprehensive use of our approach, highlighting how it might improve data mining and

machine learning methods in addition to optimization. By supplying richer and more relevant datasets, our methodology contributes to enhancing the relationship between optimization methods and machine learning/data mining methodologies, hence creating doors for more robust model training and increased generalizability by providing more data to them.

REFERENCES

[1] Vikas Palakonda and Jae-Mo Kang. Many-Objective Real-World Engineering Problems: A Comparative Study of State-of-the-Art Algorithms. *IEEE Access*, 11:111636–111654, 2023.

[2] Slim Bechikh, Maha Elarbi, and Lamjed Ben Said. Many-objective Optimization Using Evolutionary Algorithms: A Survey. In Slim Bechikh, Rituparna Datta, and Abhishek Gupta, editors, *Recent Advances in Evolutionary Multi-objective Optimization*, Adaptation, Learning, and Optimization, pages 105–137. Springer International Publishing, Cham, 2017.

[3] Ngoc Hoang Luong, Tanja Alderliesten, Arjan Bel, Yury Niatsetski, and Peter A. N. Bosman. Application and benchmarking of multi-objective evolutionary algorithms on high-dose-rate brachytherapy planning for prostate cancer treatment. *Swarm and Evolutionary Computation*, 40:37–52, June 2018.

[4] Xuning Liu, Yong Zhou, Jiaqi Zhao, Rui Yao, Bing Liu, Ding Ma, and Yi Zheng. Multiobjective ResNet pruning by means of EMOAs for remote sensing scene classification. *Neurocomputing*, 381:298–305, March 2020.

[5] Kalyanmoy Deb and Aravind Srinivasan. Innovization: innovating design principles through optimization. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, GECCO '06, pages 1629–1636, New York, NY, USA, July 2006. Association for Computing Machinery.

[6] Sunith Bandaru, Amos H. C. Ng, and Kalyanmoy Deb. Data mining methods for knowledge discovery in multi-objective optimization: Part A - Survey. *Expert Systems with Applications*, 70:139–159, March 2017.

[7] Sunith Bandaru and Kalyanmoy Deb. Automated discovery of vital knowledge from Pareto-optimal solutions: First results from engineering design. In *IEEE Congress on Evolutionary Computation*, pages 1–8, Barcelona, Spain, July 2010. IEEE.

[8] Azam Asilian Bidgoli, Shahryar Rahnamayan, Bilgehan Erdem, Zekiye Erdem, Amin Ibrahim, Kalyanmoy Deb, and Ali Grami. Machine learning-based framework to cover optimal Pareto-front in many-objective optimization. *Complex & Intelligent Systems*, 8(6):5287–5308, December 2022.

[9] Ran Cheng, Yaochu Jin, Kaname Narukawa, and Bernhard Sendhoff. A Multiobjective Evolutionary Algorithm Using Gaussian Process-Based Inverse Modeling. *IEEE Transactions on Evolutionary Computation*, 19(6):838–856, December 2015. Conference Name: IEEE Transactions on Evolutionary Computation.

[10] Jaroslav Hájek, András Szöllös, and Jakub Šístek. A new mechanism for maintaining diversity of Pareto archive in multi-objective optimization. *Advances in Engineering Software*, 41(7):1031–1057, July 2010.

[11] Santosh Tiwari, Patrick Koch, Georges Fadel, and Kalyanmoy Deb. AMGA: an archive-based micro genetic algorithm for multi-objective optimization. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, GECCO '08, pages 729–736, New York, NY, USA, July 2008. Association for Computing Machinery.

[12] Seyedali Mirjalili. Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Computing and Applications*, 27(4):1053–1073, May 2016.

[13] SOMA PRATHIBHA, B. Latha, and V. Vijaykumar. An Improved Multi-Objective Workflow Scheduling Using NSPSO With Fuzzy Rules, March 2021.

[14] Yuxin Wu, Jianjun Li, Jing Huang, Dongsheng Qing, Peng Ai, Chi, and Keqin Li. Multi-objective Optimization Model of Forest Spatial Structure Based on Dynamic Multi-Group PSO Algorithm, March 2022.

[15] HuaXin Qiu and HaiBin Duan. Multi-objective pigeon-inspired optimization for brushless direct current motor parameter design. *Science China Technological Sciences*, 58(11):1915–1923, November 2015.

[16] Xingyi Zhang, Ye Tian, Ran Cheng, and Yaochu Jin. An Efficient Approach to Nondominated Sorting for Evolutionary Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation*, 19(2):201–213, April 2015.

[17] Handing Wang and Xin Yao. Corner Sort for Pareto-Based Many-Objective Optimization. *IEEE Transactions on Cybernetics*, 44(1):92–102, January 2014. Conference Name: IEEE Transactions on Cybernetics.

[18] Qiang Long, Xue Wu, and Changzhi Wu. Non-dominated sorting methods for multi-objective optimization: Review and numerical comparison. *Journal of Industrial and Management Optimization*, 17(2):1001–1023, February 2021. Publisher: Journal of Industrial and Management Optimization.

[19] Kalyanmoy Deb and Himanshu Jain. An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems With Box Constraints. *IEEE Transactions on Evolutionary Computation*, 18(4):577–601, August 2014.

[20] Simon Huband, Luigi Barone, Lyndon While, and Phil Hingston. A Scalable Multi-objective Test Problem Toolkit. In Carlos A. Coello Coello, Arturo Hernández Aguirre, and Eckart Zitzler, editors, *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, pages 280–295, Berlin, Heidelberg, 2005. Springer.

[21] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. Scalable Test Problems for Evolutionary Multiobjective Optimization. In Ajith Abraham, Lakhmi Jain, and Robert Goldberg, editors, *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*, Advanced Information and Knowledge Processing, pages 105–145. Springer, London, 2005.

[22] Lily Rachmawati and Dipti Srinivasan. Multiobjective Evolutionary Algorithm With Controllable Focus on the Knees of the Pareto Front. *IEEE Transactions on Evolutionary Computation*, 13(4):810–824, August 2009.

[23] Amin Ibrahim, Shahryar Rahnamayan, Miguel Vargas Martin, and Kalyanmoy Deb. EliteNSGA-III: An improved evolutionary many-objective optimization algorithm. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 973–982, Vancouver, BC, Canada, July 2016. IEEE.

[24] Bin Xin, Lu Chen, Jie Chen, Hisao Ishibuchi, Kaoru Hirota, and Bo Liu. Interactive Multiobjective Optimization: A Review of the State-of-the-Art. *IEEE Access*, 6:41256–41279, 2018. Conference Name: IEEE Access.