An Empirical Study of Dynamic Triobjective Optimisation Problems

Shouyong Jiang*, Marcus Kaiser*, Shuzhen Wan†, Jinglei Guo‡, Shengxiang Yang§, and Natalio Krasnogor*
*School of Computing, Newcastle University, Newcastle upon Tyne NE4 5TG, U.K.
Email: math4neu@gmail.com, {natalio.krasnogor, marcus.kaiser}@ncl.ac.uk
†School of Computer Science and Information Technology, China Three Gorges University, China
Email: wanshuzhen@163.com
‡Department of Computer Science, Central China Normal University, China
Email: guojinglei@mail.ccnu.edu.cn
§School of Computer Science and Informatics, De Montfort University, UK
Email: syang@dmu.ac.uk

***Abstract*—Dynamic multiobjective optimisation deals with multiobjective problems whose objective functions, search spaces, or constraints are time-varying during the optimisation process. Due to wide presence in real-world applications, dynamic multiobjective problems (DMOPs) have been increasingly studied in recent years. Whilst most studies concentrated on DMOPs with only two objectives, there is little work on more objectives. This paper presents an empirical investigation of evolutionary algorithms for three-objective dynamic problems. Experimental studies show that all the evolutionary algorithms tested in this paper encounter performance degradedness to some extent. Amongst these algorithms, the multipopulation based change handling mechanism is generally more robust for a larger number of objectives, but has difficulty in deal with time-varying deceptive characteristics.**

I. Introduction

MANY real-world multi-objective optimisation problems (MOPs) change over time in a dynamic manner, such as planning [3], [28], scheduling [6], [23], control [8], [14], [30], and metabolic modelling [34]. This kind of problem is known as dynamic multi-objective optimisation problems (DMOPs). Due to dynamics, the optimisation of DMOPs is much more challenging than that of static MOPs as it has to deal with not only conflicting objectives, but also any change in objective functions or constraints. In other words, dynamic multiobjective optimisation algorithms (DMOAs) must be capable of tracking the changing Pareto-optimal front (PF) and/or Pareto-optimal set (PS) to provide a set of diverse solutions that approximates the new PF and PS over time.

Dynamic multi-objective optimisation (DMO) has attracted increasing research interest in recent years and there have been significant contributions made to this research field. So far, much effort has been mainly devoted to two main topics: benchmarking and algorithm design. The following delivers a brief review of the topics.

a) Benchmarking: Benchmark test problems are of great importance to evaluating the relative performance of DMOAs. They contribute to analysing and identifying the strengths and weaknesses of an algorithm in order to modify it and improve its performance. In the DMO literature, some static test problems, including the ZDT [33] and DTLZ [7] test suites, have been modified to develop dynamic characteristics that may appear in real life. The commonly used dynamic test suite, i.e. the FDA test suite [8], is such a case. Jin and Sendhoff [19] developed an open scheme of aggregating objective functions of existing test problems by dynamically changing weights to form a low-dimensional DMOP. Guan *et al.* [11] studied DMOPs with objective replacement, where some objectives may be replaced with new objectives during the evolution. Mehnen *et al.* [21] argued that the DTLZ and ZDT test suites are already challenging in their static version, and simpler test functions are needed to analyse the effect of dynamics in DMOPs. Hence, they suggested the DSW functions for DMOPs. A number of recent studies have introduced more test problems with diverse dynamics [1], [9], [18]. However, most of them focus on biobjective cases.

b) Algorithm Design: Confronted with DMOPs, it is natural to devise algorithms to solve them. Whenever a change is detected in the current optimisation environment, it is important that there is a good DMO algorithm available to handle the change efficiently. Simply restarting the optimisation process is too naive and inefficient in most cases unless the environmental change is extremely severe. Similar to dynamic single-objective optimisation [29], there mainly exists three popular types of change reaction methods. The first or foremost method is diversity introduction/maintenance. For example, the nondominated sorting genetic algorithm II (NSGA-II) was modified by incorporating with diversity introduction techniques, e.g. random initialisation or hypermutation, in order to handle dynamic environments [6]. In [10], a multipopulation strategy was used so as to resist diversity loss in the event of changes.

The second option is to predict the moving PF/PS. Prediction techniques have been extensively exploited for DMO in recent years. A wide variety of prediction models were built [12] to predict the future environment behaviours. However, a prediction-based method is best applicable to the situation that environmental changes follow a regular pattern. It is very likely to fail for unpredictable environmental changes.

Another way to deal with DMOPs is to speed up the convergence process of population when a change occurs. Memorising some information of past environments can be very helpful for reinitialising population for a new environment in a more promising search subspace. Such a method has been proved very useful for dynamic single-objective optimisation [2], [24], [29], but it is rarely studied in the context of DMO, possibly due to vast storage cost. Fast convergence can be also achieved by well-organised population update structure. In a recent study [17], generational and steady-state search structures were nicely incorporated to evolve the population fast toward the PF. It was shown that this method is able to track environmental changes rapidly.

Despite growing studies on the two topics mentioned above, most of them are done for biobjective cases. On one hand, it is much easier to construct biobjective test problems than triobjective ones. It is also easier to control problem properties and dynamics in the lower case. On the other hand, from the algorithms' point of view, more objective functions means an increase in search burden. Besides, it is not as straightforward as biobjective optimisation to assess and visualise the performance of algorithms in question. For these reasons, there have been significantly few studies on dynamic problems with three or more objectives. In order to have a good understanding of how much the optimisation difficulty increases and what the main challenges are in higher-dimensional (objective space) DMO, this paper presents an empirical study of several algorithms on the tri-objective dynamic optimisation.

The rest of this paper is organised as follows. Section II presents related work and a brief description of the three-objective test problems for our empirical studies. Section III describes experimental settings. Section IV presents the experimental results and analysis. Section V concludes the paper and suggests some possible directions for our future work.

II. Related Work

A. *Classification of DMO Environments*

A good classification of dynamic environments helps to understand dynamics and therefore design corresponding algorithms that can handle the dynamics. According to [8], DMOPs can be classified into four types based on the induced effect of change on the PF/PS:

- **Type I** - the PS changes over time while the PF remains stationary.
- **Type II** - both the PF and PS change over time.
- **Type III** - the PF changes over time while the PS remains stationary.
- **Type IV** - both the PF and PS remain stationary, although the objective functions or constraints may change over time.

Tantar *et al.* [26] argued that the classification by Farina *et al.* [8], although of undisputed importance, does not capture where dynamic changes in DMOPs come from. Accordingly, they proposed a cause-based classification for dynamic environments:

- **1st order** - the decision variables change over time.
- **2nd order** - the objective functions change over time.
- **3rd order** - the current values of the decision variables or the objective functions depend on their previous values.
- **4th order** - parts of or the entire environments change over time.

B. *Three-objective Dynamic Problems*

Most existing test problems in the literature are not objective scalable. Very few of them are three-objective dynamic problems. The study in [16] proposed a systematic framework to generate DMOPs with more than two objectives. Five test instances (SJY1-SJY5) were therefore created. The detailed definition for them can be found in [16]. The time t in each problem is defined as:

$$t = \frac{1}{n_t}\lfloor\frac{\tau}{\tau_t}\rfloor \tag{1}$$

where n_t represents the severity of change, τ is the iteration counter and τ_t represents the frequency of change.

In what follows, we briefly describe the dynamics or features involved in these problems.

1) SJY1: SJY1 has a linear PF which keeps stationary over time. However, its PS keeps changing over time. This problem is mainly used to test the tracking ability of algorithms on the changing PS.

2) SJY2: In SJY2, position-related variables [15] are swapped with each other when a change occurs. This swap can lead to significant diversity loss for population-based algorithms. The overall PF shape of SJY2 is a convex hyper-surface, and solutions become few when each objective value moves away from the origin. This problem not only tests the diversity performance of algorithms, but also assesses to what extent PF approximations cover the true PF.

3) SJY3: Similar to SJY2, SJY3 also has a swap function, but the swap function can swap between position-related variables and distance-related variables [15]. The interaction between two variable subsets is to some extent a severe change, which makes it difficult for an algorithm to relocate the new PS. Besides, the objective range is scaled with time, and the dynamic creates a changing number of extreme PF solutions far from or close to others.

4) SJY4: SJY4 is a problem whose PF shape can be concave or convex over time. It is important to realise that a change in the PF shape requires a good response from the distribution of the PS in order to obtain well-diversified solutions on the PF.

5) SJY5: SJY5 is a problem having special characteristics. we briefly describe its definition here to reveal its speciality:

$$\min\begin{cases} f_1 = \prod_{j=1}^{M-1}\cos(0.5\pi x_j) \\ f_{i=2:M-1} = \sin(0.5\pi x_{M-i+1})\prod_{j=1}^{M-i}\cos(0.5\pi x_j) \\ f_M = \left(\frac{1+g(x,t)}{1+\cos^2(0.5\pi x_1)}\right)^{\frac{1}{(1+g(x,t))^{B(t)}}} \end{cases} \tag{2}$$

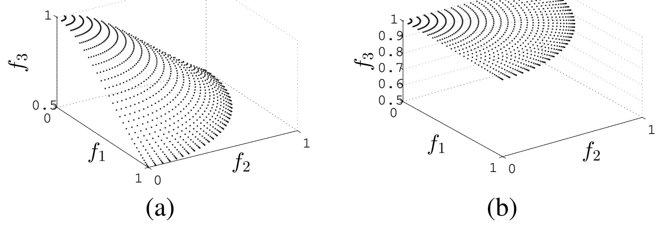
Fig. 1. PFs of SJY5 for three objectives: (a) the true PF; (b) the deceptive PF.

where $g(x,t)=\sum_{x_i\in\mathbf{x_{II}}} x_i^2$ ($\mathbf{x_{II}}=(x_M,\dots,x_n)$ is a subset of n decision variables) and $B(t)=1.5+1.2\sin(0.5\pi t)$. The search space of SJY5 is restricted to $[0,1]^n$. The PF of this problem can be described as $f_M(1+f_1^2+\cdots+f_{M-1}^2)=1$, which is illustrated in Fig. 1(a), and the PS is $x_i=0, \forall x_i\in\mathbf{x_{II}}$. From the definition, we can see that SJY5 is a Type-IV problem as both the PF and PS remain stationary. It is worth noting that SJY5 conceives deceptive properties that hinder algorithms from finding the true PF, and the difficulty changes over time. Due to the deceptive feature, algorithms are very likely to get into the local PF, as illustrated in Fig. 1(b). The following gives the proof.

For the convenience of notation, let $g(x)=1+g(x,t)$ and $\hat{f}=f_1^2+\cdots+f_{M-1}^2$. It is easy to see that $\hat{f}=\cos^2(0.5\pi x_1)$, and the last objective f_M can be simplified as

$$f_M=\left(\frac{g(x)}{1+\hat{f}}\right)^{\frac{1}{g(x)^{B(t)}}} \tag{3}$$

where f_M can be seen as a differential function of g ($1\leq g<+\infty$), written as $f_M(g)$. We can find the global minimum by using the first derivative test:

$$f_M'(g)=\left(\frac{g}{1+\hat{f}}\right)^{\frac{1}{g^{B(t)}}}\left(\frac{1}{g^{B(t)+1}}(1-B(t)\log\frac{g}{1+\hat{f}})\right) \tag{4}$$

where the critical point is $g=(1+\hat{f})\exp(\frac{1}{B(t)})$. It is clear that $f_M(g)$ is increasing on $[1,(1+\hat{f})\exp(\frac{1}{B(t)})]$ and decreasing on $[(1+\hat{f})\exp(\frac{1}{B(t)}),+\infty]$. Since $f_M(1)=\frac{1}{1+\hat{f}}<1$ and $f_M(+\infty)=\lim\limits_{g\to+\infty}f_M(g)=\exp(\lim\limits_{g\to+\infty}\frac{\log g-\log(1+\hat{f})}{g^{B(t)}})=1$, the global minimum is clearly at $g=1$. Since $f_M(g)$ has a monotonous decrease in the range $[(1+\hat{f})\exp(\frac{1}{B(t)}),+\infty]$ and this range is relatively large compared with the well/basin ($[1,(1+\hat{f})\exp(\frac{1}{B(t)})]$) of the global minimum, evolutionary algorithms are very likely to search solutions in the monotonously decreasing range, resulting in a local PF instead of the global one. Besides, the dynamic on SJY5 actually shifts the position of the critical point $g=(1+\hat{f})\exp(\frac{1}{B(t)})$ over time. As a result, the "aperture" size of the well/basin leading to the global minimum varies over time.

6) SJY6: A New Discontinous Problem: Although the above five problems capture a variety of dynamic and problem properties, none of them is discontinous or has a time-varying disconnected PF. Here, we add a new problem of this type to the SJY family, called SJY6, which is defined as

$$\min\begin{cases} f_1=(1+g(x,t))\cos^2(\frac{\pi x_1}{2})+G(t)\\ f_2=(1+g(x,t))\cos^2(\frac{\pi x_2}{2})+G(t)\\ f_3=\sum\limits_{j=1}^{2}[\sin^2(\frac{\pi x_j}{2})+\sin(\frac{\pi x_j}{2})\cos^2(p(t)\pi x_j)]+G(t)\end{cases} \tag{5}$$

wherer $g(x,t)=\sum\limits_{i=3}^{n}(x_i-G(t))^2$, $p(t)=\lfloor 6G(t)\rfloor$, and $G(t)=|\sin(0.5\pi t)|$. The search space is $[0,1]^n$. It is clear that the PS is $x_i=G(t), \forall i=3,\dots,n$. However, the PF cannot be explicitly expressed.

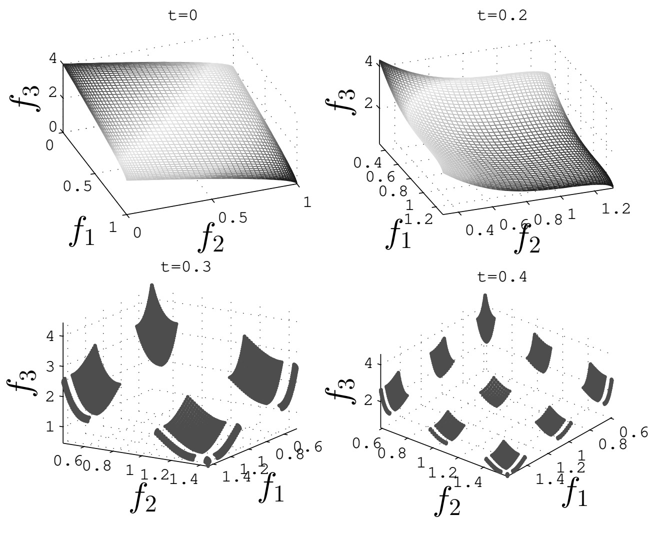

Fig. 2. PFs of SJY6 for t = 0, 0.2, 0.3, and 0.4 for three objectives.

The PF of SJY6 consists of a number of disconnected regions which is controlled by a time-dependent parameter $p(t)$ in the last objective, as illustrated in Fig. 2. Apart from disconnectivity, the overall PF also shifts over time. The PS is also dynamic, so SJY6 is a Type-II DMOP. SJY6 can be used to test an algorithm's ability to track the time-shifting of PF and maintain well-distributed subpopulations in different and varying PF subregions.

III. Experimental Settings

A. Compared Algorithms

Many static multi-objective optimisation evolutionary algorithms (MOEAs) have been adapted to handle DMOPs [4], [6], [10], [20], [25]. Other strategies, such as prediction for population reinitialisation [32], population prediction strategy (PPS) [31], and inheritance strategy [11], have also been developed for DMO. This paper compares three widely used algorithms in the DMO literature, and they are DMOPSO [20], dCOEA [10], and DNSGA-II [6]. These algorithms are relatively old compared with recently developed ones [17], but they have been widely reported to have good performance for DMOPs, particularly on 2-objective cases. Here, we use them to illustrate how 3-objective DMOPs affect EAs' search behaviour. The compared algorithms belong to different classes of meta-heuristics and each has a mechanism of change detection. All the algorithms was allowed to use 10% of population (randomly picked) to detect environmental changes. Other parameter settings of each algorithm are derived from the referenced paper.

All the algorithms were tested on the triobjective SJY problem instances.The population size was set to 300 for each algorithm. To study the effect of the frequency of change (τ_t) on each problem, n_t was set to 10, and τ_t was set to $\tau_t = 5, 10, 20, 30, 50$. For a fair comparison, the total number of changes was set to 20 during the evolution. Besides, 100 more generations were given to each algorithm before the first change to minimize the potential effect of static optimisation. Thus, the total number of generations allowed was $100 + 20\tau_t$.

B. Dynamic Performance Measures

In DMO, the PF of a DMOP is susceptible to change. Thus, the aim is not only to pursue a well-converged and well-diversified PF, but also to track the changing PF over time. Let PF(t) denote the approximated PF at time step t, so an alternative performance measure can be to assess the PF(t).

Coello and Cortés [5] proposed the concept of inverted generational distance (IGD), which estimates how far the elements in the true PF are from those in the PF approximation. The IGD measure can be used for quantifying both the diversity and convergence of an MOEA at each time step, and a set of IGD values can be obtained within a number of time windows (i.e., T_s). In this paper, we suggest the average value of the obtained IGD values for DMO assessment, which is calculated by

$$MIGD = \frac{1}{T_s}\sum_{t=1}^{T_s} IGD(t) \tag{6}$$

where $MIGD$ evaluates the average IGD metric over T_s time windows.

IV. Empirical Analysis

A. General Results

Table I presents the average rank over 30 runs for each problem and each setting of τ_t. The ranking was calculated by the Wilxcon rank-sum test on the MIGD measure at the significance level of 0.5, together with the Bonferroni correction [22] to reduce statistical Type-I errors.

For SJY1, dCOEA is clearly the best performer for all the settings of τ_t, followed by DMOPSO, and then DNSGA-II. This means, compared with DNSGA-II, dCOEA and DMOPSO has a faster convergence speed, which therefore renders better ability to track the moving PS.

SJY2 has a stationary PF, however, its PS changes over time due to the swap of pairs of position-related decision variables. Besides, there are dependencies among variables in SJY2. For smaller values of τ_t, the MIGD values of DNSGA-II is not as good as those of dCOEA, but DNSGA-II outperforms dCOEA for larger values of τ_t (that is, slower environmental changes). This indicates that DNSGA-II may be more suitable than dCOEA for dealing with variable-linkage problems if enough time is given for convergence. DMOPSO performs poorly with regard to the MIGD measure for all the settings of τ_t in this situation. The comparison on this problem suggests that fast-converging algorithms, e.g. DMOPSO, have no advantage when facing possible diversity loss. The diversity increase mechanism in DNSGA-II does not exert its effect in fast-changing environments, but is very helpful in slow-changing environments.

The amount of diversity loss in SJY3 is bigger than that in SJY2. The table shows that dCOEA is always the best performer for all the τ_t values, followed by DNSGA-II. DMOPSO ranks the last.

SJY4 challenges the uniformity of points on the PF. The comparison between DNSGA-II and dCOEA indicates that dCOEA struggles to obtain uniformly-distributed solutions for fast-changing environments, but this difficulty begins to disappear when the environmental changes become less frequent. DMOPSO performs worst on SJY4.

Interestingly, dCOEA has the worst performance for SJY5. This suggests that the time-varying deceptive feature in SJY5 poses a great challenge to multipopulation based strategies. DNSGA-II is very capable of handling this type of problem.

For the discontinuous SJY6 problem, the ranks for the three algorithms remain unchanged regardless of τ_t values. dCOEA performs the best, followed by DNSGA-II, and then by DMOPSO.

Figure 3 plots the average IGD values obtained by the three algorithms for the setting of $\tau_t = 20$ and $n_t = 10$ over 30 runs. The figure clearly illustrates that dCOEA has the best tracking performance for all the problems except SJY5, although it is affected massively by the presence of environmental changes (see the high-magnitude spikes). This suggests that dCOEA has a very good response to a change and can quickly converge toward the PF of the new environment. For SJY5, DNSGA-II is the best performer with a much lower IGD value for each environmental change compared with dCOEA and DMOPSO. It means dCOEA and DMOPSO probably get trapped into the local PF of this problem whereas DNSGA-II not.

Whilst the above analysis helps to understand the average performance of algorithms on SJY problems, it is more desirable to visually have a close look at their tracking ability. To this end, the PF approximations of the first several time steps (from $t = 0$ to $t = 0.3$) for three selected problems are depicted in Figs. 4–6. From the figures, it is easy to see that dCOEA has no difficulty in tracking the moving PS of SJY1. but loses a large amount of diversity for SJY2 and fails to locate the global PF for SJY5. DNSGA-II has good diversity in both SJY1 and SJY5, but it is not well converged to the PF. On the other hand, DMOPSO is struggling for all three problems, as evidenced by poor diversity and convergence.

It is understandable that DMOPSO performs poorly among these algorithms because it does not increase or maintain diversity in the event of change, resulting in the population not being able to adapt to the new environment quickly.

B. Impact of the Severity of Change

Beside the frequency of change, the severity of change is another major factor that influences algorithms' performance. For this reason, we have tested the algorithms at different severity levels, i.e. n_t =2, 5,10, and 20, on SJY3. The frequency of change was set to 10. The other parameters remain the same as in the previous experiment.

TABLE I. PERFORMANCE RANKING BY THE MIGD METRIC FOR SJY PROBLEMS

τ_t	rank	SJY1	SJY2	SJY3	SJY4	SJY5	SJY6
5	1st	dCOEA	dCOEA	dCOEA	DNSGA-II	DNSGA-II	dCOEA
	2nd	DMOPSO	DNSGA-II	DNSGA-II	dCOEA	DMOPSO	DNSGA-II
	3rd	DNSGA-II	DMOPSO	DMOPSO	DMOPSO	dCOEA	DMOPSO
10	1st	dCOEA	dCOEA	dCOEA	DNSGA-II	DNSGA-II	dCOEA
	2nd	DNSGA-II	DNSGA-II	DNSGA-II	dCOEA	DMOPSO	DNSGA-II
	3th	DMOPSO	DMOPSO	DMOPSO	DMOPSO	dCOEA	DMOPSO
20	1st	dCOEA	dCOEA	dCOEA	dCOEA	DNSGA-II	dCOEA
	2nd	DMOPSO	DNSGA-II	DNSGA-II	DNSGA-II	DMOPSO	DNSGA-II
	3rd	DNSGA-II	DMOPSO	DMOPSO	DMOPSO	dCOEA	DMOPSO
30	1st	dCOEA	DNSGA-II	dCOEA	dCOEA	DNSGA-II	dCOEA
	2nd	DMOPSO	dCOEA	DNSGA-II	DNSGA-II	DMOPSO	DNSGA-II
	3rd	DNSGA-II	DMOPSO	DMOPSO	DMOPSO	dCOEA	DMOPSO
50	1st	dCOEA	DNSGA-II	dCOEA	dCOEA	DNSGA-II	dCOEA
	2nd	DMOPSO DNSGA-II	dCOEA	DNSGA-II	DNSGA-II	DMOPSO	DNSGA-II
	3rd		DMOPSO	DMOPSO	DMOPSO	dCOEA	DMOPSO

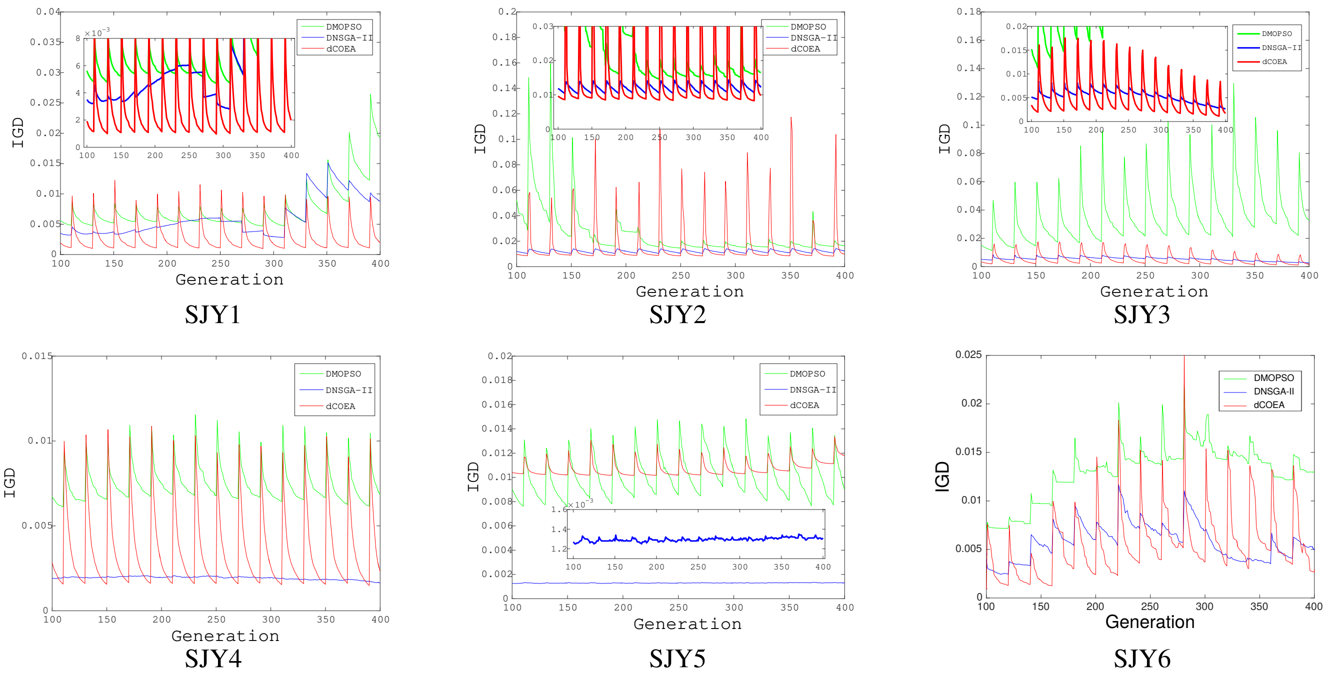

Fig. 3. The mean IGD curves of three algorithms with $\tau_t = 20$ over 30 runs on SJY1-SJY6.

Figure 7 displays the mean IGD values over 30 runs obtained by the four algorithms under different severity levels. It can be observed that, as n_t increases (i.e., the severity of change becomes smaller), all the tested algorithms achieve better performance except DMOPSO. On the other hand, DNSGA-II is very stable in response to environmental variations. The IGD values of DMOPSO and dCOEA fluctuate seriously when a new change occurs, but dCOEA is capable of overcoming the dynamics and finally converges nicely toward the changing PF at the end of each time window. DMOPSO, however, seems to lose track of the environmental changes, leading to an increase in the IGD value after each time window. The poor performance of DMOPSO can be attributed to the fact that, in the event of changes, DMOPSO does not introduce any mechanism to increase or preserve diversity (while the other three algorithms employ diversity strategies), making it unable to react to the new change promptly. This figure once again confirms that the population diversity is extremely important in DMO.

V. CONCLUSIONS AND FUTURE WORK

Whilst most existing studies have been devoted to biobjective problems in DMO, it remains unclear what challenges posed by triobjective problems and how difficult they are. This paper has presented an empirical study of some algorithms on triobjective dynamic problems. The experimental results have revealed that diversity is a again key factor that influences the performance of MOEAs when solving triobjective DMOPs. Amongst the three tested algorithms, dCOEA has the best overall performance on the SJY problems. However, dCOEA has also been found to struggle to solve problems with time-varying deceptive characteristics.

As a first step, in this paper, the analysis has been focused on triobjective DMOPs with several dynamic features. The future work will concentrate on a comprehensive theoretical analysis on dynamic many-objective problems and on various complexities, such as disconnectedness and degeneration. Besides, we will also focus on handling some open issues of DMO, e.g., change detection and dynamic performance measures.

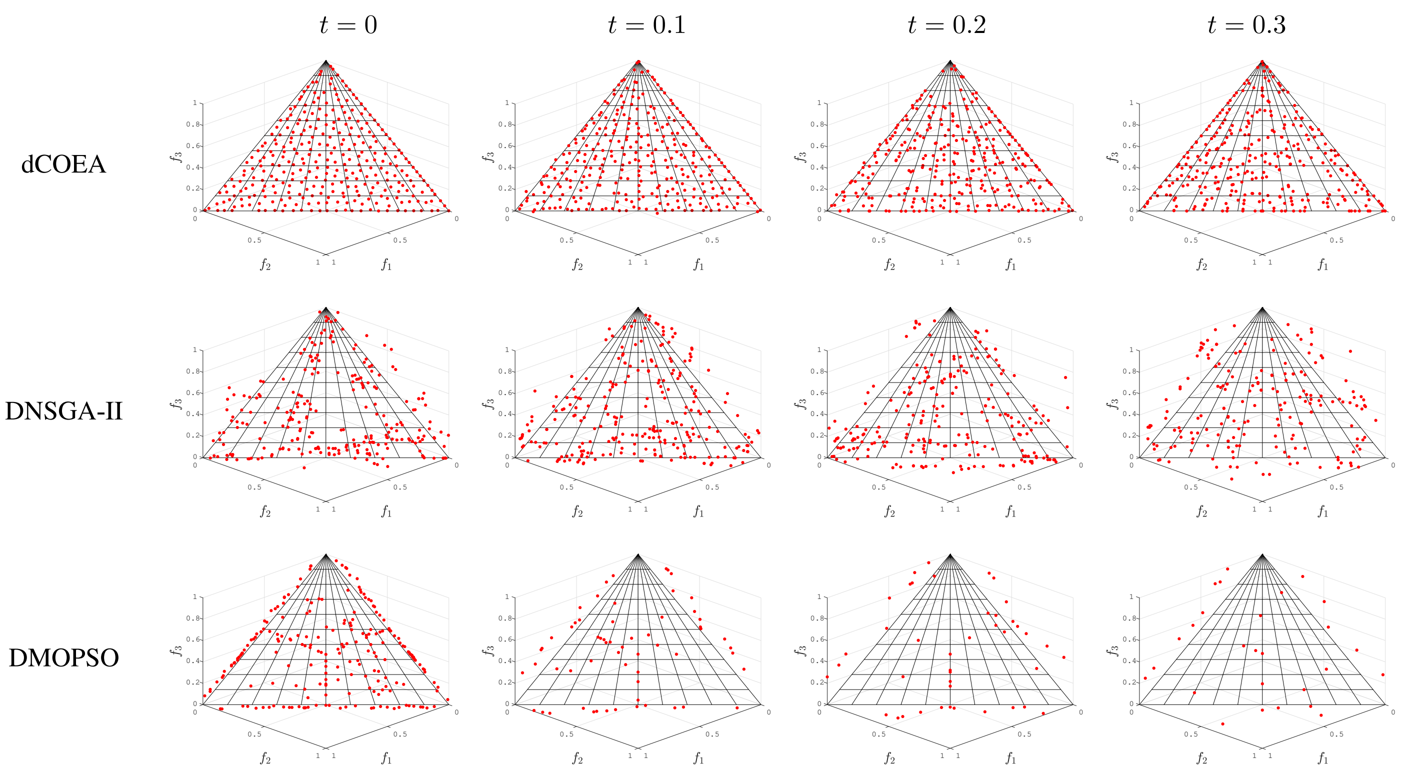

Fig. 4. PF approximations of three algorithms on SJY1 for the first four time steps.

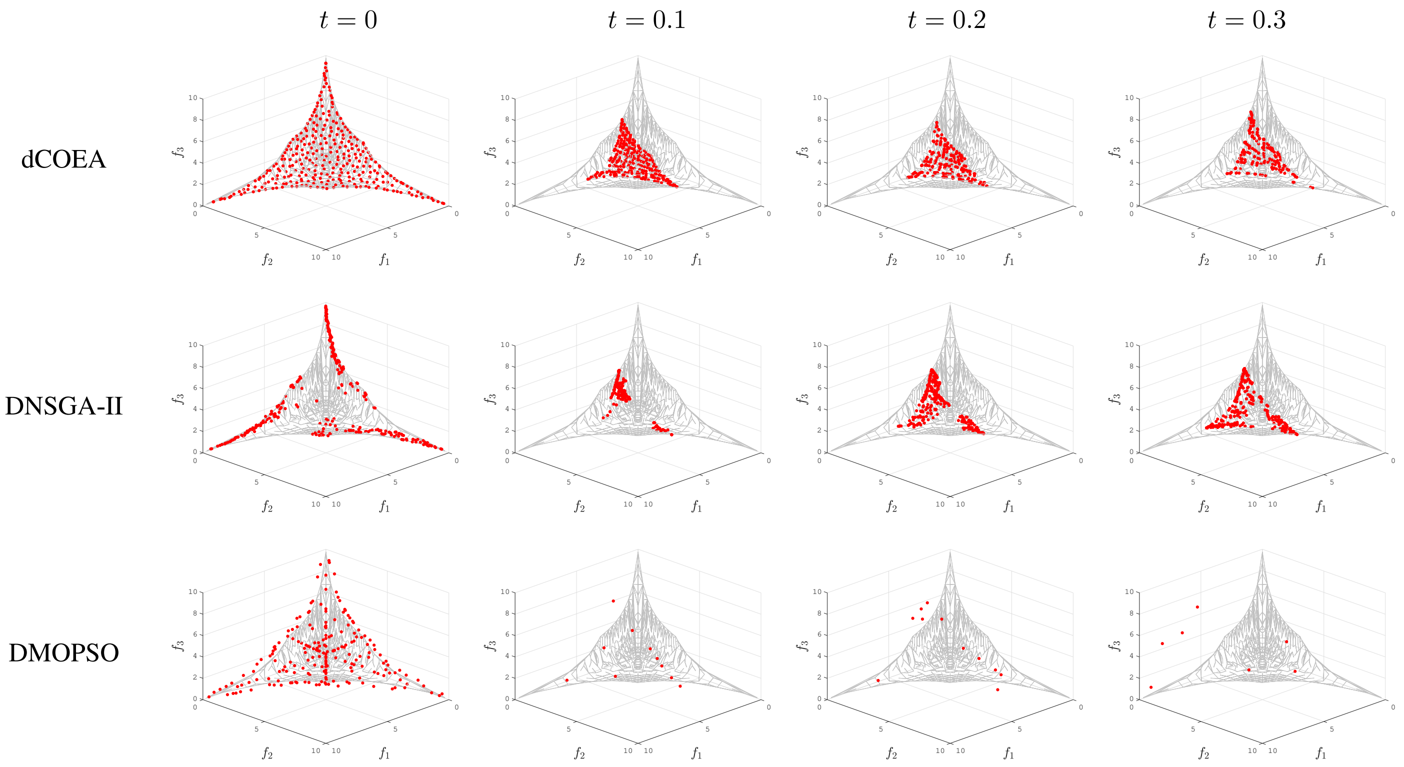

Fig. 5. PF approximations of three algorithms on SJY2 for the first four time steps.

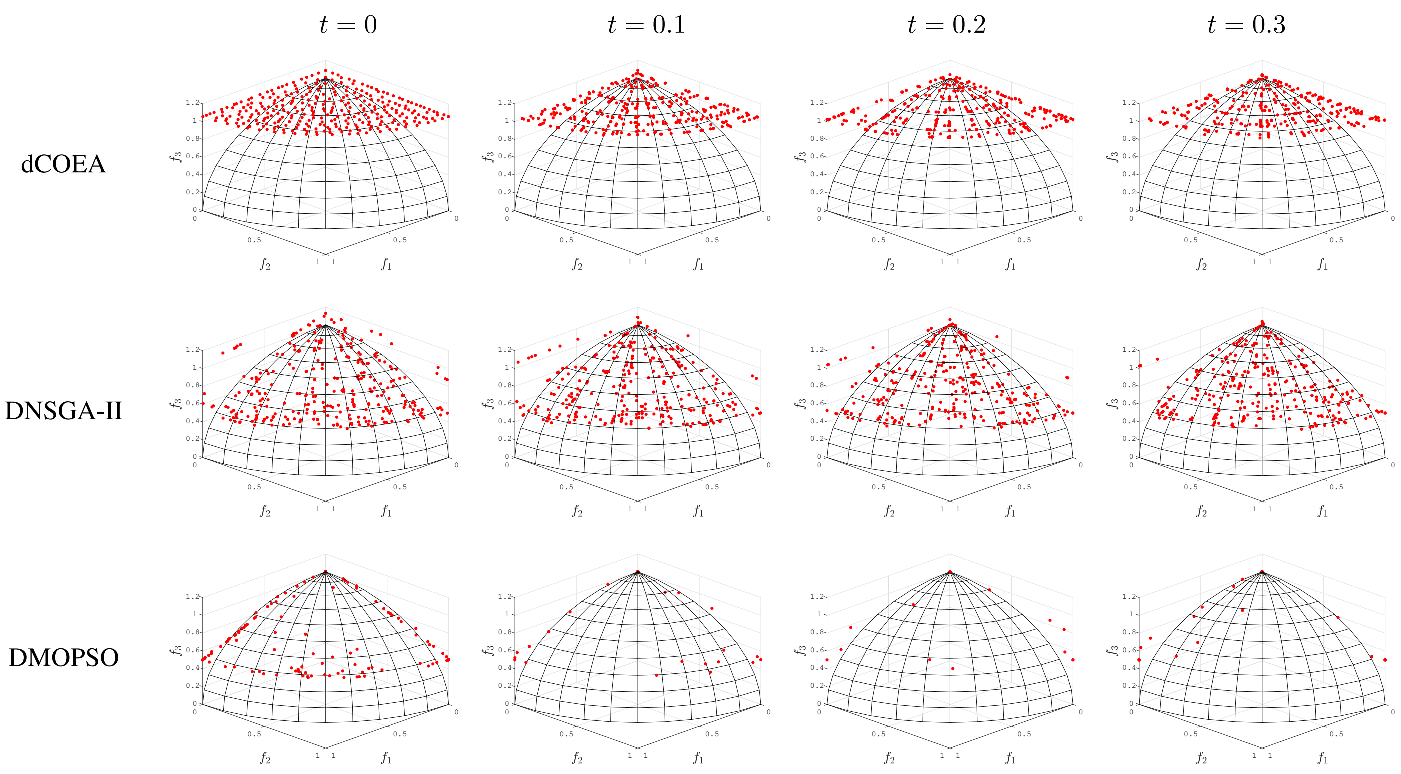

Fig. 6. PF approximations of three algorithms on SJY5 for the first four time steps.

Acknowledgements

This work was supported in part by the Engineering and Physical Sciences Research Council of U.K. under the project "Synthetic Portabolomics: Leading the way at the crossroads of the Digital and the Bio Economies (EP/N031962/1)" , and in part by the National Natural Science Foundation of China under Grant 61673331.

References

[1] S. Biswas, S. Das, P. N. Suganthan, and C. A. Coello Coello, "Evolutionary multiobjective optimization in dynamic environments: A set of novel benchmark functions," in: *Proc. 2014 IEEE Congr. Evol. Comput.*, 2014, pp. 3192–3199.

[2] J. Branke, "Memory enhanced evolutionary algorithms for changing optimization problems," in: *Proc. IEEE Congr. Evol. Comput.*, vol. 3, 1999, pp. 1875–1882.

[3] L. T. Bui, M. Zbignew, P. Eddy, and B. A. Manuel, "Adaptation in dynamic environments: A case study in mission planning," *IEEE Trans. Evolut. Comput.*, vol. 16, no. 2, pp. 190-209, 2002.

[4] M. Cámara, J. Ortega, F. de Toro, "A single front genetic algorithm for parallel multi-objective optimization in dynamic environments," *Neurocomputing* vol. 72, nos. 16–18, pp. 3570–3579, 2009.

[5] C. A. Coello Coello, 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.

[6] K. Deb, N. Rao U. B., S. Karthik, "Dynamic multi-objective optimization and decision-making using modified NSGA-II: A case study on hydro-thermal power scheduling," in: *Proc. 4th Int. Conf. Evol. Multi-criterion Optimization (EMO 2007)*, 2007, pp. 803–917.

[7] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scable test problems for evolutionary multi-objective optimization," Kanpur Genetic Algorithms Lab (KanGAL), Indian Inst. Technol., KanGAL Rep. 2001001, 2001.

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

[9] S. B. Gee, K. C. Tan, and H. A. Abbass, "A benchmark test suite for dynamic evolutionary multiobjective optimization," *IEEE Trans. Cybern.*vol. 47, no. 2, pp. 461–472, 2017.

[10] C. Goh and K. C. Tan, "A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 1, pp. 103–127, 2009.

[11] S. Guan, Q. Chen, and W. Mo, "Evolving dynamic multi-objective optimization problems with objective replacement, *Artif. Intell. Rev.*, vol. 23, no. 3, pp. 267–293, 2005.

[12] I. Hatzakis and D. Wallace, "Dynamic multiobjective optimization with evolutionary algorithms: a forward-looking approach," in *Proc. 8th Annual Conf. Genetic and Evol. Comput.*, 2006, pp. 1201–1208.

[13] M. Helbig, A. P. Engelbrecht, "Benchmarks for dynamic multi-objective optimisation algorithms", *ACM Comput. Surv. (CSUR)*, vol. 46, no. 3, pp. 1–37, 2014.

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

[15] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Trans. Evol. Comput.*, vol. 10, no. 2, pp. 477–506, 2006.

[16] S. Jiang and S. Yang, "A framework of scalable dynamic test problems for dynamic multi-objective optimization," in: *Proc. 2014 IEEE Symp. on Comput. Intell. in Dynamic and Uncertain Environments*, 2014, pp. 32–39.

[17] S. Jiang and S. Yang, "A steady-state and generational evolutionary algorithm for dynamic multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 21, no. 1, pp. 65–82, 2017.

[18] S. Jiang and S. Yang, "Evolutionary dynamic multi-objective optimization: Benchmarks and algorithm comparisons," *IEEE Trans. Cybern.*, vol. 47, no. 1, pp. 198–211, 2017.

[19] Y. Jin and B. Sendhoff, "Constructing dynamic optimization test prob-

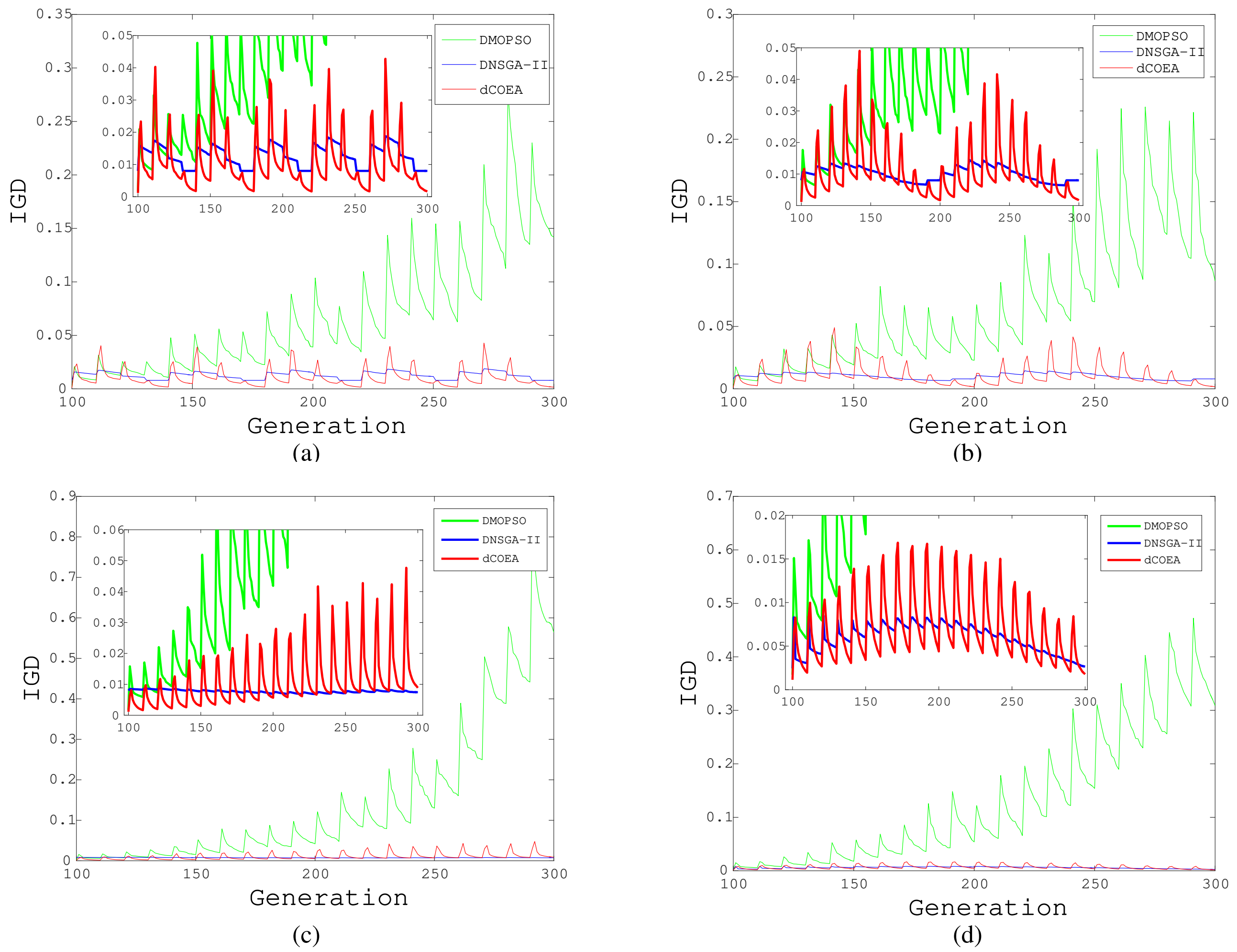

Fig. 7. The mean IGD curves obtained by three algorithms on SJY3 over 30 runs with various settings of n_t : (a) $n_t = 2$; (b) $n_t = 5$; (c) $n_t = 10$; (d) $n_t = 20$.

lems using the multi-objective optimization concept," in: *EvoWorkshops 2004: Appl. Evol. Comput.*, 2004, pp. 525–536.

[20] M. S. Lechuga, Multi-objective optimisation using sharing in swarm optimisation algorithms, Ph.D Dissertation, University of Birmingham, Birmingham, UK, 2009.

[21] J. Mehnen, G. Rudolph, and T. Wagner, Evolutionary optimization of dynamic multiobjective functions, Universitat Dortmund, Dortmund, Germany, Tech. Rep. FI-204/06, 2006.

[22] R. G. Miller, *Simultaneous Statistical Inference*, NJ, USA: Springer-Verlag, 1981.

[23] S. Nguyen, M. Zhang, M. Johnston, and K. C. Tan, "Automatic design of scheduling policies for dynamic multi-objective job shop scheduling via cooperative coevolution genetic programming," *IEEE Trans. Evol. Comput.* vol. 18, no. 2, pp. 193–208, 2014.

[24] C. Ramsey and J. Grefenstette, "Case-based initialization of genetic algorithms," in *Proc. 5th Int. Conf. Gentic Algorithms*, 1993, pp. 84–91.

[25] R. Shang, L. Jiao, Y. Ren, L. Li, and L. Wang, "Quantum immune clonal coevolutionary algorithm for dynamic multiobjective optimization," *Soft Comput.*, vol. 18, no. 4, pp. 743–756, 2014.

[26] E. Tantar, A. Tantar, P. Bouvry, "On dynamic multi-objective optimization classification and performance measures," in: *Proc. IEEE Congr. Evol. Comput.*, 2011, pp. 2759–2766.

[27] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics*, vol. 1, no. 6, pp. 80–83, 1945.

[28] P. P. Y. Wu, D. Campbell, and T. Merz, "Multiobjective four-dimensional vehicle motion planning in large dynamic environments," *IEEE Trans. Syst., Man, Cybern. Part B: Cybern.*, vol. 41, no. 3, pp. 621–634, 2011.

[29] S. Yang, "Associative memory scheme for genetic algorithms in dynamic environments," in *F. Rothlauf, et al. (eds.) EvoWorkshops 2006, LNCS*, vol. 3907, pp. 788-799, Springer, Heidelberg, 2006.

[30] Z. Zhang, "Multiobjective optimization immune algorithm in dynamic environments and its application to greenhouse control," *Appl. Soft Comput.*, vol. 8, no. 2, pp. 959–971, 2008.

[31] A. Zhou, Y. Jin, and Q. Zhang, "A population prediction strategy for evolutionary dynamic multiobjective optimization", *IEEE Trans. Cybern.* vol. 44, no. 1, pp. 40–53, 2014.

[32] A. Zhou, Y. Jin, Q. Zhang, B. Sendhoff, and E. Tsang, "Prediction-based population re-initialization for evolutionary dynamic multi-objective optimization," in: *Proc. 4th Int. Conf. Evol. Multi-Criterion Optim. (EMO 2007)*, vol. 4403, 2007, pp. 832-846.

[33] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evol. Comput.*, vol. 8, no. 2, pp. 173–195, 2000.

[34] A. R. Zomorrodi, M. M. Islam, and C. D. Maranas, "d-OptCom: Dynamic multi-level and multi-objective metabolic modeling of microbial communities," *ACS Synth. Biol.*, vol. 3, no. 4, pp. 247–257, 2014.