Benchmarking the performance of genetic algorithms on constrained dynamic problems

P. A. Grudniewski¹ · A. J. Sobey^{1,2}

Accepted: 3 July 2020 / Published online: 22 July 2020
© The Author(s) 2020

Abstract

The growing interest in dynamic optimisation has accelerated the development of genetic algorithms with specific mechanisms for these problems. To ensure that these developed mechanisms are capable of solving a wide range of practical problems it is important to have a diverse set of benchmarking functions to ensure the selection of the most appropriate Genetic Algorithm. However, the currently available benchmarking sets are limited to unconstrained problems with predominantly continuous characteristics. In this paper, the existing range of dynamic problems is extended with 15 novel constrained multi-objective functions. To determine how genetic algorithms perform on these constrained problems, and how this behaviour relates to unconstrained dynamic optimisation, 6 top-performing dynamic genetic algorithms are compared alongside 4 re-initialization strategies on the proposed test set, as well as the currently existing unconstrained cases. The results show that there are no differences between constrained/unconstrained optimisation, in contrast to the static problems. Therefore, dynamicity is the prevalent characteristic of these problems, which is shown to be more important than the discontinuous nature of the search and objective spaces. The best performing algorithm overall is MOEA/D, and VP is the best re-initialisation strategy. It is demonstrated that there is a need for more dynamic specific methodologies with high convergence, as it is more important to performance on dynamic problems than diversity.

Keywords Dynamic multi-objective optimisation · Constrained problems · Genetic algorithms · Performance benchmark · Re-initialisation

1 Dynamic optimisation problems

Many real-world problems have characteristics which change over time. Examples include forecasting of financial markets and stocks to increase profits in automated stock market investors (Bagheri et al. 2014); forecasting,

control systems and pattern recognition in traffic management to determine the scale of the traffic on the road and reduce the waiting time (Wismans et al. 2014); climate controlling systems for crop growing greenhouses (Zhang 2008) and LNG storage tank terminal design to reduce the cost of transporting LNG (Effendy et al. 2017). This is increasing the interest in optimisation of Dynamic Single-Objective Problems (Branke 2002) (DSOP) and Dynamic Multi-Objective Problems (Helbig and Engelbrecht 2015; Li et al. 2011; Farina et al. 2004) (DMOP), via Genetic Algorithms (GA).

In order to identify the mechanisms with the strongest performance on these types of problems there are a growing number of dynamic test sets such as: FDA (Farina et al. 2004), JY (Jiang and Yang 2017), UDF (Biswas et al. 2014), DSW (Mehnen et al. 2006), HE (Helbig and Engelbrecht 2014) and ZJZ (Zhou et al. 2007). These are used to replicate the most significant characteristics from different types of real-world problems. Current Dynamic Multi-Objective Problems exhibit various dynamic

Electronic supplementary material The online version of this article (<https://doi.org/10.1007/s11047-020-09799-y>) contains supplementary material, which is available to authorized users.

✉ P. A. Grudniewski
pag1c18@soton.ac.uk

A. J. Sobey
ajs502@soton.ac.uk

¹ Fluid Structure Interactions Group, University of Southampton, Southampton SO16 7QF, England, UK

² Marine and Maritime Group, Data-Centric Engineering, The Alan Turing Institute, The British Library, London NW1 2DB, England, UK

characteristics: search and objective space geometry changes, Pareto Optimal Front (POF) and Set (POS) curvature changes and shifts, discontinuities, modalities, periodicities, randomness of changes or predictabilities and variable-linkages (Helbig and Engelbrecht 2014) but all of the presented benchmarking sets are limited to unconstrained problems and are focused on solutions with continuous Pareto Optimal Fronts. Conducting benchmarking on a limited range of problem types can lead to overdevelopment of mechanisms with a strong bias towards these problems, which reduces their efficiency across the entire set of problems. Taking dynamic optimisation as an example, then since most of the current benchmarks utilise continuous characteristics there are difficulties when selecting the best solver for a constrained problem. It is also possible that the mechanism development is biased towards these problems and it is likely that these problems will require different mechanisms in the same way that static problems tend to have solvers specialised either for constrained or unconstrained problems (Zhang and Suganthan 2009). Testing on a wider set of problems is more likely to highlight potential improvements to the mechanisms and understanding the limitations of each mechanism is essential if they are to be utilised effectively in real-life applications.

Therefore, this paper introduces a novel constrained dynamic test set to improve the knowledge related to Genetic Algorithm performance on dynamic problems with the aim of broadening the benchmarking possibilities for the current state-of-the-art. A set of 15 constrained dynamic multi-objective problems are developed and 6 Genetic Algorithms are selected for their top performance on dynamic or constrained/discontinuous problems: NSGA-II (Deb et al. 2002), MOEA/D (Zhang and Li 2007), MLSGA-MTS (Grudniewski and Sobey 2018), MTS (Tseng and Chen 2009), HEIA (Lin et al. 2016) and BCE (Li et al. 2016), and their performance is compared on these problems. This performance is also tested in combination with different re-initialisation mechanisms, as these mechanisms make a considerable improvement to Genetic Algorithm performance on dynamic problems.

This paper is organised as follows: Sect. 2 presents a literature review of different Genetic Algorithms and dynamic optimisation specific mechanisms; Sect. 3 introduces a new set of constrained dynamic (CDF) multi-objective problems along with a review of currently existing sets; Sect. 4 presents a brief overview of the Genetic Algorithms and the re-initialisation mechanisms used in the benchmarking; Sect. 5 shows the results followed by conclusions in Sect. 6.

2 Increasing the diversity of the population for dynamic optimisation

In dynamic multi-objective problems, the diversity of solutions has to be maintained over the generations in order to achieve a complete set of results, the same as for static problems. However, the diversity of the gene pool must also be maintained through different time steps, meaning that diversity must be retained after convergence has occurred, unlike in static problems. To solve this problem a number of novel re-initialisation methodologies are proposed as additions to the current Genetic Algorithms, to improve performance on these dynamic problems (Nguyen et al. 2012). These methods generally enhance the diversity of the population whenever an environmental change occurs, without significant impact on the computational cost of the whole algorithm. Three major classes are found in the current state-of-the-art: niching and other diversity preservation schemes (Goh and Tan 2009), partial replacement of the population or hypermutation (Deb et al. 2007) and guided re-initialisation (Zhou et al. 2007).

In niching and other diversity preservation schemes the internal algorithm mechanisms are used in order to regain the diversity of the population after the environment changes. An example is dCOEA (Goh and Tan 2009) where a separate competitive process occurs when a dynamic change is detected, without the need to re-evaluate the whole population. Despite the positive impact on the final performance, the mechanisms from this methodology group often cannot be utilised outside of specific algorithms in which they were introduced and therefore the number of applications is limited. In addition, they show poor performance on problems with discontinuous search or objective spaces and cases where the environmental change is significant.

A second type of preservation scheme is hypermutation where a fraction of the set of potential solutions is modified by mutating the current individuals, or by replacing them with randomly generated new ones when an environmental change is detected. This method is implemented in the most commonly used Genetic Algorithm, NSGA-II (Deb et al. 2002), where the resulting algorithm is designated DNSGA-II (Deb et al. 2007), and is successfully tested on the FDA2 test function. However, the algorithm with the updated dynamic element is not compared to the original variant and therefore it is hard to determine how much benefit is gained from the hypermutation mechanism (Deb et al. 2007). Various benchmarks show that DNSGA-II struggles on more complex cases, and is outperformed by other algorithms, such as MOEA/D and dCOEA (Jiang and Yang 2017; Biswas et al. 2014), indicating that

hypermutation does not sufficiently maintain diversity over the time changes.

Re-initialisation methods are the final class of preservation scheme. Four main variants have been described by Zhou et al. (2007):

- 1 random, where the new population is initiated arbitrarily;
- 2 prediction-based, where the movement of the Pareto Optimal Set or Front is predicted, based on historical information, and the new population is sampled around the predicted location using Gaussian noise;
- 3 variation-based, where new points are created by varying the solution in the last time window with a “predicted” Gaussian noise, therefore only the information from the last time window is used, without the need to store the historical data;
- 4 the mixed method, variance and prediction (VP), where the variation method is applied to half of the population, and the prediction method to the rest.

In an initial benchmarking exercise the random variant is shown to be highly inefficient, as the algorithm needs to restart at every time change, the prediction-based method is shown to outperform the variation-based approach but the variance and prediction method shows the top performance as it can maintain benefits from both of these methods. However, all of the variants are benchmarked on a limited test set, ZJZ (Zhou et al. 2007) and FDA (Farina et al. 2004), and are only combined with a single algorithm, DMEA/PRI (Zhou et al. 2007), and therefore further testing is required to test the pervasiveness of these conclusions. The prediction-based methods are also investigated by Biswas et al. (2014) and a new strategy is introduced that relies on Controlled Extrapolation with a Pareto Optimal Front based on a distance approach (CER-POF). In this method, the positions of the “parent” solutions are evaluated based on the Pareto Optimal Front distances, instead of the Pareto Optimal Set distances as in Zhou et al. (2007), and the offspring points are sampled using Gaussian noise or a hyperbolic function, with equal probability. The proposed CER-POF method, combined with MOEA/D, is shown to outperform the prediction-based re-initialisation strategy presented by Zhou et al. (2007), and the hypermutation-based DNSGA-II algorithm (Deb et al. 2007). However, it is not compared to MOEA/D without re-initialisation, in combination with the VP method or to the Pareto Optimal Set based prediction introduced by Zhou et al. (2007), which limits the generality of the conclusions. In addition, the prediction-based re-initialisation methods assume that the change of the Pareto Optimal Set or Pareto Optimal Front follow specific patterns, which can be found and exploited to “predict” the next time step. The performance of these methods on problems where a

pattern does not exist or cannot be found easily, has not been properly evaluated despite it being the case for many real-world applications.

The re-initialisation methods have shown to increase the effectiveness of Genetic Algorithms on Dynamic Multi-Objective Problems without significant impact on the computational cost. These mechanisms outperform the hypermutation-based algorithms and exhibit comparable performance to niching-based approaches. Three re-initialisation schemes: CER-POF, VP and CER-POS, are selected as the most promising mechanisms for dynamic problems due to their combination of high performance and flexibility, where the Controlled Extrapolation with a Pareto Optimal Set based distance approach (CER-POS) Biswas et al. (2014) is introduced in order to determine the importance of different nearest-distance definitions on dynamic problems.

3 A test set for constrained dynamic problems

To increase the range of dynamic problems this paper develops a set of 15 novel two-objective Constrained Dynamic Functions (CDF),¹ which reflect the constrained and discontinuous problem types found in many applications. These modify the Constrained Functions (CF) developed for the CEC '09 multiobjective competition (Zhang et al. 2009) according to the dynamic function framework proposed by Biswas et al. (2014), as it allows straightforward modification of static functions into dynamic problems. Only two-objective problems are considered as these allow a simpler interpretation of the results, without issues of scaling, while providing enough complexity to resemble a number of real-world problems, such as (Bagheri et al. 2014; Wismans et al. 2014; Zhang 2008; Effendy et al. 2017).

3.1 General summary of dynamic problems

According to Farina et al. (2004) 4 types of dynamics can be distinguished: type I, where changes are made to the Pareto Optimal Set only; type III, where the Pareto Optimal Front is dynamic; type II, where both the Pareto Optimal Set and the Front are subject to environmental changes and type IV, where only the objective function or constraints are altered, but both the Pareto Optimal Front and the Set remain static. Part of developing a new set of constrained problems is determining to what extent the constraint

¹ These functions are available in C++ at: <https://www.bitbucket.org/Pag1c18>.

should be time dependant in relation to the Pareto Optimal Front and the Set, three cases are considered:

- (a) A static constraint where no time dependency is introduced. However, despite the constraint itself being static the effect of the constraint, and the resulting regions where there are no feasible solutions, are not constant due to the time dependency of the search and objective spaces. Therefore, the Pareto Optimal Front and Set change with respect to the resulting gaps.
- (b) A dynamic constraint where the time dependency is bonded to the dynamic change of the optimisation problem's Pareto Optimal Front and Set. In this case, the applied constraints follow the dynamic change in the pattern of the optimisation problem as the same time-dependency is applied to both of them. Therefore, the resulting gaps in both the Pareto Optimal Set and Pareto Optimal Front do not change.
- (c) A dynamic constraint where the dynamic change is independent from the change in the optimisation problem. In this case, the constraint is subject to different dynamic changes than those applied to both the Pareto Optimal Set and Pareto Optimal Front, or they remain static. Therefore, the resulting gaps on both the search and objective spaces shift independently every time step, making the problem harder to solve.

3.2 Test instances

In this paper, the CDF test set is developed to ensure a range of different problem types by considering all 3 cases, with changes to the location of the Pareto Optimal Set or Front, changes to the geometries of the search and objective space and dynamic or static constraints introduced to the variables and objectives. Due to the quantity of potential combinations for these parameters, orthogonal design is used to develop 12 representative functions as described in Table 1; this is performed according to the types of dynamic problems and constraints, changes in Pareto Optimal Front and Pareto Optimal Set, and the continuity of both Pareto Optimal Front and Pareto Optimal Set. This generates a set of problems including at least one of each of the type I-III problems with different Pareto Optimal Front and Pareto Optimal Set geometries. Additionally, one problem with a random dynamic change is introduced, as the random element greatly increases the complexity of a problem as the change cannot be predicted; and two problems of type IV are added in order to investigate the different types of time-dependency in constraints with static variable and objective spaces. All of the functions are to be minimised and have 10 decision variables.

CDF1: is generated by combining the UDF2 problem with two variable constraints; the resulting function is a Type I problem. This combination allows investigation of the behaviour of algorithms on continuous objective spaces in dynamic Pareto Optimal Set environments. However, as the constraint is also dynamic relative to the change in

Table 1 Summary of the constrained dynamic test set

Name	Type of dynamic problem	Type of constraint	Continuity	Type of POF/POS change	Constraints
CDF1	I	Dynamic	Continuous	Curvature change	2 variable constraints
CDF2	I	Static	Continuous	Curvature change; horizontal shift	1 variable constraint
CDF3	I	Dynamic	Discontinuous	Curvature change	1 variable constraint
CDF4	I	Static	Discontinuous	Vertical shift	1 objective constraint
CDF5	II	Dynamic	Continuous	Diagonal shift	1 variable constraint
CDF6	II	Static	Continuous	Diagonal shift	2 variable constraints
CDF7	II	Dynamic	Discontinuous	Curvature change; diagonal shift	1 objective constraint
CDF8	II	Static	Discontinuous	Curvature change	1 objective constraint
CDF9	III	Dynamic	Continuous	Curvature change; diagonal shift	2 variable constraints
CDF10	III	Static	Continuous	Curvature change	1 variable constraint
CDF11	III	Dynamic	Discontinuous	Horizontal shift	1 variable constraint
CDF12	III	Static	Discontinuous	Curvature change; angular shift	1 objective constraint
CDF13	II	Static	Discontinuous	Random curvature change or shift	1 objective constraint
CDF14	IV	Dynamic	Discontinuous	Continuity change	1 objective constraint
CDF15	IV	Dynamic	Discontinuous	Discontinuity shift	1 objective constraint

Fig. 1 Pareto optimal front for CDF1 over the first 11 times steps

$\sin(t)$ the infeasible regions are not observed over time, which reduces the overall complexity of the problem. The problem is described on Eq. 1 with the Pareto Optimal Front illustrated in Fig. 1.

$$\begin{aligned} f_1 &= x_1 + \frac{2}{|J_1|} \sum_{j \in J_1} (x_i - x_1^y)^2 \\ f_2 &= (1 - x_1)^2 + \frac{2}{|J_2|} \sum_{j \in J_2} (x_i - x_1^y)^2 \\ y_j &= 0.5 \left(1 + \frac{3(j-2)}{n-2} \right) + |G(t)|, \quad j = 2, \dots, n \\ x &\in [0, 1] \times [-1, 2]^{n-1} \\ g_1(x, t) &= \begin{cases} x_2 - x_1^{(0.5*2+|G(t)|-sign(k_1))\sqrt{|k_1|}} \\ k_1 = 0.5(1 - x_1) - (1 - x_1)^2 \end{cases} \\ g_2(x, t) &= \begin{cases} x_4 - x_1^{(0.5*(2+\frac{3*(4-2)}{n-2})+|G(t)|-sign(k_2))\sqrt{|k_2|}} \\ k_2 = 0.25\sqrt{1-x_1} - 0.5(1 - x_1) \end{cases} \\ POF : &\begin{cases} f_2 = (1 - f_1)^2 \\ 0 \leq f_1 \leq 1 \end{cases} \end{aligned} \quad (1)$$

CDF2: introduces a vertical shift of the Pareto Optimal Set into the CF4; the resulting problem is Type I. In comparison to CDF1, the introduced constraint is static and limits the obtainable Pareto Optimal Front values on the right-hand side of objective space. The complexity is due to the high multimodality with an uneven distribution of points across the Pareto Optimal Front. The problem is described in Eq. 2 with the Pareto Optimal Front illustrated in Fig. 2.

Fig. 2 Pareto optimal front for CDF2 over the first 11 times steps

$$\begin{aligned} f_1 &= x_1 + \sum_{j \in J_1} w_j(y_j) \\ f_2 &= 1 - x_1 + \sum_{j \in J_2} w_j(y_j) \\ y_j &= x_j - \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), \quad j = 2, \dots, n \\ w_2(z) &= \begin{cases} |z| & \text{if } z < 1.5(1 - 0.5\sqrt{2}) \\ 0.125 + (z - 1)^2 & \text{otherwise} \end{cases} \\ w_j(z) &= (z - G(t))^2, \quad j = 3, \dots, n \\ x &\in [0, 1] \times [-2, 2]^{n-1} \\ g(x, t) &= \begin{cases} \frac{a}{1 + e^{4|a|}} \\ a = x_2 - \sin\left(6\pi x_1 + \frac{2\pi}{n}\right) - 0.5x_1 + 0.25 \end{cases} \\ POF : &\begin{cases} f_2 = \begin{cases} 1 - f_1 & \text{if } 0 \leq f_1 \leq 0.5 \\ -0.5f_1 + 0.75 & \text{else if } f_1 \leq 0.75 \\ 1 - f_1 + 0.125 & \text{else if } f_1 \leq 1 \end{cases} \\ 0 \leq f_1 \leq 1 \end{cases} \end{aligned} \quad (2)$$

CDF3: is a combination of the UDF3 with a simple dynamic, variable constraint. The resulting function is a type I problem, but unlike CDF1 and CDF2, a high discontinuity of both objective and search spaces can be observed and CDF3 requires a higher diversity of the search. The problem is described in Eq. 3 with the Pareto Optimal Front illustrated in Fig. 3.

Fig. 3 Pareto optimal front for CDF3 over the first 11 times steps

$$\begin{aligned}
 f_1 &= x_1 + h + \frac{2}{|J_1|} \sum_{j \in J_1} (x_i - x_1^{y_j})^2 \\
 f_2 &= 1 - x_1 + h + \frac{2}{|J_2|} \sum_{j \in J_2} (x_i - x_1^{y_j})^2 \\
 h &= \left(\frac{0.5}{N} + \epsilon \right) |\sin(2N\pi x_1)| \\
 y_j &= 0.5 \left(2 + \frac{3(j-2)}{n-2} \right) + |G(t)|, \quad j = 2, \dots, n \\
 x &\in [0, 1] \times [-1, 1]^{n-1} \\
 N &= 10; \epsilon = 0.1 \\
 g(x, t) &= \begin{cases} x_2 - x_1^{(1+|G(t)|)} \\ \text{for } i = 0, 1, \dots, 2N \end{cases} \\
 POF &: \begin{cases} f_1 = \frac{i}{2N} \\ f_2 = 1 - \frac{i}{2N} \\ \text{for } i = 0, 1, \dots, 2N \end{cases}
 \end{aligned} \tag{3}$$

CDF4: is generated by combining the objective constraint from CF1 with a UDF2 dynamic shift; the resulting function is a Type I problem. The complexity of the problem is higher than of CDF3, due to the utilisation of the static constraint where the relative position of the infeasible region changes every time step. The problem is described in Eq. 4 with the Pareto Optimal Front illustrated in Fig. 4.

Fig. 4 Pareto optimal front for CDF4 over the first 11 times steps

$$\begin{aligned}
 f_1 &= x_1 + \frac{2}{|J_1|} \sum_{j \in J_1} (x_i - x_1^{y_j})^2 \\
 f_2 &= 1 - x_1^2 + \frac{2}{|J_2|} \sum_{j \in J_2} (x_i - x_1^{y_j})^2 \\
 y_j &= 0.5 \left(1 + \frac{3(j-2)}{n-2} \right) + |G(t)|, \quad j = 2, \dots, n \\
 x &\in [0, 1] \times [-2, 2]^{n-1} \\
 g(x, t) &= f_1 + f_2 - |\sin(N\pi(f_1 - f_2 + 1))| - 1 \\
 N &= 10 \\
 POF^* &: \begin{cases} f_2 = 1 - f_1^2 \\ 0 \leq f_1 \leq 1 \end{cases}
 \end{aligned} \tag{4}$$

CDF5: the CF5 is combined with the UDF1 constraint resulting in CDF5, which is a type II problem with a dynamic constraint. A diagonal shift in both the Pareto Optimal Set and Front is introduced, and infeasibility is not dependant on time. The high complexity of this problem is the result of the high multimodality and an uneven distribution of points across the Pareto Optimal Front, which is similar to CDF2. The problem is described in Eq. 5 with the Pareto Optimal Front illustrated in Fig. 5.

Fig. 5 Pareto optimal front for CDF5 over the first 11 times steps

$$\begin{aligned}
 f_1 &= x_1 + |G(t)| + \sum_{j \in J_1} w_j(y_j) \\
 f_2 &= 1 - x_1 + |G(t)| + \sum_{j \in J_2} w_j(y_j) \\
 y_j &= \begin{cases} x_j - 0.8x_1 \cos\left(6\pi x_1 + \frac{j\pi}{n}\right) - G(t), & j \in J_1 \\ x_j - 0.8x_1 \sin\left(6\pi x_1 + \frac{j\pi}{n}\right) - G(t), & j \in J_2 \end{cases} \\
 w_2(z) &= \begin{cases} |z| & \text{if } z < 1.5(1 - 0.5\sqrt{2}) \\ 0.125 + (z - 1)^2 & \text{otherwise} \end{cases} \\
 w_j(z) &= 2z^2 - \cos(4\pi z) + 1, \quad j = 3, \dots, n \\
 x &\in [0, 1] \times [-2, 2]^{n-1} \\
 g(x, t) &= x_2 - 0.8x_1 \sin\left(6\pi x_1 + \frac{2\pi}{n}\right) - 0.5x_1 + 0.25 - G(t) \\
 POF : \begin{cases} f_1 = y_1 + |G(t)|; & f_2 = y_2 + |G(t)| \\ y_2 = \begin{cases} 1 - y_1 & \text{if } 0 \leq y_1 \leq 0.5 \\ -0.5y_1 + 0.75 & \text{else if } y_1 \leq 0.75 \\ 1 - y_1 + 0.125 & \text{else if } y_1 \leq 1 \\ 0 \leq y_1 \leq 1 \end{cases} \end{cases}
 \end{aligned} \quad (5)$$

CDF6: the UDF1 is combined with the CF6 constraint resulting in a type II problem. This problem is similar to the CDF5 in terms of the changes in both Pareto Optimal Set and Front. Two constraints are incorporated, compared to one in the CDF5 and the utilised objective function is easier to solve due to the higher uniformity of the Pareto Optimal Front. The problem is described in Eq. 6 with the Pareto Optimal Front illustrated in Fig. 6.

Fig. 6 Pareto optimal front for CDF6 over the first 11 times steps

$$\begin{aligned}
 f_1 &= x_1 + |G(t)| + \sum_{j \in J_1} (y_j)^2 \\
 f_2 &= (1 - x_1)^2 + |G(t)| + \sum_{j \in J_2} (y_j)^2 \\
 y_j &= \begin{cases} x_j - 0.8x_1 \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), & j = 2, 4 \\ x_j - 0.8x_1 \cos\left(6\pi x_1 + \frac{j\pi}{n}\right) - |G(t)|, & j \in J_1 \\ x_j - 0.8x_1 \sin\left(6\pi x_1 + \frac{j\pi}{n}\right) - |G(t)|, & j \in J_2 - \{2, 4\} \end{cases} \\
 x &\in [0, 1] \times [-2, 2]^{n-1} \\
 g_1(x, t) &= \begin{cases} x_2 - |G(t)| - 0.8x_1 \sin\left(6\pi x_1 + \frac{2\pi}{n}\right) - \text{sign}(k_1)\sqrt{|k_1|} \\ k_1 = 0.5(1 - x_1) - (1 - x_1)^2 \end{cases} \\
 g_2(x, t) &= \begin{cases} x_4 - |G(t)| - 0.8x_1 \sin\left(6\pi x_1 + \frac{4\pi}{n}\right) - \text{sign}(k_2)\sqrt{|k_2|} \\ k_2 = 0.25\sqrt{1 - x_1} - 0.5(1 - x_1) \end{cases} \\
 POF : \begin{cases} f_1 = y_1 + |G(t)|; & f_2 = y_2 + |G(t)| \\ y_2 = \begin{cases} (1 - y_1)^2 & \text{if } 0 \leq y_1 \leq 0.5 \\ 0.5(1 - y_1) & \text{else if } y_1 \leq 0.75 \\ 0.25\sqrt{1 - y_1} & \text{else if } y_1 \leq 1 \\ 0 \leq y_1 \leq 1 \end{cases} \end{cases}
 \end{aligned} \quad (6)$$

CDF7: is generated by combining the CF1 with a UDF2 dynamic shift; the resulting function is a Type II problem. This combination allows investigation of the behaviour of algorithms on highly discontinuous spaces in dynamic Pareto Optimal Set and Front environments. However, as the constraint is also dynamic then changes in the infeasible regions are not observed over time, which reduces the overall complexity of the problem. The problem is described in Eq. 7 with the Pareto Optimal Front illustrated in Fig. 7. This function allows an investigation into the performance of algorithms based on a diagonal shift in the

Fig. 7 Pareto optimal front for CDF7 over the first 11 times steps

Pareto Optimal Front and vertical shift with curvature change of the Pareto Optimal Set on a highly discontinuous space, without relative changes to the positions of the infeasible regions over time.

$$\begin{aligned}
 f_1 &= x_1 + |G(t)| + \frac{2}{|J_1|} \sum_{j \in J_1} (x_j - x_1^{y_j} - G(t))^2 \\
 f_2 &= 1 - x_1 + |G(t)| + \frac{2}{|J_2|} \sum_{j \in J_2} (x_j - x_1^{y_j} - G(t))^2 \\
 y_j &= 0.5 \left(1 + \frac{3(j-2)}{n-2} \right), \quad j = 2, \dots, n \\
 x &\in [0, 1] \times [-2, 2]^{n-1} \\
 g(x, t) &= f_1 + f_2 - 2|G(t)| - |\sin[N\pi(f_1 - f_2 + 1)]| - 1 \\
 N &= 10 \\
 \text{POF} : \text{ points: } &\left(\frac{i}{2N} + |G(t)|, 1 - \frac{i}{2N} + |G(t)| \right), \quad i = 0, 1, \dots, 2N
 \end{aligned} \tag{7}$$

CDF8: the UDF5 is combined with the CF3 constraint to form CDF4. This problem has a similar Pareto Optimal Front pattern to CDF4. However, an angular shift of the Pareto Optimal Set with a curvature change is included and this function introduces a constraint in a more continuous objective space. In addition, the objective function is less complex but a larger search space is utilised. The problem is described in Eq. 8 with the Pareto Optimal Front illustrated in Fig. 8.

Fig. 8 Pareto optimal front for CDF8 over the first 11 times steps

$$\begin{aligned}
 f_1 &= x_1 + \frac{2}{|J_1|} \sum_{j \in J_1} (x_j - x_1^{y_j})^2 \\
 f_2 &= 1 - M(t)(x_1)^{H(t)} + \frac{2}{|J_2|} \sum_{j \in J_2} (x_j - x_1^{y_j})^2 \\
 y_j &= 0.5 \left(2 + \frac{3(j-2)}{n-2} \right), \quad j = 2, \dots, n \\
 x &\in [0, 1] \times [-1, 2]^{n-1} \\
 g(x, t) &= \sqrt{f_1} + f_2 - \sin[2\pi(\sqrt{f_1} - f_2 + 1)] - 1 \\
 \text{POF}^* : f_2 &= 1 - M(t)(f_1)^{H(t)}, \quad 0 \leq f_1 \leq 1
 \end{aligned} \tag{8}$$

CDF9: is a modified CDF6 function, with the dynamic change removed from the Pareto Optimal set and instead added to the constraints. This results in a Type III problem, with a significant curvature change in the Pareto Optimal front. The problem is described in Eq. 9 with the Pareto Optimal Front illustrated in Fig. 9.

Fig. 9 Pareto optimal front for CDF9 over the first 11 times steps

$$\begin{aligned}
 f_1 &= x_1 + |G(t)| + \sum_{j \in J_1} (y_j)^2 \\
 f_2 &= [1 - (M(t)x_1)^{H(t)}]^2 + |G(t)| + \sum_{j \in J_2} (y_j)^2 \\
 y_j &= \begin{cases} x_j - 0.8x_1 \cos\left(6\pi x_1 + \frac{j\pi}{n}\right), & j \in J_1 \\ x_j - 0.8x_1 \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), & j \in J_2 - \{2, 4\} \end{cases} \\
 x &\in [0, 1] \times [-2, 2]^{n-1} \\
 g_1(x, t) &= \begin{cases} x_2 - |G(t)| - 0.8x_1 \sin\left(6\pi x_1 + \frac{2\pi}{n}\right) - \text{sign}(k_1) \sqrt{|k_1|} \\ k_1 = 0.5[1 - (M(t)x_1)^{H(t)}] - [1 - (M(t)x_1)^{H(t)}]^2 \end{cases} \\
 g_2(x, t) &= \begin{cases} x_4 - |G(t)| - 0.8x_1 \sin\left(6\pi x_1 + \frac{4\pi}{n}\right) - \text{sign}(k_2) \sqrt{|k_2|} \\ k_2 = 0.25\sqrt{1 - (M(t)x_1)^{H(t)}} - 0.5[1 - (M(t)x_1)^{H(t)}] \end{cases} \\
 POF : \begin{cases} f_1 = y_1 + |G(t)|; & f_2 = y_2 + |G(t)| \\ y_2 = \begin{cases} [1 - z]^2 & \text{if } 0 \leq z \leq 0.5 \\ 0.5[1 - z] & \text{else if } z \leq 0.75 \\ 0.25\sqrt{1 - z} & \text{else if } z \leq 1 \end{cases} \\ z = (M(t)y_1)^{H(t)} \\ 0 \leq y_1 \leq 1 \end{cases}
 \end{aligned} \tag{9}$$

CDF10: is generated by adding curvature variation of the Pareto Optimal Front to the CF7; the resulting function is a Type III problem. Due to the utilisation of the static constraint the time complexity of the problem is higher than for CDF9, but the complexity overall is lower. The problem is described in Eq. 10 with the Pareto Optimal Front illustrated in Fig. 10;

Fig. 10 Pareto optimal front for CDF10 over the first 11 times steps

$$\begin{aligned}
 f_1 &= x_1 + \sum_{j \in J_1} (w_j(y_j))^2 \\
 f_2 &= (1 - x_1)^{H(t)} + \sum_{j \in J_2} (w_j(y_j))^2 \\
 y_j &= \begin{cases} x_j - \cos\left(6\pi x_1 + \frac{j\pi}{n}\right), & j \in J_1 \\ x_j - \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), & j \in J_2 \end{cases} \\
 w_2(z) &= w_4(z) = z^2 \\
 w_j(z) &= 2z^2 - \cos(4\pi z) + 1, \quad j = 3, \dots, n \\
 x &\in [0, 1] \times [-2, 2]^{n-1} \\
 g(x, t) &= x_2 - 0.8x_1 \sin\left(6\pi x_1 + \frac{2\pi}{n}\right) - 0.5x_1 + 0.25 - G(t) \\
 POF : \begin{cases} f_2 = \begin{cases} (1 - f_1)^{H(t)} & \text{if } 0 \leq f_1 \leq 0.5 \\ (1 - f_1)^{H(t)} - (1 - f_1)^2 + 0.5 * (1 - f_1) & \text{else if } f_1 \leq 0.75 \\ (1 - f_1)^{H(t)} - (1 - f_1)^2 + 0.25 * \sqrt{1 - f_1} & \text{else if } f_1 \leq 1 \\ 0 \leq f_1 \leq 1 \end{cases} \end{cases}
 \end{aligned} \tag{10}$$

CDF11: is a modified CDF3 function, with the dynamic change removed from the Pareto Optimal set, but added to the constraints. This results in a Type III problem, with a vertical change of the Pareto Optimal front rather than changes in the curvature. The problem is described in Eq. 11 with the Pareto Optimal Front illustrated in Fig. 11.

Fig. 11 Pareto optimal front for CDF11 over the first 11 times steps

$$\begin{aligned}
 f_1 &= x_1 + |G(t)| + h + \sum_{j \in J_1} w_j(y_j) \\
 f_2 &= 1 - x_1 + |G(t)| + h + \sum_{j \in J_2} w_j(y_j) \\
 h &= \left(\frac{0.5}{N} + \epsilon \right) |\sin(2N\pi x_1 + G(t)\pi)| \\
 y_j &= \begin{cases} x_j - 0.8x_1 \cos\left(6\pi x_1 + \frac{j\pi}{n}\right), & j \in J_1 \\ x_j - 0.8x_1 \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), & j \in J_2 \end{cases} \\
 w_2(z) &= \begin{cases} |z| & \text{if } z < 1.5(1 - 0.5\sqrt{2}) \\ 0.125 + (z - 1)^2 & \text{otherwise} \end{cases} \\
 w_j(z) &= 2z^2 - \cos(4\pi z) + 1, \quad j = 3, \dots, n \\
 x &\in [0, 1] \times [-1, 1]^{n-1} \\
 N &= 10; \epsilon = 0.1 \\
 g(x, t) &= x_2 - 0.8x_1 \sin\left(6\pi x_1 + \frac{2\pi}{n}\right) - 0.5x_1 + 0.25 \\
 POF : \begin{cases} f_1 = \frac{i - G(t)}{2N} \\ f_2 = \begin{cases} 1 - f_1 & \text{if } 0 \leq f_1 \leq 0.5 \\ -0.5f_1 + 0.75 & \text{else if } f_1 \leq 0.75 \\ 1 - f_1 + 0.125 & \text{else if } f_1 \leq 1 \end{cases} \\ \text{for } i = 0, 1, \dots, 2N \end{cases}
 \end{aligned} \tag{11}$$

CDF12: introduces angular shift and curvature variations of the Pareto Optimal Front into the CF2. A static objective constraint is introduced and the resulting function is a Type III problem, as no environmental change is applied to the Pareto Optimal Set. This function allows the behaviour in the constrained and dynamic Pareto Optimal Front environments to be evaluated. The problem is described in Eq. 12 with the Pareto Optimal Front illustrated in Fig. 12.

Fig. 12 Pareto optimal front for CDF12 over the first 11 times steps

$$\begin{aligned}
 f_1 &= x_1 + \frac{2}{|J_1|} \sum_{j \in J_1} y_j^2 \\
 f_2 &= 1 - (x_1)^{H(t)} + \frac{2}{|J_2|} \sum_{j \in J_2} y_j^2 \\
 y_j &= \begin{cases} x_j - 0.8x_1 \cos\left(6\pi x_1 + \frac{j\pi}{n}\right), & j \in J_1 \\ x_j - 0.8x_1 \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), & j \in J_2 \end{cases} \\
 x &\in [0, 1] \times [-1, 1]^{n-1} \\
 g(x, t) &= \begin{cases} \frac{a}{1 + e^{4|a|}} \\ a = \sqrt{f_1} + f_2 - \sin[2\pi(\sqrt{f_1} - f_2 + 1)] - 1 \end{cases} \\
 POF^* : f_2 &= 1 - (f_1)^{H(t)}, \quad 0 \leq f_1 \leq 1
 \end{aligned} \tag{12}$$

CDF13: the UDF8 is combined with the CF4 constraint resulting in the CDF13. This problem is type II, and uses the same objective function as CDF9. However, the constraint is dynamic as a random change is introduced. The random element greatly increases the complexity of the problem as the change cannot be predicted. The problem is described in Eq. 13 with the Pareto Optimal Front illustrated in Fig. 13.

In CDF13, several types of dynamic changes are incorporated, in a similar manner to UDF8, but only one of them is employed at a time; during the environmental change one of t_1 to t_5 is selected randomly and is increased by 1.

Fig. 13 Pareto optimal front for CDF13 over the first 11 times steps

Fig. 14 Pareto optimal front for CDF14 over the first 11 times steps

$$\begin{aligned}
 f_1 &= x_1 + |G(t_3)| + \frac{2}{|J_1|} \sum_{j \in J_1} y_j^2 \\
 f_2 &= 1 - M(t_4)(x_1)^{H(t_5)} + |G(t_3)| + \frac{2}{|J_2|} \sum_{j \in J_2} y_j^2 \\
 y_j &= x_j - \sin\left(6\pi x_1 + \frac{(j + K(t_1))\pi}{n}\right) - G(t_2), \quad j = 2, \dots, n \\
 x &\in [0, 1] \times [-2, 2]^{n-1} \\
 g(x, t) &= \begin{cases} w + f_2 - \sin[2\pi(w - f_2 + 1)] - 1 \\ w = M(t_4)(f_1)^{H(t_5)} \end{cases} \\
 POF^* : &\begin{cases} f_2 = 1 - M(t_4)(f_1 - |G(t_3)|)^{H(t_5)} + |G(t_3)|, \\ 0 + |G(t_3)| \leq f_1 \leq 1 + |G(t_3)| \end{cases}
 \end{aligned} \quad (13)$$

CDF14: introduces a time-dependent constraint into the static CF1. The additional constraint is independent of the change in objective function and the result is a Type IV problem. The time-dependency of the constraint changes the geometry of the Pareto Optimal Front and the Set over time from distinct points, through to disconnected fronts and finally to a continuous set. This means that retaining the diversity of solutions over time steps is essential, especially when the sizes of the infeasible regions are decreasing. Due to this the boundary ranges are lowered, in order to decrease the complexity. The problem is described in Eq. 14 with the Pareto Optimal Front illustrated in Fig. 14.

$$\begin{aligned}
 f_1 &= x_1 + \frac{2}{|J_1|} \sum_{j \in J_1} (x_j - x_1^{y_j})^2 \\
 f_2 &= 1 - x_1 + \frac{2}{|J_2|} \sum_{j \in J_2} (x_j - x_1^{y_j})^2 \\
 y_j &= 0.5 \left(1 + \frac{3(j-2)}{n-2}\right), \quad j = 2, \dots, n \\
 x &\in [0, 1]^n \\
 g(x, t) &= f_1 + f_2 - |\sin[N\pi(f_1 - f_2 + 1)]| - 1 + |G(t)| \\
 N &= 10 \\
 POF^* : f_2 &= 1 - f_1, \quad 0 \leq f_1 \leq 1
 \end{aligned} \quad (14)$$

CDF15: is Type IV, similar to CDF14, as the time-dependent constraint is introduced into the CF3, without any direct environmental changes in objective or variable spaces. However, in this case the position of the infeasible regions is shifting in both the search and objective spaces, instead of changing the size of them. The problem is described in Eq. 15 with the Pareto Optimal Front illustrated in Fig. 15.

$$\begin{aligned}
 f_1 &= x_1 + \frac{2}{|J_1|} \sum_{j \in J_1} y_j^2 \\
 f_2 &= 1 - x_1^2 + \frac{2}{|J_2|} \sum_{j \in J_2} y_j^2 \\
 y_j &= x_j - \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), \quad j = 2, \dots, n \\
 x &\in [0, 1] \times [-2, 2]^{n-1} \\
 g(x, t) &= f_1^2 + f_2 - \sin[2\pi(f_1^2 - f_2 + 1) + G(t)] - 1 \\
 POF^* : f_2 &= 1 - f_1^2, \quad 0 \leq f_1 \leq 1
 \end{aligned} \quad (15)$$

Fig. 15 Pareto optimal front for CDF15 over the first 11 times steps

4 Methodology

4.1 Selection of mechanisms and unconstrained test problems

The performances of different Genetic Algorithms are evaluated in combination with different re-initialization methods over the developed test instances, CDF1-15. Furthermore, eighteen bi-objective unconstrained functions are used to evaluate the differences in behaviour of the utilised solvers between the constrained and unconstrained problems. The unconstrained functions are UDF1-9 (Biswas et al. 2014), except for UDF7 as it is a three-objective problem, JY1-8 (Jiang and Yang 2017), except JY4 as the parameters given in Jiang and Yang (2017) do not reflect the problem description and the provided Pareto Optimal Front, and FDA1-3 (Farina et al. 2004). The selected set provides a range of problems including all types of dynamics, I-IV, different kinds of Pareto Optimal Front and Set changes, varying Pareto Optimal Front and Set geometries, and problem characteristics: modalities, biases and variable-linkages.

Six different Genetic Algorithms are selected for the benchmarking due to their high performance: MOEA/D, as the best algorithm for static and dynamic unconstrained problem types (Zhang and Suganthan 2009; Jiang and Yang 2017; Biswas et al. 2014); NSGA-II Deb et al. (2002) as the most commonly utilised Genetic Algorithm with high performance on static constrained and unconstrained problems; MTS (Tseng and Chen 2009) due to the high performance on constrained and unconstrained static problems while also representing a Genetic Algorithm with no crossover, it has a high potential for dynamic problems as it demonstrates rapid convergence but is currently untested on these problems; BCE (Li et al. 2016) and

HEIA (Lin et al. 2016) as examples of the current state-of-the-art of static Genetic Algorithms, with top performance on static problems and a higher emphasis on diversity; and the MLSGA-MTS hybrid (Grudniewski and Sobey 2018), in MLS1 configuration, which shows the best general performance on static problems where the addition of the MLS1 mechanisms have been shown to increase the convergence speed (Sobey and Grudniewski 2018), essential in dynamic optimisation. Each of the six selected algorithms, MOEA/D, NSGA-II, MTS, BCE, HEIA and MLSGA-MTS, are combined with the 3 different re-initialisation methodologies: VP, CER-POS and CER-POF, and compared to results without any re-initialisation mechanisms. DEMO (Farina et al. 2004), DMEA/PRI (Zhou et al. 2007) and DQMOO/AR (Hatzakis and Wallace 2006), are not selected as they are shown to be outperformed by MOEA/D (Biswas et al. 2014). DNSGA-II (Deb et al. 2007) is also not included as it is outperformed by other methods and not compatible with the selected re-initialisation mechanisms. dCOEA (Goh and Tan 2009) is not selected as it performs poorly on discontinuous search and objective spaces, while showing similar performance on other problems to MOEA/D (Biswas et al. 2014). Only Genetic Algorithms are considered, instead of the other Evolutionary Algorithms, as the principle of this paper is not to find the best solver for each kind of problem, but rather to evaluate the impact of implementing constraints into dynamic problems on Genetic Algorithms.

In addition, two constraint-handling approaches are tested against each other in order to minimise the potential bias of a single technique towards a certain Genetic Algorithm or re-initialisation methodology: the dominance-based (Deb et al. 2002) and the adaptive penalty-based (Azzouz et al. 2018) constraint-handling methods are selected as the most commonly utilised techniques.

4.2 Benchmarking parameters

The benchmarks are conducted using two different combinations of dynamic settings, the time change, T , and n_s . T denotes the time-window when the problem remains static, as the number of generations; n_s denotes the severity of change. The first set of benchmarks represents a harsh dynamic environment with both values taken to be 5. In the second set of benchmarks, developed to demonstrate optimisation in a less harsh dynamic condition, both of the parameters are set equal to 10. These two tests are conducted in order to eliminate the selection of certain parameters which promote the performance of certain algorithms or mechanisms, especially the importance of diversity and convergence rate. However, the results show similar trends in both cases, and therefore only the results for T and n_s equal to 5 are documented with the values set

to 10 documented in the supplementary material. The performance is evaluated based on the Inverted Generational Distance (IGD) (Bosman and Thierens 2003) and Hyper Volume (HV) (While et al. 2012).

In most benchmarks a relatively small population is used, around 200–500. However, as the presented constrained functions are more challenging, larger population sizes may be required to prevent the algorithm from getting stuck in an infeasible region of the search space and premature convergence. Therefore, two population sizes of 300 and 1000 are evaluated to avoid the situation where the population size is the limiting factor for the algorithm, i.e. it is not enough to develop the necessary diversity of solutions. However, the population size does not affect the trends in the results, equally reducing the overall performance of each algorithm meaning no mechanism is unfairly penalised. Therefore, the results for the 300 population sizes are attached as a data supplement but are not documented in the paper.

In most benchmarks of Genetic Algorithms on dynamic problems, the termination criterion utilised is the number of generations and the time change occurs after T generations (Jiang and Yang 2017; Biswas et al. 2014). However, the number of objective function evaluations conducted at each generation is not uniform across different methodologies. For NSGA-II it is the population size and for MOEA/D it is the number of new offspring, which is significantly lower than the parent population size, usually 0.1 of the overall population size. For the MTS and MLSGA-MTS algorithms the number of evaluations is a multiple of the population size, ranging from $5\times$ to $50\times$, as multiple local searches are used. Comparing the performance of these algorithms using the number of generations gives a wide range of different function calls and does not provide a fair benchmark. Furthermore, the total number of generations is irrelevant in most cases, as the calculation time per generation is normally relatively small in comparison to the objective function evaluation. Therefore, the selected stopping criterion is the number of fitness function evaluations and the time change occurs after a certain number of evaluations. The total number of fitness function evaluations for T and n_s equal to 5 and 10 are therefore 300,000 and 600,000 respectively, which allows 6 cycles of the UDF and CDF (Biswas et al. 2014), and 3 cycles of the JY (Jiang and Yang 2017) and FDA (Farina et al. 2004). The time change occurs after population size $\times T$ iterations and is equal to 5000 iterations in the case of T equal to 5 and 10,000 iterations in the second case. All cases are evaluated over 30 independent runs. In order to show the importance of a proper termination condition, a separate test is conducted with the commonly utilised generations termination condition, with 300 generations as the limit. The results show that MTS outperforms all of the other algorithms on

almost all of the selected functions and MOEA/D performs poorly due to the disproportionate number of fitness evaluations per generation. Therefore, the data for the generation termination condition is not included but attached as supplementary material to allow comparison to the previous literature. Even though this issue is widely known in static MOPs, the previous dynamic benchmarks have not addressed it (Jiang and Yang 2017; Biswas et al. 2014) and fitness evaluations limits have not been mentioned or obviously implemented. The obtained results show that future dynamic benchmarking exercises should use fitness evaluation limits, to allow a fair comparison of algorithms across different exercises.

5 Performance benchmarking

A summary of the top performing algorithms and re-initialization mechanisms is presented in this section for T and n_s equal to 5, using 1000 individuals in the population. Pre-benchmarking simulations show that the dynamic parameters only have a minor impact on the comparative performance of the utilised algorithms and re-initialisation strategies. In the conducted benchmarks the adaptive penalty-based constraint-handling approach leads to significantly better results in comparison to the second technique. However, there is no significant differences in comparative performance of the selected Genetic Algorithms and re-initialisation mechanisms between those constraint-handling techniques. The only exception is that HEIA shows stronger performance with the dominance-based strategy and therefore all of the results shown are from the domination-based approach. The detailed data, with minimum, maximum and average values of the IGD and HV, along with standard deviations and calculation times are added as supplementary material.

The most dominant factor in determining the performance on the dynamic problems is the selection of the Genetic Algorithm. Despite this, the results indicate that in most cases the selection of the re-initialization method can have a significant impact on the final performance. Only in a few, seemingly random, cases is the performance of the Genetic Algorithm independent of these re-initialisation methods. Interestingly, the impact of the re-initialisation strategy is not uniform across different Genetic Algorithms and problems, i.e. on the same problem the VP variant can exhibit the best performance for MTS but for BCE with VP shows the worst performance. Therefore, selection of the best strategy is difficult and further work needs to be performed to determine how this selection can be made in advance. These results are summarised in Sect. 5.1 for the re-initialisation methods and in Sect. 5.2 for the different algorithms to understand the final performance.

5.1 Re-initialisation

In order to compare the different re-initialization methods, the average rank, according to the IGD and HV metrics, of each mechanism is calculated on problems with the same characteristics and presented in Table 2 for T and n_s equal to 5. In most cases, the VP re-initialisation outperforms the other methodologies tested and has the lowest rank in total across all of the problem types, despite the lack of uptake for this method outside of the original paper (Zhou et al. 2007). The high performance of the VP method is due to the inclusion of the variance module, which is not based on pattern prediction but on random Gaussian noise. The prediction-based strategies utilize historical information from the Pareto Optimal Set at previous generations to approximate the position of the final Pareto Optimal Set in the next time step. Therefore, these methods are inefficient on problems where the Pareto Optimal Set remains static, such as type III problems: FDA2, UDF3, UDF6 and CDF9-12, and the problems where the pattern for how the objectives change cannot be predicted, such as the random change problems: UDF8, UDF9 and CDF13. This is supported by the higher performance of the variant with no re-initialisation over other methods. Industrial applications are often unpredictable, or extremely complex, and so the separation in performance of these mechanisms is likely to be even greater. Furthermore, in the vast majority of cases there is a significant difference between the ranked results according to Wilcoxon's rank sum test with a confidence level of $\alpha = 0.05$, especially comparing VP to other methods, showing its superiority.

The results show the importance of the selection of an appropriate re-initialisation mechanism. Importantly, the effect on performance of different strategies is not uniform across the Multi-Objective Evolutionary Algorithms. For example, on the two top performing Genetic Algorithm

Table 2 The average ranks of each re-initialisation method for the 1000 population size with iterations termination criterion and T and n_s equal to 5

	None	CER-POS	CER-POF	VP
Overall	2.83/3.07	2.47/2.29	2.74/2.49	1.95/2.15
Type I	3.02/3.43	2.38/2.07	2.62/2.52	1.98/1.98
TypeII	3.09/3.39	2.31/2.18	2.81/2.43	1.79/2.00
Type III	2.13/2.28	2.81/2.69	2.74/2.63	2.31/2.41
Type IV	3.17/3.08	2.33/2.33	2.58/2.25	1.92/2.33
UDF	2.79/3.21	2.46/2.31	2.81/2.40	1.94/2.08
CDF	2.78/3.06	2.40/2.07	2.72/2.51	2.10/2.37

The best mechanism in each category is highlighted in bold face. The second best is highlighted in italic face

methodologies, MTS works best with the VP method, but usually there is no statistical significance between the 4 used re-initialisation mechanisms for this methodology, whereas MOEA/D is more likely to show higher performance without any re-initialisation method. In the case of MOEA/D the authors suggest that this is caused by the MOEA/D specific mechanisms such as weight-vectors and neighbourhoods of solutions. In the re-initialisation step these features are not taken into consideration when the new population is created and the parent solutions are selected. Therefore, the overall efficiency is limited and re-initialisation mechanisms specific to MOEA/D might be necessary. This is similar to the performance of MOEA/D with the MLSGA mechanisms (Grudniewski and Sobey 2018), and it seems the MOEA/D mechanisms are more difficult to blend with additional elements.

Despite the importance of the re-initialisation method, the choice of appropriate method is often neglected in the dynamic Genetic Algorithm literature and in most cases the re-initialization method is chosen arbitrarily and not compared to the other variants (Biswas et al. 2014) or not utilized at all (Jiang and Yang 2017). Therefore, benchmarking of novel re-initialization mechanisms with other variants, on the same algorithm methodologies, and with the algorithm without any diversity preservation mechanisms is necessary, to ensure that new re-initialisation methods provide benefits over the current methods.

5.2 Genetic algorithm methodologies

In order to investigate the performance of the different algorithms, the average ranks according to the IGD and HV indicators are calculated across all of the dynamic problem types and presented in Table 3 for T and n_s equal to 5. There are significant differences between the ranked results according to Wilcoxon's rank sum test with a confidence level of $\alpha = 0.05$ in most of the presented cases. The performances of the Genetic Algorithm mechanisms are less consistent across the utilised problems than for the re-initialisation methods, and it is not easy to choose a single best method. According to the average ranks, the best overall performer is MOEA/D, while MTS and HEIA come second and third respectively. However, MOEA/D is less effective when the Pareto Optimal Set remains static, demonstrated by comparatively poor performance on Type III problems. This is caused by the static weight vectors, where the previous best point according to one weight is no longer the most suitable one after the time change. Therefore, it is likely that the points with the best values may be eliminated quickly, as their vectors are not properly assigned. This problem could be solved by reassignment of each weight after the time changes occur. MTS does not have these disadvantages and is less susceptible to the time

Table 3 The average ranks of each Genetic Algorithm for the 1000 population size with iterations termination criterion and T and n_s equal to 5

	MOEA/D	MTS	NSGA-II	MLSGA	BCE	HEIA
Overall	2.45/2.31	2.71/4.97	5.10/2.70	3.16/4.70	4.58/2.92	2.99/3.39
Type I	2.25/2.43	2.89/4.89	4.82/2.50	3.11/4.82	4.64/3.04	3.29/3.32
TypeII	1.97/1.75	2.88/5.15	5.15/2.87	3.45/4.48	4.37/3.15	3.18/3.60
Type III	3.22/3.08	2.33/4.89	5.28/2.28	2.86/5.03	4.72/2.61	2.58/3.11
Type IV	2.63/2.13	2.88/4.25	4.88/4.63	2.88/4.50	5.38/1.38	2.38/4.13
UDF	2.59/2.13	2.56/4.91	5.16/2.63	3.88/4.44	3.91/2.91	2.91/4.00
CDF	2.37/2.32	3.05/4.63	4.77/3.05	2.93/5.55	5.57/2.28	2.32/3.17

The best algorithm in each category is highlighted: in italic face according to IGD; in bold italic face for HV; and bold face with both metrics

changes due to the utilisation of multiple local searches that maintains both the convergence and diversity of the search; this results in a lack of significant differences between the implemented re-initialisation methods. High performance of these high convergence-based solvers, such as MOEA/D, further supports the findings that Dynamic Multi-Objective Problems requires high convergence of the search over diversity, unlike many static problems, which was originally suggested in Biswas et al. (2014).

The HEIA and MOEA/D exhibit the best overall performance for the constrained results. The high performance of MOEA/D on these problems is interesting as previous research shows poor results on the static constrained problems (Zhang and Suganthan 2009), and MOEA/D is outperformed by MLSGA-MTS, MTS and NSGA-II (Grudniewski and Sobey 2018). Furthermore, for most of the tested algorithms there is no significant difference in comparative performance on the CDFs than on the UDFs, on which the developed set is based. This unexpected behaviour indicates that, for dynamic problems, the fast convergence of MOEA/D is more important than the ability to discover discontinuous search and objective spaces, characteristics where the MOEA/D mechanisms perform poorly. Therefore, the constrained dynamic problems act similarly to the unconstrained cases. These findings further indicate that high convergence rate is essential when dealing with Dynamic Multi-Objective Problems.

5.3 Performance of the different algorithms on the constrained functions over time

In order to investigate the behaviour on the developed constrained set, the IGD values obtained by the benchmarked algorithms are tracked during a single run, period $t = 0$ to $t = 11.8$. This is illustrated in Fig. 17 for CDF4 and in Fig. 16 for CDF10, as problems where the behaviour is representative of these rest of the set. In all cases the VP re-initialisation is used, and only the results for MTS, MOEA/D and NSGA-II are shown as the two best and the worst performing algorithm respectively. The figures for

the other developed problems, as well as for the HV metric, are attached as supplementary data.

There is a considerable difference in performance between the top algorithms for a problem and the next best performer. Figure 16 shows that MTS is able to accurately approximate the true Pareto Optimal Front at every time step for the CDF10 problem, resulting in stable IGD values. NSGA-II is not able to properly converge on the Pareto Optimal Front whereas MOEA/D is unable to find all of the regions of the Pareto Optimal Front on the highly discontinuous search space; this is especially clear after the shift of the constraints. This behaviour of MOEA/D is expected as a similar principle is shown on constrained static problems and is due to the low performance on the Type I problems, as discussed previously. However, a different behaviour is exhibited on the CDF4 function, Fig. 17. In this case MOEA/D outperforms MTS, showing lower IGD values that remain stable at every time step, indicating that the true Pareto Optimal Front has been found, whereas the performance of NSGA-II remains the worst. This is most likely caused by a higher uniformity

Fig. 16 IGD(t) values obtained by the MOEA/D, MTS and NSGA-II on the CDF10 case

and continuity of the CDF4 problem in comparison to the CDF10. According to that CDF10 case requires a higher diversity of the search, which MOEA/D is unable to provide, unlike the MTS. The low performance of NSGA-II on both problems, and the effectiveness of MOEA/D, even over MTS on some cases, is contrasting to the trends shown on static problems. This indicates that the dynamicity is the dominant characteristic of the Dynamic Multi-Objective Problems, even over the level of discontinuities in the search spaces.

6 Conclusions

Dynamic optimisation has many applications. However, the current benchmarking problems are limited to unconstrained problems, which may benefit certain types of Genetic Algorithms. In this paper the range of dynamic test problems is expanded by proposing a set of constrained test functions. A number of combinations of current state-of-the-art algorithms and re-initialization methods are then benchmarked to determine the most effective methods for solving constrained dynamic optimisation problems. Overall MOEA/D shows the best performance on both the new constrained and traditional unconstrained test sets. The Variation and Prediction re-initialisation strategy performs the best on most problems but in some cases no significant improvement is shown over no re-initialization. Importantly, the performance of the algorithms on the constrained and unconstrained problems is similar, showing that the dynamic characteristic is dominating these problems. The following recommendations are made:

- The importance of dynamic behaviour dominates the non-continuous characteristic of the constrained problems, as there is no significant difference in the comparative performance of the tested algorithms between the constrained and unconstrained cases. The results show that Genetic Algorithms with strong convergence profiles are best at solving these problems.
- In most cases the obtained solutions are far from the true Pareto Optimal Front. As the highly dynamic problems are dominated by convergence, this puts the Genetic Algorithms at a disadvantage over other methods. Therefore, there is a need for dynamic specific methodologies with high convergence and low diversity.
- The prevalence of the predictable dynamic test problems leads to mechanisms that do well on these problems. However, in most applications the dynamic changes are unpredictable, which makes them more difficult to solve and more random elements should be included in future test sets to develop more “realistic” mechanisms.

Acknowledgements This research was supported by Lloyds Register Foundation.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Azzouz R, Bechikh S, Said LB, Trabelsi W (2018) Handling time-varying constraints and objectives in dynamic evolutionary multi-objective optimization. *Swarm Evol Comput* 39:222–248. <https://doi.org/10.1016/j.swevo.2017.10.005>
- Bagheri A, Mohammadi Peyhani H, Akbari M (2014) Financial forecasting using ANFIS networks with quantum-behaved particle swarm optimization. *Expert Syst Appl* 41(14):6235–6250. <https://doi.org/10.1016/j.eswa.2014.04.003>
- Biswas S, Das S, Suganthan PN, Coello Coello CA (2014) Evolutionary multiobjective optimization in dynamic environments: a set of novel benchmark functions. In: 2014 IEEE congress on evolutionary computation (CEC 2014), vol 1, pp 3192–3199. <https://doi.org/10.1109/CEC.2014.6900487>
- Bosman PA, Thierens D (2003) The balance between proximity and diversity in multiobjective evolutionary algorithms. *IEEE Trans Evol Comput* 7(2):174–188. <https://doi.org/10.1109/TEVC.2003.810761>

Fig. 17 IGD(t) values obtained by the MOEA/D, MTS and NSGA-II on the CDF4 case

- Branke J (2002) Evolutionary optimization in dynamic environments. Kluwer Academic Publishers, Norwell. <https://doi.org/10.1007/978-1-4615-0911-0>
- Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evol Comput* 6(2):182–197. <https://doi.org/10.1109/4235.996017>
- Deb K, Rao N UB, Karthik S (2007) Dynamic multi-objective optimization and decision-making using modified NSGA-II: a case study on hydro-thermal power scheduling. In: *Evolutionary multi-criterion optimization*, pp 803–817. <https://doi.org/10.1007/978-3-540-70928-2>
- Effendy S, Khan MS, Farooq S, Karimi IA (2017) Dynamic modelling and optimization of an LNG storage tank in a regasification terminal with semi-analytical solutions for N2-free LNG. *Comput Chem Eng* 99:40–50. <https://doi.org/10.1016/j.compchemeng.2017.01.012>
- Farina M, Deb K, Amato P (2004) Dynamic multiobjective optimization problems: test cases, approximations, and applications. *IEEE Trans Evol Comput* 8(5):425–442. <https://doi.org/10.1109/TEVC.2004.831456>
- Goh CK, Tan KC (2009) A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization. *IEEE Trans Evol Comput* 13(1):103–127. <https://doi.org/10.1109/TEVC.2008.920671>
- Grudniewski PA, Sobey AJ (2018) Behaviour of multi-level selection genetic algorithm (MLSGA) using different individual-level selection mechanisms. *Swarm Evol Comput* 44:852–862. <https://doi.org/10.1016/j.swevo.2018.09.005>
- Hatzakis I, Wallace D (2006) Dynamic multi-objective optimization with evolutionary algorithms: a forward-looking approach. In: *Proceedings of the 8th annual conference on genetic and evolutionary computation*, pp 1201–1208. <https://doi.org/10.1145/1143997.1144187>
- Helbig M, Engelbrecht AP (2014) Benchmarks for dynamic multi-objective optimisation algorithms. *ACM Comput Surv* 46(3):37:1–37:39. <https://doi.org/10.1145/2517649>
- Helbig M, Engelbrecht AP (2015) Benchmark functions for CEC 2015 special session and competition on dynamic multi-objective optimization. Technical report
- Jiang S, Yang S (2017) Evolutionary dynamic multiobjective optimization: benchmarks and algorithm comparisons. *IEEE Trans Cybern* 47(1):198–211. <https://doi.org/10.1109/TCYB.2015.2510698>
- Li C, Yang S, Pelta DA (2011) Benchmark generator for the IEEE WCCI-2012 competition on evolutionary computation for dynamic optimization problems. Technical report. <https://doi.org/10.13140/RG.2.1.3298.1842>
- Li M, Yang S, Liu X (2016) Pareto or non-Pareto: bi-criterion evolution in multiobjective optimization. *IEEE Trans Evol Comput* 20(5):645–665. <https://doi.org/10.1109/TEVC.2015.2504730>
- Lin Q, Chen J, Zhan ZH, Chen WN, Coello Coello CA, Yin Y, Lin CM, Zhang J (2016) A hybrid evolutionary immune algorithm for multiobjective optimization problems. *IEEE Trans Evol Comput* 20(5):711–729. <https://doi.org/10.1109/TEVC.2015.2512930>
- Mehnen J, Wagner T, Rudolph G (2006) Evolutionary optimization of dynamic multi-objective test functions. Technical report. <https://doi.org/10.17877/DE290R-631>, http://laral.istc.cnr.it/wiva3/atti/gsize2/papers/GSICE06_09_Wagner.pdf, arXiv:1310.2674
- Nguyen TT, Yang S, Branke J (2012) Evolutionary dynamic optimization: a survey of the state of the art. *Swarm Evol Comput* 6:1–24. <https://doi.org/10.1016/j.swevo.2012.05.001>
- Sobey AJ, Grudniewski PA (2018) Re-inspiring the genetic algorithm with multi-level selection theory: multi-level selection genetic algorithm. *Bioinspir. Biomim.* 13(5):1–13. <https://doi.org/10.1088/1748-3190/aad2e8>
- Tseng LY, Chen C (2009) Multiple trajectory search for unconstrained/constrained multi-objective optimization. In: *2009 IEEE congress on evolutionary computation (CEC 2009)*, pp 1951–1958. <https://doi.org/10.1109/CEC.2009.4983179>
- While L, Bradstreet L, Barone L (2012) A fast way of calculating exact hypervolumes. *IEEE Trans Evol Comput* 16(1):86–95. <https://doi.org/10.1109/TEVC.2010.2077298>
- Wismans L, Van Berkum E, Bliemer M (2014) Handling multiple objectives in optimization of externalities as objectives for dynamic traffic management. *Eur J Transp Infrastruct Res* 14(2):159–177
- Zhang Z (2008) Multiobjective optimization immune algorithm in dynamic environments and its application to greenhouse control. *Appl Soft Comput J* 8(2):959–971. <https://doi.org/10.1016/j.asoc.2007.07.005>
- Zhang Q, Li H (2007) MOEA/D: a multiobjective evolutionary algorithm based on decomposition. *IEEE Trans Evol Comput* 11(6):712–731. <https://doi.org/10.1109/TEVC.2007.892759>
- Zhang Q, Suganthan PN (2009) Final report on CEC '09 MOEA competition. Technical report. <http://dces.essex.ac.uk/staff/zhang/moeacompetition09.htm>
- Zhang Q, Zhou A, Zhao S, Suganthan PN, Liu W (2009) Multiobjective optimization test instances for the CEC 2009 special session and competition. Technical report
- Zhou A, Jin Y, Zhang Q, Sendhoff B, Tsang E (2007) Prediction-based population re-initialization for evolutionary dynamic multi-objective optimization. In: *Evolutionary multi-criterion optimization*. Springer, Berlin, pp 832–846. https://doi.org/10.1007/978-3-540-70928-2_62

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.