Sensor-Based Change Detection Schemes for Dynamic Multi-Objective Optimization Problems

Shaaban Sahmoud and Haluk Rahmi Topcuoglu
Computer Engineering Department, Marmara University
34722, Istanbul, Turkey
Email: ssahmoud@marun.edu.tr, haluk@marmara.edu.tr

***Abstract*—Detecting changes in a landscape is a critical issue for several evolutionary dynamic optimization techniques. This is because most of these techniques have steps to be taken as a response to the environmental changes. It may not be feasible for most of the real world problems to know a priori when a change occurs; therefore, explicit schemes should be proposed to detect the points in time when a change occurs. Although there are both sensor-based and population-based detection schemes presented in the literature for single objective dynamic optimization problems, there are no such efforts for the dynamic multi-objective optimization problems (DMOPs). This paper proposes a set of novel sensor-based change detection schemes for DMOPs. An empirical study is presented for validating the performance of the proposed detection schemes by using eight test problems which have different types and characteristics. Additionally, the proposed change detection schemes are incorporated into a dynamic multi-objective evolutionary algorithm to validate the effectiveness of the proposed change detection schemes.**

I. Introduction

The main difference between a dynamic multi-objective optimization problem (DMOP) and a stationary multi-objective optimization problem is the existence of dynamism. The dynamism is usually in the form of changes in one or more objective functions, changes in constraints or changes in parameters of the problem over time [1]. While the main goal of stationary optimization problems is to find the optimal or near-optimal solution, tracking the global optima as close as possible is the main motivation of a dynamic optimization problem. On the other hand, for a DMOP the optimization goal is to evolve a diverse Pareto front and to track changes in the Pareto front over time [2].

Since there are many real world problems which are dynamic in nature, dynamic multi-objective optimization problems (DMOP) are rapidly attracting the interest of the researchers [1]. Routing optimization problems in mobile and ad-hoc networks [3], scheduling problems [4], control problems [5], resource management problems [6] and hydro-thermal power problems [7] are selected examples of DMOPs.

Evolutionary algorithms and its dialects are among the most widely used techniques to solve dynamic optimization problems (DOPs) [8]. In literature, evolutionary computing for DOPs is called evolutionary dynamic optimization (EDO). Detecting the changes in a landscape is a critical issue for several EDO techniques, since most of them target to take an explicit action to respond to a change in the environment [8]. The EDO techniques do not perform any extra work when the changes in the environment are known; which is common in many problems that are presented in the literature. However, it may not be feasible for most of the real world problems to know a priori when a change occurs. In case of detecting the change, an EDO technique utilizes either a population-based detection strategy or a sensor-based detection strategy, where the former one considers performance degradation in the population as the indicator of a change; and the latter one utilizes reevaluation of a fixed number of sensors from the landscape during the search process [9].

In this paper, we present a set of novel sensor-based change detection schemes for dynamic multi-objective optimization problems (DMOPs). An empirical study is presented to evaluate the performance of proposed schemes by considering all types of DMOPs. Additionally, we integrate the change detection schemes with dynamic multi-objective evolutionary algorithms. In order to validate incorporation with the detection schemes, we consider the DNSGA-II-A algorithm [7], the algorithm that has diversity maintenance based extension on the well-known NSGA-II [10] algorithm. The results show that the integration of the proposed change detection schemes significantly outperforms the DNSGA-II-A algorithm.

The remainder of the paper is organized as follows. Section 2 presents a short review on DMOPs and techniques for solving DMOPs. We present our proposed change detection schemes for DMOPs in Section 3. Section 4 shows the results of the empirical study for assessing the performance of the change detection schemes and evaluates impact of the detection schemes on the performance of the DNSGA-II-A algorithm. Finally, we conclude the paper and discuss future research directions in Section 5.

II. Dynamic Multi-objective Optimization

A dynamic multi-objective optimization problem (DMOP) is an optimization problem with two or more conflicting objectives, where the objective(s), constraints or parameters of the problem change in time. Formally, a DMOP is defined as:

$$minf(x,t) = \{f_1(x,t), f_2(x,t), ...f_M(x,t)\} \quad (1)$$

$$g(x,t) < 0, h(x,t) = 0 \quad (2)$$

where x is the vector of decision variables and f is the set of objectives to be minimized with respect to time t. The two functions g(x,t) and h(x,t) define the constrains of the problem that may also change with respect to time. In case of a change

in a DMOP, at least the Pareto optimal set (POS) or the Pareto optimal front (POF) may change. The dynamic POF is the set of nondominated solutions with respect to the objective space at a given time. The set of nondominated solutions with respect to the decision space determines the dynamic POS. Since a change in the landscape may affect the existing solutions of the DMOP, an optimization algorithm should continuously track the dynamic POF. Specifically, it targets to quickly find and converge to the new POF before the next change occurs.

Based on the most common categorization of DMOPs proposed in [11], they are classified into four types depending on where the change happens, in decision space, fitness space or in both of them:

- *Type 1*. The Pareto optimal set (POS) changes over time while the Pareto Optimal front (POF) remains stationary.
- *Type 2*. Both the POF and POS change over time.
- *Type 3*. The POF changes over time while the POS remains stationary.
- *Type 4*. Both the POF and POS remain stationary.

There are a number of dynamic multi-objective evolutionary (DMOE) algorithms for solving DMOPs in the literature, which can be broadly classified into five categories [8]. *Diversity introduction* is the first category where all solutions in the population or a number of them are reinitialized randomly whenever a change is detected [7]. Updating the mutation rate based on the problem changes and diversity of the population is another example of the first category [12]. Reinitializing a portion of the population in each generation without considering the changes explicitly is an example strategy for the second category called *diversity maintenance* [13]. The algorithms of the third approach, called *multi-population* strategy, use more than one population (named as subpopulations) concurrently to solve DMOPs, where each one of the subpopulations works in separated area in the search space and may have a separate task. Two well-known examples of this category are the DVEPSO algorithm [14] and the dCOEA algorithm [2]. According to the recent research [15] multi-population algorithms outperform other types of algorithms on a various set of DMO test instances. *Memory-based* strategy is the fourth category, which uses implicit or explicit memory to store the best detected solutions and reuse them in next stages when change happens again [16]. A recent research strategy is to integrate the memory concept with other DMOE algorithms [17]. The last category is the *prediction-based* strategy where algorithms solve DMOPs by exploiting the history of the past population information in order to estimate the POF of the dynamic problem by utilizing forecasting methods [18], [19].

III. Change Detection Schemes for DMOPs

Although changes in the environment are known by the EDO algorithm for several synthetic or test problems presented in the literature, it may not be feasible for real-world problems. On the other hand, detecting the points in time when a change occurs in the landscape is a critical issue for a number of EDO techniques. Therefore, many EDO algorithms utilize either a population-based detection strategy or a sensor-based detection strategy [8], [9] to detect the changes. In a population-based approach, population behavior is considered to detect a change in the environment. Specifically, non-parametric statistical tests such as Wilcoxon-Mann-Whitney and Jensen-Shann methods are used to determine performance degradation in the population, which is considered as a change indicator [9]. For a sensor-based detection strategy, a fixed number of sensors (which are candidate solutions) are spread around the landscape and re-evaluated throughout the search process in order to detect changes. Determining a sensor-placement scheme and number of sensors needed are the main issues of a sensor-based strategy, where there are alternative schemes in the literature [20]. Although a sensor-based detection strategy requires additional fitness function evaluations, they are more favorable for the case of difficult detection scenarios [9].

Algorithm 1 Selection from different densities of POF (PPOFD)

```
begin
.
.
repeat
  g ← g + 1 /*generation number*/
  if g % change frequency = 0 then
    counter ← 0
    r ← 1 /*density rank*/
    F1 ← Determine the POF solutions
    Fs ← Sort F1 solutions according to densities
    while counter < Sensors Number do
      S ← Select individual of Rank r from Fs
      B ← Fitness functions of S
      A ← Re-evaluate(S)
      if B ≠ A then
        HandleChange() /*Change detected*/
        break
      end if
      counter ← counter + 1
      r ← r + ⌊ size of Fs / Sensors Number ⌋
    end while
  end if
  .
  .
until Termination condition is satisfied
end
```

Both population-based and sensor-based change detection strategies that presented in literature are applied on single objective dynamic optimization problems; and to the best of our knowledge, there is no work of applying and testing various change detection strategies for DMOPs. The researchers assume that the change periods are known a priori or they utilize a simple strategy such as selecting randomly a fixed percent of the population as sensors [7], [21]. In this section, we present seven sensor-based change detection schemes for DMOP, where some of them are adapted from the detections schemes designed for single objective DOPs and some of them are new schemes designed for DMOPs. The new schemes are proposed in order to locate the sensors in a DMOP such as selecting them from the POF or distribute them according to the non-domination ranks of the solutions.

The DMOPs sensor-based detection schemes proposed in this study can be classified into two main groups: *from-population* and *non-population* methods. The former group includes schemes that select the sensors from the population with respect to different selection schemes; and the latter group looks for the other parts of the landscape that are outside of the population. Sensors are visited and evaluated one by one

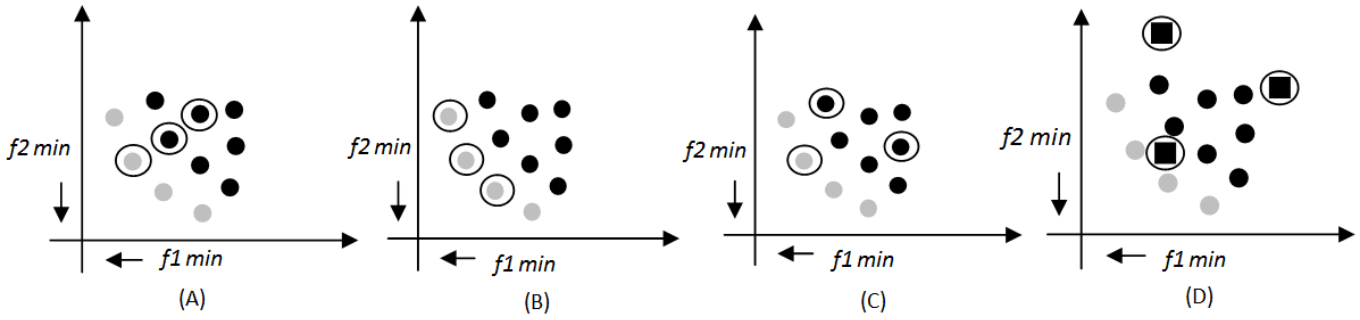

Fig. 1. Sensor based change detection schemes for a two-objective optimization problem. Each scheme uses three sensors where points are the members of the population, and points with the circles are the selected sensors

Algorithm 2 Non-population distributed solutions (NP3)

```
begin
  ⋮
repeat
   g ← g + 1 /*generation number*/
   c ← 1
   while c < Sensors Number do
      S_c ← Initialize an individual uniformly
      B_c ← Fitness functions of S_c
      c ← c + 1
   end while
   if g % change frequency = 0 then
      c ← 1
      while c <= Sensors Number do
         A_c ← Re-evaluate(S_c)
         if B_c ≠ A_c then
            HandleChange() /*Change detected*/
            break
         end if
         c ← c + 1
      end while
   end if
   ⋮
until Termination condition is satisfied
end
```

for all sensor-placement schemes proposed in this section. In case of a differentiation from its previous fitness values (a change in any objective function) for a sensor evaluation, it is an indicator of change and evaluations of remaining sensors are skipped; otherwise, the next sensor is evaluated. In this section, the proposed change detection schemes are presented, which include pseudocodes of only two schemes due to space limitations.

A. From-Population Methods

- *Random set from the population (PR).* In this method a predefined number of solutions from the population are randomly selected and re-evaluated. If there is a change at least in one sensor, then an environment change is considered. Due to multi-objective nature, all objective function values must be compared for each sensor. In Figure 1C, three sensors (i.e., the points which have a circle) are selected based on this scheme for a two-objective optimization problem. It is a simple mechanism but sometimes it may not cover all the search regions.
- *Random set from POF (PPOF).* The POF is the most important solution set in the population. This method selects a fixed number of solutions randomly from the POF set and reevaluates them to test a change occurrence (see Figure 1B). Selecting solutions from the POF provides better distribution in the search space, since the DMOE algorithms target to distribute the solutions to cover all POF regions during the evolving process.
- *Selection from different domination ranks (PRank).* This method targets to distribute the tested sensors based on the rank of that sensor (solution) in its population. It may generate various combinations based on the number of ranks and the number of sensors. The simplest case is to select one sensor from each rank, since the number of sensors is limited (see Figure 1A).
- *Selection from different densities of POF (PPOFD).* The sensors are also selected from the POF set not randomly but based on a distribution to cover the entire POF region. We consider crowding distance in order to perform a distribution mechanism without additional running time to the dynamic multi-objective evolutionary algorithm. The crowding distance for each solution is already computed by many MOEAs such as NSGA-II; and Algorithm 1 describes the steps of this method. As shown in the algorithm, first the POF is determined and then it is sorted using the crowding distance of the solutions. Then, simple selecting mechanism is applied to pick sensors from different density levels.

B. Non-Population Methods

- *One time random initialization (NP1).* This method randomly initializes fixed number of sensors to detect the environment change. The sensors are initialized once only before the algorithm starts running; then, those sensors are re-evaluated to test the change occurrence. In Figure 1D, three sensors (marked as black squares) are chosen randomly, which are outside of the population. The selected sensors can be from any part of the landscape including the regions that do not contain any candidate solutions.
- *Random initialization for each test (NP2).* This

TABLE I. SELECTED DMO TEST PROBLEMS BASED ON THEIR TYPES

Problem Type	Selected Problems
Type 1	FDA4 [11], SJY1 [15]
Type 2	FDA5 [11], dMOP2 [2]
Type 3	SJY4 [15], dMOP1 [2]
Type 4	SJY5 [15], T1 [22]

method is the same as the previous one except that the sensors are re-initialized randomly in every generation. This method adds more randomness to sensors selection process since the sensors are re-initialized periodically to scan different areas in the search space.

- *Distributed solutions (NP3).* Instead of initializing the sensors randomly as in the previous two methods, this method targets to spread the sensors over the decision space in order to cover all regions. Depending on the selected number of sensors, the decision space is divided into equal size fractions; and sensors are placed into those fractions. As in the NP1 method, the sensors are initialized once only before the start of execution. Algorithm 2 presents the pseudocode of this method.

IV. EXPERIMENTAL STUDY

In this section, we present the test problems for the empirical study of our change detection schemes, which is followed by the metrics used for the performance evaluation. Then, the results of empirical study is given in two consecutive subsections.

A. Test Problems

In order to compare performance of the proposed sensor-based change detectors, eight dynamic multi-objective test problems are selected by considering two test problems from each DMOP type given in Section 2. The names of the test problems are given in Table I.

The test problems given in Table I are selected from four different test suites. The first test suite is the FDA test problems [11], where the authors created it by adapting the static DTLZ [23] problems. The FDA test problems are the most widely used problems for testing the performance of DMOE algorithms. In our experimental study, we consider two test problems (FDA4 and FDA5) from this test suite. They are both non-convex and scalable, i.e. we can easily generate test problems with any number of objective functions. In our experiments, we also consider test problems from the SJY framework [15], which construct scalable dynamic test problems of all four types; and the dynamism can easily be added and controlled. In this paper, three test problems (SJY1, SJY4 and SJY5) are selected from SJY framework. The SJY1 test problem has a linear POF; and the POF shape of the SJY4 problem changes gradually during the execution from concave to convex. The POF of the SJY5 test problem has also a non-convex shape. Additionally, we use T1 test problem proposed in [22], where the dimension of the decision variables can change over time. The last two test problems are the dMOP1 and the dMOP2 that proposed by Goh and Tan [2], which were designed based on the FDA problems. The formulation of each problem can be accessed in the related reference given in Table I.

The time t for each test problem is defined as,

$$t = \frac{1}{n_t} * \frac{\tau}{fr} \tag{3}$$

where n_t represents the severity of change, τ is the iteration count and fr is the frequency of change.

B. Comparison Metrics

In order to compare the quality of solutions of the change detectors, we consider two metrics.

- *The true positive rate (TP).* This metric measures the percentage of the changes that are correctly detected in the environment. A high TP rate that is close to 1 indicates that almost all of the changes are caught by the change detection scheme.

$$TPrate = \frac{correctly\ identified\ changes}{total\ number\ of\ changes} \tag{4}$$

- *The average number of invoked sensors (sAvg).* This metric measures the average number of sensors used to detect a change in the environment. Normally, the value of sAvg is between 1 and number of detectors considered. Since, each sensor needs one further function evaluation to detect the change in the dynamic environment, a high sAvg value indicates high overhead of the detections scheme, which means that the need of more sensors will cost more function evaluations. Therefore a small value of this metric is desired.

As part of the experimental study, the proposed change detection schemes are incorporated with the DNSGA-II-A algorithm. In order to evaluate the significance of the schemes on the DNSGA-II-A algorithm, we use the following three metrics:

- *The mean inverted generational distance (mIGD)* [24]. It is derived from IGD [25] and computes the total average generational distance for all generations of the resulted solutions which can be calculated as:

$$\text{mIGD} = \frac{1}{g_n} * \sum_{t=1}^{g_n} IGD_t \tag{5}$$

where g_n is the number of generations.

- *The mean inverted generational distance just before the change (mIGDB).* This metric is developed to compute the performance of the DMOE algorithms just before the next change [17]. While $mIGD$ computes the total average generational distance for all generations, this metric computes the last generational distance value just before the change. The $mIGDB$ metric can be calculated from this equation:

$$\text{mIGDB} = \frac{1}{C_n} * \sum_{t=1}^{C_n} IGD_t \tag{6}$$

where C_n is the number of changes and IGD_t is calculated just before the next change.

- *Scotts spacing (SS).* This metric [26] is developed to measure the distribution of the discovered Pareto front. It measures how evenly the members in POF are distributed, and it is computed from this equation:

$$\text{SS} = \sqrt{\frac{1}{n-1}\sum_{1}^{n}(D_i - D_m)^2} \qquad (7)$$

where D_i is the Euclidean distance between the ith member in POF and its nearest member in POF and D_m is the average value of D_i.

C. Performance Evaluation of the Change Detection Schemes

In this section, we present results of an empirical study for performance evaluation of seven proposed sensor-based change detection schemes. In our experimental study, the terms NP1, NP2 and NP3 indicate the one time random initialization, random initialization for each test and distributed solutions based change detection schemes, respectively. Additionally, the terms PR, PPOF, PRANK and PPOFD refer to random selection from population, random selection from POF, selection from different domination ranks and selection from different densities of POF based detection schemes, respectively.

In our experiments, the number of sensors is set to 4, unless otherwise stated. The severity of change is set to 100 and the term t for each dynamic test problem is computed using equation 3. Frequency of change is equal to 10;; and there are three objectives for each test problem. For each run, the stopping criterion is 1000 generations; and 50 independent runs are performed for each detection scheme on each test. A one-way ANOVA test on the TP rates is performed to demonstrate whether the performance variations among the detection schemes are statistically significant or not.

Figure 2 is the notched boxplot of the TP rates of the seven detection schemes for eight dynamic multi-objective test problems considered. Each boxplot in a figure correspond to a detection scheme with lines at the lower quartile, median and upper quartile data values, where the whiskers are the lines extending from each end of the box to show the extent of the rest of the data.

The results given in Figure 2 clearly show that the non-population sensor based change detection methods outperform the from-population sensor based methods in most of the test problems. Specifically, both the NP1 and the NP3 cases are better than the other methods in 7 out of 8 test problems. Among the from-population methods, the PPOFD scheme is the best one but it is still not as good as the NP1 and the NP3 schemes. Sensors that selected with from-population schemes may be confined in a smaller region of the fitness space. Conversely, the non-population sensor based methods select random values without restrictions which may make it capable of covering landscape of all objective functions. Additionally, the changes in "type 2" test problems can be detected by 100% due to the changes in both POF and POS in such problems.

In another experiment, average number of sensors needed to detect changes (sAvg) is measured for each change detection scheme proposed in this study. Table 2 shows sAvg values of 100 indepedent runs of each case for eight test problems. Based on the table, the non-population sensor based methods are better than the from-population based methods, since they need only one sensor evaluation for most of the cases. This situation is different from the case of single objective DOPs [20]; since, for each sensor re-evaluation on a DMOP, there is m function re-evaluation, where m is the number of objective functions. The existence of more than one objective function enhances the detection performance of the sensors and reduces the need of more sensor re-evaluation processes.

TABLE II. AVERAGE NUMBER OF SENSORS INVOKED FOR PROPOSED CHANGE DETECTION SCHEMES (MAXIMUM OF 4 SENSORS).

Test Problem	NP1	NP2	NP3	PR	PPOF	PRank	PPOFD
SJY1	1.00	1.00	1.00	1.16	1.15	1.27	1.19
FDA4	1.00	1.00	1.00	1.27	1.27	1.20	1.31
dMOP2	1.00	1.00	1.00	1.18	1.10	1.01	1.10
FDA5	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SJY4	1.01	1.00	1.00	1.50	1.49	1.20	1.37
dMOP1	1.00	1.00	1.00	1.46	1.43	1.45	1.54
SJY5	1.02	1.00	1.00	2.46	2.46	3.16	2.90
T1	1.00	1.00	1.00	1.00	1.00	1.00	1.00

The effect of increasing the number of sensors on performance is examined in the next experiment. Figure 3 shows the average TP rates of the detection schemes for four dynamic multi-objective test problems (one test problem from each type) by varying the number of sensors. The results show that firing two sensors is enough to detect the changes with the NP1, the NP2 and the NP3 schemes for the FDA4 and the FDA5 test problems. Although, increasing number of sensors may improve performance (i.e. an increase in TP rates) when sensors are selected from-population, the rate of improvement is not high. The best improvement is achieved for the SJY5 test problem, where the TP rate for the PPOFD scheme increases from 11% to 36% when number of sensors is increased from 2 to 4.

Performance of the change detection schemes are evaluated by varying the number of objectives considered in the test problems. We consider one test problem of each type (the FDA4 for type 1, the FDA5 for type 2, the SJY4 for type 3, and the SJY5 for type 4) in this experiment; and the number of objectives is varied by using the values from the set {2, 3, 5, 8}. As in the previous tests, the number of sensors is set to 4 and the parameter n_t for the severity of change is set to 100.

TABLE III. THE AVERAGE TP RATES OF DETECTION SCHEMES BY VARYING THE NUMBER OF OBJECTIVE FUNCTIONS CONSIDERED IN EACH TEST PROBLEM.

Test Prob.	# of Obj.	NP1	NP2	NP3	PR	PPOF	PRank	PPOFD
FDA4	2	**100.0**	**100.0**	**100.0**	79.7	81.0	80.6	80.4
	3	**100.0**	**100.0**	**100.0**	84.2	84.2	84.3	85.1
	5	99.0	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
	8	98.6	99.0	99.0	**100.0**	**100.0**	**100.0**	**100.0**
FDA5	2	99.9	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
	3	99.9	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
	5	99.4	**100.0**	99.9	**100.0**	**100.0**	**100.0**	**100.0**
	8	99.0	**100.0**	99.0	**100.0**	**100.0**	**100.0**	**100.0**
SJY4	2	96.7	92.7	**97.0**	74.0	74.1	74.2	72.2
	3	**96.2**	91.5	96.0	75.4	75.2	75.0	74.5
	5	**96.0**	89.1	**96.0**	57.7	57.7	55.7	58.2
	8	**95.1**	83.0	95.0	60.6	60.6	59.3	85.5
SJY5	2	**88.8**	51.8	88.0	4.5	4.2	4.5	10.8
	3	**89.0**	53.7	88.0	7.9	7.9	7.6	19.7
	5	**90.0**	54.0	88.0	7.7	7.7	7.7	32.2
	8	89.9	40.2	**91.0**	13.8	13.8	12.8	37.0

The average TP rates of all methods by varying the number

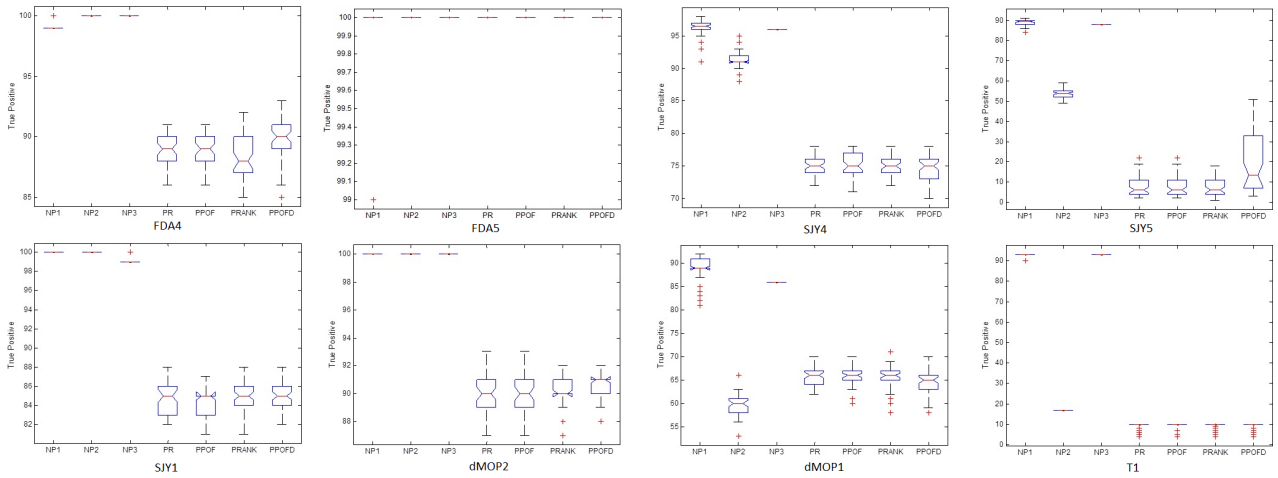

Fig. 2. One-way ANOVA test results as a notched boxplot of the proposed change detection schemes for DMOPs with a 95% confidence interval.

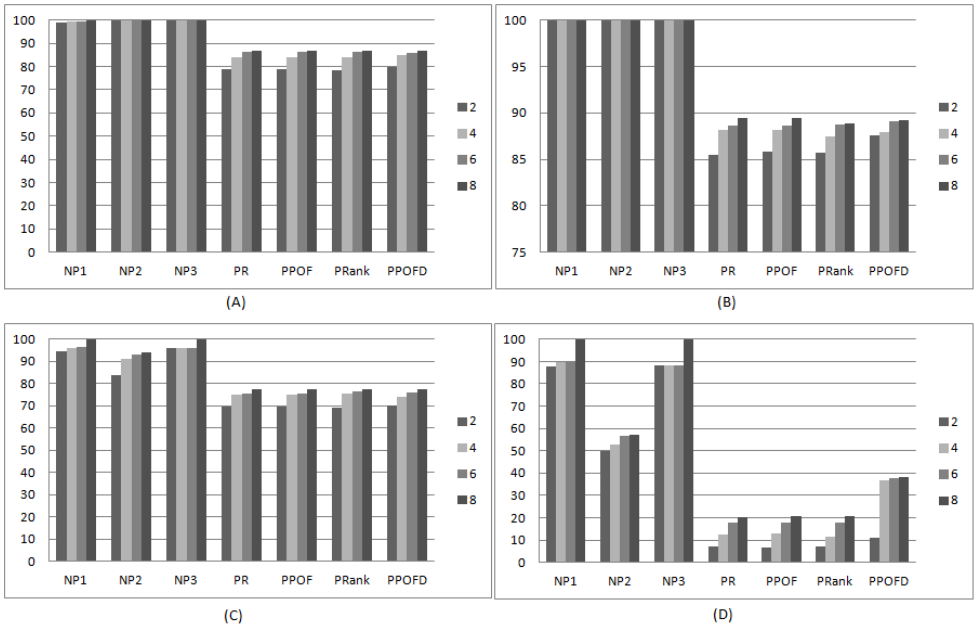

Fig. 3. Average TP rates of the change detection schemes for different number of sensors. (A) FDA4 test problem, (B) FDA5 test problem, (C) SJY4 test problem, (D) SJY5 test problem.

of objectives are given in Table 3. Sensor detection schemes based on the from-population strategy give good results and slightly outperform the non-population schemes, when higher number of objectives are considered for type 1 and type 2 test problems. On the other hand, the non-population schemes significantly outperform the from-population schemes for type 3 and type 4 test problems. This result is probably because of the nature of the type 3 and type 4 problems which do not contain a significant change in their POF and POS (in type 4 there is no change, where in type 3 the change is small). According to that, the from-population schemes concentrate on the POF region of the fitness space during the algorithm run and consequently they may not be able to identify most of the environmental changes. Additionally, the performance of from-population schemes improves with the increase of objectives for the SJY5 test problem.

D. Incorporate the Change Detection Strategies with a Dynamic Multi-Objective Evolutionary Algorithm

It is observed that selecting sensor based on non-population schemes outperform selecting sensors based on from-population schemes, with respect to both average TP rates and average number of invoked sensors to detect the changes. In this section, we propose extensions on a dynamic multi-objective evolutionary algorithm by utilizing the change detection schemes proposed in this study. We consider DNSGA-II-A algorithm, which is an algorithm that introduces diversity in a population when there is a change [7], [15]. Specifically, it replaces fixed portion of the population (20% in our experimental study) with randomly generated new solutions whenever there is a change in the environment. As in the previous experiments, population size is set to 100; a crossover probability of 0.7 and a mutation probability of

TABLE IV. PERFORMANCE COMPARISON OF THE DNSGA-II-A ALGORITHM INCORPORATED WITH THE CHANGE DETECTION SCHEMES.

Metric	Test Problem	NP1	NP2	NP3	PR	PPOF	PRank	PPOFD	DNSGA-II
mIGD	FDA4	0.11497 (9.1E-07)+	0.11512 (7.8E-07)+	0.11525 (3.9E-07)+	0.11511 (6.4E-07)+	0.11502 (6.4E-07)+	**0.11482 (5.7E-07)**	0.11483 (7.2E-07)	0.11511 (6.4E-07)+
	FDA5	0.14162 (4.1E-05)	**0.14038 (4.2E-05)**	0.14262 (3.8E-05)+	0.14262 (3.8E-05)+	0.14262 (3.8E-05)+	0.14262 (3.8E-05)+	0.14262 (3.8E-05)+	0.14262 (3.8E-05)+
	SJY4	0.10180 (8.2E-05)	0.10219 (7.8E-05)	0.09958 (3.7E-05)+	0.09986 (3E-05)+	0.10081 (4.1E-05)+	**0.09903 (3.7E-05)**	0.09950 (4.9E-05)	0.09986 (3.0E-05)
	SJY5	0.06364 (2.8E-06)+	0.06375 (2.1E-06)+	0.06315 (2.3E-06)	0.06339 (2.7E-06)+	**0.06275 (1.9E-06)**	0.06294 (3.1E-06)	0.06321 (2.5E-06)	0.08783 (6.5E-05)+
mIGDB	FDA4	0.11552 (7.6E-07)	0.11567 (6.8E-07)+	0.11581 (3.1E-07)+	0.11566 (4.8E-07)	0.11564 (5.1E-07)	0.11551 (4.9E-07)	**0.11549 (4.5E-07)**	0.11566 (4.8E-07)
	FDA5	**0.13963 (1.4E-05)**	0.14048 (1.9E-05)	0.14158 (1.5E-05)+	0.14158 (1.5E-05)+	0.14158 (1.5E-05)+	0.14158 (1.5E-05)+	0.14158 (1.5E-05)+	0.14158 (1.5E-05)+
	SJY4	0.09094 (4.4E-05)+	0.09128 (4.4E-05)+	0.08923 (2.6E-05)	0.08947 (1.8E-05)+	0.08963 (2.5E-05)	**0.08878 (2.3E-05)**	0.08888 (2.2E-05)	0.08947 (1.8E-05)+
	SJY5	0.06248 (2.4E-06)+	0.06239 (1.5E-06)+	0.06167 (2.2E-06)	0.06217 (2.3E-06)+	**0.06163 (1.8E-06)**	0.06176 (3E-06)	0.06192 (2.6E-06)	0.08279 (5.6E-05)+
SS	FDA4	0.03768 (1.9E-07)+	0.03756 (3.1E-07)	0.03761 (1.2E-07)+	0.03756 (3.1E-07)	**0.03752 (1.5E-07)**	0.03756 (1.9E-07)	0.03759 (1.2E-07)+	0.03766 (3.1E-07)+
	FDA5	0.05358 (1.6E-05)	**0.05270 (9.5E-06)**	0.05445 (1.7E-05)	0.05445 (1.7E-05)	0.05445 (1.7E-05)	0.05445 (1.7E-05)	0.05445 (1.7E-05)	0.05445 (1.7E-05)+
	SJY4	0.05255 (1.8E-05)+	**0.05088 (1.5E-05)**	0.05213 (2.2E-05)	0.05139 (2.1E-05)	0.05127 (1.6E-05)	0.05141 (1.5E-05)	**0.05091 (1.4E-05)**	0.05139(2.1E-05)
	SJY5	0.03189 (6.5E-07)	0.03206 (5.6E-07)	0.03188 (1.2E-06)	0.03187 (7.1E-07)	0.03187 (7.4E-07)	**0.03179 (3.5E-07)**	0.03185 (8E-07)	0.03325 (2.6E-06)+

A "+" sign in the right side of the values indicates that the best algorithm performs significantly better than the corresponding algorithm.

1/n (where n is the number of variables) are used in the algorithm. The number of variables is equal 10; and real-valued representation is considered.

In this experiment, we consider one problem from each type as in the previous experiments. Table IV compares the performance of the seven proposed change detection algorithms that are incorporated with the DNSGA-II-A algorithm and the performance of the DNSGA-II-A algorithm without using any change detecting mechanism. The mean inverted generational distance (mGID), mean inverted generational distance before the change (mGIDB) and Scotts spacing metrics are used to measure the performance of the eight cases. The value of each metric on each problem is given as mean and standard deviation values. Additionally, the change detection scheme that outperforms other alternatives is given in bold at each row; therefore, there is one bold case for each problem and metric pair. On the other hand, a "+" sign in a cell under the column of a detection scheme indicates that the results of the best algorithm in the corresponding row statistically outperforms the scheme presented in the corresponding column. As an example, for the mIGD metric on FDA4 test problem (i.e., the first row of the table), the PRank scheme is the best strategy and it statistically outperforms 6 other alternatives (since there are 6 "+" signs in the cells of the same row) and it outperforms the PPOFD scheme.

Table V presents pairwise comparison of the DNSGA-II-A algorithm and change detection schemes incorporated with the DNSGA-II-A algorithm, based on the results given in Table IV. Each cell in the table has three numbers for pairwise comparison between the scheme given in the corresponding row and the scheme given in the corresponding column, where the numbers are for the comparisons of the mGID, the mGIDB, the Scotts spacing values, respectively. Here, each number can be a value from 0 to 4, where 4 is the number of test problems considered in Table IV. As an example, in the first cell of the seventh row (i.e. the cell of the PPOFD row and the NP1 column), we have (2,1,1) values, which indicate that PPOFD scheme statistically outperforms the NP1 scheme in 2 problems for the mGID metric, in 1 problem for the mGIDB metric and also in 1 problem for the Scotts spacing metric. The last column in the table sums up the results of each row, where the scheme of the highest sum is the best one. The PRank and PPOFD schemes are the best two schemes, since they have the highest values in the last column.

When the experiments in previous section are considered, the non-population based schemes (NP1, NP2 and NP3 schemes) significantly outperformed from-population schemes with respect to the TP rates and the sAvg values. On the other hand, the results given in Table IV show that the integrations of the from-population schemes (i.e., PPOF, PRank and PPOFD) with the DNSGA-II algorithm outperform the integrations of non-population schemes in most of the cases presented. The non-population schemes give better results only for the Type 2 problem, the FDA5 test. In addition, the results show that the proposed change detection algorithms enhance the performance of the DNSGA-II algorithm since the DNSGA-II algorithm (without any change detection mechanism) has the lowest value in Table V. Actually, when the change detection algorithm detects every change even if it is very small (as in the non-population schemes), it fires the process of re-initializing 20% of population in the DNSGA-II algorithm, which may degrade the performance of the algorithm. Therefore, in spite of the good true positive results that the non-population sensor schemes gained, the overall performance of the evolutionary algorithm that use these methods is not as good as the performance of the evolutionary algorithm that use the from-population change detection scheme. Conforming to our results, it is recommended to use from-population based change detection schemes (i.e. the PPOF, the PRank and the PPOFD schemes), since those schemes can enhance the performance of the DMOE algorithm by detecting only the important changes and not every environmental change (even it is very small) as in the non-population schemes.

V. CONCLUSION

In this paper, we investigated the performance of seven different sensor-based detection schemes (NP1, NP2, NP3, PR, PPOF, PRank and PPOFD schemes) on eight dynamic multi-objective test problems. The proposed schemes can be categorized into two groups: from-population schemes which select the sensors from the population with respect to different selection schemes and non-population schemes which select sensors from other parts of the landscape that are outside of the population. Results of our empirical study show that non-population schemes outperforms the from-population schemes with respect to TP rate values and number of invoked sensors.

Additionally, we proposed a hybridization technique by incorporating proposed sensors-detection schemes with the DNSGA-II-A algorithm, which is a dynamic multi-objective evolutionary algorithm that introduces diversity when there is a change in the environment. Based on the results of the experimental study, the performance of the from-population sensor based schemes such as the PPOF, the PRank and the PPOFD schemes are better than the non-population schemes in most of the cases considered. Furthermore, the results of experiments validate that the proposed change detection schemes enhance

TABLE V. PAIRWISE COMPARISON OF THE CHANGE DETECTION SCHEMES FOR THREE METRICS AND FOUR TEST PROBLEMS (BASED ON THE RESULTS GIVEN IN TABLE IV)

	NP1	NP2	NP3	PR	PPOF	PRank	PPOFD	DNSGA-II	Total
NP1	x	0,1,1	2,2,0	2,1,0	2,1,0	1,1,0	1,1,0	1,1,2	20
NP2	0,0,2	x	2,1,1	2,1,0	2,1,0	1,1,0	1,1,1	1,1,2	21
NP3	1,2,1	1,2,2	x	1,2,0	0,0,0	0,0,0	0,0,0	1,2,2	17
PR	0,0,2	0,1,1	0,1,1	x	0,0,0	0,0,0	0,0,1	0,0,3	10
PPOF	1,2,2	1,3,1	0,1,1	1,2,0	x	0,0,0	0,0,1	1,2,3	22
PRank	2,2,2	2,3,1	2,1,1	3,2,0	2,0,0	x	0,0,1	2,2,3	31
PPOFD	2,1,1	2,3,1	2,1,0	3,2,0	2,0,0	0,0,0	x	2,2,2	27
DNSGA-II	0,0,1	0,1,0	1,1,0	1,0,0	1,0,0	0,0,0	0,0,0	x	6

the performance of the DNSGA-II-A algorithm significantly.

A further extension is to utilize the hybridization of the proposed schemes with other dynamic multi-objective evolutionary algorithms, in addition to the DNSGA-II-A algorithm. Population-based change detectors used for the single objective DOPs in the literature will be adopted for the DMOPs as part of another future work. Furthermore, it is planned to investigate various detection mechanisms for severity of changes in DMOPs and to incorporate those mechanisms with the dynamic multi-objective evolutionary algorithms presented in the literature.

REFERENCES

[1] S. Yang and X. Yao, *Evolutionary Computation for Dynamic Optimization Problems*. Springer-Verlag, 2013.

[2] C.-K. Goh and K. C. Tan, "A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 1, pp. 103–127, 2009.

[3] D. Constantinou, "Ant colony optimisation algorithms for solving multi-objective power-aware metrics for mobile ad hoc networks," Ph.D. dissertation, University of Pretoria, 2011.

[4] C.-L. Chen and W.-C. Lee, "Multi-objective optimization of multi-echelon supply chain networks with uncertain product demands and prices," *Computers & Chemical Engineering*, vol. 28, no. 6, pp. 1131–1144, 2004.

[5] R. P. Hämäläinen and J. Mäntysaari, "Dynamic multi-objective heating optimization," *European Journal of Operational Research*, vol. 142, no. 1, pp. 1–15, 2002.

[6] S. Palaniappan, S. Zein-Sabatto, and A. Sekmen, "Dynamic multiobjective optimization of war resource allocation using adaptive genetic algorithms," in *SoutheastCon 2001. Proceedings. IEEE*. IEEE, 2001, pp. 160–165.

[7] K. Deb, U. Bhaskara Rao N., and S. Karthik, "Dynamic multi-objective optimization and decision-making using modified nsga-ii: a case study on hydro-thermal power scheduling," in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2007, pp. 803–817.

[8] J. Branke, *Evolutionary Optimization in Dynamic Environments*. Norwell, MA, USA: Kluwer Academic Publishers, 2001.

[9] H. Richter, "Detecting change in dynamic fitness landscapes," in *2009 IEEE Congress on Evolutionary Computation*. IEEE, 2009, pp. 1613–1620.

[10] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[11] M. Farina, K. Deb, and P. Amato, "Dynamic multiobjective optimization problems: test cases, approximations, and applications," *IEEE Trans. Evolutionary Computation*, vol. 8, no. 5, pp. 425–442, 2004.

[12] H. G. Cobb, "An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environments," DTIC Document, Tech. Rep., 1990.

[13] J. J. Grefenstette *et al.*, "Genetic algorithms for changing environments," in *PPSN*, vol. 2, 1992, pp. 137–144.

[14] M. Greeff and A. P. Engelbrecht, "Solving dynamic multi-objective problems with vector evaluated particle swarm optimisation," in *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*. IEEE, 2008, pp. 2917–2924.

[15] S. Jiang and S. Yang, "A framework of scalable dynamic test problems for dynamic multi-objective optimization," in *Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), 2014 IEEE Symposium on*. IEEE, 2014, pp. 32–39.

[16] Y. Wang and B. Li, "Investigation of memory-based multi-objective optimization evolutionary algorithm in dynamic environment," in *2009 IEEE Congress on Evolutionary Computation*. IEEE, 2009, pp. 630–637.

[17] S. Sahmoud and H. R. Topcuoglu, "A memory-based nsga-ii algorithm for dynamic multi-objective optimization problems," in *European Conference on the Applications of Evolutionary Computation*. Springer, 2016, pp. 296–310.

[18] A. Zhou, Y. Jin, Q. Zhang, B. Sendhoff, and E. Tsang, "Prediction-based population re-initialization for evolutionary dynamic multi-objective optimization," in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2007, pp. 832–846.

[19] A. Muruganantham, Y. Zhao, S. B. Gee, X. Qiu, and K. C. Tan, "Dynamic multiobjective optimization using evolutionary algorithm with kalman filter," *Procedia Computer Science*, vol. 24, pp. 66–75, 2013.

[20] L. Altin and H. R. Topcuoglu, "Performance evaluation of sensor-based detection schemes on dynamic optimization problems," in *Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), 2014 IEEE Symposium on*. IEEE, 2014, pp. 24–31.

[21] S. Jiang and S. Yang, "A steady-state and generational evolutionary algorithm for dynamic multiobjective optimization," 2012.

[22] L. Huang, I. H. Suh, and A. Abraham, "Dynamic multi-objective optimization based on membrane computing for control of time-varying unstable plants," *Information Sciences*, vol. 181, no. 11, pp. 2370–2391, 2011.

[23] K. Deb, "Multi-objective genetic algorithms: Problem difficulties and construction of test problems," *Evolutionary computation*, vol. 7, no. 3, pp. 205–230, 1999.

[24] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated pareto sets, moea/d and nsga-ii," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 284–302, 2009.

[25] C. A. C. Coello and N. C. Cortés, "Solving multiobjective optimization problems using an artificial immune system," *Genetic Programming and Evolvable Machines*, vol. 6, no. 2, pp. 163–190, 2005.

[26] J. R. Schott, "Fault tolerant design using single and multicriteria genetic algorithm optimization." DTIC Document, Tech. Rep., 1995.