A Survey of Evolutionary Continuous Dynamic Optimization Over Two Decades – Part A

Danial Yazdani, *Member, IEEE,* Ran Cheng, *Member, IEEE,* Donya Yazdani, Jürgen Branke, *Member, IEEE,* Yaochu Jin, *Fellow, IEEE,* and Xin Yao, *Fellow, IEEE,*

***Abstract*—Many real-world optimization problems are dynamic. The field of dynamic optimization deals with such problems where the search space changes over time. In this two-part paper, we present a comprehensive survey of the research in evolutionary dynamic optimization for single-objective unconstrained continuous problems over the last two decades. In Part A of this survey, we propose a new taxonomy for the components of dynamic optimization algorithms, namely, convergence detection, change detection, explicit archiving, diversity control, and population division and management. In comparison to the existing taxonomies, the proposed taxonomy covers some additional important components, such as convergence detection and computational resource allocation. Moreover, we significantly expand and improve the classifications of diversity control and multi-population methods, which are under-represented in the existing taxonomies. We then provide detailed technical descriptions and analysis of different components according to the suggested taxonomy. Part B of this survey provides an in-depth analysis of the most commonly used benchmark problems, performance analysis methods, static optimization algorithms used as the optimization components in the dynamic optimization algorithms, and dynamic real-world applications. Finally, several opportunities for future work are pointed out.**

***Index Terms*—Unconstrained continuous dynamic optimization, Evolutionary algorithms, Multi-population, Change detection, Response component, Taxonomy.**

P_A-I. Introduction

CHANGE is an unavoidable part of many optimization problems, and adaptation is necessary to tackle them. To solve an optimization problem in a dynamic environment, it is important that the algorithm can efficiently find an optimal solution and also track it over time after environment changes. The ubiquity of dynamic optimization problems (DOPs) demands extensive research into the design and development of algorithms capable of dealing with environmental changes [1].

Inspired by biological evolution and natural self-organized systems, evolutionary algorithms (EA) and swarm intelligence (SI) methods have been vastly used for optimizing DOPs due to their natural capability in dealing with environmental changes [2]. Indeed, both classes have been successfully applied to DOPs with various environmental and dynamic characteristics [1], [3]. However, one cannot directly apply them to tackle DOPs as these methods are originally designed for optimization in static environments and cannot cope with the challenges of a DOP alone. Hence, they are usually used together with some other *components* to form dynamic optimization algorithms (DOAs). Solving DOPs using EAs and SIs has been a hot research topic in the last two decades. Figure P_A-1[1] depicts the growth in the number of scientific articles published in this field.

Several studies have reviewed various aspects of DOPs, including surveys [1], [3]–[5], books [6]–[9], and Ph.D. theses [10], [11]. Among the surveys, [4] (2005) briefly reviews optimization in dynamic environments as a part of uncertain environments. In this survey, components of DOAs are classified into four main categories: generating diversity after a change, maintaining diversity over time, using multi-population, and memory based approaches (including *implicit* and *explicit memory*). Cruz et al. [5] provided the first survey focusing exclusively on DOPs. This survey uses the same taxonomy introduced in [4]. In [1] (2012), for the first time, DOP performance indicators is reviewed. This survey also provides a brief review of the DOP benchmarks. Although this survey uses the same taxonomy as in [4] to study DOAs, it also considers three extra classes: prediction, change detection, and self-adaptation. The latest DOP survey published in 2017 [3] focuses on SI methods adopted to tackle DOPs while also briefly covers performance indicators, benchmark problems, and real-world applications. This survey is mostly focused on single-objective unconstrained DOPs with continuous and discrete environments and gives a brief introduction to multi-objective and constrained DOPs. It also uses the same taxonomy introduced in [4] to classify DOAs components, but it further classifies multi-population DOAs into fixed, dynamic, and adaptive numbers of subpopulations.

Several types of DOPs are investigated in the literature, such as combinatorial DOPs [12], continuous DOPs [13], multi-objective DOPs [14]–[16], and constrained DOPs [17]. In [1], [3]–[5] several types of DOPs including constrained/unconstrained, continuous/discrete, and single/multi-objective, are covered together. Each of these areas has a vast literature with specific challenges, methods, benchmark

Danial Yazdani, R. Cheng and X. Yao are with Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China (e-mails: danial.yazdani@gmail.com, chengr@sustc.edu.cn, xiny@sustc.edu.cn). X. Yao is also with the CERCIA, School of Computer Science, Birmingham B15 2TT, United Kingdom.

Donya Yazdani is with the Advanced Reasoning Group, Department of Computer Science, Aberystwyth University, Aberystwyth, Ceredigion SY23 3DB, United Kingdom (email: d.yazdani@aber.ac.uk).

J. Branke is with the Operational Research and Management Sciences Group in Warwick Business school, University of Warwick, Coventry CV4 7AL, United Kingdom (email: Juergen.Branke@wbs.ac.uk).

Y. Jin is with Department of Computer Science, University of Surrey, Guildford, Surrey GU27XH, United Kingdom (email: yaochu.jin@surrey.ac.uk).

[1]Labels of all sections, equations, tables, and figures in Part A and Part B of this survey are prefixed with P_A and P_B, respectively.

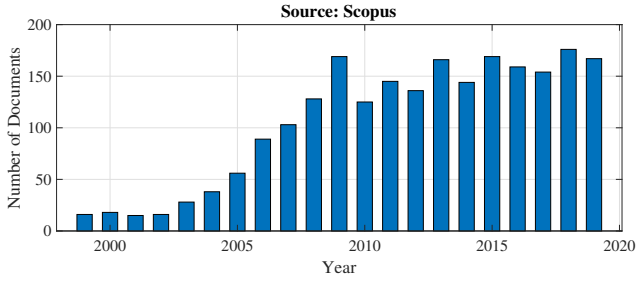

Fig. $\mathrm{P_A}$-1. Publication trend in evolutionary dynamic optimization over the last two decades. This results show the number of documents containing the relevant phrases to the evolutionary dynamic optimization in their title, abstract, or index terms.

problems, and performance analysis metrics. Considering that there is a trade-off between the comprehensiveness of a review and the range of the topics covered, many important details are missed. To address the need for more detailed reviews, some surveys have narrowed their covered topics to specific areas to provide more technical details. For example, [18] covers the single class of self-adaptive approaches, [19] reviews hyperheuristics for DOPs, [14] and [15] focus on multi-objective DOPs, and [12] reviews combinatorial DOPs. Following the same line of work, in this two-part survey, we focus on *single-objective unconstrained continuous DOPs*. Being extremely popular, this class of DOPs has a large literature with various methods, benchmark problems, and performance analysis metrics with real-world applications. The designed DOAs for this DOPs are shown to be easily extendable to other important types of DOPs such as constrained DOPs [17] and robust optimization over time (ROOT) [20].

In this two-part paper, we present a comprehensive survey of the research in evolutionary single-objective unconstrained continuous optimization[2] over the last two decades. This survey helps researchers to gain a bird's eye view of the field and to learn about its major trends, algorithms, benchmark problems, performance analysis metrics, real-world applications, shortcomings, and potential future directions.

In the first part of this survey, we first describe different classes of DOPs. Then, we propose a new *comprehensive* taxonomy for the components of DOAs that are developed to address DOPs' challenges. The taxonomy poses the following advantages in comparison to the existing ones:

- It covers some important components of DOAs which were not included in the previous taxonomies, such as convergence detection and resource allocation methods.
- The classification of diversity control methods is significantly improved. Unlike the existing taxonomies, the purpose of the diversity control methods, which is either to address the global or local diversity loss issues, is taken into consideration. In addition, the proposed taxonomy covers some additional classes of diversity increasing methods (which were ignored in the previous taxonomies) that are triggered when some conditions are met.
- The multi-population methods which are the most popular and effective DOAs are fully classified in our taxonomy by their population structure and the components used to create and manage subpopulations.

We then provide in-depth technical descriptions of the developed components in DOAs according to the suggested taxonomy.

In the second part of this survey [21], we first study *baseline functions* and *dynamics* of DOP benchmarks separately. To review the baseline functions, we classify them into basic static functions, moving peaks, and the composition of basic static functions. Then, we provide a review of the commonly used performance indicators and plots that have been used for analyzing and comparing the performance of DOAs. We study the DOP performance indicators by classifying them into two groups of *fitness/error based* and *efficiency based* ones. Then, we study the static optimization algorithms (especially EAs and SI methods) used as the *optimization components* in DOAs. After providing a review of real-world applications of DOAs, we conclude the survey by discussing some of the current challenges, the gap between academic research and real-world problems, and some potential future research directions.

To identify the relevant papers for this survey, we conducted a search using various publication search engines, including ScienceDirect, IEEE Xplore, ACM Digital Library, Springer, CiteSeerX, Wiley, Taylor & Francis, Scopus, Google Scholar, etc. For the search keywords, we used different combinations of terms such as <dynamic, uncertain, time-varying, non-stationary>, <environment, optimization, problem>, and <evolutionary algorithm, swarm intelligence algorithm, metaheuristic>. Thereafter, we selected references that match the scope of this survey which are published after 1999. More than 200 journal papers, book chapters, conference papers, Ph.D. theses, and master's dissertations have been selected for our survey. Note that, all presented statistics in this survey, such as pie-charts, are extracted from this collection.

The organization of this two-part paper is as follows:

Part A: Section (§) $\mathrm{P_A}$-II provides background information, including problem definition and different types of DOPs. § $\mathrm{P_A}$-III describes the proposed taxonomy for the components of DOAs. The first part of this survey is concluded in § $\mathrm{P_A}$-IV.

Part B: § $\mathrm{P_B}$-I provides an introduction to Part B of this survey. The commonly used benchmark problems in the DOP literature are reviewed in § $\mathrm{P_B}$-II. The performance indicators and plots used for analyzing the performance of DOAs are reviewed in § $\mathrm{P_B}$-III. The static optimization algorithms used as the optimization components in DOAs are reviewed in § $\mathrm{P_B}$-IV. § $\mathrm{P_B}$-V reviews real-world DOPs. § $\mathrm{P_B}$-VI discusses the challenges of the field's current status and provides some potential future research directions. Finally, § $\mathrm{P_B}$-VII concludes the second part of this survey.

$\mathrm{P_A}$-II. Background

We start by describing a DOP. Without loss of generality, an unconstrained single-objective DOP can be defined as:

$$F(\vec{x}) = f\left(\vec{x}, \vec{\alpha}^{(t)}\right), \qquad (\mathrm{P_A}\text{-}1)$$

[2]For brevity, we use the term *DOPs* to refer to this specific type of problems from now on.

where f is the objective function, $\vec{x}$ is a solution in the search space, $\vec{\alpha}$ is a vector of time-varying objective function control parameters, and $t \in [0, T]$ is the time index. In real-world problems, $\vec{\alpha}$ can be environmental parameters which change over time, such as temperature, costs, workload, or available resources. In DOP benchmarks, for example the moving peaks benchmark (MPB) [22], [23], $\vec{\alpha}$ includes the parameters of peaks, such as width, height, and center position which all change over time (see § $\mathrm{P_B}$-II-A1). $\vec{\alpha}$ can also include other time-variant environmental parameters like the domain of variables, number of variables, and variable interactions [10], [11].

Most existing works in the DOP literature consider the DOPs whose environmental changes happen only in discrete time, i.e., $t \in \{1, \ldots, T\}$. For a DOP with T environmental states, there is a sequence of T stationary environments:

$$\left\langle f(\vec{x}, \vec{\alpha}^{(1)}), f(\vec{x}, \vec{\alpha}^{(2)}), \ldots, f(\vec{x}, \vec{\alpha}^{(T)}) \right\rangle. \qquad (\mathrm{P_A}\text{-}2)$$

It is commonly assumed that there is a degree of similarity between the successive environments in ($\mathrm{P_A}$-2), i.e., the changes are not highly severe. In cases where the environmental changes are very severe, the information from the previous environments are less useful [22]. In such circumstances, it may be best to consider each environment as an independent static optimization problem, and to reinitialize the optimization algorithm after each environmental change. In this paper, similar to the majority of DOP literature, we focus on the DOPs whose dynamic is expressed with similarity between environmental states as the case in many real-world optimization problems [6], [10], [22].

DOPs can be categorized according to various characteristics. In the literature, several classifications have been introduced for DOPs [1], [24]–[29]. By properly classifying DOPs and identifying their specific features and challenges, we can apply appropriate DOAs for optimizing them. Moreover, identifying the strengths and weaknesses of DOAs in solving different classes of DOPs will help researchers to develop more efficient DOAs. In the following of this section, we describe some important classifications of DOPs based on different criteria.

In [26], Eberhart and Shi classify DOPs according to the changes in the position and fitness value of the global optimum. They introduce three classes of DOPs where:

- The position of optimum remains unchanged while its fitness value changes over time,
- The optimum position changes while its fitness value remains unchanged, and
- Both position and fitness value of the optimum change over time.

The first class is relatively easier for optimization methods to solve since there is no need to track the optimum after environmental changes. For both the second and third classes, however, the algorithms need to track the optimum after environmental changes. For the second class, if the optimum fitness value is obtained, it can be used as a reference in future environments to avoid *immature convergence*. Hence, it might be less challenging than the third class where both position and fitness value change over time. According to our readings, the majority of the DOP literature focuses on the third class.

Another classification of DOPs is proposed by De Jong [27], where the change severity, change frequency, and change patterns have been taken into account. Inspired by the characteristics of some real-world DOPs, De Jong's classification divides DOPs into four groups:

- DOPs with drifting landscapes: in these problems, change severity of the environment is very gradual, but the change frequency is high. This class reflects the real-world DOPs whose environmental changes are caused by aging equipment or minor changes in the quality of raw materials.
- DOPs with significant morphological changes: in these problems, the fitness of each region over the search space can either increase or decrease after environmental changes. Therefore, some low-quality regions from the previous environment can become regions with high fitness values and vice versa. Such problems are quite challenging since the global optimum can appear in a region that had a low quality in the previous environment. Similar properties can be seen in ompetitive marketplaces where the areas with the highest profit change due to the change of demands and the competition level. Such characteristics are simulated in moving peaks based DOP benchmarks (see § $\mathrm{P_B}$-II-A1).
- Periodical DOPs: the environmental changes in these DOPs show reappearing/cyclic patterns where the landscape visits a finite set of states repeatedly. This type of DOPs' dynamic is reflective of problems involving repeating states such as seasonal changes (e.g., energy consumption level over seasons) and hourly demand rate changes (e.g., rush hours in the morning).
- DOPs with abrupt changes: in these problems, the change severity is usually huge and the similarity between successive environments is very low. In real-world DOPs, this type of dynamics is the result of cataclysmic events such as failure(s) in a part(s) of the system.

Later, Duhain and Engelberth [25] classify DOPs based on two levels of change severity and change frequency:

- Quasi-static: DOPs with low change severity and frequency,
- Progressively: DOPs with low change severity and high change frequency (similar to the drifting landscapes in the De Jong's classification),
- Abrupt: DOPs with high change severity and low change frequency (note that in De Jong's classification, abrupt DOPs covers all problems with high change severity while the change frequency is not considered), and
- Chaotic: DOPs with high change severity and frequency.

Among the above classes, chaotic DOPs are the most challenging. They make the tracking operation very difficult since the new optimum can be far away from the previous and the available computational resources in each environment are very limited due to high change severity and frequency.

Besides the change severity and frequency, Branke and Schmeck [24] also use some other criteria, such as predictabil-

ity, change visibility, and aspect of change for classifying DOPs. For the predictable DOPs, some aspects of the problem follow a regular pattern, so they can be learned and then predicted. Periodical DOPs belong to this group of problems, hence, an algorithm can predict the position of the next optimum according to the historical information [30]. Similarly, in some DOPs where the optimum moves toward a fixed (i.e., unchanging) direction (e.g., linear changes in [31], see ($\mathrm{P_B}$-8)), the optimum trajectory is predictable. Another factor that can be predicted in some DOPs is the moment of the next environmental change. DOAs can learn these predictable characteristics and use them to improve the performance. In contrast, in unpredictable DOPs, the environmental changes do not follow any specific spatial or temporal pattern (i.e., they have random changes). Visibility of changes is another important criterion to classify DOPs. In DOPs with visible changes, the DOAs are informed about the occurrence of environmental changes. However, DOAs have to detect the changes in DOPs with invisible environmental changes. Another criterion to classify DOPs is the *aspects of changes*, which indicates which parts of the problem, such as dimension, objective function, and/or constraints, change over time.

Homogeneity/heterogeneity of DOPs is another criterion for classifying DOPs [10], [11], [29]. The heterogeneity of DOPs can be investigated from different aspects. The following DOPs can be considered as heterogeneous DOPs:

- DOPs whose different regions of the landscape change with different levels of severity. For instance, where some parts of the landscape change more severely than others [11], or where some parts of the landscape remain unchanged while the rest change [32].
- DOPs whose change severity changes over time.
- DOPs whose change frequency changes over time.
- Modular DOPs whose subfunctions have different characteristics, for example different change severities, change frequencies, and dimensions [11].

In heterogeneous DOPs, the problem characteristics can change over time, or can be different from region to region in the landscape. Consequently, tackling heterogeneous DOPs can be challenging since the DOAs need to adapt to different regions/environments with different characteristics over time. If a DOP has none of the above properties, it is considered homogeneous. For example, a DOP with fixed change frequency over time can be considered a homogeneous DOP (from this specific aspect).

$\mathrm{P_A}$-III. Taxonomy of the components of DOAs

Tackling a DOP is challenging, especially if it has higher change frequency and/or environmental change severity. On the one hand, DOAs need to locate the optimal solution quickly; on the other hand, they have to face environmental changes and properly react to them. It becomes even more challenging when only limited computational resources are available between successive environmental changes. Generally, the response actions that a DOA should perform after a change occurrence must address the following issues:

- Global diversity loss: this issue occurs due to the intrinsic nature of EAs and SIs converging to promising regions. In such circumstances, the exploration capability deteriorates considerably.
- Local diversity loss: during exploitation, individuals[3] of a (sub-)population usually collapse to an optimum. Consequently, even if the optimum shifts slightly after an environmental change, the tracking capability can still be hindered.
- Limited computational resources: usually, there are limited available computational resources, which in turn put limitations on the number of fitness evaluations during each environment. Thereby, controlling the usage of the computational resources and avoiding wasting them are essential.
- Outdated memory: after each environmental change, the stored fitness values which were calculated based on the previous environment, will become outdated. To address this issue, all solutions whose fitness values have been stored must be reevaluated in each new environment [1].

Generally, efficient DOAs are complex algorithms that use several components to address the challenges of DOPs. In Figure $\mathrm{P_A}$-2, we depict a general taxonomy for the components of DOAs by aggregating the existing taxonomies [1], [3]–[5]. Considering the current status of the field, the existing taxonomies suffers from the following shortcomings:

- Several important components of DOAs are not considered, such as convergence detection and resource allocation methods.
- The class of diversity control components is underrepresented. In particular, the components that increase diversity are only limited to those which are triggered after environmental changes. All the components used to increase diversity when some specific conditions are met, are ignored. The aim of the diversity control component is not precise, i.e., it is not clear whether they have been designed to address local or global diversity loss.
- Multi-population DOAs, as the most efficient and popular class of DOAs, have not been properly classified. Classifying such complex algorithms only on the basis of the number of subpopulations does not fully cover their structural differences.

As the surveys are usually written based on taxonomies, shortcomings of a taxonomy reflect on the whole survey.

In this section, we propose a taxonomy for the components of DOAs, which addresses the aforementioned shortcomings of the existing taxonomies. Figure $\mathrm{P_A}$-3 illustrates the proposed taxonomy. According to this taxonomy, DOAs' components are classified into the following classes:

- Convergence detection: these components do not address any challenges of DOPs directly. However, their importance relies in their application in many diversity control, population division and management, and resource allocation components.
- Change detection: many components of DOAs trigger an explicit response to a change, thus requiring to know when the environment has changed. In many real-world

[3]The term *individual* is used equivalently to a candidate solution in different optimizers, e.g., *particle* in particle swarm optimization [33].

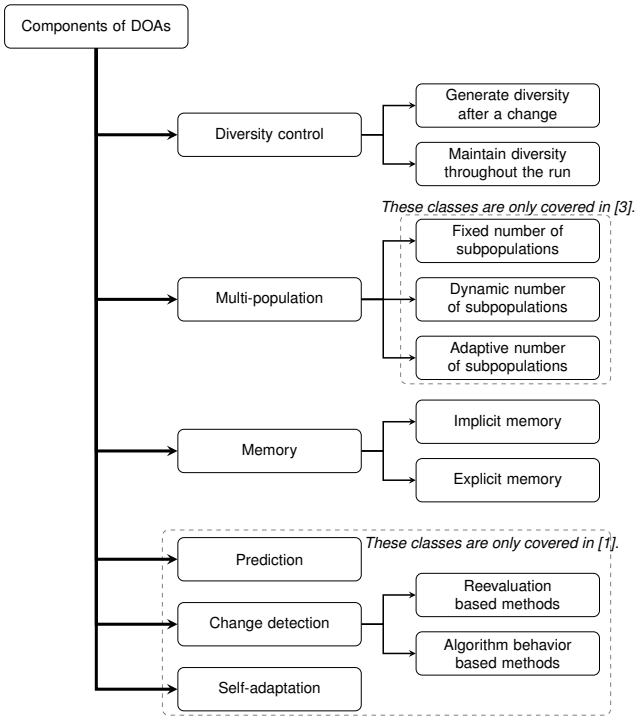

Fig. $\mathrm{P_A}$-2. Aggregation of all provided taxonomies of the components of dynamic optimization algorithms (DOAs) in the existing surveys [1], [3]–[5].

problems such information is known, but if it is not known, it has to be detected.

- Explicit archiving: the historical information obtained from previous environments, especially the location of the promising regions (optima), can be useful to accelerate the tracking process in each new environment. Some DOAs use *explicit memory* for archiving the historical information.
- Diversity control: to design an efficient DOA, global and local diversity loss issues must be addressed to maintain the quality of exploration and exploitation.
- Population division and management: two different approaches have been used in DOAs to manage the population, i.e., bi-population and multi-population. Moreover, we further classify multi-population approaches according to the used population clustering methodology, clustering frequency, homogeneity of subpopulations, using constant or variable values for subpopulation number and population size, and computational resource allocation.

Note that unlike the illustrated taxonomy in Figure $\mathrm{P_A}$-2, we do not consider any independent classes for implicit memory, self-adaptation, and prediction methods in our proposed taxonomy due to the following reasons. First of all, the class of implicit memory is outdated. It has been pointed out in [22] (1999) that explicit memory is more efficient and easier to understand than implicit memory, hence, these methods have been rarely used in the state-of-the-art DOAs. Second, the majority of the self-adaptive and prediction methods belong to the other classes of components, such as diversity control. For example, [34], [35] use self-adaptation and prediction methods in their diversity control component.

In the following subsections, we review the indicated classes of components in Figure $\mathrm{P_A}$-3.

A. Convergence detection

Convergence detection methods are used in different decision-making processes in DOAs, e.g., to determine whether a promising region has been found, whether the exploitation has been carried out (convergence to a peak summit), or whether the diversity has dropped. Although these methods do not directly address any DOP challenges, they are crucial to trigger some other components such as diversity control, subpopulation management, and computational resource allocation. These components start to work when a convergence, either in a population or in a subpopulation, has been detected. As can be seen in Figure $\mathrm{P_A}$-3, we classify these components into three groups which are discussed in the rest of this subsection.

1) Fitness monitoring: These convergence detection methods determine the convergence status of a population based on the individuals' fitness values. One commonly used method to detect convergence of a population is to monitor the fitness value of its best found position. It is decided that the population has converged if the fitness value of its best found position has not improved over the previous k iterations:

$$\begin{cases} Converged & f(\vec{\mathbf{g}}^{*(i)}, \vec{\alpha}^{(t)}) = f(\vec{\mathbf{g}}^{*(i-k)}, \vec{\alpha}^{(t)}) \\ Unconverged & f(\vec{\mathbf{g}}^{*(i)}, \vec{\alpha}^{(t)}) > f(\vec{\mathbf{g}}^{*(i-k)}, \vec{\alpha}^{(t)}) \end{cases}, \tag{$\mathrm{P_A}$-3}$$

where $\vec{\mathbf{g}}^{*(i)}$ is the best found position in the ith iteration. This method is first introduced in [36] and then has been used in several DOAs [37]–[41]. A considerable flaw is that it only can detect convergence if the best found position has not improved. If the best found position has improved only slightly, no convergence will be detected. The proposed method in [42] has addressed this shortcoming by modifying ($\mathrm{P_A}$-3) to:

$$\begin{cases} Converged & f(\vec{\mathbf{g}}^{*(i)}, \vec{\alpha}^{(t)}) - f(\vec{\mathbf{g}}^{*(i-k)}, \vec{\alpha}^{(t)}) \leq \epsilon \\ Unconverged & f(\vec{\mathbf{g}}^{*(i)}, \vec{\alpha}^{(t)}) - f(\vec{\mathbf{g}}^{*(i-k)}, \vec{\alpha}^{(t)}) > \epsilon \end{cases}, \tag{$\mathrm{P_A}$-4}$$

where ϵ is a positive constant. Note that ($\mathrm{P_A}$-3) is a special case of ($\mathrm{P_A}$-4) where $\epsilon = 0$. In [43], it is proposed that if an entire population have not improved at least l times during a predefined number of iterations, then the population is considered as converged. In [44], it is assumed that a population has converged if the standard deviation of the fitness values of its individuals is less than a threshold.

2) Spatial size monitoring: Another way to decide upon convergence status of a population is to use its spatial size. To this end, if the spatial size of a population is less than a predefined threshold, then there is a convergence. Note that in some works, the spatial size is denoted as diversity, radius, or size of a population. In the following of this subsection, we describe some commonly used methods to calculate the spatial size of a population.

In [13], the spatial size of a population a is calculated by:

$$\mathfrak{s}_a = \max_{d \in \{1, \cdots, D\}} \left(\max_{i,j \in a} |x_{i,d} - x_{j,d}| \right), \tag{$\mathrm{P_A}$-5}$$

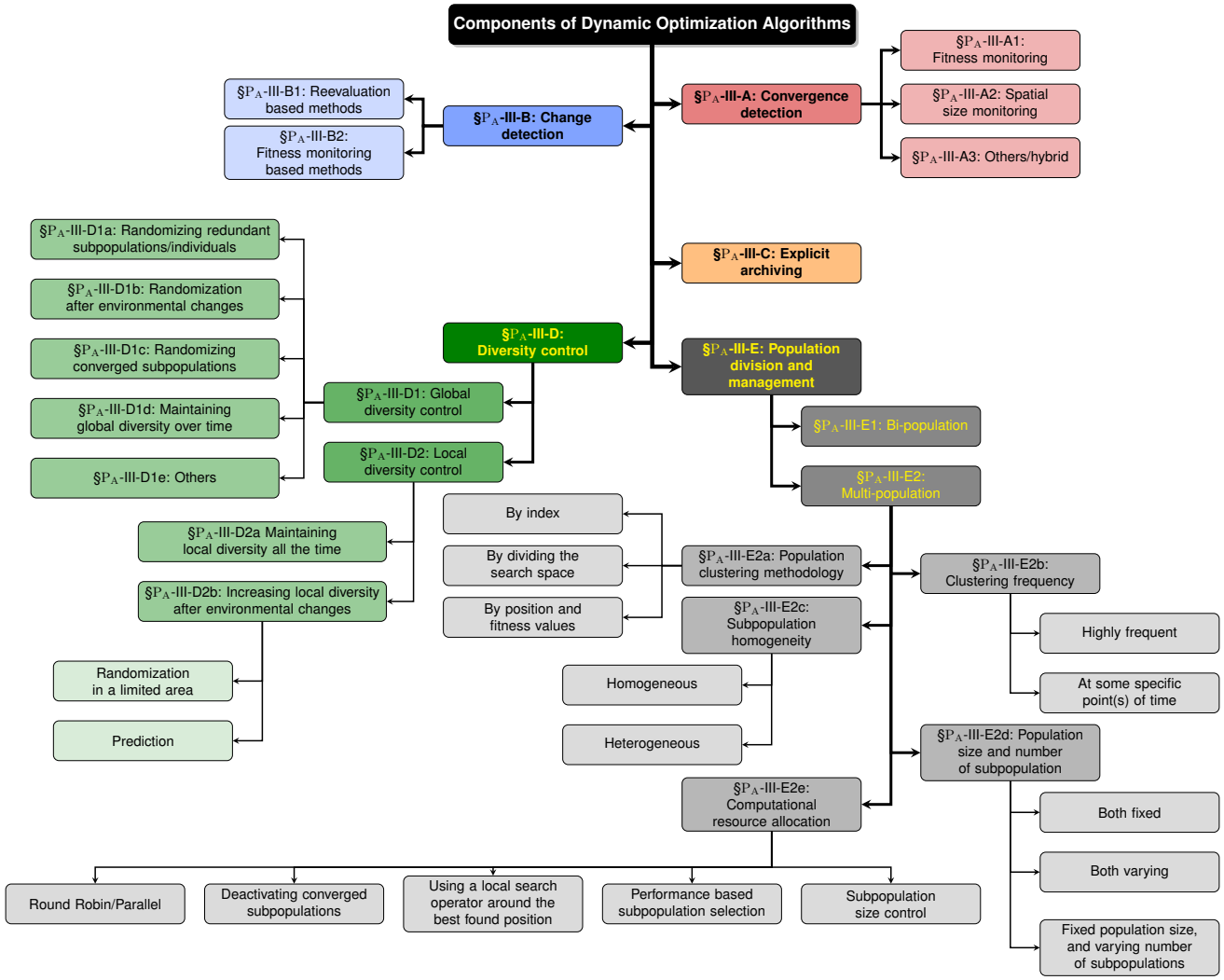

Fig. $\mathrm{P_A}$-3. Taxonomy of the components of dynamic optimization algorithms.

where D is the number of dimensions, and $x_{i,d}$ is the dth dimension of the ith individual. In fact, ($\mathrm{P_A}$-5) calculates the spatial size as the largest distance, along any axis, between any two individuals of a population. In [45], the spatial size of the population a is calculated as the maximum Euclidean distance between its best individual and other members, which is formulated as:

$$\mathfrak{s}_a = \max_{i \in a} \|\vec{\mathbf{g}}^* - \vec{x}_i\|, \qquad (\mathrm{P_A}\text{-}6)$$

where $\vec{\mathbf{g}}^*$ is the best found position (e.g., Gbest in PSO [46] or the best individual in DE [47]). In [48], the spatial size of the population a is calculated as the largest Euclidean distance between its individuals, which is formulated as:

$$\mathfrak{s}_a = \max_{i,j \in a} \|\vec{x}_i - \vec{x}_j\|. \qquad (\mathrm{P_A}\text{-}7)$$

This method has also been used in [48]–[50]. In [51], the spatial size of a population is calculated in a similar way to ($\mathrm{P_A}$-7), but the infinity norm distance replaces the Euclidean distance. Li and Yang [52] propose to determine the spatial size of the population a as the average Euclidean distance of all individuals to the center of the population:

$$\mathfrak{s}_a = \frac{1}{n_a} \sum_{i \in a} \|\vec{c}_a - \vec{x}_i\|, \qquad (\mathrm{P_A}\text{-}8)$$

where n_a is the number of individuals in the population a, and $\vec{c}_a$ is its center position, which is the average position of all members ($\frac{\sum_{i \in a} \vec{x}_i}{n_a}$). This method has also been used in several DOAs, including [53]–[55]. In [56], the spatial size of a population is calculated similar to ($\mathrm{P_A}$-8), but the center position is replaced by the best found position in the population.

3) Other/hybrid: There are some other proposed methods that do not fall into the above classes. In [57], a similar method to ($\mathrm{P_A}$-4) is used, but instead of monitoring the progress of the best found position based on its fitness values, its position is monitored. In this method, if $\|\vec{\mathbf{g}}^{*(i)} - \vec{\mathbf{g}}^{*(i-k)}\|$ is less than a threshold, then the subpopulation is considered as converged. Nasiri and Meybodi [58] use a hybrid method for convergence detection by monitoring both fitness and relocation distance of the best found position. It is assumed that a population

has converged if its best found position has not improved or relocated during the last k iterations (according to two different thresholds).

In some DOAs, the criteria for determining the convergence status is defined based on the optimization component's parameters. In [59], a local search operator is used as the optimization component, and it is assumed that it has converged if the value of its step size parameter is less than a threshold. In the PSO based DOA in [42], it is assumed that a subpopulation has converged if the velocity values of all particles of a subpopulation in all dimensions lie inside a predefined range $[-L, L]$.

4) Discussion on convergence detection: We have classified the convergence detection methods into two main classes of fitness and spatial size based methods. One main issue of the fitness based methods is that they are error-prone, i.e., they can wrongly decide an unconverged population is converged and vice versa. The convergence detection methods in ($\mathrm{P_A}$-3) and ($\mathrm{P_A}$-4) are less reliable since they only focus on the best found position which cannot represent the convergence status of a population accurately. Among the fitness based methods, the proposed method in [44] is more accurate as it considers the standard deviation of all individuals' fitness values. However, this method is not accurate in the regions with low gradients. In comparison to fitness based methods, the spatial size based methods are more accurate as they evaluate the convergence status of the population based on the positions of individuals. However, these methods are more computationally complex.

Overall, despite the importance of the convergence detection methods, they have rarely been investigated in the DOP literature. Up to now, there has been little study that compares different convergence detection methods or analyzes their effectiveness in different scenarios.

B. Change detection

The majority of DOAs are change-dependent with some components triggered after environmental changes, to provide necessary reactions such as addressing the *outdated memory issue* by reevaluating all solutions. Effectiveness of these DOAs relies on immediate reactions after environmental changes. Change detection components are suitable for DOAs that are designed for optimizing the DOPs with apparent or detectable environmental changes, which is the case in many real-world problems. We classify the change detection methods into reevaluation based and fitness monitoring based ones.

1) Reevaluation based change detection methods: The most commonly used methods to detect environmental changes are reevaluation based. In these methods, a number of solutions, called *detectors*, are reevaluated frequently (usually in every iteration), and if the obtained fitness values are different from the previous values, an environmental change is detected. The reevaluation based methods are first introduced by Branke in [22], where solutions in an archive are reevaluated in each iteration. Different versions of reevaluation based methods are used in the DOP literature where the detectors are chosen differently. Some suggest to take a fixed solution as the detector [60], or the best found position [61]. Others suggest to use a predefined number of randomly chosen individuals' positions [62], the best found position of each subpopulation [63], some randomly initialized solutions (not from the population) [64], and some randomly initialized solutions plus some randomly chosen individuals from the population [65].

2) Fitness monitoring based change detection methods: These methods try to detect environmental changes by monitoring the evaluated fitness values. In [61], the fitness values of the best and the second best individuals are monitored, and if both have not changed over a predefined number of iterations, it is assumed that a change has happened. In [58], the average of all evaluated fitness values from the last environmental change is monitored. If this value is becoming worse for a predefined number of fitness evaluations, the algorithm reevaluates a single detector to make sure an environmental change has happened. Observing the average fitness values is also used in [66], where if these values fluctuate considerably, it is assumed that an environmental change has happened.

3) Discussion on change detection: In many real-world DOPs, the occurrence of environmental changes is obvious, and algorithms are informed about them [10]. For example, the arrival of new orders, fault in a part of the system, change in temperature, change in the number of resources, and change in costs, are detected by sensors, operators, and agents, which inform the DOAs. Consequently, designing reaction based DOAs without any change detection component is reasonable.

Existing change detection methods by monitoring the behaviors of the evaluated fitness values over time are not proper methods as they are error-prone [2], [67]. These methods can miss an environmental change or detect a false one. On the one hand, detecting a false environmental change will result in an unnecessary reaction, which is costly and leads to a performance drop. On the other hand, missing an environmental change can significantly deteriorate the performance of DOAs since they cannot react to an unidentified change.

Unlike the fitness monitoring based components that do not need any additional fitness evaluation, the reevaluation methods consume computational resources by frequently reevaluating the detectors. However, they are more accurate and reliable. It is shown in [67] that the reevaluation methods are capable of performing *robust 100% detection* if a sufficient number of detectors are used. However, it should be mentioned that using too many detectors is prohibitively costly as they consume a considerable amount of computational resources.

Environmental changes in continuous DOPs can be detected easily when the change is global. In such a circumstance, reevaluating a single solution can accurately detect environmental changes [11], [42]. However, if only some parts of the environment change, detecting change becomes more challenging. Therefore, in DOPs with local environmental changes, using small numbers of detectors for change detection would be error-prone. It is shown in [32] that the rate of successful change detection deteriorates when the overall size of the affected regions in the environmental changes decreases. Consequently, to tackle DOPs with local environmental changes, algorithms must use larger numbers of detectors. To this end, the best found position of each subpopulation can be used as

the detector [13]. Therefore, if the local environmental changes happen in the covered areas by the DOA, the change can be detected.

A few DOAs are change-independent [32], [53], [68], i.e., their components and structures are not designed to react to environmental changes. This class of DOAs are suitable for the DOPs whose environmental changes are hard or impossible to detect. The structure and procedures of these DOAs are independent of any knowledge about the environmental changes.

C. Explicit archiving

Assuming there are some similarities between successive environments [6], it is common in DOAs to use the obtained information from previous environments to accelerate the process of discovering the optimum in the new environment. In DOAs, this information usually contains the location(s) of promising region(s), i.e., peak(s) in the fitness landscapes. This information is passed between consecutive environments and is often updated during each environment. To use historical information, DOAs usually use three main components: promising regions coverage, implicit memory [22], and explicit archiving (also called explicit memory). Promising regions coverage components try to keep a number of individuals around the discovered promising regions in the previous environments. These components are related to the population structure of the DOAs which will be reviewed in § $\mathrm{P_A}$-III-E. Some DOAs use implicit memory, such as diploidy methods [22], to use historical information. In this survey, we do not cover these methods as they have been rarely used in the last two decades to solve continuous DOPs. Finally, a number of DOAs use the explicit archiving components to store the locations of the discovered promising regions. The archived solutions are retrieved and injected into the population at some predefined points of time, which are usually right after environmental changes. Consequently, there will be some individuals around the previously discovered promising regions in order to accelerate the optimum tracking process. In the following of this subsection, we review the explicit archiving components from three different perspectives: *a*) what solutions to store and when, *b*) what solutions to delete and when, and *c*) what solutions to retrieve and when.

a) What solutions to store and when: In [69], a bi-population DOA, which consists of an explorer subpopulation and an exploiter subpopulation, is introduced. These subpopulations collaborate through an archive. When the explorer has converged, it sends its best found position to the archive. In [70], a multi-population method is introduced where the best found position by each subpopulation is sent to the archive at the end of each environment. In [71], the best found position of each subpopulation is sent to the archive once it has converged. This approach has been applied in some subsequent works such as [55], [72], [73]. In [74], the best found positions by subpopulations are sent to the archive after each environmental change.

In [59], an archiving method is proposed for single-population DOAs. In this method, when the optimization component has converged, its computational budget has been consumed, or an environmental change has happened, the best found solution is sent to the archive. Another archiving method for single-population methods is proposed in [75] where the best found position by the population is sent to the explicit archive when it has converged, or after environmental changes.

b) What solutions to delete and when: Usually, an archive has a finite capacity, hence, several methods have been designed to remove solutions if the archive is full. The replacement methods are designed based on: age of the archived solutions, distance between archived solutions or between archived solutions and a newly arrived solution, and comparing the fitness of the archived solutions and a newly arrived solution.

In [76], an aging based method is introduced where the oldest archived solution is replaced by the new one. In [77], a fitness based method is used where the worst archived solution is replaced by the new one. In [78], if there is an archived solution whose Euclidean distance to the new solution is less than a threshold, it will be replaced by it. Otherwise, if there is no close existing archived solution, the oldest solution can be replaced by the new entry based on a predefined probability. In [75], three different conditional replacement methods are designed, where the oldest, closest, or worst archived solution is replaced by a better new solution. In [55], [74], in the first step, archived solutions that are *similar* to the newly arrived solution (i.e., they are closer than a threshold), are determined. Then, if there are any similar archived solutions, the fitness of the worst one is compared to that of the new entry and it will be replaced if it is worse. Otherwise, if there is no similar archived solution, then the new entry is compared with the worst archived solution based on fitness and will replace the worst archived solution if it is better. In a simpler approach [79], the new solution is added to the archive in the first step, then the inferior of the two closest solutions is removed.

To efficiently manage the archive, some methods have been introduced to remove outdated and redundant solutions. Since the promising regions usually move after environmental changes, it is necessary to replace their previous archived positions with the updated ones. In [69], to update the outdated archived positions of the promising regions, when two solutions are closer than a threshold, the inferior one is removed. This idea is based on the assumption that each promising region's successive positions are relatively close. In [73], if the new entry solution is closer than a threshold to an archived solution, the old solution will be replaced by the new one (even if the archive is not full). In [80], similar solutions are not permitted in the archive, and if two solutions are closer than a threshold, the older one is removed. In [59], a new solution is added to the archive if it is better than the closest archived solution; otherwise, it will be discarded.

c) What solutions to retrieve and when: Usually, after environmental changes, the archived solutions are used to locate some individuals around the previously discovered promising regions. In [70], each individual is randomized around a solution from the archive which is chosen randomly with a predefined probability. In the introduced bi-population

DOA in [69], two methods for using archived solutions are designed:

- Archive-based resetting: all solutions in the archive are reevaluated and the best one is passed to the exploiter subpopulation if it is better than the exploiter's best found position.
- Archive-based migrants: all solutions in the archive are copied into the worst individuals of the exploiter subpopulation.

In [76], archived solutions participate in the selection process of the parents for crossover in the EA. In the multi-population DOA proposed in [78], there are also two suggested methods for using the archived solutions:

- A predefined number (n) of the best archived solutions are distributed almost uniformly among all m subpopulations and replace their worst individuals.
- A clustering procedure is performed on the archived solutions, and the n best cluster heads (the best solution in a cluster) are distributed almost uniformly among all m subpopulations and replace their worst individuals. The applied clustering method is originally designed to divide the population into subpopulations in [81]. In this method, the number of clusters is determined adaptively according to the distribution of solutions.

1) Discussion on the explicit archiving: In this subsection, we have reviewed how historical information is used in some DOAs using explicit archiving components. As pointed out in [22], using explicit archiving is suitable only for a class of DOPs where the optimum returns to a previous location, or previous environments reappear periodically. To tackle such DOPs, using archived historical information will make the DOAs to switch to the global optimum in the new environment instantaneously [1]. Furthermore, if the movement of the promising regions or peaks can be predicted, the archived solutions can be used as the training dataset for the prediction methods [35], [82].

However, using an explicit archive is questionable for solving DOPs whose environmental changes are random (i.e., unpredictable) [6]. In fact, efficient management of an explicit archive is a complex task. In addition to managing full memories and removal of redundant solutions, the archived solutions will be outdated after each environmental change. As a consequence, each archived solution, which represents a location of a promising region, must be updated by performing exploitation. This exploitation process can be performed by injecting archived solutions into the population and updating them by removing older similar (close) solutions [69], [78], or by performing independent local search on the archived solutions after each environmental change [37], [83], [84]. In some multi-population DOAs, after discovering a converged subpopulation, its best found position is archived, then the subpopulation is either randomized to search for undiscovered promising regions, or removed to avoid wasting computational resources [55], [72], [73]. However, the explicit archiving components can be replaced by a simple deactivation method where the converged subpopulations are deactivated until the next environmental change [42], [49]. Then, the DOA will use promising regions coverage approach instead of the explicit archiving. The effectiveness and simplicity of passing the locations of the discovered promising regions by covering them with individuals/subpopulations, make them popular among researchers. Based on our readings, the number of DOAs that use the individuals/subpopulations to cover promising regions is around four times larger than the number of those using the explicit archives.

D. Diversity control

Diversity loss is one of the most critical challenges of DOPs and one of the main reasons behind the inefficiency of static optimization algorithms in solving DOPs, because converged populations are incapable of efficiently searching for the new optimum after an environmental change. In this survey, we classify the components that have been designed to address the diversity loss into two main groups according to whether they address local diversity loss or global diversity loss. On the one hand, the methods that address global diversity loss, try to increase global diversity across the search space in order to improve the exploration capability of DOAs. On the other hand, the methods that address local diversity loss increase the exploitation capability.

1) Global diversity control: Maintaining global diversity across the search space or increasing it when necessary is vital for DOAs. Multi-population methods address the global diversity loss issue by using several subpopulations over the search space. However, they are in need of some other components to control the subpopulations in order to cover the search space efficiently. In addition, even multi-population methods cannot cover the whole search space. This results in some parts of search space to remain uncovered which potentially contain the new global optimum. Hence, multi-population methods will also end up using some additional components to control global diversity systematically. As can be seen in Figure $\mathrm{P_A}$-3, in this survey, we classify applied components for addressing the global diversity loss issue into five groups, which are described in the following of this part.

a) Randomizing redundant subpopulations/individuals: Using one subpopulation with a reasonable number of individuals is usually enough to cover a promising region and tracking it. Therefore, using more than one subpopulation or using a large number of individuals often deteriorates the performance of the DOAs. One reason behind this is the global diversity dropping as a result of concentrating several subpopulations (or many individuals) in the same promising regions, which causes an overcrowding issue. Besides, these redundant subpopulations and individuals waste valuable computational resources. To handle this issue, DOAs usually randomize redundant subpopulations/individuals to increase the global diversity. In the following of this part, these methods are described.

Branke [85] proposes the first method for removing redundant subpopulations/individuals. In this DOA, which is called Self Organizing Scouts (SOS), a multi-population approach is used where there is a parent subpopulation (explorer) and several child subpopulations (exploiters). The

child subpopulations are responsible for covering and tracking the promising regions which have been discovered by the parent subpopulation. If any parent's individuals lie inside a child subpopulation's search region, its individuals will be randomized. Moreover, if the best individual of a child subpopulation lies inside the search region of another child subpopulation, its individuals will be randomized and rejoin the parent subpopulation. The aforementioned randomization processes in SOS prevent overcrowding and maintain global diversity.

In [63], if the Euclidean distance between two subpopulations' best found positions (e.g., Gbest in PSO) becomes less than a predefined radius r_{excl}, then the subpopulation with better best found position is kept and the other one will be randomized. This method that acts based on the Euclidean distance between subpopulations' best found positions, is known as *exclusion* method and is used in many DOAs [70], [86]–[88]. Parameter r_{excl} is defined differently in the literature. In [63], r_{excl} is defined as a constant value. In [13], the value of r_{excl} is calculated according to the number of peaks, the search range, and the number of dimensions. However, the number of peaks is usually not known due to the black-box assumption of the problems. Later on, in [89], the number of peaks is replaced by the number of discovered peaks. In [90], individuals are clustered using fuzzy c-mean [91], where the number of clusters is set to the number of subpopulations. Then, for each cluster, the minimum distance to other clusters is calculated. The average of these distances is used for setting r_{excl}. Authors claim that doing so, the environment conditions (the current peaks' parameters) and the distribution of individuals are considered for r_{excl} setting.

A modified version of the exclusion method is proposed in [92]. In this version, if two subpopulations enter the mutual exclusion area of each other, before randomizing the inferior one, the *midpoint* between their best found positions is obtained. If the fitness of the midpoint is worse than both best found positions, neither should be re-initialized, because the subpopulations are likely residing on two peaks whose distance to each other is less than r_{excl}. This method has been further modified in [93], where in addition to the midpoint, two points close to each of the best found positions of the involved subpopulations are also taken into consideration. If the fitness of at least one of the three points is less than the best found positions, neither is re-initialized. Otherwise, the inferior subpopulation will be re-initialized. Kordestani et al. [94] use a new method for randomizing the redundant subpopulations to increase the global diversity. After exclusion detection, the individuals of the inferior subpopulation are re-initialized one by one. After randomizing an individual, its distance to all subpopulations' best found positions are calculated. The re-initialized individual is accepted if it does not reside in the exclusion area of any other subpopulations.

In [95], a multi-population DOA, denoted as *speciation* based DOA, is proposed that uses a radius based clustering method to decide on subpopulations. The clustering is carried out every iteration. In this method, when two subpopulations reside on the same peak, they will get closer by converging to the peak summit. Afterward, the individuals of the inferior subpopulation will join the superior one once they enter its clustering radius. Thereafter, if the number of individuals in a subpopulation is more than a predefined threshold, the redundant inferior individuals are randomized across the search space. Note that merging subpopulations and randomizing redundant individuals to avoid overcrowding is usually used in the clustering based multi-population DOAs in which the clustering process is carried out frequently (usually in every iteration) [81], [95], [96]. This method is modified in [97], where it uses another threshold for the minimum number of randomized individuals. In this DOA, if the number of randomized redundant individuals is less than the minimum number, individuals are removed one by one from the inferior subpopulations to reach that number.

In [98]–[100], the search space is divided into hypercubes using cellular automata (CA) [101]. To maintain the global diversity, when the number of individuals in a cell (hypercube) passes a threshold, the worst individuals are randomized.

b) Randomization after environmental change: In many environmental change reaction based DOAs, the global diversity is increased after each environmental change. The proposed single-population DOA in [102] increases the global diversity after environmental changes using a random immigration method. In the bi-population DOA in [22], the explorer subpopulation is re-initialized after each environmental change. Randomizing a predefined portion of the population of a single-population DOA after each environmental change is introduced in [61]. Karimi et al. [103] propose a single-population method with varying population size $N \in [N_{\min}, N_{\max}]$. The global diversity is increased by inserting $N_{max} - N$ random individuals across the search space after each environmental change.

For some multi-population DOAs, global diversity is increased by randomizing some inferior subpopulations or inferior individuals in each subpopulation after each environmental change. In [52], the best individual of each subpopulation is kept, and the rest are randomized across the search space. Afterward, the clustering procedure is run to form new subpopulations. In [104], a predefined percentage of the inferior individuals in each subpopulation are randomized across the search space. In [105], randomizing the inferior of similar individuals, which are the ones whose distance is less than a threshold, is used to increase the global diversity. In [79], the individuals of each subpopulation whose fitness values are less than the median of the fitness values in the subpopulation, are randomized. In [106], subpopulations are divided into two groups of better and worse subpopulations. The individuals of the subpopulation in the worse group are randomized across the search space. After that, the randomized individuals are clustered to form new subpopulations.

Another group of DOAs that increases the global diversity after environmental changes are the ones that use an explicit archive to store the locations of the discovered promising regions (see § $\mathrm{P_A}$-III-C). In [71], the archived solutions are copied into some individuals, and the rest of them are randomized across the search space. In [107], a predefined number of individuals of each subpopulation are replaced with better archived solutions, and the rest are randomized

across the search space, which results in increasing the global diversity.

c) Randomizing converged subpopulations: One commonly used method in many DOAs is to randomize converged subpopulations (see § $\mathrm{P_A}$-III-A for convergence detection methods) to improve the global diversity and exploration capability. In multi-population DOAs with an adaptive number of subpopulations [11], there is usually a free (non-converged) subpopulation. Once the free subpopulation has converged to a promising region (e.g., a peak), it creates a subpopulation to continue exploitation on the discovered promising region. Then, the free subpopulation will be randomized across the search space to search for uncovered promising regions. Therefore, the number of subpopulations adapts to the number of discovered peaks in these DOAs. This type of multi-population approach is introduced by Blackwell [87], and has been used in many DOAs [42], [51], [108], [109].

In some explicit archiving based DOAs (see § $\mathrm{P_A}$-III-C), when a subpopulation has converged, it sends its best found position to the archive, then it will be randomized [72], [110], [111]. In [43], the converged non-best subpopulations are randomized. In this method, the subpopulation with the best found position among all subpopulations is not randomized. Consequently, this subpopulation continues performing the exploitation around the best found position while the other converged subpopulations are randomized to increase the exploration capability of the DOA.

The anti-convergence method is introduced by Blackwell and Branke [13], which increases the global diversity when all subpopulations have converged. In such a situation, the subpopulation with the worst best found position is randomized. By performing the anti-convergence mechanism, the DOA sacrifices coverage of the discovered promising region with the worst fitness in the current environment to increase the global diversity. This method has been used in several DOAs [45], [112], [113].

d) Maintaining global diversity over time: Another way to address the global diversity loss issue is to use some optimization components that maintain a high degree of global diversity over time. Consequently, in DOAs that use such optimization components, the challenge of global diversity loss is addressed automatically [114]–[117]. For example, Crowding DE (CDE) [118] has been used in several DOAs, such as [40], [73], [110], [119]. In the aforementioned DOAs, CDE is responsible for locating the promising regions, exploration, and maintaining the global diversity.

e) Other methods: Various methods have been used in DOAs that cannot fit into the above classes. Moreover, these methods have rarely been used in DOAs, so we have not provided separate classes for them.

In a group of DOAs, some aging based methods are used for randomizing individuals/subpopulations. In [70], if the age of a non-best individual is more than a threshold, it is randomized with a given probability. In addition, if the age of the best individual exceeds another threshold, the subpopulation will be randomized with another given probability. An aging method is used in [120], where the number of iterations that an individual used to be the best b or worst w in its subpopulation is counted. When b reaches a threshold, the subpopulation containing the individual gets randomized. Furthermore, if w reaches another threshold, the individual is randomized.

In [55], when the spatial size of a subpopulation decreases, its number of individuals is adaptively decreased (until a minimum threshold is reached) since fewer individuals are enough to perform exploitation. Then, after every predefined number of fitness evaluations, the algorithm performs a procedure for increasing the global diversity using the removed individuals.

In some DOAs, the global diversity is increased when some specific conditions are met [121]. These methods are usually used in the DOAs that are change-independent. In [32], when the ratio between the remained individuals to the initial value of the overall number of individuals becomes less than a threshold, a re-diversification process is performed. In this process, all the best found positions by converged subpopulations are kept, and the rest of the individuals are randomized. Removing redundant individuals in the population is performed by other components of the DOA. A similar approach is applied in [68], but with different conditions. In this method, a time period parameter δ is defined. If the ratio of the difference between the number of subpopulations in the current fitness evaluation FE and $FE - \delta$ is less than a threshold, the algorithm will perform a diversity increasing process. This condition is changed in [53] to the moment when the average radius of non-stagnating subpopulations is less than a predefined percentage of the search range.

2) Local diversity control: When a (sub)population has converged to an optimum position (i.e. peak summit), its individuals are extremely close to each other. Consequently, after an environmental change, when the optimum position has changed, the population will be incapable of tracking it. To counteract this issue, the following methods have been defined.

a) Maintaining local diversity over time: Maintaining the local diversity inside each subpopulation at a predefined level is a common approach to address the local diversity loss. One way to maintain local diversity is the *collision avoidance* [63]. In this method, *charged individuals* are used whose update rules include an acceleration part to avoid collision and to create repulsion between individuals. In [122], a *disperse* method is proposed which calculates the distance between all individuals and the best found position. Afterward, a predefined fraction of the closer individuals update their positions in the opposite direction.

In [63], so-called *quantum individuals* are generated uniformly at random in a hyperball whose center is the best found position by a subpopulation. The radius of the hyperball is defined with a parameter r_{cloud}. If a quantum position is better than the best found position (in terms of fitness value), the quantum position is copied to the best found position. Using quantum individuals alongside the standard individuals of an optimization component is used in several DOAs to maintain the local diversity of each subpopulation over time [13], [45], [48], [120]. *Brownian individuals* [53], [55], [86], [88] are similar to quantum ones, but a Gaussian distribution is used to generate positions.

Another method to maintain local diversity of subpopulations is to add noise to the position of the individuals in each

iteration. In [86], an entropic model is used where a Gaussian step is added to individuals every iteration. In [100], a random factor is added to the velocity update rule of PSO, which results in maintaining the local diversity.

Another way suggested in the literature is to first allow the local diversity to decrease upto a threshold, then increasing it. Consequently, the local diversity of each subpopulation will remain in a predefined range. In [45], [96], when the local diversity of a subpopulation becomes less than a threshold, half of the neutral individuals turn into quantum ones to increase the local diversity.

b) Increasing local diversity after environmental changes: Some DOAs allow the subpopulations to continue converging to the optima to increase their exploitation capability. Then, after each environmental change, local diversity of subpopulations is increased to address the local diversity loss. We classify these components into the ones that randomize the individuals in a limited area and the ones that locate the individuals according to a predicted optimum position.

Randomization in a limited area: In [89], quantum individuals are used for a limited number of iterations after each environmental change to increase the local diversity of subpopulations. Hashemi and Meybodi [99] suggest to turn half of the standard individuals to quantum ones for a predefined number of iterations after each environmental change. In [123], a predefined number of the worst individuals of each subpopulation are randomized around the best found position with Gaussian distribution. The individuals of each subpopulation are randomized around its best found position with uniform distribution in [42], [90]. In [93], [111], a random number generated with normal distribution is added to the positions of all individuals. In [109], the individuals of each subpopulation are relocated around the best found position using a chaotic mapping method. Woldesenbet and Yen [76] propose to use a *variable relocation* method for adapting converged individuals in the new environment based on the estimated change severity.

Prediction: In [35], [82], a component is developed to predict the next position of the optimum by learning the correlations between successive environments. After each environmental change, some individuals are located on/around the predicted position of the optimum to accelerate the tracking process. This prediction component can be considered as a local diversity increasing method that systematically increases the local diversity of each subpopulation toward a predicted optimum position. Such methods are effective in the DOPs which are predictable, such as those whose local optima (peaks) movements are correlated with their previous movements.

3) Discussion on diversity control: An efficient DOA must address both global and local diversity loss. To address the global diversity loss, DOAs usually use more than one method. Many DOAs, especially the multi-population ones, use methods that randomize the redundant subpopulations/individuals (see § $\mathrm{P_A}$-III-D1a). The reason is that these methods not only increase global diversity, but they also prevent wastage of the computational resources by avoiding overcrowding situations. A considerable portion of these methods are exclusion ones, which guarantee that each promising region or peak is covered by not more than one subpopulation. However, a shortcoming of the exclusion methods is that they usually cannot distinguish two subpopulations on the same peak from two subpopulations on two peaks that are very close. Some works such as [92], [93] try to address this issue by using *midpoints*.

Randomizations across the search space after environmental changes are usually suitable for single and bi-population DOAs to increase their global diversity. However, in multi-population DOAs, the global diversity is usually maintained by randomizing redundant subpopulations/individuals and some converged sub-populations. To further increase the global diversity, some multi-population DOAs use also post environmental change randomizations across the search space, such as [104], [106]. A consequence of using randomizations across the search space after environmental changes for multi-population DOAs is that some subpopulations may lose track of the promising regions that they are covering. More specifically, these subpopulations may migrate to better promising regions that can be covered by other subpopulations.

As mentioned in § $\mathrm{P_A}$-III-D1c, Some DOAs randomize converged subpopulations. One such group of DOAs are those whose number of subpopulations is adapted to the number of discovered promising regions [42], [87]. In these algorithms, a free subpopulation(s) is used as the explorer and once it has converged, the DOA assumes that a promising region has been discovered. Thereafter, a tracker/exploiter subpopulation is generated in the area, and the free subpopulation will be randomized. This type of DOA is one of the most efficient optimization algorithms in the DOP field [11]. In some DOAs where the converged subpopulation is randomized without creating an exploiter subpopulation or storing the position in an archive, the DOA will lose track of the discovered promising region. This also happens in the DOAs that use the anti-convergence method where the algorithm will lose track of the current worst discovered region, which can contain the global optimum after environmental changes.

The DOAs that use optimization components which preserve a high degree of global diversity are not very efficient because a considerable amount of computational resources are wasted for performing movements that result in maintaining the global diversity. This can be also observed in DOAs that try to maintain local diversity of their subpopulations over time. In these methods, a significant amount of computational resources are used to maintain local diversity [1]. The methods that increase local diversity after environmental changes are more effective since they consume considerably less computational resources. However, these methods cannot be used for DOPs whose environmental changes are undetectable [32].

One crucial matter in using the methods that address the local diversity loss, is degree of the local diversity that should be preserved or increased in each subpopulation. On the one hand, if the local diversity is maintained/increased too high, the exploitation capability will deteriorate. On the other hand, if the local diversity is very low, the tracking speed will deteriorate. One suggestion is to maintain/increase the local diversity based on the *shift severity* values (i.e., the peak

relocation length). In [20], [51], the shift severity of each peak is estimated by calculating the distances between the best found positions by each tracker subpopulation at the end of successive environments.

E. Population division and management

We classify the main population division and management components applied in DOAs into two main classes: bi-population and multi-population. The number of subpopulations in bi-population DOAs is fixed and cannot be changed unless the structure of the algorithm is modified. However, the number of subpopulations in multi-population DOAs is a variable that can be set by either user or internal procedures of the DOAs. Note that the single-population DOAs use the default population structure of the static optimization algorithm. Therefore, we do not consider them in this section since they do not use any specific population division and management approach to tackle DOPs. Single-population DOAs usually try to tackle DOPs using other components such as diversity control [124] and explicit archive [59], [75].

1) Bi-population DOAs: Bi-population DOAs use two subpopulations where one is usually used for exploration, and the other is responsible for exploitation. In [22], subpopulations communicate through an explicit archive. The explorer subpopulation is responsible for discovering promising regions and storing them in the archive. The exploiter subpopulation is responsible for using fit old solutions from the archive and performing exploitation around the best found position. Therefore, using the explicit archive, this DOA is capable of using historical information of the discovered promising regions to accelerate tracking process. This framework has also been applied in [83].

Another way to use historical information of the discovered promising regions is to cover multiple promising regions using some optimization components that are designed to maintain global diversity and locate multiple optima, such as CDE [118]. CDE has originally been designed for detecting multiple peaks in static multimodal optimization. These type of optimization components are usually used as the explorer subpopulation in bi-population DOAs. CDE is used in several DOAs [40], [110], [119] to locate and cover multiple promising regions. After each environmental change, CDE's individuals are reevaluated and its current best found position is sent to an exploiter subpopulation. In [114], a modified PSO is used which acts similar to CDE in locating multiple promising regions. Similar approaches have been used with other optimization algorithms [115]–[117].

2) Multi-population DOAs: Multi-population DOAs are the most effective and flexible methods to tackle DOPs [1], [125]. These DOAs use several subpopulations where the number of subpopulations is a parameter that can be set by either the user or adaptively. As discussed in § $\mathrm{P_A}$-III-C1, multi-population DOAs are the most suitable and popular methods to pass the historical information of the discovered promising regions to each new environment by covering them using individuals/subpopulations. In the following of this part, we review the multi-population methods from various aspects.

a) Population clustering methodologies: In the DOP literature, different clustering methods have been used to divide the main population into subpopulations. The most commonly used clustering methods can be classified based on index, search space dividing, and individuals' positions/fitness. In each cluster, the optimization process is usually done independently from other clusters. To this end, the position of each individual is updated according to its position and those in the same cluster. In other words, the *attractors* in each cluster are chosen form its individuals.

By Index: In this commonly used method, the individuals of each subpopulation are clustered according to their indices. For example, if ten individuals are supposed to form two subpopulations, then the individuals with indices $1-5$ form the first subpopulation and the ones with indices $6-10$ form the second. Such a population division approach does not add any computational burden to DOAs. However, it does not consider the attributes of individuals. This population division approach was first introduced by Blackwell and Branke [63], and has been used by many DOAs [1]. Note that the subpopulation membership for individuals in these DOAs is usually fixed over time.

Dividing search space: In this approach, the search space is divided into hypercubes. Then, either the individuals in each hypercube or some neighbor hypercubes, form a subpopulation. These methods have been introduced by Hashemi and Meybodi [98], where the search space is divided into hypercubes, which are cells of a CA [101]. The information of the individuals in each hypercube is stored in the memory of the corresponding cell. In each cell, the local attractor is defined as the best individual in the cell and its neighbor cells. This method has been used in several DOAs [99], [100], [126].

By position and fitness values: In some multi-population DOAs, the population is divided into subpopulations using clustering methods that work based on the position and fitness value of individuals. In some of these clustering methods, only the individuals' positions are considered. K-means and fuzzy c-means, which are two popular data clustering methods, are utilized in some DOAs for clustering individuals based on their positions [71], [73]. In these methods, the number of subpopulations is an input parameter of the clustering method.

In [32], [68], [127], a clustering method is used with an upper bound threshold for the maximum number of individuals $n_{\max}$ in each subpopulation/cluster. In the beginning, each individual forms a cluster, and the Euclidean distances between all pairs of clusters, which are individuals in this step, are calculated. Thereafter, the clustering method finds the closest clusters whose accumulative number of individuals is not more than $n_{\max}$ and merges them. After merging two closest clusters, the distances between all clusters are updated. Note that the distance between two clusters is the Euclidean distance between their two closest individuals. The aforementioned steps are repeated until there will be no cluster with only one individual. In the clustering method in [53], similar to the aforementioned method, each individual forms a cluster in the beginning. After that, the closest clusters are merged until the *sum of intra-cluster distances* becomes less than the *sum of inter-cluster distances* [128].

Some clustering methods take the fitness values of individuals into account as well [38], [81], [84]. In these methods, some better individuals usually become cluster heads. In [95], a clustering method is used that considers the position of the individuals and their fitness ranks. This method works based on a predefined radius r_s. In the first step, all individuals are sorted based on their fitness values and form a sorted list of L. There is a set of cluster heads (seeds) S and a set of members M, which are both empty in the beginning. The best individual in L is assigned as the first cluster head, hence is removed from L and is added to S. The main procedure after this step is that if the Euclidean distance between the best individual in L and all the cluster heads in S is not less than r_s, then this individual is removed form L and added to S. Otherwise, it will be added to M and removed from L. This procedure is repeated until $L = \emptyset$. Thereafter, individuals in M are assigned to the best cluster head in S whose Euclidean distance is less than r_s.

In [129], the population is clustered using a *nearest better clustering* method, which works based on graphs. First, the Euclidean distances between all individuals are calculated. Then, each individual is connected to its nearest better (i.e., with better fitness value) individual, which creates a spanning tree. Afterward, the average of all edges $\bar{e}$ in the tree is calculated, and the edges whose lengths are larger than $\bar{e} \cdot \theta$ are removed, where θ is a positive adaptive parameter that is defined according to the current iteration and the maximum iteration. The remaining connected individuals belong to the same subpopulation.

In the clustering method in [38], [84], [130], all individuals are listed in a list L. Then, the best individual in L and its $m-1$ nearest individuals are chosen as a subpopulation with the size m, and then all will be removed from L. This process is repeated until L becomes empty.

Here, we focus on the most significant clustering methods applied in the literature. Some other clustering methods used in the literature to form subpopulations based on the position and fitness value of the individuals can be found in [37], [41], [55], [76], [131].

b) Clustering frequency: Multi-population DOAs can be categorized according to how frequent they perform clustering. It can be highly frequent, or only at some specific point(s) in time. In the DOAs with index based clustering, usually, the population is either divided at the beginning of optimization or when a subpopulation is generated. In either case, there will be no change during optimization [13], [89], hence the individual memberships in subpopulations are permanent. In DOAs that use clustering methods based on the positions, distances, ranks, and/or fitness values, the individuals are either re-clustered every iteration [81] or at some specific points in time, which is usually after environmental changes [127]. In the DOAs that divide the search space into hypercubes, the search space is initially divided into hypercubes that will not change later. However, the individual memberships will change over time by moving from one hypercube to another one. Therefore, these DOAs do not perform any clustering, but the lists of individuals in different hypercubes are updated every iteration.

c) Homogeneity of subpopulations: In some multi-population DOAs, subpopulations are homogeneous, i.e., the optimization component and its parameter settings (e.g., individual numbers), and the role of subpopulations are identical. Some multi-population DOAs with index-based clustering use homogeneous subpopulations [13], [86]. In addition, multi-population DOAs that use clustering methods in which the number of individuals is predefined as an input parameter, have homogeneous subpopulations [38], [84], [130]. Another group of DOAs use heterogeneous subpopulations, where the optimization component or the parameter settings of subpopulations are determined based on their specific tasks/roles. In [42], the free subpopulation is responsible for performing exploration that has a different parameter setting compared to the exploiter subpopulations. In [132], a fast evolutionary programming is used for locating promising regions and performing exploration, while PSO subpopulations are used to exploit and track the discovered promising regions (i.e., it uses two different optimization components).

d) Population size and number of subpopulations: The population size and the number of subpopulations in a DOA can be fixed or variable depending to the used components and the population structure. Among multi-population DOAs, the ones with fixed population size and the number of subpopulations are the most inflexible. One main reason is that they cannot adapt the number of subpopulations to the number of discovered promising regions in the search space. The proposed framework in [13] uses a fixed number of subpopulations with fixed individual memberships in subpopulations. As a result, the overall size of the population is fixed. In this group of DOAs [48], [86], [94], the performance is dependent to the defined number of subpopulations, which needs to be tuned for different DOPs.

In most DOAs that use the clustering methods based on individuals' attributes (e.g., position and fitness value), the population size is fixed [37], [56], [72], [81], [127]. However, according to the clustering method, their number of subpopulations can vary over time. Usually, DOAs that use a clustering method whose input parameters include the number of subpopulations or the subpopulation size, have a fixed population size and number of subpopulations. Otherwise, the number of subpopulations changes over time according to the distribution of individuals [32], [37], [56], [81], [127]. DOAs whose overall population size is fixed but their number of subpopulations can vary, are more flexible than the DOAs with fixed population size and subpopulation number. Note that this flexibility is limited since there are usually thresholds for the minimum and maximum numbers of individuals in each subpopulation.

Finally, DOAs with varying population size and the number of subpopulations are the most flexible multi-population DOAs. In these DOAs, such numbers are usually adapted to the number of discovered promising regions. On the one hand, these DOAs use smaller populations and fewer subpopulations when the DOPs consist of few promising regions. On the other hand, they produce more individuals and subpopulations when there are many promising regions in the search space. Besides, they can adapt these numbers during optimization where the

number of promising regions in the search space changes over time. Some of these DOAs use an adaptive number of subpopulations, homogeneous subpopulations, and index-based clustering, which are introduced in [87]. Li et al. [53], [68] propose DOAs with varying numbers of subpopulations and individuals, which use position based clustering.

e) Computational resource allocation: Computational resource allocation methods are one of the main components of multi-population DOAs. The aim of these methods is to monitor and manage the consumption of computational resources by different subpopulations. In the proposed taxonomy, these methods are classified into five groups which are reviewed in the following of this part.

Round Robin/Parallel: In many multi-population DOAs, the computational resources are allocated equally to different subpopulations. Therefore, depending on the DOA implementation, the subpopulations are either run in a parallel or concurrent manner using the classic Round Robin method. Using this method, in each iteration of the DOA, the optimization process of each subpopulation is run for one internal iteration. Generally, the uniform allocation of computational resources is not usually an effective approach as the role of subpopulations, their task achievements, and their priorities are not taken into consideration.

Deactivating converged subpopulations: In some DOAs, to avoid wasting computational resources due to unnecessary exploitation, the subpopulations that have converged to the local optimum are deactivated until the next environment. Kamosi et al. have proposed the idea of deactivation in [49], which was originally inspired by the removal of converged subpopulations in [52] to avoid *over-exploitation*. This idea has been used in several DOAs [104], [113], [133]. In [42], the best subpopulation is not involved in the deactivation method since its output directly affects the output of the DOA. Therefore, this subpopulation continues to perform exploitation to improve the quality of the best found solution in each environment.

Using a local search operator around the best found position: As stated before, the effectiveness of exploitation around the best found position directly affects the performance of the DOA. Considering this, in some DOAs, additional computational resources are allocated to the best subpopulation in order to strengthen the local search around the best found position. To this end, these DOAs use a local search operator on the best found position (among all subpopulations) in each iteration. This idea was first introduced by Rezazadeh et al. [106] and has been used in many DOAs [50], [72], [115].

Performance based subpopulation selection: In some DOAs, the computational resources are allocated to the subpopulations with higher *performance*. The performance of a subpopulation can be defined in different ways. In [88], a subpopulation's performance is determined according to the fitness value of its best found position. A predefined number of superior subpopulations use the whole computational resources for a certain period after each environmental change, while the remaining inferior subpopulations remain deactivated during this period.

In [92], the performance of each subpopulation is defined based on two factors: 1) the improvement of subpopulation's best found position in the last iteration that it was active, and 2) the difference between the fitness of the subpopulation's best found position and the worst of the best found position among all subpopulations. In each iteration, after determining the performance of subpopulations, the subpopulation with the highest performance is run. This subpopulation continues running until its performance drops below that of another subpopulation. The primary goal of this resource allocation method is to run subpopulations that are residing on the best promising regions (peaks) earlier in each environment. This method has been improved in [36], where a penalty value is incorporated with the performance measuring of each subpopulation. The penalty value is deducted from the performance value of a subpopulation when there is no improvement in its best found position in the previous iterations.

In [134], the performance of each subpopulation is defined according to its *success rate* and the fitness of its best found position. In this method, the success rate is the number of improved individuals in the last iteration divided by the subpopulation size. In each iteration, the subpopulation with the highest performance is chosen to be run.

Subpopulation size control: Controlling the number of individuals in subpopulations (i.e., size of a subpopulation) is another way to manage computational resource consumption. Note that these methods are different from those multi-population DOAs with varying subpopulation size as a result of clustering. Herein, each subpopulation size is determined according to the fitness of its best found position, where better subpopulations will have larger numbers of individuals and vice versa. Assigning more individuals to more promising regions is a way to allocate more computational resources to strengthen the exploitation around such regions. This idea is first introduced by Branke [85], where larger numbers of individuals are assigned to more promising regions. In [74], [88], more individuals are assigned to the better subpopulations using a migration approach. In this approach, individuals are removed from the inferior subpopulations and added into the superior ones. Removing unnecessary individuals around promising regions is another way to prevent computational resource wastage. In all reviewed methods in § P_A-III-D1a, where the redundant individuals/subpopulations are randomized, the main goal is to increase the global diversity. However, if the redundant individuals/subpopulations are removed instead of being randomized, the main aim of the method will be more efficient computational resource allocation [42], [127].

3) Discussion on population division and management: In this part, we have reviewed DOAs according to the population division and management components they adopt. Multi-population DOAs outperform single and bi-population approaches [1], [11] since their structure is more flexible for addressing diversity loss using the diversity control components. In addition, they are quick in reacting to environmental changes and tracking the global optimum as they usually are capable of covering multiple promising regions.

Different clustering methods used in the literature to form subpopulations were reviewed in § P_A-III-E2a. Except for the

search space division approaches, other clustering methods are commonly used in designing DOAs. Each class of clustering methods has its own strength and weak points. For example, the index-based clustering approaches are easy to implement and without additional computational load. However, they do not consider the attributes of individuals, such as their positions. On the other hand, the clustering methods that consider such attributes are computationally heavy as they need to calculate several Euclidean distances. These methods need some input parameters, such as the number of subpopulations, maximum subpopulation size, or subpopulation radius, which affect their flexibility and performance. For example, the clustering methods that use a radius to identify clusters (e.g., [81]) are less flexible as the value of radius needs to be tuned for different problems. Similarly, clustering methods that need the number of clusters as their input are less flexible since the number of subpopulations is fixed and needs to be tuned for different problems. Up to now, although many clustering methods have been used in the literature to form subpopulations, there has been little study that investigates, analyzes, and compares the effect of different clustering methods on the performance of DOAs.

Among multi-population DOAs, those with varying population size and the number of subpopulations are the most efficient [53], [89]. In these DOAs, such numbers are usually adapted to the number of discovered promising regions. However, these DOAs are challenged in DOPs with large number of promising regions since generating large numbers of subpopulations and individuals slowdowns the optimization process.

Computational resource allocation methods are one of the most important components of multi-population DOAs. Their role becomes even more important under specific circumstances, such as when there is a large number of subpopulations or when the environmental change frequency is high. Despite the importance of these methods, little attention has been given in designing systematic methods which consider several factors including problem characteristics (such as the number of discovered promising regions and change frequency), the role of subpopulations, and the task achievements of subpopulations.

F. Discussion on the proposed taxonomy

In this section, we have reviewed the components used for designing DOAs in the literature. We defined these components according to the classification provided in the proposed taxonomy shown in Figure $\mathrm{P_A}$-3. In this taxonomy, we have covered some important classes of components called convergence detection and computational resource allocation that have not been considered in previous taxonomies. In addition, for the first time, we have classified the components used for managing subpopulations in multi-population DOAs. We also have improved the classification of diversity control components by considering some of their characteristics, such as their area of effect, which was never considered before. Components of each class have some specific strengths and weaknesses, hence they can show different effectiveness for different DOPs. However, the compatibility of different components with each other, and their effectiveness in overcoming challenges of different types of DOPs have rarely been studied in the literature.

We started by reviewing the convergence detection components which are mostly used by multi-population DOAs. We have classified these components into two main classes of fitness and spatial size monitoring methods. Both classes may generally detect false convergence; however, the spatial size monitoring methods are more accurate and reliable. Note that spatial size monitoring methods are computationally complex as they usually calculate many distances. Usually, some other components and procedures of DOAs are triggered once a subpopulation has converged, such as creating a new subpopulation, increasing diversity, and archiving a position in an explicit archive.

We then studied the change detection components by classifying them into reevaluation and fitness monitoring based methods. According to our readings, fitness monitoring based methods are more error-prone in detecting environmental changes in comparison to reevaluation based methods. However, depending on the considered DOPs' characteristics, reevaluation based methods may need a larger number of detectors,, which results in consuming a lot of computational resources (i.e., fitness evaluations). For example, in a DOP with a large search space (e.g., high dimensional DOPs and/or the ones with vast search domains) and local environmental changes (i.e., some parts of the landscape involve with changes), we will need a large number of detectors across the search space to accurately detect changes.

Many real-world problems involve noise [4]. In noisy DOPs, we cannot directly use reevaluation based change detection components, because the noise changes the fitness values of detectors over time. To adapt these components to noisy environments, we can compare the intensity of the detectors' fitness fluctuations with the estimated amount of disturbance caused by noise. Therefore, these components become capable of detecting environmental changes which are more severe than noise. However, if the noise intensity differs over the landscape or if it changes over time, reevaluation based methods will become significantly deficient. In such circumstances, it might be a generally better idea to use change-independent DOAs whose components are not triggered by the environmental changes [32], [53], [135]. Finally, it is worth mentioning that using change detection components is unnecessary in many real-world DOPs since the DOAs get informed about the environmental changes.

Another important class of components studied in this survey is explicit archiving. These components are suitable for reappearing/cyclic DOPs where the optimum returns to a previous position or previous environments reappear periodically. Using explicit archiving components, DOAs will be capable of archiving historical information and reusing them for accelerating the tracking process. In addition to reappearing/cyclic DOPs, explicit archiving components can be used for DOPs where the optimum trajectory is predictable. In such DOPs, stored solutions in the archive(s) can be used as the training dataset for prediction methods to estimate the

next position of the optimum, the direction of changes, or the expected area that will contain the optimum [35], [82]. Note that explicit archiving components are not useful for solving unpredictable DOPs [6].

This survey has further classified the diversity control components based on their area of effect which can be either a limited area (i.e., local diversity) or across the search space (i.e., global diversity). On the one hand, the local diversity control components are suitable for circumstances where the optimum relocates slightly after environmental changes. In such circumstances, increasing/maintaining the local diversity of a (sub-)population according to the expected optimum's relocation length, ameliorates the tracking performance. In other words, increasing/maintaining local diversity improves the exploitation capability of a (sub-)population after environmental changes. On the other hand, if the optimum position can appear/relocate in/to far away or uncover regions of the search space, DOAs need to increase/maintain their global diversity to improve their exploration capability. Therefore, to perform efficient exploitation and exploration after environmental changes, both local and global diversity control components should be used in the structure of the DOAs.

As reviewed in § $\mathrm{P_A}$-III-D, some global and local diversity increasing components are triggered after environmental changes. These components are effective for tackling DOPs with detectable/visible environmental changes [1]. However, they cannot be used in DOPs with undetectable environmental changes [32], [135]. These diversity control components are also not suitable for DOPs with very high change frequencies, such as progressive and drifting DOPs that almost continuously change over time (see § $\mathrm{P_A}$-II). In such DOPs, the frequent increase of diversity prevents the algorithm from converging.

We then have reviewed the population division and management components used to create and manage subpopulations in multi-population DOAs. Population division and management components provide a better framework to apply diversity control rather than the single-population methods. Using diversity control components, multi-population DOAs are usually capable of covering multiple moving promising regions. According to our readings, most developed DOAs, especially in the last decade, apply multi-population approaches. It should be mentioned that the popularity of multi-population methods can be related to the popularity of moving peaks based DOP benchmarks (see § $\mathrm{P_B}$-II). In fact, our investigations indicate that the majority of the multi-population DOAs are somehow tailored to tackle these types of DOPs. As the effectiveness of most multi-population DOAs has not been investigated in solving real-world DOPs (see § $\mathrm{P_B}$-V), their performance in realistic applications is not entirely clear.

Finally, we have reviewed computational resource allocation components that are mostly used to manage computational resource consumption of subpopulations in multi-population DOAs. Any multi-population DOA uses at least the simplest type of this component: Round Robin. For DOPs with sufficient computational resources in each environment (e.g., the ones with low change frequencies), using Round Robin seems enough. Such DOPs have an adequate amount of computational resources in each environment, hence subpopulations can fulfill their tasks (such as tracking optimum) in each environment by using simple Round Robin. On the other hand, in DOPs where there is a limited amount of available computational resources for each environment (e.g., DOPs with higher change frequencies or larger environments), using more advanced computational resource allocation components can considerably improve the performance of DOAs [11].

In the following of this subsection, we will discuss how to generally develop a DOA by assembling the components illustrated in Figure $\mathrm{P_A}$-3. Note that the following discussion is derived from our investigations on the most popular and efficient DOAs in the field.

To develop a DOA, first, we need to decide on the population structure and choose the population division and management components. Currently, multi-population DOAs are the most popular approaches in the field. As discussed in § $\mathrm{P_A}$-III-E3, multi-population components with adaptive number of subpopulations and overall population size are the most flexible and efficient methods [53], [89].

We then must choose a resource allocation component (see § $\mathrm{P_A}$-III-E2e). In the simplest form, the classic *Round Robin* method is used. In cases where the available computational resources in each environment is limited, e.g., when the number of subpopulations is large (due to a large number of promising regions/peaks) and/or the change frequency is high, other computational resource allocation methods should be used. For example, in [42], the DOA uses the *deactivation* component for converged subpopulations, and the Round Robin method for active subpopulations. This DOA also uses the *local search operator* component around the best found position. Thus, it benefits from three computational resource allocation components simultaneously.

All DOAs need diversity control components to improve/maintain their exploration and exploitation capabilities after environmental changes. Unless a DOP's environmental changes are undetectable or its environment continuously changes over time, the components that increase the local diversity after environmental changes (see § $\mathrm{P_A}$-III-D2b) are good options. For the global diversity control component of a multi-population DOA, we should use a component that randomizes redundant subpopulations/individuals (see § $\mathrm{P_A}$-III-D1a). Usually, using components that randomize the converged subpopulations and redundant subpopulations/individuals can address global diversity loss (i.e., improve/maintain exploration capability) adequately [42], [89].

In case we use any diversity control and population division and management components that are triggered after a subpopulation convergence, we should also add a convergence detection component. In addition to the components that are triggered after environmental changes, DOAs need to reevaluate their candidate solutions after environmental changes to address the outdated memory issue. Consequently, a change detection component is needed unless the DOA is informed about the environmental changes. In cases of DOPs with undetectable environmental changes, we cannot use any component that is triggered by environmental changes. To address the outdated memory issue in DOAs which are designed for

such DOPs, candidate solutions are frequently reevaluated over time [32]. Another type of DOPs which need us to add some additional components to the DOA, are predictable DOPs. For improving the performance of DOAs in solving such DOPs, we usually use explicit archiving components to benefit from the historical information. Finally, a static optimization algorithm, such as PSO or DE, is embedded into the DOA framework as the *optimization component* (see § $\mathrm{P_B}$-IV).

Choosing a proper set of components to build a DOA for tackling a specific DOP can be done by trying different combinations of components and analyzing their performance using different performance indicators (see § $\mathrm{P_B}$-III-A). However, one main issue is that the characteristics of DOPs such as change frequency, change severity, largeness (based on the domain of variables or number of dimensions), predictability, and change detectability may change over time in some heterogeneous DOPs. Therefore, selecting a proper set of components for all environments of such DOPs may be impossible. One way to address this challenge is to use hyperheuristic approaches [19], [136], [137], where the algorithm tries to choose the most proper set of components according to the current performance of the DOA.

$\mathrm{P_A}$-IV. Conclusion

In this two-part survey, we provide a review of the research done in the field of single-objective unconstrained continuous dynamic optimization problems (DOPs) in the last two decades. Our main aim is to help researchers to gain an overall perspective of the current status of the field. In the first part of this survey, we studied dynamic optimization algorithms (DOAs). To tackle a DOP, DOAs should continually find and track the changing optimum while handling the specific challenges posed by the problem. An efficient DOA is usually a complex algorithm which is constructed by assembling several components. The complexity of these algorithms makes them hard-to-understand. To improve the understandability of the structure of DOAs, we have proposed a comprehensive taxonomy which identifies and classifies the components of DOAs. Based on the proposed taxonomy, we then provided an in-depth technical description of each class of components.

In Part B of this survey, we review the DOP benchmarks, the performance indicators and plots used for analyzing and comparing the performance of DOAs, the static optimization algorithms used as the optimization components in DOAs, and applications of DOAs for the real-world problems. Part B is concluded by a discussion on the potential future directions in the field.

References

[1] T. T. Nguyen, S. Yang, and J. Branke, "Evolutionary dynamic optimization: A survey of the state of the art," *Swarm Evol. Comput.*, vol. 6, pp. 1 – 24, 2012.

[2] T. T. Nguyen and X. Yao, "Continuous dynamic constrained optimization—the challenges," *IEEE Trans. Evol. Comput.*, vol. 16, no. 6, pp. 769–786, 2012.

[3] M. Mavrovouniotis, C. Li, and S. Yang, "A survey of swarm intelligence for dynamic optimization: Algorithms and applications," *Swarm Evol. Comput.*, vol. 33, pp. 1 – 17, 2017.

[4] Y. Jin and J. Branke, "Evolutionary optimization in uncertain environments-a survey," *IEEE Trans. Evol. Comput.*, vol. 9, no. 3, pp. 303–317, 2005.

[5] C. Cruz, J. R. González, and D. A. Pelta, "Optimization in dynamic environments: a survey on problems, methods and measures," *Soft Comput.*, vol. 15, no. 7, pp. 1427–1448, 2011.

[6] J. Branke, *Evolutionary optimization in dynamic environments*. Springer Science & Business Media, 2012, vol. 3.

[7] S. Yang and X. Yao, Eds., *Evolutionary Computation for Dynamic Optimization Problems*. Springer-Verlag Berlin Heidelberg, 2013, vol. 490.

[8] E. Alba, A. Nakib, and P. Siarry, Eds., *Metaheuristics for dynamic optimization*. Springer-Verlag Berlin Heidelberg, 2013.

[9] R. W. Morrison, *Designing evolutionary algorithms for dynamic environments*. Springer Science & Business Media, 2013.

[10] T. T. Nguyen, "Continuous dynamic optimisation using evolutionary algorithms," Ph.D. dissertation, University of Birmingham, 2011.

[11] D. Yazdani, "Particle swarm optimization for dynamically changing environments with particular focus on scalability and switching cost," Ph.D. dissertation, Liverpool John Moores University, Liverpool, UK, 2018.

[12] S. Yang, Y. Jiang, and T. T. Nguyen, "Metaheuristics for dynamic combinatorial optimization problems," *IMA Journal of Management Mathematics*, vol. 24, no. 4, pp. 451–480, 2013.

[13] T. Blackwell and J. Branke, "Multiswarms, exclusion, and anti-convergence in dynamic environments," *IEEE Trans. Evol. Comput.*, vol. 10, no. 4, pp. 459–472, 2006.

[14] C. Raquel and X. Yao, "Dynamic multi-objective optimization: a survey of the state-of-the-art," in *Evolutionary computation for dynamic optimization problems*. Springer, 2013, pp. 85–106.

[15] R. Azzouz, S. Bechikh, and L. B. Said, "Dynamic multi-objective optimization using evolutionary algorithms: a survey," in *Recent advances in evolutionary multi-objective optimization*. Springer, 2017, pp. 31–70.

[16] R. Azzouz, "Evolutionary approaches for dynamic multi-objective optimization," Ph.D. dissertation, Computer Science Department, University of Tunis, 2017.

[17] C. Bu, W. Luo, and L. Yue, "Continuous dynamic constrained optimization with ensemble of locating and tracking feasible regions strategies," *IEEE Trans. Evol. Comput.*, vol. 21, no. 1, pp. 14–33, 2016.

[18] P. Novoa-Hernández, C. C. Corona, and D. A. Pelta, "Self-adaptation in dynamic environments– a survey and open issues," *International Journal of Bio-Inspired Computation*, vol. 8, no. 1, pp. 1–13, 2016.

[19] T. Macias-Escobar, B. Dorronsoro, L. Cruz-Reyes, N. Rangel-Valdez, and C. Gómez-Santillán, "A survey of hyper-heuristics for dynamic optimization problems," in *Intuitionistic and Type-2 Fuzzy Logic Enhancements in Neural and Optimization Algorithms: Theory and Applications*. Springer, 2020, pp. 463–477.

[20] D. Yazdani, T. T. Nguyen, and J. Branke, "Robust optimization over time by learning problem space characteristics," *IEEE Trans. Evol. Comput.*, vol. 23, no. 1, pp. 143–155, 2018.

[21] D. Yazdani, R. Cheng, D. Yazdani, J. Branke, Y. Jin, and X. Yao, "A survey of evolutionary continuous dynamic optimization over two decades – part B," *IEEE Transactions on Evolutionary Computation*, 2021.

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

[23] D. Yazdani, M. N. Omidvar, R. Cheng, J. Branke, T. T. Nguyen, and X. Yao, "Benchmarking continuous dynamic optimization: Survey and generalized test suite," *IEEE Trans. Cybern.*, pp. 1 – 14, 2020.

[24] J. Branke and H. Schmeck, "Designing evolutionary algorithms for dynamic optimization problems," in *Advances in Evolutionary Computing*, A. Ghosh and S. Tsutsui, Eds. Springer Natural Computing Series, 2003, pp. 239–262.

[25] J. G. O. L. Duhain, "Particle swarm optimisation in dynamically changing environments - an empirical study," Master's thesis, University of Pretoria, Pretoria, South Africa, 2012.

[26] R. C. Eberhart and Y. Shi, "Tracking and optimizing dynamic systems with particle swarms," in *IEEE Congr. Evol. Comput.*, vol. 1. IEEE, 2001, pp. 94–100.

[27] K. De Jong, "Evolving in a changing world," in *International Symposium on Methodologies for Intelligent Systems*. Springer, 1999, pp. 512–519.

[28] J. G. O. L. Duhain and A. P. Engelbrecht, "Towards a more complete classification system for dynamically changing environments," in *IEEE Congr. Evol. Comput.* IEEE, 2012, pp. 1–8.

[29] K. Weicker, "Performance measures for dynamic environments," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2002, pp. 64–73.

[30] B. Nasiri, M. Meybodi, and M. Ebadzadeh, "History-driven particle swarm optimization in dynamic and uncertain environments," *Neurocomputing*, vol. 172, pp. 356 – 370, 2016.

[31] P. Angeline, "Tracking extrema in dynamic environments," in *Evolutionary Programming VI*, P. Angeline et al., Ed. Springer Lecture Notes in Computer Science, 1997, vol. 1213, pp. 335–345.

[32] C. Li and S. Yang, "A general framework of multipopulation methods with clustering in undetectable dynamic environments," *IEEE Trans. Evol. Comput.*, vol. 16, no. 4, pp. 556–577, 2012.

[33] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-International Conference on Neural Networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[34] P. Novoa-Hernández, C. C. Corona, and D. A. Pelta, "Self-adaptive, multipopulation differential evolution in dynamic environments," *Soft Comput.*, vol. 17, no. 10, pp. 1861–1881, 2013.

[35] X. Liu, Z. Zhan, and J. Zhang, "Neural network for change direction prediction in dynamic optimization," *IEEE Access*, vol. 6, pp. 72 649–72 662, 2018.

[36] M. C. du Plessis and A. P. Engelbrecht, "Differential evolution for dynamic environments with unknown numbers of optima," *Journal of Global Optimization*, vol. 55, no. 1, pp. 73–99, 2013.

[37] W. Zhang, W. Zhang, G. G. Yen, and H. Jing, "A cluster-based clonal selection algorithm for optimization in dynamic environment," *Swarm Evol. Comput.*, vol. 50, p. 100454, 2019.

[38] W. Luo, R. Yi, B. Yang, and P. Xu, "Surrogate-assisted evolutionary framework for data-driven dynamic optimization," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 3, no. 2, pp. 137–150, 2019.

[39] J. K. Kordestani, H. A. Firouzjaee, and M. Reza Meybodi, "An adaptive bi-flight cuckoo search with variable nests for continuous dynamic optimization problems," *Applied Intelligence*, vol. 48, no. 1, pp. 97–117, 2017.

[40] J. K. Kordestani, A. Rezvanian, and M. R. Meybodi, "Cdepso: a bi-population hybrid approach for dynamic optimization problems," *Applied Intelligence*, vol. 40, no. 4, pp. 682–694, 2014.

[41] W. Luo, J. Sun, C. Bu, and H. Liang, "Species-based particle swarm optimizer enhanced by memory for dynamic optimization," *Appl. Soft Comput.*, vol. 47, pp. 130 – 140, 2016.

[42] D. Yazdani, B. Nasiri, A. Sepas-Moghaddam, and M. R. Meybodi, "A novel multi-swarm algorithm for optimization in dynamic environments based on particle swarm optimization," *Appl. Soft Comput.*, vol. 13, no. 4, pp. 2144–2158, 2013.

[43] D. Bose, S. Biswas, S. Kundu, and S. Das, "A strategy pool adaptive artificial bee colony algorithm for dynamic environment through multi-population approach," in *Swarm, Evolutionary, and Memetic Computing*, B. K. Panigrahi et al., Ed. Springer Berlin Heidelberg, 2012, pp. 611–619.

[44] T. T. Nguyen, I. Jenkinson, and Z. Yang, "Solving dynamic optimisation problems by combining evolutionary algorithms with kd-tree," in *Conference on Soft Computing and Pattern Recognition*. IEEE, 2013, pp. 247–252.

[45] X. Li, J. Branke, and T. Blackwell, "Particle swarm with speciation and adaptation in a dynamic environment," in *Genet. Evol. Comput. Conf.* ACM, 2006, pp. 51–58.

[46] M. R. Bonyadi and Z. Michalewicz, "Particle swarm optimization for single objective continuous space problems: a review," pp. 1–54, 2017.

[47] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, 2010.

[48] K. Trojanowski, "Properties of quantum particles in multi-swarms for dynamic optimization," *Fundamenta Informaticae*, vol. 95, no. 2-3, pp. 349–380, 2009.

[49] M. Kamosi, A. B. Hashemi, and M. R. Meybodi, "A hibernating multi-swarm optimization algorithm for dynamic environments," in *Nature and Biologically Inspired Computing*. IEEE, 2010, pp. 363–369.

[50] A. Sepas-Moghaddam, A. Arabshahi, D. Yazdani, and M. M. Dehshibi, "A novel hybrid algorithm for optimization in multimodal dynamic environments," in *Int. Conf. Hybrid Intell. Syst.* IEEE, 2012, pp. 143–148.

[51] D. Yazdani, M. N. Omidvar, J. Branke, T. T. Nguyen, and X. Yao, "Scaling up dynamic optimization problems: A divide-and-conquer approach," *IEEE Trans. Evol. Comput.*, vol. 24, no. 1, pp. 1–15, 2019.

[52] C. Li and S. Yang, "A clustering particle swarm optimizer for dynamic optimization," in *IEEE Congr. Evol. Comput.* IEEE, 2009, pp. 439–446.

[53] C. Li, T. T. Nguyen, M. Yang, M. Mavrovouniotis, and S. Yang, "An adaptive multipopulation framework for locating and tracking multiple optima," *IEEE Trans. Evol. Comput.*, vol. 20, no. 4, pp. 590–605, 2016.

[54] R. Liaw and C. Ting, "Incorporating fitness inheritance and k-nearest neighbors for evolutionary dynamic optimization," in *IEEE Congr. Evol. Comput.* IEEE, July 2018, pp. 1–8.

[55] X. Luo, Z. Wang, R. Guan, Z. Zhan, and Y. Gao, "A distributed multiple populations framework for evolutionary algorithm in solving dynamic optimization problems," *IEEE Access*, vol. 7, pp. 44 372–44 390, 2019.

[56] W. Luo, J. Sun, C. Bu, and R. Yi, "Identifying species for particle swarm optimization under dynamic environments," in *Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2018, pp. 1921–1928.

[57] D. Yazdani, M. R. Akbarzadeh-Totonchi, B. Nasiri, and M. R. Meybodi, "A new artificial fish swarm algorithm for dynamic optimization problems," in *IEEE Congr. Evol. Comput.* IEEE, 2012, pp. 1–8.

[58] B. Nasiri and M. R. Meybodi, "History-driven firefly algorithm for optimisation in dynamic and uncertain environments," *International Journal of Bio-Inspired Computation*, vol. 8, no. 5, pp. 326–339, 2016.

[59] M. Mavrovouniotis, F. Neri, and S. Yang, "An adaptive local search algorithm for real-valued dynamic optimization," in *IEEE Congr. Evol. Comput.* IEEE, 2015, pp. 1388–1395.

[60] A. Carlisle and G. Dozier, "Adapting particle swarm optimization to dynamic environments," in *International conference on Artificial Intelligence*, 2000, pp. 429–434.

[61] X. Hu and R. C. Eberhart, "Adaptive particle swarm optimization: detection and response to dynamic systems," in *IEEE Congr. Evol. Comput.*, vol. 2. IEEE, 2002, pp. 1666–1670.

[62] A. Carlisle and G. Dozler, "Tracking changing extrema with adaptive particle swarm optimizer," in *World Automation Congress*, vol. 13. IEEE, 2002, pp. 265–270.

[63] T. Blackwell and J. Branke, "Multi-swarm optimization in dynamic environments," in *Applications of Evolutionary Computing*, G. R. Raidl et al., Ed. Lecture Notes in Computer Science, 2004, vol. 3005, pp. 489–500.

[64] W. Du and B. Li, "Multi-strategy ensemble particle swarm optimization for dynamic optimization," *Inf. Sci.*, vol. 178, no. 15, pp. 3096 – 3109, 2008.

[65] A. Meier and O. Kramer, "Prediction with recurrent neural networks in evolutionary dynamic optimization," in *Applications of Evolutionary Computation*, K. Sim and P. Kaufmann, Eds. Springer International Publishing, 2018, pp. 848–863.

[66] B. Niu, Q. Liu, and J. Wang, "Bacterial foraging optimization with memory and clone schemes for dynamic environments," in *Advances in Swarm Intelligence*, Y. Tan et al., Ed. Springer International Publishing, 2019, pp. 352–360.

[67] H. Richter, "Detecting change in dynamic fitness landscapes," in *IEEE Congr. Evol. Comput.* IEEE, 2009, pp. 1613–1620.

[68] C. Li, S. Yang, and M. Yang, "An adaptive multi-swarm optimizer for dynamic optimization problems," *Evol. Comput.*, vol. 22, no. 4, pp. 559–594, 2014.

[69] H. Wang, D. Wang, and S. Yang, "Triggered memory-based swarm optimization in dynamic environments," in *Applications of Evolutionary Computing*, M. Giacobini, Ed. Springer Berlin Heidelberg, 2007, pp. 637–646.

[70] J. Brest, A. Zamuda, B. Boskovic, M. S. Maucec, and V. Zumer, "Dynamic optimization using self-adaptive differential evolution," in *IEEE Congr. Evol. Comput.* IEEE, 2009, pp. 415–422.

[71] U. Halder, D. Maity, P. Dasgupta, and S. Das, "Self-adaptive cluster-based differential evolution with an external archive for dynamic optimization problems," in *Swarm, Evolutionary, and Memetic Computing*, B. K. Panigrahi et al., Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 19–26.

[72] H. Wang, S. Yang, W. Ip, and D. Wang, "A memetic particle swarm optimisation algorithm for dynamic multi-modal optimisation problems," *International Journal of Systems Science*, vol. 43, no. 7, pp. 1268–1283, 2012.

[73] R. Mukherjee, G. R. Patra, R. Kundu, and S. Das, "Cluster-based differential evolution with crowding archive for niching in dynamic environments," *Inf. Sci.*, vol. 267, pp. 58 – 82, 2014.

[74] W. Wu, D. Xie, and L. Liu, "Heterogeneous differential evolution with memory enhanced brownian and quantum individuals for dynamic optimization problems," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 32, no. 02, p. 1859003, 2018.

[75] Y. Bravo, G. Luque, and E. Alba, "Global memory schemes for dynamic optimization," *Natural Computing*, vol. 15, no. 2, pp. 319–333, 2015.

[76] Y. G. Woldesenbet and G. G. Yen, "Dynamic evolutionary algorithm with variable relocation," *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 500–513, 2009.

[77] A. M. Turky and S. Abdullah, "A multi-population harmony search algorithm with external archive for dynamic optimization problems," *Inf. Sci.*, vol. 272, pp. 84 – 95, 2014.

[78] T. Zhu, W. Luo, and L. Yue, "Combining multipopulation evolutionary algorithms with memory for dynamic optimization problems," in *IEEE Congr. Evol. Comput.* IEEE, 2014, pp. 2047–2054.

[79] R. Vafashoar and M. R. Meybodi, "A multi-population differential evolution algorithm based on cellular learning automata and evolutionary context information for optimization in dynamic environments," *Appl. Soft Comput.*, p. 106009, 2019.

[80] H. Nakano, M. Kojima, and A. Miyauchi, "An artificial bee colony algorithm with a memory scheme for dynamic optimization problems," in *IEEE Congr. Evol. Comput.* IEEE, 2015, pp. 2657–2663.

[81] D. Parrott and X. Li, "Locating and tracking multiple dynamic optima by a particle swarm model using speciation," *IEEE Trans. Evol. Comput.*, vol. 10, no. 4, pp. 440–458, 2006.

[82] X. Liu, Z. Zhan, T. Gu, S. Kwong, Z. Lu, H. B. Duh, and J. Zhang, "Neural network-based information transfer for dynamic optimization," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2019.

[83] W. Zhang, M. Zhang, W. Zhang, Y. Meng, and H. Wu, "Innate-adaptive response and memory based artificial immune system for dynamic optimization," *International Journal of Performability Engineering*, vol. 14, no. 9, p. 2048, 2018.

[84] X.-F. Liu, Y.-R. Zhou, X. Yu, and Y. Lin, "Dual-archive-based particle swarm optimization for dynamic optimization," *Appl. Soft Comput.*, p. 105876, 2019.

[85] J. Branke, T. Kaussler, C. Schmidt, and H. Schmeck, "A multi-population approach to dynamic optimization problems," in *Evolutionary Design and Manufacture*. Springer, 2000, pp. 299–307.

[86] R. Mendes and A. S. Mohais, "DynDE: a differential evolution for dynamic optimization problems," in *IEEE Congr. Evol. Comput.*, vol. 3. IEEE, 2005, pp. 2808–2815.

[87] T. Blackwell, *Particle Swarm Optimization in Dynamic Environments*. Springer Berlin Heidelberg, 2007, pp. 29–49.

[88] M. C. du Plessis and A. P. Engelbrecht, "Improved differential evolution for dynamic optimization problems," in *IEEE Congr. Evol. Comput.* IEEE, 2008, pp. 229–234.

[89] T. Blackwell, J. Branke, and X. Li, "Particle swarms for dynamic optimization problems," in *Swarm Intelligence: Introduction and Applications*, C. Blum and D. Merkle, Eds. Springer Lecture Notes in Computer Science, 2008, pp. 193–217.

[90] I. Rezazadeh, M. R. Meybodi, and A. Naebi, "Adaptive particle swarm optimization algorithm in dynamic environments," in *Computational Intelligence, Modelling and Simulation*. IEEE, 2011, pp. 74–79.

[91] J. C. Bezdek, R. Ehrlich, and W. Full, "Fcm: The fuzzy c-means clustering algorithm," *Computers & Geosciences*, vol. 10, no. 2-3, pp. 191–203, 1984.

[92] M. C. du Plessis and A. P. Engelbrecht, "Using competitive population evaluation in a differential evolution algorithm for dynamic environments," *European Journal of Operational Research*, vol. 218, no. 1, pp. 7–20, 2012.

[93] L. Xiao and X. Zuo, "Multi-depso: A de and pso based hybrid algorithm in dynamic environments," in *IEEE Congr. Evol. Comput.* IEEE, 2012, pp. 1–7.

[94] J. K. Kordestani, M. R. Meybodi, and A. M. Rahmani, "A note on the exclusion operator in multi-swarm pso algorithms for dynamic environments," *Connection Science*, pp. 1–25, 2019.

[95] D. Parrott and Xiaodong Li, "A particle swarm model for tracking multiple peaks in a dynamic environment using speciation," in *IEEE Congr. Evol. Comput.*, vol. 1. IEEE, 2004, pp. 98–103.

[96] S. Bird and X. Li, "Using regression to improve local convergence," in *IEEE Congr. Evol. Comput.* IEEE, 2007, pp. 592–599.

[97] W. Luo, X. Lin, T. Zhu, and P. Xu, "A clonal selection algorithm for dynamic multimodal function optimization," *Swarm Evol. Comput.*, vol. 50, p. 100459, 2019.

[98] A. B. Hashemi and M. R. Meybodi, "Cellular pso: A pso for dynamic environments," in *Advances in Computation and Intelligence*, Z. Cai et al., Ed. Springer Berlin Heidelberg, 2009, pp. 422–433.

[99] A. B. Hashemi and M. R. Meybodi, "A multi-role cellular pso for dynamic environments," in *International CSI Computer Conference*. IEEE, 2009, pp. 412–417.

[100] A. Sharifi, V. Noroozi, M. Bashiri, A. B. Hashemi, and M. R. Meybodi, "Two phased cellular pso: A new collaborative cellular algorithm for optimization in dynamic environments," in *IEEE Congr. Evol. Comput.* IEEE, 2012, pp. 1–8.

[101] J. Kari, "Theory of cellular automata: A survey," *Theoretical Computer Science*, vol. 334, no. 1, pp. 3 – 33, 2005.

[102] K. Trojanowski and Z. Michalewicz, "Searching for optima in non-stationary environments," in *IEEE Congr. Evol. Comput.*, vol. 3, 1999, pp. 1843–1850.

[103] J. Karimi, H. Nobahari, and S. Pourtakdoust, "A new hybrid approach for dynamic continuous optimization problems," *Appl. Soft Comput.*, vol. 12, no. 3, pp. 1158 – 1167, 2012.

[104] P. Novoa-Hernández, D. A. Pelta, and C. C. Corona, *Improvement Strategies for Multi-swarm PSO in Dynamic Environments*. Springer Berlin Heidelberg, 2010, pp. 371–383.

[105] S. K. Nseef, S. Abdullah, A. Turky, and G. Kendall, "An adaptive multi-population artificial bee colony algorithm for dynamic optimisation problems," *Knowledge-Based Systems*, vol. 104, pp. 14 – 23, 2016.

[106] I. Rezazadeh, M. R. Meybodi, and A. Naebi, "Particle swarm optimization algorithm in dynamic environments: Adapting inertia weight and clustering particles," in *European Symposium on Computer Modeling and Simulation*. IEEE, 2011, pp. 76–82.

[107] X. Zuo and L. Xiao, "A de and pso based hybrid algorithm for dynamic optimization problems," *Soft Comput.*, vol. 18, no. 7, pp. 1405–1424, 2013.

[108] F. B. Ozsoydan and A. Baykasoglu, "A multi-population firefly algorithm for dynamic optimization problems," in *Conference on Evolving and Adaptive Intelligent Systems*. IEEE, 2015, pp. 1–7.

[109] L. Shen, L. Xu, R. Wei, and L. Cao, "Multi-swarm optimization with chaotic mapping for dynamic optimization problems," in *2015 8th International Symposium on Computational Intelligence and Design (ISCID)*, vol. 2. IEEE, 2015, pp. 132–137.

[110] R. I. Lung and D. Dumitrescu, "A collaborative model for tracking optima in dynamic environments," in *IEEE Congr. Evol. Comput.* IEEE, 2007, pp. 564–567.

[111] S. Biswas, D. Bose, and S. Kundu, "A clustering particle based artificial bee colony algorithm for dynamic environment," in *Swarm, Evolutionary, and Memetic Computing*, B. K. Panigrahi et al., Ed. Springer Berlin Heidelberg, 2012, pp. 151–159.

[112] P. Novoa, D. A. Pelta, C. Cruz, and I. G. del Amo, "Controlling particle trajectories in a multi-swarm approach for dynamic optimization problems," in *Methods and Models in Artificial and Natural Computation. A Homage to Professor Mira's Scientific Legacy*, J. Mira et al., Ed. Springer Berlin Heidelberg, 2009, pp. 285–294.

[113] I. G. del Amo, D. A. Pelta, and J. R. González, "Using heuristic rules to enhance a multiswarm pso for dynamic environments," in *IEEE Congr. Evol. Comput.* IEEE, 2010, pp. 1–8.

[114] X. Zheng and H. Liu, "A different topology multi-swarm pso in dynamic environment," in *International Symposium on IT in Medicine Education*, vol. 1. IEEE, 2009, pp. 790–795.

[115] B. Nasiri and M. R. Meybodi, "Speciation based firefly algorithm for optimization in dynamic environments," *Int. J. Artif. Intell.*, vol. 8, no. S12, pp. 118–132, 2012.

[116] S. Abdullah, S. K. Nseef, and A. Turky, "An interleaved artificial bee colony algorithm for dynamic optimisation problems," *Connection Science*, vol. 30, no. 3, pp. 272–284, 2017.

[117] A. Turky, S. Abdullah, and A. Dawod, "A dual-population multi operators harmony search algorithm for dynamic optimization problems," *Computers & Industrial Engineering*, vol. 117, pp. 19 – 28, 2018.

[118] R. Thomsen, "Multimodal optimization using crowding-based differential evolution," in *IEEE Congr. Evol. Comput.*, vol. 2. IEEE, 2004, pp. 1382–1389.

[119] R. I. Lung and D. Dumitrescu, "Evolutionary swarm cooperative optimization in dynamic environments," *Natural Computing*, vol. 9, no. 1, pp. 83–94, 2010.

[120] S. Das, A. Mandal, and R. Mukherjee, "An adaptive differential evolution algorithm for global optimization in dynamic environments," *IEEE Trans. Cybern.*, vol. 44, no. 6, pp. 966–978, 2014.

[121] H. G. Cobb and J. J. Grefenstette, "Genetic algorithms for tracking changing environments," in *International Conference on Genetic Algorithms*. Morgan Kaufmann Publishers Inc., 1993, pp. 523–530.

[122] C. Hu, X. Wu, Y. Wang, and F. Xie, "Multi-swarm particle swarm optimizer with cauchy mutation for dynamic optimization problems," in *Advances in Computation and Intelligence*, Z. Cai et al., Ed. Springer Berlin Heidelberg, 2009, pp. 443–453.

[123] P. Novoa-Hernández, C. C. Corona, and D. A. Pelta, "Efficient multi-swarm pso algorithms for dynamic environments," *Memetic Computing*, vol. 3, no. 3, pp. 163–174, Aug 2011.

[124] J. K. Kordestani, A. Rezvanian, and M. R. Meybodi, "An efficient oscillating inertia weight of particle swarm optimisation for tracking optima in dynamic environments," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 28, no. 1-2, pp. 137–149, 2015.

[125] C. Li, T. T. Nguyen, M. Yang, S. Yang, and S. Zeng, "Multi-population methods in unconstrained continuous dynamic environments: The challenges," *Inf. Sci.*, vol. 296, pp. 95 – 118, 2015.

[126] V. Noroozi, A. B. Hashemi, and M. R. Meybodi, "Cellularde: A cellular based differential evolution for dynamic optimization problems," in *Adaptive and Natural Computing Algorithms*, A. Dobnikar et al., Ed. Springer Berlin Heidelberg, 2011, pp. 340–349.

[127] S. Yang and C. Li, "A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments," *IEEE Trans. Evol. Comput.*, vol. 14, no. 6, pp. 959–974, 2010.

[128] C. D. Manning, P. Raghavan, and H. Schutze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[129] W. Luo, B. Yang, C. Bu, and X. Lin, "A hybrid particle swarm optimization for high-dimensional dynamic optimization," in *Simulated Evolution and Learning*, Y. Shi et al., Ed. Cham: Springer International Publishing, 2017, pp. 981–993.

[130] S. Kundu, D. Basu, and S. S. Chaudhuri, "Multipopulation-based differential evolution with speciation-based response to dynamic environments," in *Swarm, Evolutionary, and Memetic Computing*, B. K. Panigrahi et al., Ed. Springer International Publishing, 2013, pp. 222–235.

[131] A. Nickabadi, M. M. Ebadzadeh, and R. Safabakhsh, "A novel particle swarm optimization algorithm with adaptive inertia weight," *Appl. Soft Comput.*, vol. 11, no. 4, pp. 3658 – 3670, 2011.

[132] C. Li and S. Yang, "Fast multi-swarm optimization for dynamic optimization problems," in *International Conference on Natural Computation*, vol. 7. IEEE, 2008, pp. 624–628.

[133] D. Yazdani, B. Nasiri, A. Sepas-Moghaddam, M. Meybodi, and M. Akbarzadeh-Totonchi, "mNAFSA: a novel approach for optimization in dynamic environments with global changes," *Swarm Evol. Comput.*, vol. 18, pp. 38 – 53, 2014.

[134] J. K. Kordestani, A. E. Ranginkaman, M. R. Meybodi, and P. Novoa-Hernández, "A novel framework for improving multi-population algorithms for dynamic optimization problems: A scheduling approach," *Swarm Evol. Comput.*, vol. 44, pp. 788 – 805, 2019.

[135] C. Li, S. Yang, and M. Yang, "Maintaining diversity by clustering in dynamic environments," in *IEEE Congr. Evol. Comput.* IEEE, 2012, pp. 1–8.

[136] B. Kiraz, A. Ş. Etaner-Uyar, and E. Özcan, "Selection hyper-heuristics in dynamic environments," *Journal of the Operational Research Society*, vol. 64, no. 12, pp. 1753–1769, 2013.

[137] S. A. van der Stockt and A. P. Engelbrecht, "Analysis of selection hyper-heuristics for population-based meta-heuristics in real-valued dynamic optimization," *Swarm Evol. Comput.*, vol. 43, pp. 127–146, 2018.