Artificial Intelligence Review (2021) 54:2323–2409
https://doi.org/10.1007/s10462-020-09906-6

Performance assessment of the metaheuristic optimization algorithms: an exhaustive review

A. Hanif Halim[1] · **I. Ismail**[1] · **Swagatam Das**[2]

Published online: 6 October 2020

Abstract
The simulation-driven metaheuristic algorithms have been successful in solving numerous problems compared to their deterministic counterparts. Despite this advantage, the stochastic nature of such algorithms resulted in a spectrum of solutions by a certain number of trials that may lead to the uncertainty of quality solutions. Therefore, it is of utmost importance to use a correct tool for measuring the performance of the diverse set of metaheuristic algorithms to derive an appropriate judgment on the superiority of the algorithms and also to validate the claims raised by researchers for their specific objectives. The performance of a randomized metaheuristic algorithm can be divided into efficiency and effectiveness measures. The efficiency relates to the algorithm's speed of finding accurate solutions, convergence, and computation. On the other hand, effectiveness relates to the algorithm's capability of finding quality solutions. Both scopes are crucial for continuous and discrete problems either in single- or multi-objectives. Each problem type has different formulation and methods of measurement within the scope of efficiency and effectiveness performance. One of the most decisive verdicts for the effectiveness measure is the statistical analysis that depends on the data distribution and appropriate tool for correct judgments.

Keywords Metaheuristics · Population-based optimization · Performance metric · Performance indicator · Single and multi-objective optimization · Continuous optimization · Discrete optimization

✉ Swagatam Das
swagatam.das@isical.ac.in

A. Hanif Halim
halim_hanif@ymail.com

I. Ismail
idrisim@utp.edu.my

[1] Electrical and Electronic Engineering, Universiti Teknologi PETRONAS, Tronoh, Perak, Malaysia

[2] Electronics and Communication Science Unit, Indian Statistical Institute, 203 B. T. Road, Kolkata, India

1 Introduction

The family of stochastic search and optimization algorithms has a unique characteristic of randomness, where an algorithm executes different paths towards the best solution by the same input. This attributed the applicability of the algorithms to a wide range of optimization problems. The stochastic algorithms can be further divided into two categories: *heuristic* and *metaheuristic* algorithms. Both methods are based on the same concept, which is to find the solution by some kind of guided trial and error (Yang 2010). Heuristics are mostly problem-dependent and for various problems, different heuristics can be defined. A metaheuristic method, on the other hand, makes almost no prior assumption about the problem, can integrate several heuristics inside, and is usually described in terms of a set (commonly known as a *population*) of candidate solutions to the problem. Thus, metaheuristics can be applied to a wide range of problems that they treat as black-boxes. Some examples of heuristic algorithms are nearest neighbor search and tabu search, whereas some well-known metaheuristic algorithms are Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Cuckoo Search (CS), and Harmony Search (HS) algorithms. The metaheuristic algorithms employ certain trade-offs between randomization and local search. Randomization offers a good alternative for the algorithm to escape from local optima and explore the search on the global scale, whereas the local search mechanism exploits the search towards finer search regions zeroing on an optimum. These properties are simplified in a definition of *exploration* (diversification of the search over a large search volume) and *exploitation* (intensification of the search in a smaller volume) mechanisms that favor the metaheuristic algorithms to be suited for most of the global optimization problems. However, due to their stochastic nature, metaheuristic algorithms do not guarantee the best solution (global optima in terms of optimization problems) and every trial may result in a spectrum of approximate or near-optimal solutions. A good metaheuristic algorithm is expected to find an acceptably good quality solution within a given computational budget on a wide spectrum of problems.

The quality of each metaheuristic algorithm is measured with criteria that reflect the exploration and exploitation abilities in finding a global optimum. There are numerous measurements of the algorithmic performance for metaheuristics in diverse optimization scenarios as can be found in the literature. Some pieces of literature such as Rardin and Uzsoy (2018), Ivkovic et al. (2016) and Memari et al. (2017) divided the assessments into two main concepts which are the solution quality and computation speed. Others include robustness as another criterion (Heliodore et al. 2017; Hamacher 2014). Despite these concepts, the performance measure of a metaheuristic algorithm can be generalized into two broad categories: efficiency and effectiveness measures. Several works also generalized the same concepts (Bartz-Beielstein 2005; Yaghini et al. 2011). The efficiency is referred to as the number of resources required by the algorithm to solve any specific problem. Some algorithms are more efficient than others for a specific problem. The efficiency is generally related to time and space, such as speed and the rate of convergence towards a global optimum. As a general example, two algorithms (algorithms A and B) performed equally well in terms of the final solution quality. However, algorithm A has a faster execution time due to a lesser amount of computations involved, compared to algorithm B that has more code segments, maybe with nested looping. Thus, in the perspective of algorithm complexity, algorithm A has a better performance compared to B. On the other hand, the effectiveness is a measure that relates to the solutions returned by an algorithm. The effectiveness reveals the capability of the

stochastic algorithm within a given number of trials such as the number of optimal solutions, count of global or local optima, and the comparative statistical analysis to judge the significance of the results. In essence, the existing literature related to the quality of a metaheuristic algorithm performed various analyses to compare and validate their performance within the scope of efficiency and effectiveness. Nonetheless, numerous discussions on review papers related to performance metrics are mostly problem-specific and limited to specific criteria. Some of the review papers are briefly summarized in Table 1.

In addition to the reviews on specific problem type as in Table 1, there also exist several discussions and review papers related to the statistical evaluation on the performance analysis. Each paper specifies the analytical methodology and some of these proposed new approaches in comparison to the alternatives as shown in the following Table 2.

Despite each scope proposed by others, this paper covers and extends the review on important performance measures for each problem type that includes single- and multi-objective as well as continuous and discrete optimization problems. Furthermore, this paper also discusses the scope of applications as summarized in Table 1 and other recent problems discussed in the present literature on single-objective problems concerning the effectiveness and efficiency point of view. For the multi-objective problems, effectiveness and efficiency are described in more general terms. To ensure significant and qualitative metrics for each problem type, comprehensive surveys from reputed venues and well-cited publications are carried out, which include various journal articles, book chapters, and conferences. The reviewed publications have been selected mostly based on their effectiveness in the relevant areas.

This paper is further organized as follows: Sect. 2 discusses the scope of a single objective for continuous and discrete problems. Section 3 discusses the performance measures for multi-objective problems, followed by future challenges of performance measures in Sects. 4 and 5 with the conclusion.

2 Single-objective optimization problems

In essence, the goal of single-objective optimization is to find the best solution that corresponds to either minimum or maximum value of a single objective function. To date, numerous measures on single-objective problems for both continuous and discrete search spaces have been proposed in the literature. This section discusses the efficiency and effectiveness measures for both domains since some of the measures apply to both. Any specific metric related to only continuous or discrete problems is duly remarked in the section or discussed in individual sub-sections. The review for continuous problems is usually related to the constrained and unconstrained (bound-constrained) function optimization. In discrete domain, most of the algorithm comparisons are applied for the combinatorial problems that include such as the Assignment Problem (AP), Quadratic Assignment Problem (QAP), Travelling Salesman Problem (TSP), Travelling Thief Problem (TTP) (Bonyandi et al. 2013), Knapsack Problem (KP), Bin-Packing Problem (BP), Graph Coloring Problem, Scheduling Problem, and Orienteering Problem (OP). The performance measure for these problems may be similar in certain aspects such as the time measurement and convergence. Other metrics of relative performance may differ due to the nature of the objective function of each problem.

Table 1 Samples of performance review from other literature

References	Scope of review	Brief description
Hellwig and Beyer (2019)	Constrained optimization problem	Review on the experimental principles and performance. Quality indicators are categorized into efficiency, effectiveness, and variability of solutions
Whitley et al. (1996)	Unconstrained optimization	Reviewed and proposed methodologies for comparing the effectiveness of evolutionary algorithms on test function optimization
Nguyen et al. (2012)	Dynamic optimization problem	Classifies the performance measures into optimality-based (referred to as the algorithm's ability to find the closest optimum solution) and behavior-based (referred to as the performance of algorithm behavior that is useful to the dynamic environments)
Gunantara (2018)	Multi-objective Optimization Problems (MOPs)	Reviewed the MOP performance measures based on Pareto optimal front and scalarization method
Audet et al. (2018)		A comprehensive review of 57 MOP performance indicators and partitioned into cardinality, convergence, distribution, and spread
Riquelme et al. (2015)		A brief review of the advantages/disadvantages of 54 performance metrics related to MOPs. The metrics are summarized into two groups. The first group is related to cardinality, accuracy, and diversity; whereas the second group is based on the approximation set: either unitary or binary metrics. The paper also ranks the 10 most used metrics based on the number of citations from 2005 to 2013
Okabe et al. (2003)		Overview of MOP metrics with advantages/disadvantages for cardinality-based, distance-based accuracy, volume-based accuracy, distribution, and spread. The paper highlighted not to rely on a single metric for conclusions and also proposed several metrics that can be applied for the problem of more than three objectives
Mirjalili and Lewis (2015)	Robust MOP	Review several MOP metrics and proposed metrics related to uncertainties that describe convergence, uniformity, and number of obtained robust as well as non-robust Pareto optimal solutions
Yu et al. (2018)	Ensemble methods on MOP	Review five popular MOP metrics and two ensemble methods for algorithm ranking

Table 2 Samples of statistical methods for algorithm evaluation from other literature

References	Scope of review	Brief description
Chiarandini et al. (2007)	Statistical evaluation as performance metrics	Review different scenarios of metaheuristics assessment and further divides statistical analysis into two models: the Univariate model that concerns on either solution-cost or run-time analysis, and the multivariate model that concerns both solution-cost and run-time measures. The paper discusses a detailed description of the cumulative distribution and statistical comparison test
Beiranvand et al. (2017)	Evaluation and reporting method for algorithm comparison	Review the practical methods of reporting the algorithm results in tabular, graphical, and profile measures. Three types of profiles are discussed include that performance profile, accuracy profile, and data profile. The paper highlights the suitability as well as the advantages/disadvantages of each method
Corani and Benavoli (2015)	Bayesian test	The paper review several frequentist and Bayesian inferential test for the accuracy of two competing algorithms. The scope of the paper includes single- and multiple data sets. Among the discussed tests are the t-test, correlated t-test, signed-rank test, Bayesian signed-rank test, and Poisson-binomial test. The paper highlighted limitations encountered by each test and further introduced correlated Bayesian t-test
Calvo et al. (2019)		The paper review Wilcoxon-Mann–Whitney test variants and comparison with Bayesian counterparts. The comparison is further analyzed using evolutionary algorithms over a set of 23 discrete optimization problems in several dimensions

2.1 Efficiency measure

The efficiency measure is related to the algorithm's response towards finding the optimal solution. It is heavily related to the computational speed, rate of convergence, and the time to find accepted optimal solutions. Rardin and Uzsoy (2018) highlighted that this criterion has attracted more attention in the literature than the quality of the solution. This has also driven more concerns in the field of parallel metaheuristic as the technology of parallel computing advanced (Nesmachnow 2014). The typical measurement of algorithmic efficiency is the graph representation of fitness convergence. Some other methods being used in the literature include the convergence rate (Senning 2015; Dhivyaprabha et al. 2018; Hamacher 2007; Paul et al. 2015; He 2016), algorithm complexity (Aspnes 2017) and several statistical measurements, see for example, from Chiarandini et al. (2007), Hoos (1998), Ribeiro et al. (2009), Hansen et al. (2016) and Derrac et al. (2014). In what follows, we discuss some basic measures related to the efficiency of the metaheuristic algorithms.

2.1.1 Rate of convergence

The rate of convergence is a measure of how fast the algorithm converges towards optimum per iteration or sequence. Some of the theoretical studies on convergence of the stochastic algorithms are based on the Markov chain process (Yang 2011) by estimating the eigenvalues of the state transition matrix or specifically the second largest eigenvalues of the matrix. Nonetheless, this method is complicated and difficult to estimate (Ming et al. 2006; Suzuki 1995). There is another method of evaluating the convergence, which is using an iterative function. In general, the function is equivalent to the rate of fitness change as expressed below (Senning 2015; Dhivyaprabha et al. 2018):

$$Conv.rate = \frac{\left|f_{opt} - f_i\right|}{\left|f_{opt} - f_{i-1}\right|}, \tag{1}$$

with f_{opt} as the optimum fitness value, f_i as the best value at ith iteration or step of function evaluation, and f_{i-1} as the best value at previous ith iteration or step of function evaluation. Equation (1) is a basic representation of the convergence rate. The convergence rate is applicable for both continuous and discrete domains. Based on the review from numerous literature, there are mainly four representations of convergence pattern for test function optimization as summarized in Table 3.

The first metric is the fitness convergence that refers to the dynamic change of fitness value concerning the steps. The convergence is usually represented as a graph showing the variation of fitness/cost value with either time unit, the number of iterations, or function evaluations (denoted as FEVs). This method shows straightforward information on the algorithm performance and most literature added this as one of their performance indicators. Nonetheless, it is limited to the scale of a feasible solution to the problem, such as towards 0 for $f_{opt} = 0$, or towards -1 for $f_{opt} = -1$. An example of fitness convergence is in Fig. 1. For an unknown optimum value, which is usually true for practical cases, the convergence rate is characterized by the ratios of consecutive errors (Senning 2015) as follows:

$$Conv.rate\ unknown\ optimum = \frac{\left|f_{i+1} - f_i\right|}{\left|f_i - f_{i-1}\right|}, \tag{2}$$

Table 3 Convergence representation in the literature

#	Convergence representation	Description
1	Fitness convergence (Sadollah et al. 2018)	Current fitness of problem per iteration or computation time, (R_1, R_2, R_3)
2	Convergence progressive rate (Liu et al. 2017a)	The absolute difference between optimum R_1 and current fitness, R_1
3	Logarithmic convergence rate (Salomon 1998)	logarithmic difference between optimum $R_1(A, B, U, p) = \int_{u \in U} C(A, R, u)p(u)du$, and current fitness, u
4	Average convergence rate He and Lin (2016)	Average of geometric convergence in k trials between $p(u)$ and current fitness, $u \in U$

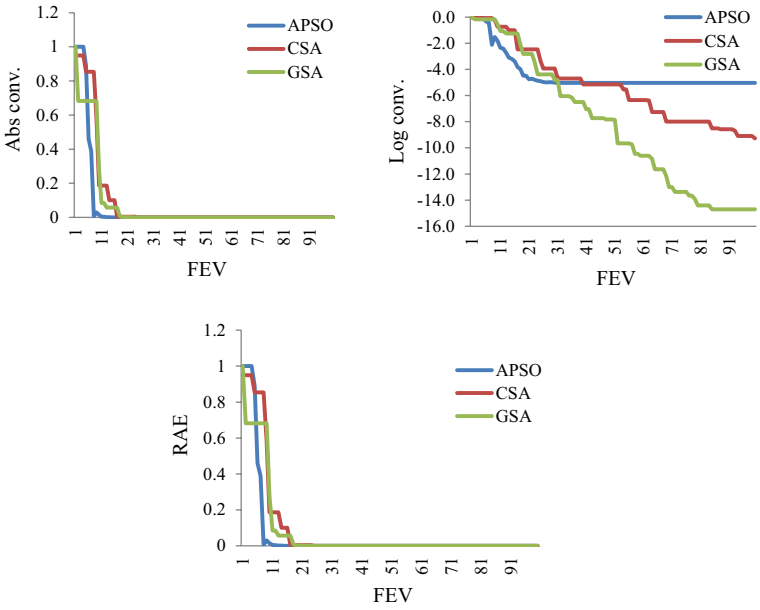


Fig. 1 Comparison of absolute convergence (upper left), logarithmic convergence (upper right), and convergence based on relative approximation error (RAE) (lower)

with f_{i+1},f_i, and f_{i-1} as the fitness of the next, current, and previous ith iteration. The second measure is the convergence progressive rate that is equivalent to the absolute difference $\left|f_i - f_{opt}\right|$ as implemented by Liu et al. (2017a). Measurement without known f_{opt} can be expressed with $\left|f_i - f_{i-1}\right|$ that represents the relative change of error with respect to the iterations or FEVs. This method is suitable for dynamic optimization problems as the objective changes over time. An example of such an application is in De Sousa Santos et al. (2019) that used this metric as a convergence in the stopping criterion. The third measure is the logarithmic convergence rate (Salomon 1998; Mortazavi et al. 2019) that is defined by $log\left|f_i - f_{opt}\right|$. The logarithmic convergence measures the dynamic fitness change throughout the iteration (He and Lin 2016). The curves of absolute and logarithmic convergence are depicted in Fig. 1. The figure compares convergence rate for Accelerated Particle Swarm Optimization (APSO) (Yang et al. 2011), Crow Search Algorithm (CSA) (Askarzadeh 2016), and Gravitational Search Algorithm (GSA) (Rashedi et al. 2009). As observed in the figure, the logarithmic convergence magnifies the absolute convergence pattern. As observed for CSA and GSA, both algorithms converged towards an optimum solution before 30th FEVs. However, the logarithmic convergence magnifies the pattern and reveals that GSA converged to its optimum solution on 86th FEVs, whereas CSA is still fine-tuning its convergence in 100th FEVs. The logarithmic convergence is also proposed by IEEE CEC (Liang et al. 2006) as a guideline for optimization algorithm competition by using the run-length distribution of $\log\left(f(x) - f_{opt}\right)$ concerning FEVs. For an unknown global optimum, the global optimum f_{opt} is replaced with the best of the run error (Das 2018).

The fourth type of convergence representation is the average convergence rate that specifically measures the rate of fitness change as proposed by He and Lin (2016).

The formulation is a geometric rate of fitness change between sequences of f_i to f_{opt} as defined in Eq. (3):

$$R(i) = 1 - \left(\left| \frac{f_{opt} - f_1}{f_{opt} - f_0} \right| \dots \left| \frac{f_{opt} - f_i}{f_{opt} - f_{i-1}} \right| \right)^{\frac{1}{i}} \equiv 1 - \left(\left| \frac{f_{opt} - f_i}{f_{opt} - f_0} \right| \right)^{\frac{1}{i}}, \tag{3}$$

where f_0 and f_1 is the initial fitness and fitness by $i=1$ respectively. The convergence rate $R(i) = 1$ if $f_i = f_{opt}$. This method was also implemented by Dhivyaprabha et al. (2018). An important point needs to be highlighted considering the observations in some papers related to convergence curve comparison. Usually, the convergence curves of a specific problem with n algorithms are compared and summarized in one chart. However, if the experiment is carried out with a huge number of FEVs or iterations, the presentation may look tedious and difficult to interpret. For this reason, a representable chart with a shorter number of evaluations or a zoomed version of the convergence curve is more appropriate. The objective is to compare and deduce the convergence trend of each algorithm. This shorter version chart can be attached at the side of the convergence with full FEVs. This applies to both continuous (such as constrained or unconstrained) and discrete such as combinatorial problems. An example of such a problem is shown in Fig. 2. The figure depicts the optimization comparison of 6 algorithms on berlin52 TSP problem as demonstrated by Halim and Ismail (2019). As observed in the figure, the highest convergence rate for all of the algorithms is in the first 500–700 FEVs, even though the experiments

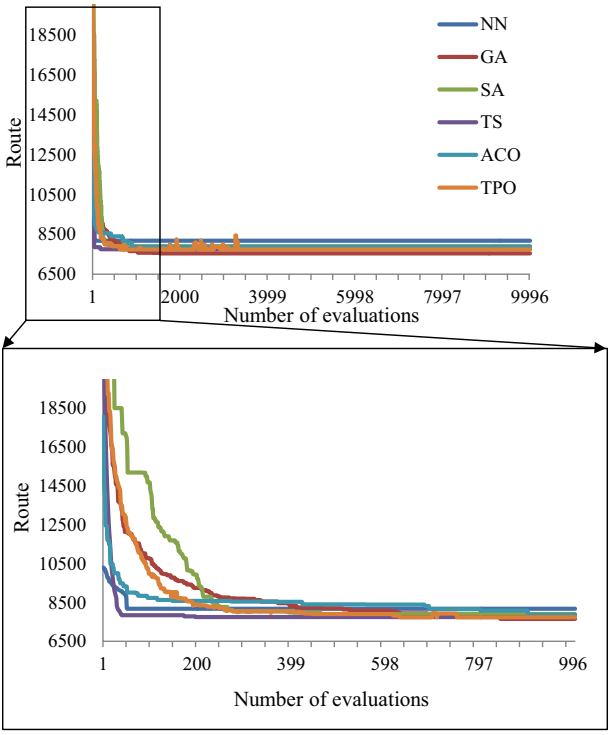


Fig. 2 Convergence curve of TSP (Halim and Ismail 2019). (Full forms of the algorithms used should be mentioned in the legend)

were run for 10,000 FEVs. Thus, the essential part to indicate which algorithms react faster towards a better solution is observed in this initial period. The latter period (say from 1000 to 10,000 FEVs) is for long term response of the algorithm and it is more appropriate to be represented in a tabulated solution. Some other examples are from Kiliç and Yüzgeç (2019) and Jalili et al. (2016) that superimposed the convergence curve of a shorter number of iterations into the original curve. Some examples of convergence curve with long FEVs that may need to be improved are from Degertekin et al. (2016) as well as El-Ghandour and Elbeltagi (2018).

The graphical representation of convergence is equivalent to the fitness rate of the algorithm and it is problem-dependent. In TSP, fitness is usually demonstrated by distance traveled over time that decreased towards an optimum solution. However, in TTP, the profit is increased towards the optimum solution. In Bin-packing problems, a graphical correlation between algorithms for initial and final solutions related to the number of items required (either in percentage or instance categories) with respect to the number of bins is one of the fruitful comparisons for relative efficiency, which is presented partly in Santos et al. (2019). Such graphical comparison can describe the relative performance of each algorithm (Grange et al. 2018) and is used for numerous practical Bin-packing problems such as the optimization of the last-mile distribution of nano-stores in large cities (Santos et al. 2019) and e-commerce logistic cost (Zhang et al. 2018).

Further enhancement of convergence rate comparison in continuous problems is using relative error (Hamacher 2007). The similar formulation is defined with a different name such as the index of error rate (Ray et al. 2007; Paul et al. 2015), RAE (He 2016) and the relative difference between best values as demonstrated by Agrawal and Kaur (2016) and Kaur and Murugappan (2008). The formulation at the ith generation is defined as follows.

$$E(i) = 1 - \frac{F_i}{f_{opt}}, \tag{4}$$

with F_i being the fitness value of ith generation or FEVs and f_{opt} as the optimum solution. The values from RAE pretty much exhibit a similar trend with the absolute convergence rate as shown in Fig. 1. Another convergence related index is the percentage of average convergence (Paul et al. 2015) as follows:

$$\textit{Average convergence } \% \ = 1 - \frac{\overline{f_x} - f_{opt}}{f_{opt}} \times 100, \tag{5}$$

with $\bar{f}_x$ as the average fitness. The index is used to measure the quality of the initial population since a good average convergence of the initial population increases the convergence speed with better exploration (Kaur and Murugappan 2008).

Apart from the convergence curve, other problem-related convergence measures were also proposed for combinatorial problems such as for TTP and TOP (Thief Orienteering Problem): the convergence of an algorithm A over algorithm B can be formulated by a fraction of average value found by algorithm A to the best solution found by either algorithm as shown in Santos and Chagas (2018) as follows:

$$A_x = \frac{\bar{A}}{max\left(A_{best}, B_{best}\right)}, \tag{6}$$

with $\bar{A}$ as the average objective value of algorithm A, A_{best} and B_{best} as the best solution found by both the algorithms respectively. This average convergence can be categorized by the different number of items and item relation types from the knapsack problem. To compare the convergence of algorithms by different problem types, an approximate of convergence relation between current best generation (or FEVs, or iteration) with the total number of generation calculated as follows:

$$C_{\text{relation}} = \frac{\mathrm{C_G}}{\mathrm{T_G}}, \tag{7}$$

with $\mathrm{C_G}$ as the current generation that corresponds to the best solution of an algorithm and $\mathrm{T_G}$ as the total number of generations. Hence, lower C_{relation} reflects a faster convergence of the algorithm. An example of implementation can be found in Zhou et al. (2019). A similar method can be adopted concerning the number of dimensions. A trend of convergence behavior can be observed by plotting the number of *succesful evaluation/dimension* over the number of dimensions. A higher slope corresponds to faster convergence by each dimension (Nishida et al. 2018). Another metric related to the convergence in design optimization is proposed by Mora-Melia et al. (2015). The metric is a combination of two formulations of rates defined as follows:

$$\mathrm{E} = \frac{\eta_{\text{quality}}}{\eta_{\text{convergence}}}, \tag{8}$$

where η_{quality} is the effectiveness of success rate equivalent to the fraction of successful run over the total run represented as $\eta_{\text{quality}} = (\text{Successful run}/\text{total run})$. The second term, $\eta_{\text{convergence}}$ is equivalent to the speed of convergence that is either time or number of FEVs to compute the final solution. An example of the application of this metric is observed in El-Ghandour and Elbeltagi (2018) that also used for the water distribution network problem, relatively similar problem application as Mora-Melia et al. (2015). This metric seems to be universal and can be applied to other problem types.

The convergence measure for the dynamic environment has different formulation due to the change of objective functions and constraints over time, which also reflects most of the real-world problems. These changes influence the optimization process throughout the measured time (Mavrovouniotis et al. 2017). There are several convergence measures proposed in the literature such as *offline error* (Yang and Li 2010), *modified offline error* and *modified offline performance* (Branke 2002), *staged accuracy,* and *adaptability* (Trojanowski and Michalewicz 1999). The offline error measures the average of differences between the best solution by an algorithm before the environment change and optimum value after the change defined as follows:

$$Offline_{error} = \frac{1}{K}\sum_{k=1}^{K}\left(h_k - f_k\right), \tag{9}$$

with f_k as the best solution found by the algorithm before the kth environmental change, h_k as the changed optimum value at the kth environment and K as the total number of environments. Examples of applications on this measure can be found in Zhang et al. (2020). The *staged accuracy* measures the difference between the current best of the population 'before change 'generation and the optimum value and averaged over the entire run, whereas the *adaptability* measure the difference between the current best individual of each generation

and the optimum averaged value throughout the run. Both formulations are shown in (10) and (11) respectively.

$$P_{StageAccr} = \frac{1}{K}\sum_{k=1}^{K}\left|f_{opt}^{(G_k)} - f_i^{(G_k)}\right|, \tag{10}$$

$$P_{adaptability} = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{G_k}\sum_{g=1}^{G_k}\left|f_{opt}^{(G_k)} - f_i^{(G_k)}\right|, \tag{11}$$

where K is the number of changes, G_k is the number of generations within stage k, $f_{opt}^{(G_k)}$ is the optimum value of each change or stage, and $f_i^{(G_k)}$ is the current best individual value in the population of ith generation. Other distinguished measures for dynamic environments are the Recovery Rate (RR) and the Absolute Recover Rate (ARR) proposed by Nguyen and Yao (2012). RR is denoted as a response speed for an algorithm to recover from an environmental change towards converging to a new solution before the next change occurs as in Eq. (12):

$$RR = \frac{1}{m}\sum_{i=1}^{m}\frac{\sum_{j=1}^{p(i)}\left[f_{best}(i,j) - f_{best}(i,1)\right]}{p(i)\left[f_{best}(i,p(i)) - f_{best}(i,1)\right]}, \tag{12}$$

where $f_{best}(i,j)$ as the fitness of the best feasible solution since the last change discovered by the algorithm until the jth generation of change period i, m is the number of changes and $p(i), i = 1 : m$ as the number of generations by every change of period i. RR=1 if the algorithm able to recover and converges towards a solution immediately after the change. On the other hand, RR=0 if the algorithm is unable to recover after the environment change. Another measure, ARR is used to analyze the response speed of an algorithm to converge towards global optimum as in Eq. (13):

$$ARR = \frac{1}{m}\sum_{i=1}^{m}\frac{\sum_{j=1}^{p(i)}\left[f_{best}(i,j) - f_{best}(i,1)\right]}{p(i)\left[f_{global}(i,p(i)) - f_{best}(i,1)\right]}, \tag{13}$$

with f_{global} as the global optimum of the respective problem. The rating of *ARR is similar to RR* whereas $ARR = 1$ if the algorithm able to recover and converges towards the global optimum. Both RR and ARR can be further described in a graphical representation (RR–ARR diagram in Fig. 3) to understand the convergence/recovery behavior of the algorithm as illustrated by Nguyen and Yao (2012). The RR–ARR diagram consists of a diagonal line and points of RR/ARR scores by each algorithm. The score is represented in x and y coordination of the point below the diagonal line. Each point's position corresponds to the characteristics of each algorithm. If the point lies on the diagonal line (such as algorithm A), it can recover from dynamic change and converged to a new global optimum. For other algorithms, such as algorithm B that lies closer to diagonal in more right side shows characteristics of faster recovery towards global optimum, whereas algorithm C is more likely converged to the local optimum with faster recovery and algorithm D is more likely not converged yet to an optimum solution with slower recovery (Nguyen and Yao 2012). In most cases, the RR–ARR measures can be used together to indicate if an algorithm can converge to the global optimum within a defined time frame between changes and also reveal how fast the algorithm requires to converge (Yang and Yao 2013).

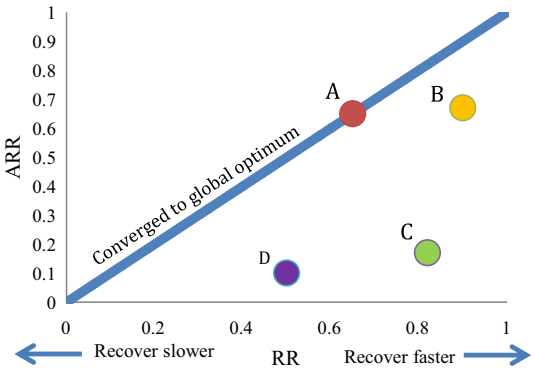


Fig. 3 An example of the RR–ARR diagram (Nguyen and Yao 2012)

2.1.2 Diversity

Another measure related to the convergence behavior of an algorithm is the diversity. Diversity is referred to as a distribution of the algorithm's population in the search space and it reflects the information of exploration and exploitation throughout the iteration. Higher diversity shows the ability of exploration, whereas the lower diversity of the population indicates an exploitative tendency of the search. Therefore, diversity measure affects the convergence behavior of the algorithm significantly and a good diversity will avoid premature convergence. The diversity of a population-based algorithm is heavily related to the speed, thus it is an efficiency-related measure. To date, numerous diversity measures are introduced in the literature. Cheng et al. (2014) introduced dimension-wise population diversity as formulated in Eq. (14):

$$Div = \frac{1}{D}\sum_{j=1}^{D} Div_j, \tag{14}$$

with D being the total number of dimensions and Div_j as the solution diversity based on the L_1 norm for the jth dimension. Other measurements are the diversity of population-entropy (Yu et al. 2005), diversity of average distance around swarm center (Krink et al. 2002), diversity of normalized average-distance around swarm center (Tang et al. 2015), and the diversity based on average of the average distance around the particles in the swarm (Olorunda and Engelbrecht 2008). This method extends the concept of the distance around the swarm center, where each swarm particle is denoted as a center and the average over all these distances is calculated. The diversity measure based on entropy (Yu et al. 2005; Tang et al. 2015) divides the search space into Q areas of equal size with Z_i search agents in a population of size N. The probability of search agents situated in ith area is then determined by Z_i/N. The population-entropy diversity for the continuous optimization problem is then defined as:

$$E(t) = -\sum_{j=1}^{Q} \frac{Z_i}{N} \log_e \left(\frac{Z_i}{N}\right). \tag{15}$$

In combinatorial problems such as TSP, the diversity measure based on entropy is formulated based on the number of edges, thus defined as the edge entropy. The edge entropy of a population is measured as (Tsai et al. 2003):

$$edge_{entropy} = -\sum_{e \in X} \frac{F(e)}{N} \log_2 \left(\frac{F(e)}{N} \right). \tag{16}$$

where $X = E(s_1) \cup E(s_2) \cup \ldots \cup E(s_n)$ is the series of edges, $F(e)$ is the number of edges, e is the current edge and N is the population size. Higher edge entropy implies higher population diversity. Another related measure is the edge similarity of a population is defined as (Tsai et al. 2003):

$$edge_{similarity} = \frac{2}{n(n-1)} \sum_{i}^{n} \sum_{j=1, j \neq i}^{n} \left| T_{i,j} \right|, \tag{17}$$

with $T_{i,j}$ as the number of edges shared by pairs (s_i, s_j). Both equations imply a diverse population by larger edge entropy and low edge similarity. Both equations can be represented with a curve versus the number of generations or iterations and plotted with the compared algorithms. A faster decrease of edge entropy from high value denotes a faster reduction of diversity towards the solution. Several examples of the metric application can be observed in Tsai et al. (2004) and Nagata (2006).

Another diversity measures for continuous problems highlights the diversity of search particles around the swarm center. The measure is based on the average distance around the swarm center (Krink et al. 2002) is shown as follows:

$$D_{Swarm\ center}(S) = \frac{1}{|S|} \sum_{i=1}^{|S|} \sqrt{\sum_{j=1}^{D} \left(p_{ij} - \bar{p}_j \right)^2}, \tag{18}$$

where S is the swarm, $|S|$ is the swarm size, D is the dimensionality of the problem, p_{ij} is the jth value of the ith particle and $\bar{p}_j$ is the jth value of the average point of particles $\bar{p}$. A lower $D_{Swarm\ center}(S)$ value shows convergence around the swarm center, whereas higher values indicate a higher dispersion of search agents away from the center.

Another concept similar to above was defined by Riget and Vesterstrøm (2002), but with further normalization concerning the swarm diameter, L. The formulation of the normalized average distance around the swarm center is shown in Eq. (19). This method is used in most of the literature such as by Mortazavi et al. (2019) and Aziz et al. (2014). Apart from swarm diameter, this normalization method can also be replaced with a swarm radius.

$$D_{Swarm\ center\ N}(t) = \frac{1}{|N||L|} \sum_{i=1}^{N} \sqrt{\sum_{j=1}^{N} \left(p_{ij} - \bar{p}_d \right)^2}, \tag{19}$$

with p_{ij} as the jth value of the ith search agent and the dth value of the average point $\bar{p}_d$ of all search agents. The value $\bar{p}_d$ also denotes the center of the swarm in the dth dimension. N is the population size and L is the longest diagonal length in the search space or the swarm diameter. A graphical representation of diversity concerning the FEVs can be used for observing the diversity of the search agents during the attraction–repulsion phase as a countermeasure to avoid the premature convergence as depicted in (Riget and Vesterstrøm (2002). As a definition, the swarm diameter is equal to the maximum distance between any two search agents, whereas the swarm radius is defined by the distance between the swarm center and the furthest path of the search agent from the swarm center. The swarm diameter is calculated as follows:

$$|L| = \max_{(i \neq j) \in [1,|N|]} \left(\sqrt{\sum_{k=1}^{Dim} \left(x_{ik} - x_{jk} \right)^2} \right), \tag{20}$$

where x_{ik} and x_{jk} are the kth dimension of the ith and jth search agent's position respectively. The swarm radius is defined as shown in Eq. (21) with $\bar{x}_k$ as the average of kth dimension from the swarm center. Both swarm diameter and radius can be used in diversity measures. Large values denoting highly dispersed search agents, whereas lower value showing convergence.

$$|R| = \max_{i \in [1,|N|]} \left(\sqrt{\sum_{k=1}^{Dim} \left(x_{ik} - \bar{x}_k \right)^2} \right). \tag{21}$$

Another concept of diversity is the average of average distances around the swarm center (Olorunda and Engelbrecht 2008) that evaluates the average distance around each search agent in the swarm that is using each particle as a center, then calculating the average overall distances as formulated in Eq. (22):

$$D_{all} = \frac{1}{|N|} \sum_{i=1}^{N} \left(\frac{1}{|N|} \sum_{j=1}^{|N|} \sqrt{\sum_{k=1}^{Dim} \left(x_{ik} - x_{jk} \right)^2} \right), \tag{22}$$

where the second term inside the bracket denotes the average distance around search agent x_i. This method indicates the average dispersion of all search agents in the swarm relative to each agent in the swarm. Another recently introduced diversity measurement is O-diversity (Chi et al. 2012). The method calculates the average distance around the global optimum point O without including the outliers that may affect the accuracy judgments of convergence or divergence at each specific state. The O-diversity is defined as follows:

$$D_O = \frac{1}{Dim} \sum_{k=1}^{Dim} \frac{1}{(N - N_0)} \sum_{i=1}^{N} \left(\left| x_{ik} - O_k \right| - S_O \right), \tag{23}$$

where N_0 is the number of outliers, S_O is the sum of outliers on dimension k and O_k is the optimal point at the kth dimension. The O-diversity D_O is also defined as the population position diversity (Chi et al. 2012). Declining value of D_O indicates better optimization performance as the algorithm reaches the global best value O.

The dynamics of the diversity move of each search agent also has been considered in the literature. These measures defined as swarm coherence (Hendtlass and Randall 2001), which is defined as:

$$S_C = \frac{V_S}{\bar{V}}, \tag{24}$$

with V_S as the speed of swarm center and $\bar{V}$ as the average agent's speed. The swarm coherence is a ratio of swarm center speed concerning the average speed of all agents in the swarm. The speed of the swarm center is defined as in the following equation (Hendtlass 2004).

$$V_S = \left\| \frac{\sum_{i=1}^{N} \vec{V}_i}{N} \right\|, \tag{25}$$

with $\vec{V}_i$ as the velocity of each agent over N number of swarm and x is a distance norm such as Euclidean distance. Larger S_C shows a higher center velocity and implies a high percentage of search agents that traverse with the same velocity vector (in the same direction) in high acceleration. Conversely, lower values of swarm center show either a high percentage of the search agents are traversing in the opposite direction or still traversing in the same direction but with a much slower speed. For the average agent's speed, larger values show that the search agents on average are making large changes compared to current positions, which is also denoted with higher exploration. On the contrary, smaller values of average speed denote that the search agents are wandering around their relative proximity and the neighborhood bests, hence also exploitation. In a perspective of swarm coherence, a lower value shows either the agents are highly dispersed in the search space or they are traversing in relatively small direction. A higher swarm coherence denoting either the swarm is converging or they are traversing in the same resultant direction.

The diversity measure in discrete optimization shows the capability of an algorithm to converge to local optimum at early iterations. An example from Tsai et al. (2014) proposed the diversity measure as an average in a search space for TSP as follows:

$$\bar{S}^t = \frac{1}{2n} \sum_{i=1}^{n} \sum_{j=1}^{n} e_{i,j}^t, \tag{26}$$

where n is the number of cities and $e_{i,j}^t$ as the edge between city i and j. $e_{i,j}^t = 1$ if an edge between both cities exists and $e_{i,j}^t = 0$ otherwise. The value $\bar{S}^t$ represents the average path of all cities in t generations. As an overview comparison, a curve of diversity by different algorithms can be plotted against the number of generations. Usually, the diversity of an effective algorithm increases in the early step of generations, and later become smaller as it converging towards the optimum.

2.1.3 Combinatorial problem-specific metrics

Some of the efficiency metrics related to the combinatorial problems are discussed in this sub-section. The first metric is the sub-optimum solution. The speed of an algorithm to reach an optimal solution can be measured by summing up the sub-solution of optimal results (such as TSP, TTP, or OP). As an instance, , $\varphi = \{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ are the optimal sub-solution found by an algorithm at a specific generation. Thus, the rate of edge optimum at generation t can be expressed as follows (Tsai et al. 2014):

$$O_t = \sum_{i=1}^{n} O_{i,j}^t, \tag{27}$$

with n being the number of instances, and $O_{i,j}^t$ as the condition of sub-solution existence; $O_{i,j}^t = 1$ refers to the existence of an optimum sub-solution edge between pairs i and j, whereas $O_{i,j}^t = 0$ is otherwise. The second metric related to the combinatorial problem is tour improvement. The average improvements by an algorithm can be compared with other algorithms with Eq. (28) as follows:

$$\bar{x}_{impr} = \frac{1}{N}\sum_{i=1}^{N}\left[f_A(s_i) - f_B(s_i)\right], \tag{28}$$

where N is the population size, s_i as the ith individual in the population, f_A as the proposed algorithm and f_B as the benchmarked algorithm. The motivation for this measure is to observe any improvement of the proposed algorithm against a benchmarked algorithm. This comparison can also be made between the older and improved algorithms as used by Tsai et al. (2004). For the Bin-packing problem, a convergence of total cost with the number of bins for compared algorithms can be highlighted as showed by Zhang et al. (2018). Any algorithm that converged faster towards lower cost is considered as more efficient.

The efficiency measure related to the Travelling Thief Problem (TTP): TTP consists of the combination between Travelling Salesman Problem (TSP) and Knapsack Problem (KP) as proposed by Bonyandi et al. (2014). The optimization for TTP need a tradeoff between each sub-problem (TSP and KP) since higher picking item of KP resulted in the higher total value of items, but also resulted in reducing speed and increases the renting rate. The combination of each sub-problem leads to the variation of TTP instances such as TSP related routes and KP-types: *uncorrelated, uncorrelated with similar* weights, and *bounded strongly correlated* types. Other TTP instances are item factor (number of items per city in each TSP-KP combination) and renting rate that links between each sub-problem. The performance measures between algorithms can be described graphically by plotting the gain of tour time, f with the total value, g, and the best solution must be compromised between both gains as demonstrated by Bonyandi et al. (2013). Furthermore, the importance of TSP and KP can be dynamically interchanged throughout the process. Reducing the value of the renting rate might reduce the contribution of the TSP component to the final objective value, whereas increasing the renting rate might lead to less significant of the total profit items to the impact of final objective value (Bonyandi et al. 2014).

For Orienteering problem, OP, the significant performance measure is the computation time between nodes or also defined as service time (Li and Hu 2011), the set of visited nodes in a tour, and the score of visiting each node to maximize the fitness as in the following equation:

$$max \sum_{i=1}^{n}\sum_{j=1}^{n} S_i x_{ij}, \tag{29}$$

with S_i as the associate score of node i, and $x_{ij} = 1$ if the tour visits node j immediately after visiting node i, $x_{ij} = 0$ otherwise. The algorithms shall visit each prescribed node within a defined maximum time. Thus, an algorithm that fails to score within maximum time is considered an underperformer. A unique measure of the Lin-Kernighan (LK) algorithm for solving TSP is using the search effort. The LK algorithm and its variants consist of edge exchanges in a tour and this procedure consumes more than 98% of the algorithm's runtime (Smith-Miles et al. 2010). The search effort of the LK-based algorithms is measured based on the count of edge exchange occurs during the search and it is independent of the hardware, compiler, and programming language used (Smith-Miles et al. 2010). Further readings on LK-based algorithms and its efficiency measures can be found in Van Hemert (2005) and Smith-Miles and Van Hemert (2011).

2.1.4 Computation cost

The computation cost of an algorithm is dependent mainly on the number of iterations and population size (Polap et al. 2018). Each algorithm is executed until exceeding the number of iterations or until no improvement is achieved (global optimum solution). Based on Rardin and Uzsoy (2018), computation time is crucial and increased more attention than the solution quality due to different hardware and software technology. Jackson et al. (1991) and Barr et al. (1995) discussed several factors that need to be considered to evaluate the computation time. The time measurement can be captured in mainly three parts of the algorithm execution: time to best-found solution, total computation time, and timing per phase (Barr et al. 1995). The time to best solution is referred to as the computation time required by the algorithm to find the global optimal solution. The total computation time is referred to as the run time of an algorithm until it is terminated due to its stopping criteria. Thus, the time to the best solution may be lesser than the total computation time, and this performance can be easily demonstrated in a classical convergence of fitness versus time curve. The third computation time (the time per phase) measurement is another quality option that is referred to as the timing of each defined phase such as initial population, improved version, the sequence of a collaborative hybrid algorithm, and percentage to the global optimum.

There are several methods proposed in the literature on capturing the runtime of algorithms in continuous and discrete problems. Some literature (Conn et al. 1996; Benson et al. 2000) proposed performance calculation based on the average and cumulative sum over all the problems. However, this may bias the results for small numbers of difficult problems as it may dominate the whole results (Dolan and Moré 2002). Another approach is by comparing the medians and quartiles between solver times (Bongartz et al. 1997). This method is superior to the average method since it did not bias the results, however, the information of trends between one quartile to the next may lead to uncertain assumptions (Dolan and Moré 2002). Some researchers capture the computation time descriptively. An example of such a method is the ratio of computation time to converge towards 5% of the best-found solution (Barr et al. 1995) as formulated in Eq. (30):

$$T_{ratio} = \frac{Time\ to\ converge\ within\ 5\%\ \ global\ optimum}{Time\ to\ global\ optimum}. \tag{30}$$

Another method of comparing the ratio of computation time with others is by rating the percentage of time consumed by an algorithm to the best runtime (Billups et al. 1997). The rating is categorized by *competitive* or *very competitive*. The rating is defined as *competitive* if the time consumed by the proposed algorithm is $t \leq 2T_{min}$ and *very competitive* if $t \leq \frac{4}{3}T_{min}$, with T_{min} as the minimum time obtained among all the algorithms on a specific problem. However the definition of classification limit (*competitive* or *very competitive*) is depending on the researcher's arbitrary option that may have some tightness or looseness in question. Some pieces of literature compare computation cost via a ratio between an unsuccessful and a successful run that concerns on the runtime of the algorithm. The proposed method is defined with average runtime, aRT (Hansen et al. 2012), which is formulated as in the following equation:

$$aRT = \frac{\left(n_u + n_s\right)}{n_s}, \tag{31}$$

with n_u as the number of unsuccessful runs and n_s as the number of successful runs. The runtime did not necessarily refer to the computation time; it can also be correlated with the number of Function Evaluations or FEVs. It is not recommended to use the number of iterations to benchmark the computation steps. The number of FEVs is usually taken as a reliable measure of the computation complexity, provided there is no hidden cost inside the algorithm. The number of iterations for different algorithms may perform different amounts of computation in their inner loops. Furthermore, comparing using CPU time is inappropriate and it is preferable to use the more abstract FEVs for comparing stochastic algorithms (Hellwig and Beyer 2019).

In discrete optimization problems, one of the most commonly used for runtime measure is the first hitting time (FHT) (Nishida et al. 2018) that is defined as the number of function evaluations required by the algorithm to reach the first hit of the optimum solution. Lehre and Witt (2011) defined the hitting time measurement by considering the time required to reach bound of optimum $\varepsilon > 0$ for which $\left|f(x) - f\left(x_{opt}\right)\right| < \varepsilon$ that denoted as the expected FHT with respect to ε. The growth of expected FHT is also bounded by a polynomial in $1/\varepsilon$ and the number of dimensions of the specific problem (Lehre and Witt 2011). The performance of FHT can be plotted against the number of dimensions, thus a comparison of FHT between algorithms with respect to dimensions can be analyzed graphically.

Another useful method of comparing the computation time between algorithms for both continuous and discrete domains is using a Box and Whisker plot as illustrated in Fig. 4. The Box and Whisker plot summarizes important features of the data and helps to demonstrate the comparison among the algorithms. The definition of each section (Fig. 4 right) on the box-whisker plot is summarized in Table 4. The analysis in the scope of computation time with the number of dimensions and variables is also another performance measure for algorithm efficiency. Ideally, the computation time increases with the number of algorithmic variables. Saad et al. (2017) demonstrate the relationship between the computation time in seconds and FEVs with the number of variables in various algorithms. This comparison was able to show which algorithm has a better performance for what number of variables.

During comparison among different algorithms, computing system between each comparison instance must be taken into careful consideration. Precise time measurement for algorithm comparison is crucial. Some literature mentioned the method of capturing the computation time in high resolution as demonstrated by Shukor et al. (2018), Ngamtawee

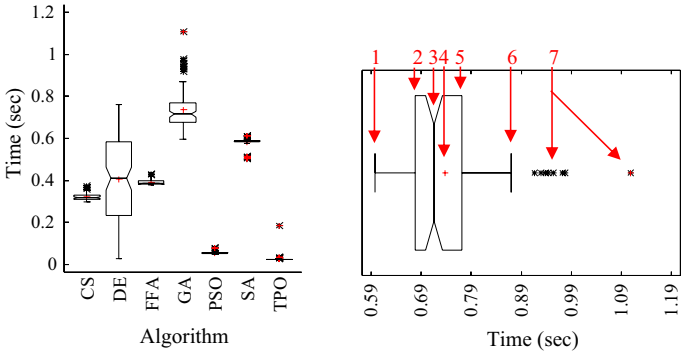


Fig. 4 Box-plot for comparison between algorithms (left) and its definition (right)

Table 4 Box-plot description

Number	Description
1	Whisker represents the smallest value of data: ¼th quadrant of data
2	The minimum range of middle 50% of data
3	Median value and notch. Median notch shows an approximate of $100(1-\alpha)$ % of the confidence interval for median: ½ quadrant of data
4	Sample mean
5	The maximum range of middle 50% of data: ¾th quadrant of data
6	Whisker represent the biggest value of data: ¼th quadrant of data with bigger values
7	Outlier points that have 1.5 times the inter-quartile range (box width). Any points that lie more than three times of interquartile range is denoted as far outside points (Tukey 1977) and represented with plus signs superimposed on point symbol.

and Wardkein (2014) and Bahameish (2014), where the elapsed time was captured in Nanoseconds using JAVA function code *System.nanoTime()*. Other literature such as from Pan et al. (2019) measured and presented the time in milliseconds.

2.1.5 Comparing different platforms

The combinatorial NP-hard problems usually consume higher computation cost and directly influence the computer CPU. Thus, it is not straightforward to distinguish the computational time of each method that being compared if it is executed from different computer configurations. Numerous researchers compared their algorithm's computation with other published papers especially in the Vehicle Routing Problem and Orienteering Problem. The benchmarked comparison is the processor efficiency rate or CPU speed. There are plentiful benchmark programs available, but it is important to keep in mind that the performance of various processors depends on many external factors such as compiler efficiency and type of operation used for measurement. However, it is fairly enough to attach as a reference in the paper if it meant to indicate a rough estimation of the computational speed of different platforms. Some examples of benchmark comparisons are using *a million of the floating-point operations per second*, Mflop/s (Lo et al. 2010; Rouky et al. 2019). Other study benchmarked CPU speed using *a million instructions per second*, MIPS (Attiya et al. 2020; Jena et al. 2020). Another example is using System Performance Evaluation Cooperative (later is defined as the Standard Performance Evaluation Corporation), SPEC (Fenet and Solnon 2003; Crainic et al. 2011; Wu et al. 2018). Both MIPS and Mflop/s are relatively easy to understand and measurable. The MIPS metric measures the number of CPU instructions per unit time. It is usually used to compare the speed of different computer systems and to derive the execution time. It is mathematically defined as follows (Dawoud and Peplow 2010):

$$MIPS = \frac{\text{Instruction count} * \text{Clock rate}}{\text{CPU clocks} * 10^6}, \tag{32}$$

where CPU clocks denote the total number of clock cycles for program execution. The execution time can be derived with a known value of MIPS as follows: $\textit{Execution time} = \textit{Instruction count}/\left(MIPS * 10^6\right)$. In contrast, the Mflops metric

measures the floating-point operations per million of execution time or mathematically defined as follows (Dawoud and Peplow 2010).

$$MFLOPS = \frac{\text{Number of floating} - \text{point operations}}{\text{Execution time} * 10^6}. \tag{33}$$

The metric signifies floating point operations such as addition, subtraction, multiplication, and division that applied to numbers represented by single or double precision. The data are specified in program language using keywords such as float, real or double. It is necessary to keep in mind the drawbacks of these methods. The Mflop/s metric depends on the type of floating-point operations present in the processor and it treats the floating-point operation for addition, subtraction, multiplication, or division equally. Practically, the complexity of floating-point division is much higher and time-consuming than the floating-point addition. Some of the problems for MIPS are its nonlinearity behavior and inconsistent with the correlation of performance (Noergaard 2013). Furthermore, the metric can also be inconsistent even comparing processors of the same architecture due to the possibility of not measuring throughout processor performance for instance I/O or interrupt latency (Dawoud and Peplow 2010). The SPEC method is more rigorous than MIPS and Mflop/s. The SPEC method evaluates the rate metric by measuring the time required to execute each benchmark program on a tested system and normalized the time measured for each program by the required execution time. The normalized values are then averaged based on the geometric mean. Nonetheless, this method has shortcomings such that the geometric mean is not linearly related to the program's actual execution time and it was shown unreliable metric where a given program executes faster on a lower SPEC rating (Dawoud and Peplow 2010). Another possible benchmark comparison is CoreMark (Ibrahim 2019). The method is mostly used to indicate the processor performance in microcontroller technology and it is more reliable than Mflop/s and MIPS (Embedded 2011). As far as our knowledge goes, there is no literature benchmarked with this method yet.

2.1.6 Algorithm complexity

The efficiency of the algorithm for the CPU time can also be measured by its complexity. The complexity of an algorithm's performance is related to space and time (Aspnes 2017; Dawoud and Peplow 2010). Space and time complexities quantify the amount of memory and time taken by an algorithm to run as a function of the length of the input. Numerous factors can affect the time and space complexity such as hardware, operating system, processor, compiler software, and many more, which may not be included in the algorithm performance comparison. The main concern of complexity in algorithm performance is how the algorithm is executed. An excellent tool for comparing the asymptotic behavior of algorithms' complexity is the Big-O notation (Brownlee 2011; Cormen et al. 2001). This method provides a problem independent way of characterizing an algorithms space and time complexity. As an example is a coding of nested-loop as follows.

In the worst case in Fig. 5a, the ***for*** loop runs n times, then the *counter*++ will run for $0 + 1 + 2 + \cdots + (n-1) = \frac{n*(n-1)}{2}$. Therefore the time complexity for an asymptotic upper bound of the algorithm will be $\mathrm{O}(n^2)$ with O denotes as the Big-O-notation. The computation under O-notation will ignore the lower order terms since it is insignificant for larger input. However, in Fig. 5b, the algorithm computes counter++ with $n + n/2 + n/4 + \cdots + 1 = 2 * n$. Thus, the time complexity of the algorithm will be O(n) since higher order is '1'. Therefore the time complexity for the algorithm in Fig. 5b

```
int counter = 0
for (int i = 0; i < n; i++)
for (int j = 0; j < i; j++)
counter++;
```

(a) algorithm with higher complexity

```
int counter = 0,
for (int i = n; i > 0; i /=2)
for (int j = 0; j < i; j++)
counter++;
```

(b) algorithm with lower complexity

Fig. 5 Pseudocode of for-loop with different algorithm complexity

is more efficient. The simplification of the algorithm is not a recent topic and was discussed in numerous works of literature such as (McGeoh 1996) and to date such as from Zhan and Haslbeck (2018), Rakesh and Suganthan (2017). The complexity experiment shall be tested on a predefined function with several trials and the CPU time for each trial. Some standard guidelines of algorithm complexity comparison as presented in Hansen et al. (2012) for Black-Box Optimization Benchmarking (BBOB) and Suganthan et al. (2005), Awad et al. (2016) for the Congress on Evolutionary Computation (IEEE CEC). For BBOB competition, the participants shall run the complexity test on *Rosenbrock function* with defined dimensions. The setup, coding language, compiler, and computational architecture during experimenting also need to be recorded. For CEC competition (CEC17), the complexity analysis is shown in Fig. 6. The time for executing the program in Fig. 6 is denoted as $T0$. Then the same procedure is repeated to evaluate time $T1$ with 200,000 evaluations of the same D dimension of specific test function (Function 18 from CEC17). The procedure is repeated five times with the same function (Function 18 from CEC17) to evaluate time $T2$. The five values are then averaged to evaluate $\hat{T}2$. All of the results $\left(T0, T1, \hat{T}2\right)$ are tabulated and these procedures are calculated for 10, 30, and 50 dimensions to observe the algorithm complexity's relationship with dimension.

The complexity measure is also widely used in discrete optimization. Different kinds of literature define the time complexity according to the expectation of the algorithm's performance. As an example, Ambati et al. (1991) proposed time complexity of GA-based algorithms with $O(n \log n)$, whereas Tseng and Yang (2001) shows the time complexity of GA is $O\left(n^2 ml\right)$ and Tsai et al. (2014) proposed with $O(nml)$ with n as the number of cities, m as the number of chromosomes, and l as the number of generations.

Fig. 6 Pseudocode for measuring the algorithm's complexity (IEEE CEC 2017)

```
x = 0.55
for i = 1:1000000
   x = x + x;
   x = x/2;
   x = x * x;
   x = sqrt(x);
   x = log x;
   x = exp(x);
   x = x/(x + 1);
end
```

2.1.7 Statistical analysis

There are several statistical analyses proposed in the literature for measuring the efficiency of algorithms. These are empirical cumulative distribution functions and ordered alternative tests.

2.1.7.1 Empirical cumulative distribution function, ECDF The performance measure of T runtime from experiments of metaheuristic algorithms on a specific problem can be described by a probability that is equivalent to the cumulative distribution function (Chiarandini et al. 2007; Hoos 1998). The cumulative distribution of sampled data $\boldsymbol{T_1}, \ldots, \boldsymbol{T_n}$ is then characterized by empirical cumulative distribution function (ECDF) that is equivalent to Eq. (34). The distribution is also denoted as a run time distribution.

$$F_n(t) = \frac{1}{n} \sum_{i=1}^{n} I\left(T_i \leq t\right), \tag{34}$$

where n is the number of sampled data and $I(\cdot)$ denotes the indicator function. This general formulation hold for both censored (if a time limit is defined before reaching the optimum solution) and uncensored data. An example of the ECDF curve of the SA algorithm for solving the combinatorial problem is depicted in Fig. 7 (top). The mid-line of ECDF represents the median of the overall solution. The ECDF also can be used to compare between algorithms as shown in Fig. 7 (bottom), where three algorithms (PSO, TPO, and DE) are superimposed in one chart and revealed that DE outperformed the other two algorithms, whereas TPO performed better than PSO due to the probability,$P_r\left(T_{DE} \leq T_{TPO} \leq T_{PSO}\right)$.

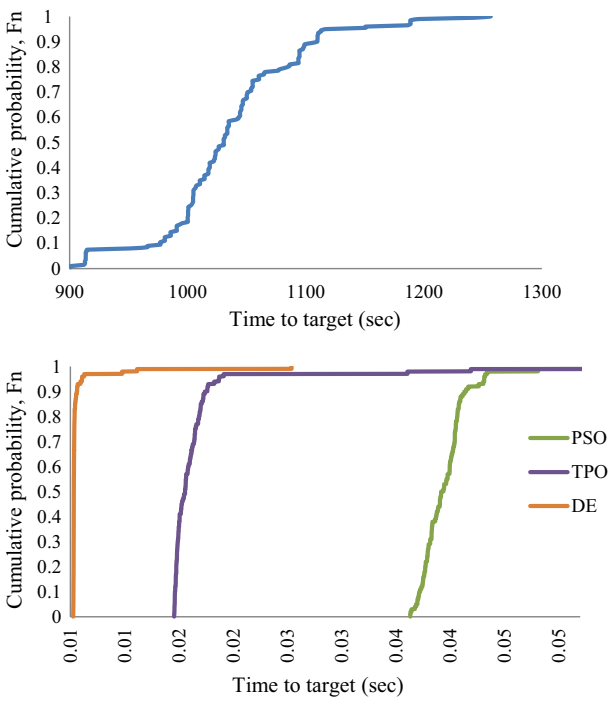


Fig. 7 ECDF as single algorithm measurement (top) and comparison between algorithms (bottom)

Some literature benchmarked ECDF as a performance indicator (Ribeiro et al. 2009; Hansen et al. 2016).

2.1.7.2 Ordered alternative test The ordered alternative test consists of non-parametric multiple tests that assume the null hypothesis of equal trends or medians and the alternative hypothesis of series of treatments of unequal trends or medians. There are three types of ordered alternatives used in metaheuristic performance analysis that include Page test, Terpstra-Jonckheere test, and Match test as simplified in the following subsections.

(a) Page test (Derrac et al. 2014; Page 1963)

Method construct order from k treatments on N samples and ranked from the best (with 1) to the worst (with k). The number of k treatments depends on the number of cut-points of N samples. Then the Page L statistic as in Eq. (35) is computed using the sum of ranks. An *alternative ranks procedure* is applied for algorithms that reached an optimum solution before the end of total cut-points (either *algo1* or *algo2* or both) (Derrac et al. 2014). Advantage: two graphical instances: (1) convergence in a discrete manner based on the cut-points, (2) deviation of ranks between two algorithms concerning the cut-points.

$$L_{Page} = \sum_{j=1}^{k} jR_j = R_1 + 2R_2 + \cdots + kR_k, \tag{35}$$

with R_j as the rank of jth of k measures from N samples. The sum of ranks values R_j will follow an increasing order of measured algorithm convergence. Thus for an increasing order, the null hypothesis will be rejected in favor of the alternative. Example of Page test from (Derrac et al. 2014) that compares two algorithms in 10 cut-points regular intervals on a minimization problem is shown in Fig. 8.

(b) Terpstra–Jonckheere test, TF test (Terpstra 1952; Jonckheere 1954)

Method Essentially based on Mann–Whitney U statistic, where U is obtained for each pair of samples and added. For k samples, the U statistic is calculated for each of $\frac{k(k-1)}{2}$ pairs and ordered. Then the test statistic is computed by summing each U statistic as in Eq. (36). Advantage: powerful alternative hypothesis with ordered medians of either decreasing or increasing pattern.

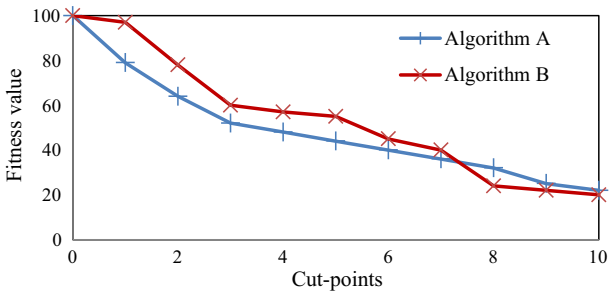


Fig. 8 Page test for comparison between algorithms (Derrac et al. 2014)

$$W_{TJ} = \sum_{j=1}^{k} U_j = U_1 + U_2 + \cdots + U_k, \tag{36}$$

where U is the Mann–Whitney U statistic for an individual sample with $j = [1, 2, \ldots, k]$. For large sample sizes, the null distribution of W_{TJ} approaches a normal distribution. Thus, calculation of mean (μ) and standard deviation (σ) is necessary to determine the critical value with $W_{TJ} \leq \mu - z\sigma - 1/2$. An example of metaheuristic algorithms analysis using the TJ test is by Obagbuwa and Adewumi (2014).

(c) Match test (Neave and Worthington 1988)

Method Similar to Page and TJ test but the calculation is based on rank-sums. Determined by the number of matches of ranks with the expected ranks and half of near matches. Null hypothesis similar to other non-parametric methods and alternative hypotheses similar to the TJ test. The test is computed by ranking a row from 1 to k and ties are assigned as average ranks. Each rank is compared with the expected rank, defined as the column index. A match is counted if the rank equals the column index. Every non-match that lies between $0.5 \leq |r_i| \leq 1.5$ is counted as a near match, with r_i as individual rank. The test statistic is calculated as in Eq. (37):

$$L_{\text{Match}} = L_1 + \frac{1}{2}(nm), \tag{37}$$

with L_1 as the number of matches and nm as the number of near matches. Similar to the TJ test, large sample sizes approach a normal distribution, thus the mean and standard deviation is calculated and the critical value is determined with $L_{Match} \geq \mu + z\sigma + 1/2$ with z as the upper tail critical value from normal distribution and $1/2$ as a continuity correction.

2.2 Effectiveness measure

2.2.1 The effectiveness rate of solution

Variants of rate measures to demonstrate the algorithm effectiveness are defined in the literature such as the successful convergence, feasible rate, FEVs by successful runs, success rate, and performance. These methods are clustered as the effectiveness rate of solution since they are based on the same foundation of rate measurement.

2.2.1.1 Successful convergence In essence, the successful convergence can be presented by calculating either count or percentage of local or global optimum under a defined number of trials and can be represented in tabular or graphical form. The interpretation from this analysis is straightforward, which is to show which algorithm has a higher percentage and frequency of solutions towards the near optimum. This measure applies to both continuous and discrete domains. Some literature defined a threshold for a specific problem and the algorithm is accepted as successful if it converges to a lower or equal to the threshold (minimization); defined as a percentage of success rates (Kiran 2017).

Typical function optimization, especially for the benchmark functions in various CEC competitions like the CEC 2014 (Liang et al. 2013), requires the measures of error value such that $\left(f_i(x) - f_{opt}\right)$ with $f_i(x)$ as the current solution found by algorithm and f_{opt} as the

optimum solution of the respective test function. Most of the papers presented their absolute results after the end of FEVs or after the maximum function of evaluation is elapsed. The obtained error values are then stored for n runs and the statistical indices (such as mean and standard deviation) are ranked and further analyzed with the statistical inference method. This process is denoted as the static comparison and some examples are in Chen et al. (2010) and Epitropakis et al. (2011). The main disadvantage of this type of static comparison and ranking is due to the fixed computational budget and the ranking might be different if another computational budget is used for the comparison. A superior method is calculated based on several rankings on several cut-points within a defined computational budget, which is also referred to as the dynamic comparison and ranking (Liu et al. 2017b). A good example of such a method applies to the CEC competitions (Liang et al. 2013) that require function values at several cut-points. In the CEC 2014 competition, the computational budget is limited to $MaxFEV = 10000 * D$ with D as the dimension. The dynamic ranking is calculated in 14 cut-points with $(0.01, 0.02, 0.03, 0.05, 0.1, 0.2, \ldots, 0.9, 1.0) * MaxFEV$ for each run. From this dynamic, the algorithm performance can be identified with respect to the path of the cut-points. This method applies to both continuous and discrete domains.

In various combinatorial problems, the percentage of improvement is usually defined as the rate of differences between the best-known optimum found by algorithms and the global optimum solution. Most of the combinatorial problems are already set with the global optimum value. Examples for most of the TSPs are defined in http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/tsp/ and for TTPs in https://cs.adelaide.edu.au/~ec/research/combinatorial.php. Thus, the general algorithm effectiveness can be calculated easily by the difference between found solutions for the prescribed objective value as follows:

$$\mathrm{AE}\% = \frac{p_{\mathrm{algorithm}} - p_{\mathrm{opt}}}{p_{opt}} \times 100, \tag{38}$$

with AE% as the percentage of algorithm effectiveness, $p_{\mathrm{algorithm}}$ as the optimum solutions found by algorithm and p_{opt} as the prescribed optimum value of the particular problem. It is to highlight that some papers presented the effectiveness such as Zhou et al. (2019) with the deviation of the best solution found by the algorithm concerning the prescribed optimum value:

$$\Delta\% = \frac{p_{best} - p_{opt}}{p_{best}} \times 100, \tag{39}$$

where p_{best} as the best solution found by the algorithm. This expression is straightforward, however, it did not represent the true algorithm characteristic as it only shows the best solution found by the algorithm and maybe the algorithm converges in large standard deviation within n number of runs. The absolute calculation of Eq. (39) is not wrong and can be inserted as one of the metrics; however, two other metrics that represent the overall algorithm effectiveness must also be presented. These are the average of converged solutions within n number of trials and the variation of solutions generated by the algorithm. Appropriate representation is by calculation of the difference between the averages of solution within n trials with the optimum result as expressed below:

$$\mathrm{AE}_{\mathrm{average}}\% = \frac{p_{average} - p_{\mathrm{opt}}}{p_{opt}} \times 100, \tag{40}$$

with $p_{average}$ as the average of the solutions of the algorithm in n many trials. Some examples of recent papers using such metrics are Ali et al. (2020) and Campuzano et al. (2020). A similar metric is also used for the Crew Scheduling Problems (CrSPs) that measure the percentage gap between solutions found by the algorithm with respect to the best-known solution (García et al. 2018). Most of the literature on combinatorial problems defined the difference between the proposed algorithm and the best value or another algorithm of the same portfolio as the gap percentage. In the TTP problem, the gap percentage can be divided into several categories such as the number of cities, number of picked items, and the ratio between numbers of items concerning the cities as demonstrated by Bonyandi et al. (2014). With these comparisons, more understanding of algorithm performance over each instance can be introduced. The same formulation is also applicable in the QAP problem, where it is defined as the percentage excess that is equivalent to the average of best-known solutions over n number of trials (Merz and Freisleben 2000). The second important metric is the variation or spread of the solutions, which will be discussed in Sect. 2.2.5. Another useful measure for both continuous and discrete problems is the relative effectiveness with respect to the benchmarked solution in either time or distance relation. The formulation is simplified as the following equation:

$$\Delta p = \frac{p_A - p_B}{p_B} \times 100, \tag{41}$$

with p as either distance or computation time and p_A as the solution of the proposed algorithm and p_B as the benchmarked algorithm that has a better performing solution. A positive value of Δp denotes that the benchmarked algorithm is superior to the proposed algorithm. Some examples of literature using this feature are from Skidmore (2006), Tsai et al. (2014) and Silva et al. (2020) for TSP optimization and Wu et al. (2020) for QKP.

2.2.1.2 Asymptotic performance ratio In combinatorial problems especially for Bin packing problem, such as Balogh et al. (2015) presented an asymptotic performance ratio denoted as the ratio of an optimal number of bins converge by an algorithm to the optimum solution as defined in the following equation:

$$R_{Asymp}(A) = \lim_{n \to \infty} sup \left\{ \max_{L: Opt(L)=n} \left\{ \frac{A(L)}{n} \right\} \right\}, \tag{42}$$

where L as input number of bins used by algorithm A to pack and $Opt(L)$ as the number of bins in an optimal solution. Zehmakan (2015) presented a graphical comparison of this ratio between algorithms over a defined number of problem instances. Thus, a better algorithm shows a lower ratio over the problem instances.

2.2.1.3 Feasible rate The feasible rate is an independent run that generates at least one feasible solution. This metric applies to both continuous and discrete problems, where it is equivalent to the number of feasible trials divided by the total number of trials as shown in Eq. (43). A higher feasible rate shows that more solutions reached the feasible region of the search space, thus denoting a better performance. This metric is one of the standard procedures for CEC competition (Suganthan et al. 2005; Liang et al. 2006). Other literature defined this term as feasibility probability (Mezura-Montes et al. 2010).

$$Feasible\ rate = \frac{Number\ of\ feasible\ runs}{Number\ of\ total\ run}. \tag{43}$$

2.2.1.4 Average number of function evaluation for optimality (AFESO) AFESO is determined based on the average of FEV of each successful trial that reaches close to the neighborhood of $f\left(x_{opt}\right)$ (Das 2018) with the formulation as:

$$AFESO = \frac{1}{SuR}\sum_{i=1}^{SuR} FEVs_i, \tag{44}$$

with SuR as the number of successful runs and $FEVs_i$ as the function evaluation at the ith run. A lower AFESO has better performance since it denotes a lower average cost required by the algorithms to reach the near-optimum solution.

2.2.1.5 Success rate and performance A successful run is an independent run with absolute difference between the best solutions $f(x)$ and optimum $f\left(x_{opt}\right)$ that less than a defined threshold value. Liang et al. (2006) suggests a success condition with $f(x) - f\left(x_{opt}\right) \leq 0.0001$. The success rate and success performance are defined as in (45) and (46). Note that FEV is the number of function evaluations. Both measures are standard procedure for CEC competition (Suganthan et al. 2005; Liang et al. 2006).

$$SR = \frac{Number\ of\ succesfull\ runs}{Number\ of\ total\ run}, \tag{45}$$

$$SP = \frac{(Mean\ FEV\ of\ succesfull\ run)(number\ of\ total\ run)}{Number\ of\ succesfull\ run}. \tag{46}$$

Note that SR is also used in the formulation of an efficiency metric (Mora-Melia et al. 2015) as defined in Eq. (8) that represents the effectiveness rate of SR over FEVs. The SP metric from Eq. (46) can also be evaluated with the combination of the probabilities of convergence with $AFESO$. The metric estimates the speed and reliability of the algorithm (Mezura-Montes et al. 2010), a lower SP denotes a better combination of speed and consistency, thus reliability of the algorithm as shown in the following equation.

$$SP = \frac{AFESO}{P}, \tag{47}$$

with P as the probability of convergence, this is calculated by the ratio of the number of successful runs to the total number of runs.

2.2.1.6 Scoring rate of best to worst solution A relatively similar measure as the success rate is the score of the best concerning the worst converged solution or simply *best solution*/*worst solution*. This metric describes the improvement ratio of the algorithm and generally indicates the coverage of solutions generated by the algorithm in its specific problem space. Such a metric can be usually applied in the design and combinatorial optimization problems in order to compare the potential capability of the algorithm in solving the problem. The algorithm with a higher score is denoted as poorer compared to the lower ratio. Examples such as Adekanmbi and Green (2015) and Lee et al. (2019a) utilize a ratio

of best to worst solution as an indicator of algorithm improvement in engineering optimization and water distribution problems. Another example is Santos et al. (2019) in the combinatorial Bin-packing problem that evaluates the ratio between the best and worst solutions of total bins. The average and variation of this ratio may also indicate the effectiveness of the algorithm over the others for n trials of solutions.

Another related quality indicator is the scaled median and scaled average performance as described by Wagner et al. (2018). In this method, the best and worst objective scores are defined as boundaries of solutions interval and the actual scores are mapped within [0, 1] with 1 associated as the highest score. The scaled performance is quite similar to the median plot that normalized within [0, 1]. This metric has a better overview when the comparison is carried out with many algorithms.

2.2.2 Measures of profile

This section discusses several performance metrics based on the profile of an algorithm. There are mainly six performance metrics based on characteristics or profiles proposed in the literature. Each method has an essentially similar perception of indicator but with slight differences.

2.2.2.1 Performance profile (Dolan and Moré 2002) The performance profile is defined with computational cost $t_{p,s}$ that obtained in each pair of problems and algorithms. The method illustrates a percentage of problems solved by computation cost that can be referred to as time, FEVs, or other cost-related units. Thus, larger $t_{p,s}$ reflects the worst performance. Then a performance ratio that is proportional to the computation cost is defined as:

$$r_{p,s} = \frac{t_{p,s}}{min\{t_{p,s} : s \in S\}}, \tag{48}$$

with s as the solver (or algorithm) and S as the set of algorithms. In other words, the performance ratio is identified by dividing the computation time of an algorithm by the minimum computation time returned from all algorithms. Then the performance profile of solver s is defined as:

$$\rho_s(\tau) = \frac{1}{|P|} size\{p \in P : r_{p,s} \leq \tau\}, \tag{49}$$

with $|P|$ as the number of elements of the test set P and $\rho_s(\tau)$ as the portion of time that corresponds to the performance ratio $r_{p,s}$ of algorithm $s \in S$ within $\tau \in \mathbb{R}$. Note that $\rho_s(\tau)$ is the cumulative distribution function of the performance ratio. The best algorithm $s \in S$ is represented by a higher value of $\rho_s(\tau)$. The performance profile $\rho_s(\tau)$ compares different algorithm versus the best algorithm that has the highest $\rho_s(\tau)$. Some examples of performance profiles can be observed in Beiranvand et al. (2017), Monteiro et al. (2016) and Vaz and Vicente (2009). To derive the second best algorithm, a chart of performance profile needs to be drawn without the first best performer. Some drawbacks are that the criteria for passing and failing of convergence are flexible that may change the profile itself. Furthermore, the performance profile did not provide sufficient information for expensive optimization problems (Moré and Wild 2009). The main advantage of this method is it combines speed and success rate in one graphical form. The main information gained from the performance profiles is to show how the proportion of solved solution increases by

increasing of performance ratio. Liu et al. (2017b) improved the profile with the confidence interval by adding the upper and lower bound of the confidence interval in the profile chart to observe the variances generated by the algorithms.

2.2.2.2 Operational characteristic and operational zones (Sergeyev et al. 2017) The operational zone is a further development of performance profile. The number of successful solutions within several trials can be represented graphically using the operational zone (Sergeyev et al. 2017). The method was originated from the idea of operational characteristic for comparing deterministic algorithms introduced by Grishagin (1978). The operational characteristic is a non-decreasing function that indicates the number of problems solved after each FEV within the total number of trials. The operational zone consists of n operational characteristics performed by the metaheuristic algorithm with n as the total number of trials since a set of n operational characteristics resulted in n patterns or zone of the metaheuristic solutions. The zone is then extracted with upper- and lower bounds, representing the worst- and best case solutions and an average of operational characteristic for all runs can be estimated. The graphical construction of operational characteristics and the operational zone is shown in Fig. 9 (inspired by Sergeyev et al. 2018). The operational zone with average characteristics can be used to compare several algorithms in each chart. The example of operational zones in Fig. 9 consists of n number of trials from an algorithm that shaded between two red curves, these red curves represent the best (upper boundary) and the worst (lower boundary) trial of the measure algorithm. Then an average of the algorithm performance that relates the number of solved problems concerning the number of trials can be estimated as a middle line (represented in the blue curve). The quality of compared metaheuristic algorithms can be compared based on the average line and the size of the shaded area, which represents the total n trials of that particular algorithm. The lower the size denotes the lesser variance of solutions and better reliability of the algorithm. As an example, if the lower bound of algorithm b is higher than the operational zone of algorithm a, then it can be concluded that algorithm b outperformed algorithm a. The measurement of this method also shows no significant difference if the algorithms are executed with a different number of trials (Sergeyev et al. 2018).

2.2.2.3 Data profile (Moré and Wild 2009) It is a modified version of the performance profile for a comparison of derivative-free algorithms (Moré and Wild 2009). The information from the data profile shows the percentage of problems solved in a given tolerance of τ time within the budget of k FEVs. It is assumed that the numbers of FEVs are increased by a

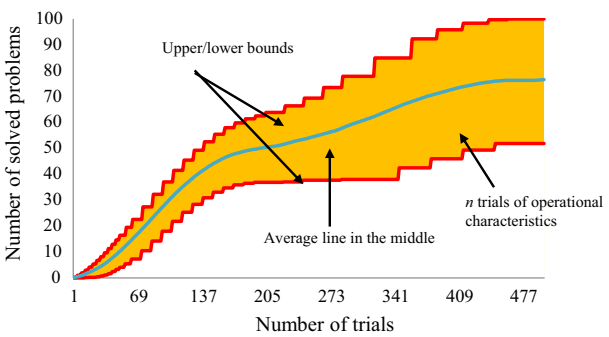


Fig. 9 Operational characteristics and operational zone

higher number of variables to satisfy the convergence criteria. The data profile is suitable for optimization problems particularly with a high computational burden and is defined as follows:

$$d_s(k) = \frac{1}{|P|} size\left\{ p \in P : \frac{t_{p,s}}{n_p + 1} \leq k \right\}, \tag{50}$$

where $t_{p,s}$ as the number of FEVs required to satisfy the convergence test, n_p is the number of variables in the problem $p \in P$, and d_s as the percentage of solved problems in $k\left(n_p + 1\right)$ FEVs. The data profile approximates the operational characteristic if the term $\frac{t_{p,s}}{n_p+1}$ is replaced by the number of iterations. Some examples of a data profile can be observed in Beiranvand et al. (2017) and Hellwig and Beyer (2019). In a nutshell, the data profile shows how the proportion of solved solution increases by increasing the relative measure of the computational budget. Liu et al. (2017b) improved the data profile with a confidence interval to observe the variances generated by the algorithms.

2.2.2.4 Accuracy profile (Hare and Sagastizábal 2006) The accuracy profile (Hare and Sagastizábal 2006) is designed for a fixed cost dataset. Fixed-cost is referred as a final optimization error $f(x) - f\left(x_{opt}\right)$ that is fixed after running the algorithm for a certain period, the number of FEVs, or iterations. The accuracy profile is based on the accuracy measure of fixed-cost datasets as follows:

$$\gamma_{p,s} = \begin{cases} -f_{acc}^{p,s}, & -f_{acc}^{p,s} \leq M, \\ M, & -f_{acc}^{p,s} > M, \end{cases} \tag{51}$$

with $\gamma_{p,s}$ as the accuracy measure, $f_{acc}^{p,s} = \log_{10}\left(f\left(\bar{x}_{p,s}\right) - f\left(x_{popt}\right)\right) - \log_{10}\left(f\left(x_p^0\right) - f\left(x_{popt}\right)\right)$, $\bar{x}_{p,s}$ is the solution obtained by algorithm s on problem p, x_{popt} is the optimum solution and x_p^0 is the initial point of problem p. The term $f_{acc}^{p,s}$ in the equation above is interpreted as negative since the improvements from starting $f\left(x_p^0\right)$ towards global optimum shall be decremented. The performance of the accuracy profile is then calculated as in Eq. (52).

$$R_s(\tau) = \frac{1}{|P|} size\left\{ \gamma_{p,s} | \gamma_{p,s} \geq \tau, p \in P \right\}, \tag{52}$$

The accuracy profile $R_s(\tau)$ is a proportion of problems that algorithm $s \in S$ able to solve within an accuracy of τ of the best solution. Some examples of an accuracy profile can be observed in Beiranvand et al. (2017).

2.2.2.5 Function profile (Vaz and Vicente 2009) It is a modified version of the data profile that shows the number of FEVs required to achieve some level of global optimum (Vaz and Vicente 2009). The reason for the modification is due to the characteristic of stochastic algorithms that did not necessarily produce a sequence monotonically decreasing towards best value (Vaz and Vicente 2009). The function profile is formulated as follows:

$$\rho_s(\Upsilon) = \frac{1}{|P|} size\left\{ p \in P : r_{p,s} < \Upsilon \right\}, \tag{53}$$

with $r_{p,s}$ as the average number of FEVs taken by algorithm s to solve problem p for $p \in P$ and $s \in S$. The value of $r_{p,s}$ is condition-based, $r_{p,s} = +\infty$ (denote as failure) if the algorithm is unable to find a feasible solution of problem p within a defined relative error ε with $\left(f_{p,s} - f_{p,L}\right) / \left|f_{p,L}\right| > \varepsilon$. The value $f_{p,s}$ is the objective function obtained by algorithm s on problem p and $f_{p,L}$ represents the best objective function obtained by all algorithms for problem p. The value $\rho_s(\Upsilon)$ is equal to the function profile of algorithm or solver $s \in S$ as a fraction of problems where the number of the objective function is lower than Υ.

2.2.2.6 Performance metric based on stochastic dominance (Chicco and Mazza 2019) This metric is generally defined as the Optimization Performance Indicator based on Stochastic Dominance, OPISD. The method is based on the first-order stochastic dominance of CDF between two algorithms. The first-order stochastic dominance is defined as follows: the solutions obtained by an algorithm A has the first-order stochastic dominance over the solutions obtained by algorithm B if and only if the CDF of any solutions obtained by algorithm B lie on the right side of CDF from solutions of algorithm A as shown in Fig. 10 and expressed mathematically as follows:

$$F_A^{(H)}(y) \geq F_B^{(H)}(y), \tag{54}$$

with H as the number of solutions from each algorithm and F as the CDF constructed for each algorithm A and B sorted in ascending order of variable y.

The OPISD metric modifies the dominance formulation with $\mathrm{F}_{\mathrm{ref}}^{(\mathrm{H})}(\mathrm{y}) \geq \mathrm{F}_{\mathrm{algo}}^{(\mathrm{H})}(\mathrm{y})$ with $\mathrm{F}_{\mathrm{ref}}^{(\mathrm{H})}(\mathrm{y})$ as a reference CDF and $\mathrm{F}_{\mathrm{algo}}^{(\mathrm{H})}(\mathrm{y})$ as the CDF of the algorithm. The reference CDF is constructed differently based on either with known or unknown global optimum. With known global optimum, the reference CDF is fixed and calculated in absolute term, whereas in unknown cases, the entry of reference CDF may vary depending on H number of solutions and on the context of calculation such as computation time limit. The metric OPISD is then defined by firstly calculating the area between the reference CDF and the algorithm CDF as follows:

$$\mathrm{A}_{\mathrm{A,G}}^{(\mathrm{H})}(\mathrm{y}) = \frac{1}{\mathrm{H}} \int_{\mathrm{z}=0}^{\mathrm{H}} \left(\mathrm{y}_{\mathrm{A}}^{(\mathrm{H})}(\mathrm{z}) - \mathrm{y}_{\mathrm{G}}\right), \tag{55}$$

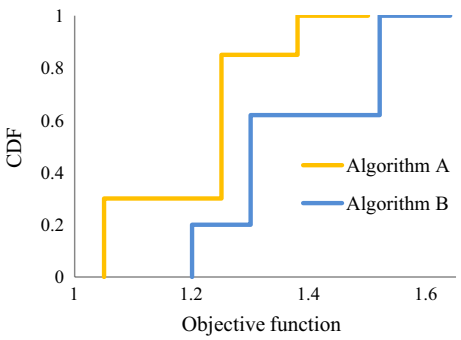


Fig. 10 First-order stochastic dominance

$$A^{(H)}_{A,R}(y) = \frac{1}{H}\int_{z=0}^{H}\left(y^{(H)}_{A}(z) - y^{(H)}_{ref,R}(z)\right), \tag{56}$$

with $A^{(H)}_{A,G}(y)$ referred to as the area calculation for known global optimum, where it depends on the absolute y_G and $A^{(H)}_{A,R}(y)$ for unknown optimum that depends on the relative change of reference,$y^{(H)}_{ref,R}(z)$. The OPISD metric is then determined by each absolute (known optimum) or relative (unknown optimum) as in Eqs. (57) and (58) respectively.

$$OPISD^{(H)}_{G} = \frac{1}{1 + A^{(H)}_{A,G}}, \tag{57}$$

$$OPISD^{(H)}_{R} = \frac{1}{1 + A^{(H)}_{A,R}}. \tag{58}$$

The OPISD value is then ranked by the different algorithms and higher value denotes better performance as the metric is inversely proportional to the area between the algorithm's CDF and the reference CDF.

2.2.3 Scalability

The algorithm efficiency reduces as the dimensionality of search space increases. The main reason for this reduction is due to the rapid growth of the search space hyper-volume by a higher number of dimensions, which then reduces the convergence speed of most of the algorithms. Some literature presents the scalability measure that plots the average error and standard deviation of the algorithm's solutions over the dimensions with a fixed limit of FEVs (Kim et al. 2016). Others present the algorithmic performance with the number of dimensions individually. A more comprehensive and simplified method is presented by Das et al. (2009). The number of FEVs required by algorithms to reach the optimum solution within a defined threshold is plotted with respect to the number of dimensions as shown in Fig. 11 (inspired by Das et al. 2009). The plot shows an increasing trend of required FEV by higher dimension, thus a lower inclined curve shows better robustness of algorithm against dimensionality.

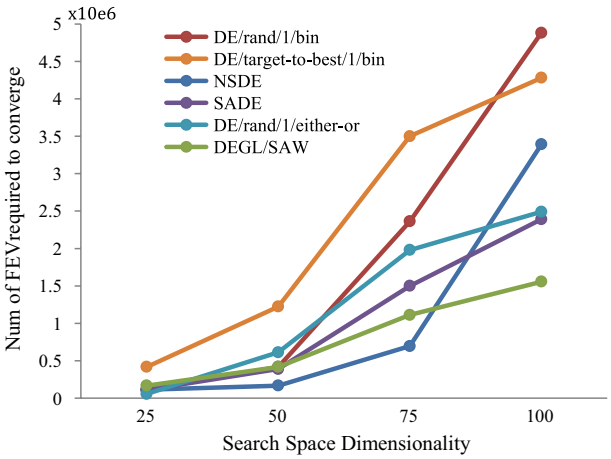


Fig. 11 Scalability measure of algorithms (adopted from Das et al. 2009)

Based on the figure, it is clear to indicate that DEGL/SAW has the best performance against the dimensionality range of [10; 50; 75; 100] compared to other algorithms.

The dimensionality can also be compared against the population size particularly related to the large scale global optimization (LSGO) problem (Li et al. 2013). The performance comparison can be measured in either constant or with dynamic size such as ($n = 2d$) as demonstrated by Bolufé et al. (2015) in multiple dimensions d=5, 10, 50, 100, and usually tabulated in increasing order of dimensions by each problem. Inspired from the idea of fitness difference as demonstrated by Bolufé et al. (2015) that measure the average relative performance between two algorithms as follows:

$$Diff\% = \frac{\left(\bar{f}_{algorithm_a} - \bar{f}_{algorithm_b}\right)}{max\left(f_{algorithm_a}, f_{algorithm_b}\right)}, \tag{59}$$

with $\bar{f}_{algorithm_a}$ and $\bar{f}_{algorithm_a}$ as the average fitness of algorithms a and b respectively and divided by the maximum fitness difference between both. A negative value denotes that algorithm a performed better than the others. Using this formulation, a database of fitness differences with respect to dimensionality can be measured by calculating the Eq. (47) over multiple dimensions. Then a relationship curve of relative performance with the number of dimensions can be plotted.

2.2.4 The 100-digits accuracy

It is a unique performance evaluation based on the digits of the solution. The method was proposed by Trefethen (2002) in conjunction with the Society for Industrial and Applied Mathematics (SIAM) with the idea to test high accuracy computing of optimization algorithms. The evaluation is denoted as the 100-Digits challenge designed to solve 10 hard problems to 10 digits of accuracy. Each correct digit is awarded one point making the maximum score of 100. A similar 100-Digit challenge is also proposed currently in CEC 2019 (Price et al. 2018) with the same concept of 10 test functions to compute each function minimization to 10 digits of accuracy without being limited by time. The 10 problems shall be solved with one algorithm with a limited tuning of the control parameter for each function. The criteria for 10-digits accuracy is defined as follows: suppose the optimum solution is 1.000000000, thus a solution of 1.924235666 denotes a one-digit accuracy (‘1’) and solution of 1.003243567 denotes a solution accuracy of 3 correct digits (‘1.00’). Each test function is limited to predefined dimensions and range by the CEC organization. The score for each function is the average number of correct digits in the best of 25 out of 50 trials. The results for each test function optimization shall be recorded in a standard requirement by CEC 2019 with: the number of FEVs that each trial utilized to reach 1, 2,…, 10-digit level of accuracy, a table with the number of trials in a total of 50 that found n correct digits, an average number of correct digits in the best 25 runs and the total score (the sum of scores of all 10 functions) also need to be included along with the values of two control parameters that are used for tuning for each function.

2.2.5 Statistical analysis

Another crucial measure of algorithm performance is statistical analysis. It is essential to include a comprehensive statistical comparison among the algorithms to generalize the effectiveness of exploration and exploitation and to deduce a conclusion on which

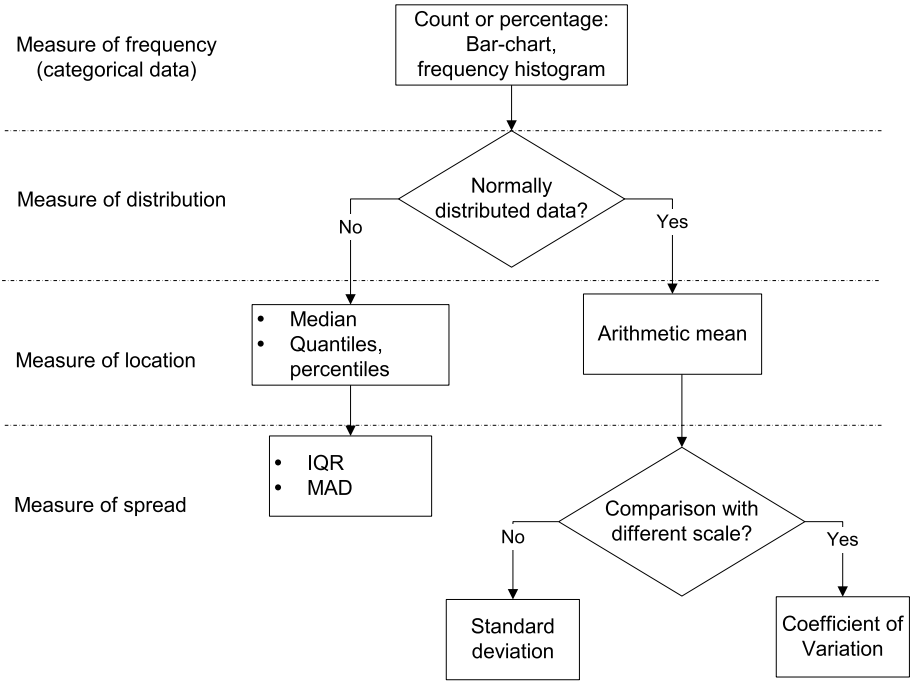


Fig. 12 Univariate descriptive statistics

algorithm performed better in a specific problem. This method is highly recommended to ensure that conclusions obtained from the corresponding tests are not biased by the researcher's intention or by chance. Thus, correct descriptive and inferential statistics are important to ensure that the conclusion been made are correct.

2.2.5.1 Descriptive statistic The descriptive statistic describes the raw data in a more meaningful way with simple interpretation and can be divided into four clusters that include the measure of frequency, distribution, central tendency, and spread of the data. Several numerical statistical indices usually used for each cluster is summarized in Fig. 12 that divides the type of data into normal and non-normal type The indices can be represented in a graphical form such as simple bar-chart or tabular form, which most of the literature usually highlight the best value of each statistical index among the algorithms. A standard procedure defined by IEEE Congress on Evolutionary Competition, CEC (Suganthan et al. 2005) include termination algorithm by error $f(x) - f_{optimum} = 10^{-8}$ with descriptive of best value, mean, median, worst, and standard deviation.

Most of the literature presented their algorithms' capability through univariate statistics that cover at least some or all of the measures summarized in Fig. 12. The frequency measure describes the number of global optimum or the frequency of optimum solutions within a defined threshold found by the algorithm. In combinatorial problems, the count of the optimum solution is a type of frequency measure that can be presented in tabular form or simple graphical form. Some examples from literature such as Zhou et al. (2019) for comparing four algorithms on six TSP instances, Cubukcuoglu et al. (2019) for comparison of 10 instances QAP with various island-based algorithms and Fomeni et al. (2020)

summarized the number of solved instances for Quadratic Knapsack Problem. In a Bin packing problem, general information of performance is shown by the number of instances for which optimal solutions are found over the total bins, as presented by Santos et al. (2019). The total bins are referred to as the total number of required bins for each algorithm. Other examples such as the number of solved TSP instances in minimum time or gap as presented by Campuzano et al. (2020) and the frequency of solved TSP instance within n number of trials (Silva et al. 2020) are also considered as a piece of good information for the algorithm effectiveness.

The distribution measure is essential for understanding the data type represented by the algorithm solutions, which then lead to the decision of selecting the appropriate method of measuring central tendency and spread of the solutions. For normally distributed data, the average of solutions can be calculated using the arithmetic mean with $\bar{x} = \sum_{i=1}^{n} x_i/n$, whereas for non-normal data is calculated with median instead. The median is the middle value of an ordered statistics, which is referred to as the sequence of data in non-decreasing order $x_1 \leq x_2 \leq \cdots \leq x_n$ with x_1 and x_n as the smallest and largest value respectively. In mathematical formulation, the median is defined as:

$$\tilde{x} = \begin{cases} x_{\left(\frac{(n+1)}{2}\right)}, & if\ n\ =\ odd, \\ \frac{\left(x_{\left(\frac{n}{2}\right)} + x_{\left(\frac{n}{2}+1\right)}\right)}{2}, & if\ n\ =\ even. \end{cases} \tag{60}$$

The median also corresponds to the 0.5 percentiles of data and is more resistant against outliers as compared to mean. The quantiles or percentiles divide the data into 100 equally scale, while deciles divide the data into 10 equally scaling such that 10% of the data represents D_1, 20% of data for D_2 and so forth. Another data fraction index for location is the quartiles that divide data into four equal divisions, such that 25% of the data represents Q_1, 50% for Q_2, 75% of data for Q_3 and the last quartile with Q_4. These data fraction indicator is an alternative of location measure that gives the researcher more insights on the percentage of data concerning the location associated with the probability of finding the solution. For random variable x from a population, the quantiles q_p is defined as $P(x \leq q_p) \geq p$ and $P(x \geq q_p) \leq 1 - p$. In other words, the probability of achieving a solution that equals or outperforms q_p is greater than or equal to p (Ivkovic et al. 2016). Unlike the arithmetic mean, the quantiles also can be used to find out the percentage of quality for an algorithm if it is unable to solve a solution with probability r with the condition $p < 1 - r$. The quantiles measure for location has the advantage of interpreting the solutions over a range of algorithm executions. Some references on using these measures can be found in Ivkovic et al. (2016) and Woolway and Majozi (2019) as well as Pierezan et al. (2019). The measure of the spread of normally distributed data by using the standard deviation is the most widely accepted method among the researchers. The standard deviation is defined as the root mean square of deviation as shown in the following equation:

$$s = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} \left(x_i - \bar{x}\right)^2}, \tag{61}$$

where x_i as the observed individual value of sample data,$\bar{x}$ as the mean of sample data for n number of samples. The variance is equal to the square of standard deviation. The standard deviation is an appropriate variability measure for comparison of different algorithms that

are defined in the same units and with nearly equal means. However, it is not appropriate to benchmark standard deviation if the means between the algorithms are relatively large or each algorithm is defined with different units. The suitable method for such comparison is by measuring the coefficient of variation, COV that determines the relative dispersion of algorithms that expressed as follows:

$$COV = \frac{s}{Mean} \times 100. \tag{62}$$

The COV is a unit-free measure and usually presented as a percentage since it measures the ratio of dispersion, s over mean. This measure is suitable if one to compare the variability of the proposed algorithm with other algorithm's solution presented in other literature. An example is from Boryczka and Szwarc (2019) that used COV as an additional indicator of algorithm effectiveness for Asymmetric TSP. For non-normal data, the spread can be determined based on the inter-quartile range, IQR that corresponds to the middle half of the data between Q_3 and the Q_1 as shown in Eq. (63). This is also equivalent to the box range of the box plot as illustrated in Fig. 13. IQR is a robust estimate of data spread.

$$IQR = Q3 - Q1. \tag{63}$$

Another useful dispersion measure for non-normal data is the median absolute deviation, MAD (Hampel 1974) defined as follows:

$$MAD = \frac{1}{n} \sum_{i=1}^{n} |x_i - m|, \tag{64}$$

where n is the number of population, x_i is the individual value and m is the median. If the underlying distribution is approximately normal with large data, then the formulation can be simplified by multiplication of a constant 1.483 as given as in the following equation and denoted as scaled MAD or MADe (Ellison et al. 2009).

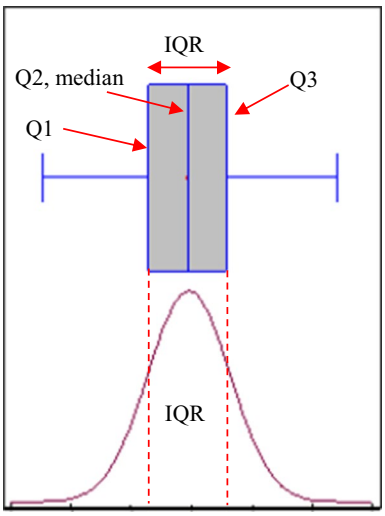


Fig. 13 IQR representation

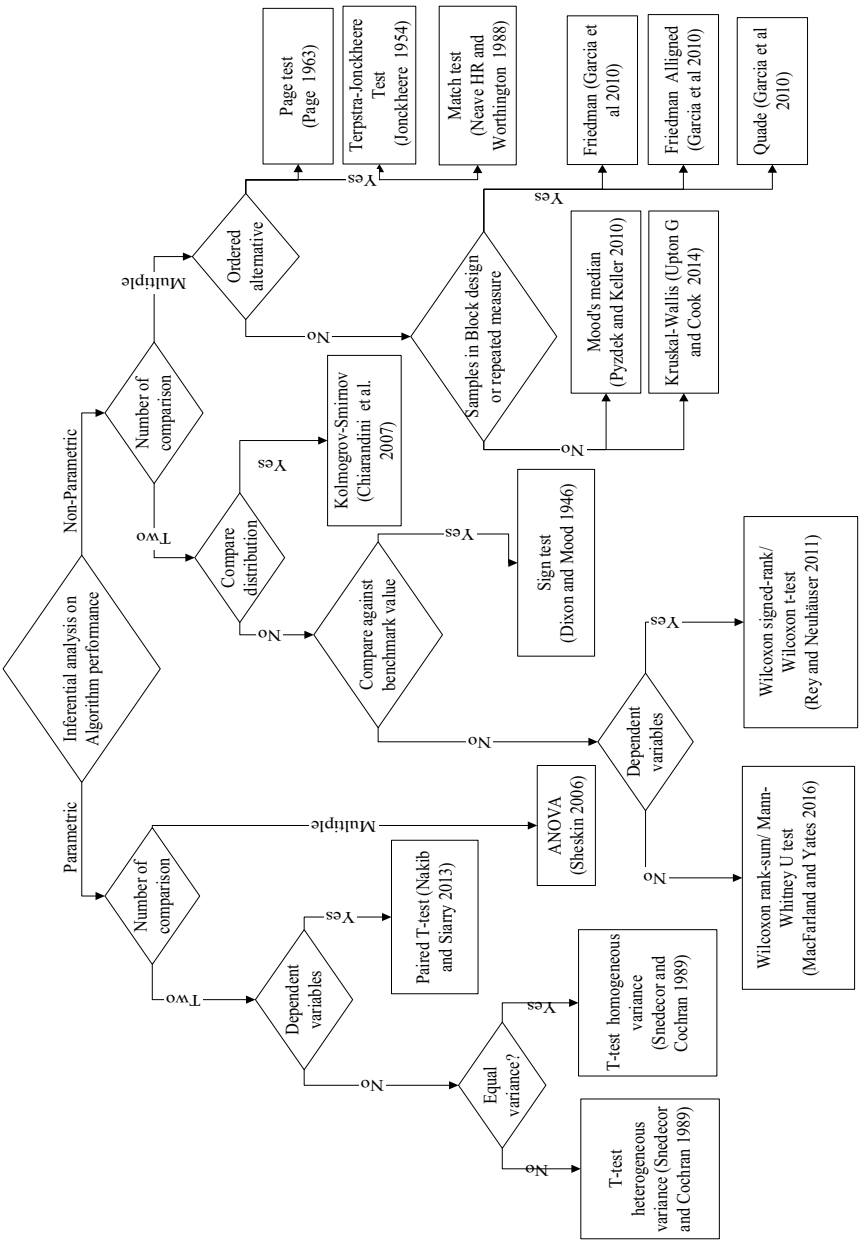


Fig. 14 Flow chart of options parametric and non-parametric test

$$MADe = 1.483MAD. \tag{65}$$

The constant 1.483 is a correction factor to make MAD unbiased to the normal distribution. The MADe method is a robust estimator and not unduly affected by extreme values even though few data points make the distribution to be skewed (Olewuezi 2011).

2.2.5.2 Inferential statistic The inferential statistic is another scope of statistical measure where it is important to make generalizations about the relationship of sample data from each algorithm and to observe the differences between them. In regard of modern performance comparison between optimization algorithms, this branch is mainly divided into two categories, which are the frequentist and Bayesian tests (Carrasco et al. 2017, 2020) The frequentist is considered as a 'classical approach' that further divided into parametric and nonparametric tests with decision scope of hypothesis tests, whereas the Bayesian test concerns on prior and posterior distributions based on the probability of both conditions. Figure 14 summarizes the suitable method for frequentist statistical analysis that depends on the type of data (parametric and nonparametric) and the number of comparisons (two or multiple sets or ordered alternative).

There are three main attributes for parametric tests: independence of data, normality, and Heteroscedasticity (Sheskin 2006). Under theoretical analysis, if one or more of the assumptions are not met, a non-parametric test is the appropriate alternative for the comparison analysis. The selection of tests under both parametric and non-parametric methods is depending on the number of n algorithms being compared; multiple comparisons are the option for $n > 2$ and two-sample comparisons otherwise. As shown in Fig. 14, the parametric two samples comparison (also denoted as t test) are further divided into paired t-test and independent t-test. The paired t-test (Nakib and Siarry 2013) is appropriate if the variables are dependent on each other. Examples of using this condition are by comparing the same algorithm on two problem instances or comparison of previous and an improved version of the same algorithm. The selection of independent t-test is also divided into two criteria that depend on the homogeneity of variance between both samples (Snedecor and Cochran 1989). The t-test determines the differences between the two population means by comparing sample standard deviations and means. Even though this test requires normality conditions, it is fairly robust to violation of assumption when the sample sizes of both samples are equal or greater than 30. Another mostly implemented parametric multiple comparison test is the analysis of variance or ANOVA that compares the number of variables and determines whether significant differences between variances and means are observed. Generally, ANOVA is suitable if three main conditions of parametric are met. However, findings from Blanca et al. (2017) suggest that ANOVA is still suitable for the non-parametric type of data due to the robustness of the F-test that control Type I error if the distributions have values of skewness and kurtosis ranging from -1 to $+1$. The Type I error is referred to as the error committed if the null hypothesis,h_0 is rejected when it is true for acceptance. ANOVA determines the differences by computing the F-ratio, a ratio of variation between and within data with a posthoc test. Among the popular posthoc tests are LSD, Tukey-HSD, Scheffe, Bonferroni, SNK, and Duncan.

On the non-parametric counterparts, the selections of two sample comparisons rely on the type of comparison. The first option is whether to compare the distribution of the data, such as comparing the ECDF of two algorithms. This is can be done by using the Kolmogorov–Smirnov test (Chiarandini et al. 2007). This method considers the maximal difference between each cumulative distribution and the distribution statistic is determined

by permutation methods (Conover 1980). Also, this test can show whether there exists a *statistical dominance* between two distributions and able to determine general differences instead of the location differences such as mean or median (Chiarandini et al. 2007). The second selection is whether to compare the algorithm against its benchmark solution, which can be analyzed using a sign test (Dixon and Mood 1946). In essence, the sign test is a one-sample test of median that can be used to compare: (1) the algorithm solution against the known global optimum value (hypothesized value), (2) previous and improved version of the algorithm, and (3) for ordered categorical data of rank that is not based on a numerical scale such as non-numeric quality measure comparison of an algorithm to the reference. Further non-parametric two-sample comparisons are the Wilcoxon rank-sum test (or also defined as Mann–Whitney U test) and the Wilcoxon signed-rank test (also defined as Wilcoxon t-test). The former method is suitable for independent samples, whereas the second variant is otherwise. The Wilcoxon signed-rank test (Rey and Neuhäuser 2011) is a pair-wise test to detect significant differences between the two samples, which are reflected in two algorithms. The method is more powerful even under the appropriately paired t-test (García et al. 2009) and the advantage of this test is no dependency of Type I error on the assumption of population normality (Fahoome 2002). This test is suitable to compare the previous and improved versions of the same algorithm. The Wilcoxon rank-sum test compares two sample medians that correspond to unpaired t-test (MacFarland and Yates 2016). This method is more powerful compared to the sign test for comparison of ordered categorical data.

The non-parametric multiple comparisons in Fig. 14 comprises two methods of comparisons: ordered variables and comparison between samples. For ordered variables comparison, three methods have been introduced and implemented in algorithm comparisons, which are the Page test, Terpstra–Jonkheere test, and Match test. These methods are appropriate for comparing the convergence performance between algorithms by evaluating the differences among each algorithm's best value at several points of the search. In contrast to the multiple sample comparisons such as Mood's median (MM), Kruskal–Wallis (KW), Friedman and Quade tests, the ordered alternatives compare several populations with an ordered hypothesis as an extension of the one-sided test. The multiple sample comparisons are appropriate for a general alternative where at least two independent populations differ in averages (either mean or median depending on parametric or non-parametric tests). This test did not identify the pairwise group differences or the trend of these differences, whereas the ordered alternative specifies the order of differences or trends among groups (Fahoome 2002).

The non-parametric multiple sample comparisons are further divided into either with or without block design or repeated measures. Under this condition, the MM and KW are appropriate for independent samples, and especially KW is analogous to the one-way ANOVA of the parametric test. The MM test is used to determine whether the medians of all n algorithms are equal. It is more robust against outliers than KW-test and is appropriate for the preliminary stage of analysis but is less powerful (due to wider confidence interval) for analyzing data from several distributions including normal distribution and data should only include one categorical factor. Other options such as the Friedman test, Friedman aligned test, and Quade test are suitable for k related samples and analogous to the two-way ANOVA. The Friedman test is quite similar to KW-test except it is for randomized block design or repeated measure. The observed values within each block are replaced by the equivalent ranks. The Friedman test differentiates the algorithms by its sum of ranks, whereas KW-test differentiates between algorithms by the average rank. The Aligned Friedman Ranks is based on an aligned performance by firstly evaluating the

average performance that is denoted as the value of location. Then the differences between each algorithm's performances with the value of the location are obtained and repeated in each combination of algorithms and problems: referred to as aligned observation or score. Another possible option for multiple samples is the Quade test. This method is unique due to the ranking method that based on the weightage such as different difficulty or differences registered in the algorithm performance, while other methods (discussed before) are based on similar ranking size. It is to be noted that most of the recent combinatorial optimization problems reasoning their comparison in the non-parametric comparisons such as the Sign test (Dahmani et al. 2020) and Wilcoxon test (Akhand et al. 2020; Ali et al. 2020; Zhong et al. 2019).

2.2.6 Bayesian test

The alternative to hypothesis testing is the Bayesian test. The application of the Bayesian statistical method in algorithm performance analysis is relatively small compared to traditional hypothesis test or frequentist test. Some of the applied Bayesian methods in the literature are (1) correlated Bayesian t-test, (2) Bayesian sign test, (3) Bayesian Wilcoxon signed-rank test and (4) Bayesian Friedman test. The correlated Bayesian t-test can be implemented to compare two algorithms on multiple data sets (Corani and Benavoli 2015). The test considers the correlation among data of both algorithms with a covariance matrix and assumes a normal-Gamma distribution as prior distribution of the difference between algorithms and Student distribution as the posterior distribution of the parameter. The possibility of no significant difference between algorithms is determined by a region of practical equivalence (denoted as *rope*) and it is defined with bounds of $[r_{min}, r_{max}]$. The probabilities of the algorithms' relationships are derived based on *rope*. An example of a relationship is $P(algo_A = algo_B) = P(\mu \in rope)$ or $P(algo_A << algo_B) = P(\mu < r_{min})$. The analysis of probabilities is straightforward and the limits can be varied according to the circumstances.

The Bayesian sign test is a non-parametric comparison based on the Dirichlet process (Benavoli et al. 2014). The inference for Bayesian sign test is constructed by firstly, develop the posterior probability density function as a linear combination of Dirac's delta centered on the observation with weights (derived from Dirichlet distribution), then the posterior probability function is approximated as a posterior probability of the belonging parameter to each region of interest (Carrasco et al. 2017, 2020).

The Bayesian Wilcoxon signed-rank test consider two independent observations Z and Z', where Z is the difference between paired data (or two algorithms solution) and $Z \geq Z'$ (Benavoli et al. 2014). Assuming that both observations come from F cumulative distribution, then the Bayesian Wilcoxon signed-rank test is approximated with prior and posterior distribution $F \sim DP(\alpha, G_0)$ and $F \sim DP(\alpha_n, G_n)$ respectively with n as the number of observations Z fall in a defined area. The posterior distribution is obtained by sampling the weights of DP (Carrasco et al. 2017).

3 Multi-objective optimization algorithms

Unlike single-objective, the goal of multi-objective optimization is to find the best solutions that comprehend good feasible solutions of n objective functions, with $n \geq 2$. Mathematically speaking, the multi-objective problem, MOP can be formulated as follows (Ehrgott 2005):

$$\min/max f_1(x), f_2(x), \ldots, f_n(x), x \in U, \tag{66}$$

with x as the solution, n as the number of objective functions, U as the feasible solutions, $f_n(x)$ as the nth objective function and *min/max* as the combination of objectives. Generally, different objectives contradict each other and a compromised solution needs to be accepted. This also resulted in an infinite number of optimal compromises solutions, which is denoted as Pareto set. The corresponding points that performed the best solution among Pareto set are defined as the Pareto front. Two known options can be used for analyzing MOP, the first method is based on scalarization and the second method is Pareto-optimality (Weck 2004; Gunantara and Hendrantoro 2013; Peitz and Dellnitz 2018).

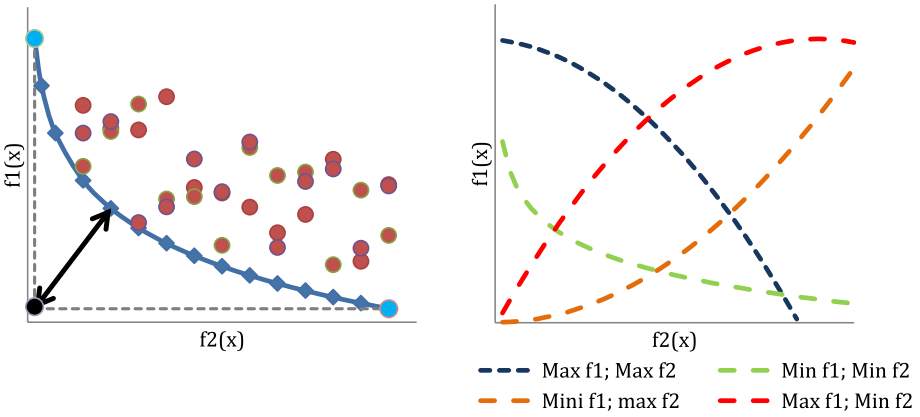
The scalarization method aggregates all MOPs into a scalar function incorporated with the fitness function (Gunantara and Hendrantoro 2013) and the function is formulated into a single solution using weights. A typical formulation of the scalarization method is based on the weighted sum approach as follows (Murata et al. 1996):

$$F(\boldsymbol{x}) = w_1 f_1(x) + w_2 f_2(x) + \cdots + w_n f_n(x), \tag{67}$$

with w_n as the weight and $f_n(x)$ the objective function of the nth objective respectively. The weights are defined before the optimization process and determine the solution of the fitness function. The assignment of weight is also dependent on the performance priority; larger weight value is denoted as a higher priority compared to the smaller value (Gunantara 2018). There are many types of weights formulation for scalarization methods such as equal weight, rank order centroid weight, rank-sum weight, ϵ-constraint method, Chebychev method, and boundary intersection method (Gunantara 2018; Emmerich and Deutz 2018). The performance of the scalarization method of multiple runs can be analyzed using cumulative distribution function, CDF as applied in practical example by Gunantara (2018). However the scalarization method usually suffers from shortcomings that include information relating to the original problem such as dominance relationship might be lost, and difficulty of choosing the right weighting scheme (Hale et al. 2020). Generally, all scalarization methods have in common that the Pareto set is approximated by a finite set of Pareto optimal points. The Pareto optimal points are computed by solving scalar subproblems. The second MOP options are the Pareto-optimality that aims to obtain the whole Pareto front without a combination of objective functions. Comparing both methods, the Pareto methods require a much longer time compared to the scalarization method. Gunantara and Hendrantoro (2013) found that the Pareto method takes 4.4 times more computation compared to the scalarization method. The main reason is that the Pareto method evaluates all the possible pairs in the optimization evaluation, whereas the optimization evaluation via the scalarization method is done randomly based on the number of populations and iterations.

The multi-objective problems consist of sets of solutions that need to be optimized concerning the constraints. This conception of optimality is closely related to the notion of dominance. A solution $\boldsymbol{x}$ dominates another solution vectors $\boldsymbol{x}'$, $[\boldsymbol{x}, \boldsymbol{x}' \in \vartheta]$ with ϑ contains all feasible solutions if it is at least as good as the latter for each objective and strictly better for at least one objective. Mathematically speaking, the vector $\boldsymbol{x}$ dominate $\boldsymbol{x}'$ is described as $\boldsymbol{x} \succ \boldsymbol{x}'$. Thus, a set of non-dominated solutions with respect to ϑ is defined as Pareto optimal set,P^*. The corresponding Pareto optimal set of the objective function then reveals the Pareto front of a problem.

The Pareto front is the result of mapping Pareto optimal set P^* to the objective space, Γ as shown in Eq. (68). A non-dominated solution set obtained by the algorithm should



(a) Dominated and non-dominated solutions **(b)** Two objectives with different goals

Fig. 15 Pareto optimal set representation

approximate the true Pareto front and able to cover the true Pareto front as widely as possible.

$$\mathrm{PF} = \{F(\boldsymbol{x}) \in \Gamma | \boldsymbol{x} \in P^*\}. \tag{68}$$

Generalization using PF is generally applicable in both continuous and discrete multi-objective problems. For discrete problems such as the Knapsack problem, the Pareto front is determined by plotting the non-dominated solutions of each knapsack in each axis, and the algorithm located to the furthest stretch is the best compared to others. The Pareto front can be plotted by different problem instances and item sizes. The notions of dominated, non-dominated solutions, and Pareto front, PF are shown in Fig. 15a. The set of dominated solutions are indicated with red points and the set of non-dominated solutions are indicated with blue points, PF is then derived with the blue curve.

Based on the figure, two important terms need to be noted, which are the anchor point and utopia point. The anchor point represents the best point of each objective function (displayed with light blue points), whereas the utopia point, f^0 (also denoted as an ideal point) signifies the intersection of the minimum of both objective functions (if both objectives aim at minimization) as shown in black point on the figure. Generally, the utopia point is not attainable in the Pareto set since it lies beyond the attainable area of both objectives. The next best-compromised candidate is the corresponding Pareto solution that lies as close as possible to the utopia point (denoted with an arrow in Fig. 15a. The closeness is implying with the minimum Euclidean distance from the utopia point to the corresponding PF as defined in the equation below (Chang 2015):

$$D(x) = \left\| f(x) - f^0 \right\| = \sqrt{\sum_{j=1}^{q} \left[f_j(x) - f_j^0 \right]^2}, \tag{69}$$

with f_j^0 as the component of the utopia point in the criterion space and $f_j(x)$ as the closest point on the Pareto front.

Table 5 Outperformance relations

Weak outperformance	*A* weakly outperforms *B* if all points in *B* are equal to or dominated by *A* with at least one point in *A* that not contained in *B*
Strong outperformance	*A* strongly outperforms *B* if all points in *B* are dominated by *A* and some points in *B* are dominated by point in *A*
Complete outperformance	*A* completely outperforms *B* if each point in *B* is dominated by point in *A*

The pattern of PF varies by the different combinations of objective functions as depicted in Fig. 15b. The figure illustrates PF of four different combinations of bi-objective problems with "Max" as a maximization problem and "Min" as a minimization problem respectively. There are numerous multi-objective measures proposed in the literature. Nonetheless, no single measure can entirely capture the total MOP performance as some of the metrics reveals the effectiveness and others related to the efficiency performance. Thus, appropriate measures must be selected upon MOP claims to ensure useful analysis and findings. Hansen and Jaskiewicz (1998) introduced quality aspects of MOP based on the approximation to the true Pareto front that defined as outperformance relations. This method shows a relationship between two sets of internally non-dominated objective vectors *A* and *B* as summarized in Table 5. The complete outperformance is the strongest, whereas the weak outperformance is the weakest of the relations.

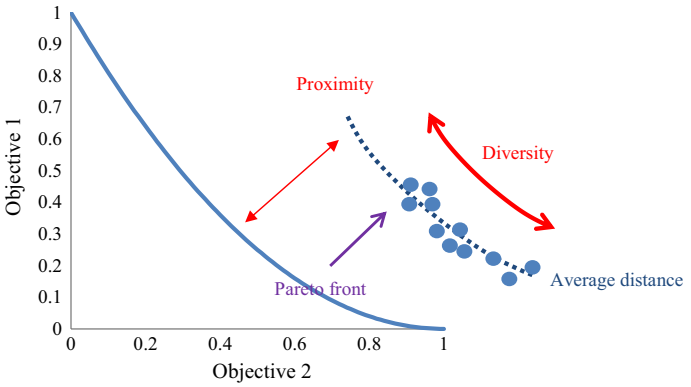


Fig. 16 Effectiveness measures for MOPs

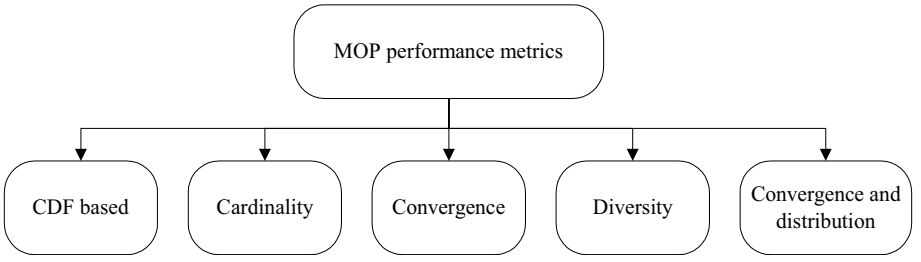


Fig. 17 The main categorization of performance metrics for MOP

The MOP metrics can be categorized in numerous ways. Earlier literature such as Zitzler et al. (2000) proposed three measurable goals: (1) minimization of the distance between non-dominated front to PF, (2) good distribution of solutions in objective space, and (3) maximization of the non-dominated front with a wide range of values by each objective. Later literature such as Okabe et al. (2003) categorized MOP measures in terms of cardinality, distance, volume, distribution, and spread. Coello et al. (2010) divide the performance measures into convergence, diversity, and hybrids indicators. Jiang et al. (2014) and Cardona and Coello (2020) categorized the metrics into capacity, convergence, diversity, and convergence-diversity measures. Riquelme et al. (2015) divide the metrics into (1) quality measure that includes cardinality, accuracy, and diversity metrics or (2) the number of approximation sets involved: unary and binary metrics. Audet et al. (2018) divide the categories into cardinality, convergence, distribution-spread, and convergence-distribution base metrics. All of the categories above cover mainly on the effectiveness of the algorithm in MOP with the objective of better diversity and proximity of solutions towards PF as shown in Fig. 16.

As a general conception and based on literature as discussed previously such as Audet et al. (2018), Cardona and Coello (2020). Requilme et al. (2015) and Jiang et al. (2014), the main categories of MOP metrics can be defined briefly as in Fig. 17.

The CDF based metric is referred to as the algorithm performance based on its cumulative distribution of solutions. The cardinality metrics are referred to as the number of the solution found by the algorithm. A higher number of solutions denote better performance. The convergence metric reflects the distance of solution sets towards PF. The diversity metric measures the distribution and the spread of the solutions. The convergence and distribution metric covers both performance indicators of distance and spread of the solutions towards PF. Since the general goal for MOP measures is the quality of solutions concerning the PF, there is no specific cluster for the effectiveness and efficiency category. Some of the metrics from the category in Fig. 17 did reflect the efficiency and mostly effectiveness of the algorithm's solution. The most applied performance metrics in MOP are summarized as the following sub-sections in chronological order, however, they should not be considered as a complete list. Each measure is clustered according to the category defined in Fig. 17.

3.1 Cumulative distribution for MOP

The cumulative distribution is one of the good measures for MOP performance due to the stochasticity of the algorithms and variation of solutions that may dominate or weakly dominate PF approximation in more than one objective. Two main measures under this category are the Empirical Attainment Function, EAF, and average Runtime Attainment Function, aRTA as described in the following sub-section.

3.1.1 Empirical attainment function, EAF (Fonseca and Fleming 1996)

The attainment function is a generalization of ECDF that is used for analyzing and comparing stochastic algorithms performance in MOP. Unlike most of the MOP performance indicators, this method illustrates the algorithm performance in graphical form. The attainment function describes the location of algorithm solution in objective space and it is estimated empirically, thus denoted as empirical attainment function, EAF. Mathematically, the EAF is defined as in the following equation:

$$\alpha(z) = \frac{1}{n} \sum_{i=1}^{n} I\left(X^i \preccurlyeq \{z\}\right), \tag{70}$$

with X^i as the ith non-dominated set approximated Pareto front or the frequency of attaining z by its respective approximation set $X^1, \ldots, X^n$ of an algorithm in n runs with $\preccurlyeq$ denotes as the weak Pareto dominance relation between sets and $I(x)$ as an indicator function that maps the objective space to $[0, 1]$ with $I(x) : \mathbb{R}^d \mapsto \{0, 1\}$ or Eq. (71).

$$I(x) = \begin{cases} 1, & \text{if } x \text{ is true,} \\ 0. & \text{otherwise.} \end{cases} \tag{71}$$

In general, the EAF is a function α that estimate the probability of being dominated by the approximated PF within the objective space $\mathbb{R}^d$. The attainment surface method separates the boundary of objective space into two regions. The first region is attained or dominated objective function by the algorithm and the second is the region that is not attained by the algorithm. This measure is formalized in the concept of k%-attainment surface that corresponds to $k/100$ percentiles of empirical frequency distribution, which corresponds to $k = i * 100/n$ with $i = 1, \ldots, n$ runs of the algorithm (Ibánez et al. 2010). An example, median attainment is referred to as region attained by 50% of the runs, whereas 25%, as well as 75%, correspond to 25th and 75th percentiles of attainment or the first- and third quartile fronts respectively. Besides, the region between both percentiles also corresponds to the inter-quartile region that analogous to the inter-quartile range for the single-objective problem (Ferrer et al. 2012). An example of EAF on the bi-objective problem is shown in Fig. 18. EAF plot can also be used for comparing algorithms by plotting each EAF for each algorithm side by side and the location of objective vectors for each algorithm can be compared.

The attainment function of each algorithm can also be compared by its differences, denoted as Differential Empirical Attainment Function, Diff-EAF. The method is relatively simple and easy to understand based on its graphical description that plots the EAF differences in a single chart. As an example, two algorithms are measured on solving the same bi-objective problems with n runs. Then the Diff-EAF can be calculated as in the following equation (Ibánez et al. 2010).

$$\delta_n^{A-B}(z) = \alpha_n^A(z) - \alpha_n^B(z). \tag{72}$$

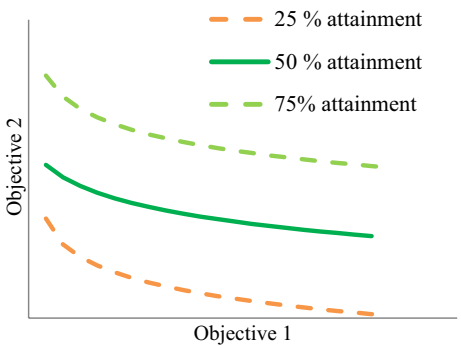


Fig. 18 EAF with 25%, 50% (median) and 75% attainment

A positive difference in the corresponded area shows that algorithm A is better compared to B and otherwise for a negative difference. Thus, the graphical representation of Diff-EAF can point out which algorithm is better to which part of the solution space. Visualizing the EAF chart for 2-D problems is straightforward using simple line plots or heat maps. However, for 3-D problems, the visualization is challenging due to the rectangular cuboid facets and areas. There are several approaches for the picture the 3-D attainment surfaces such as grid-based sampling, slicing, maximum intensity projection, and direct volume rendering (Filipič and Tušar 2018). Refer to Ibánez et al. (2010), Minella et al. (2011) and Tušar and Filipič (2014) for further description and calculation of EAF difference in 2-dimensional problems and 3-dimensional problems (Tušar and Filipič 2014).

3.1.2 Average runtime attainment function, aRTA (Brockhoff et al. 2017)

An alternative to the EAF method, Brockhoff et al. (2017) proposed the average runtime attainment function, aRTA to measure the expected runtime of solution that weakly dominates PF. This method is a generalization of attainment function that based on a target vector $z \in \mathbb{R}^d$ to collect the runtime,$T(z)$ as the minimum number of function evaluations to obtain the solution that weakly dominates z. The function *aRTA* is then evaluated over n trials of the algorithm with n runtimes:$T_1(z), \ldots, T_n(z)$ and with n_s successes with the following equation:

$$aRTA(z) = \frac{1}{n_s} \sum_{i=1}^{n} T_i(z). \tag{73}$$

The notion of *aRTA* is relatively similar to the calculation of average runtime, *aRT* of single problem optimization as proposed by Hansen et al. (2012) in Eq. (31). The *aRTA* function maps $\mathbb{R}^d$ to positive real numbers,$\mathbb{R}^+$ with a color map as demonstrated in Brockhoff et al. (2017). The advantage of this method compared to EAF is that *aRTA* can capture the algorithm's performance over multiple runs and over time, whereas EAF captures only on n defined times. Further information can be referred to in the respective paper. To compare *aRTA* of two algorithms, a ratio function of both *aRTA* values need to be calculated and plotted over the search space z. This is analogous to the Diff-EAF on comparing two algorithms via subtraction. Suppose a comparison between algorithm A and algorithm B, the ratio is calculated as follows:

$$aRTA_{ratio}(z) = \frac{aRTA^B(z)}{aRTA^A(z)}. \tag{74}$$

3.2 Cardinality measure

Cardinality quantifies the number of non-dominated solutions returned by an algorithm. In general, larger cardinality that sufficiently describes the set is desired; too many numbers of solutions might overwhelm the decision process. The cardinality based measure is appropriate if there is a high probability to find a significant percentage of non-dominated

solutions (Hansen and Jaskiewicz 1998). Some of the cardinality based measures are briefly described in the following sub-sections.

3.2.1 *C*-Metric (Zitzler and Thiele 1998)

The C-metric (also denoted as Coverage of Two Sets) is proposed by Zitzler and Thiele (1998) that referred to as the coverage of two sets of decision vectors such that A and B as $A, B \subseteq X$. The function C maps the ordered pair (A, B) to the interval $[0, 1]$ with the following equation.

$$C(A, B) = \frac{|\{b \in B; \exists a \in A : a \succcurlyeq b\}|}{B}. \tag{75}$$

The notion $C(A, B) = 1$ is referred to as all points in B dominated by or equal to or weakly dominated (Knowles and Corne 2002) points in A. In the opposite, $C(A, B) = 0$ shows that none of the points in B are covered by set A. The measure using C-metric can be represented using box-plot for n number of runs with m different algorithms. A higher C-values denotes a higher coverage of an algorithm on a non-dominated solution and this can be represented in the percentage of coverage or C-metric (Zitzler and Thiele, 1999). The C-metric can give information on the quality between approximation sets A and B and is a widely accepted binary metric. Nonetheless, the usage of this metric decreased in recent years as the implementation of hypervolume and ε-measure reflects more aspects of quality between sets A and B at lower computational cost (Riquelme et al. 2015). Also, C-values are often difficult to interpret if the solution set of A and B are not comparable.

3.2.2 C_{1R}-Metric (Hansen and Jaskiewicz 1998)

This metric measures the ratio of points found in the reference set, R over the cardinality of the Pareto set approximation. In other words, this metric indicates the ratio of found solutions in R. The metric is defined with A as approximation sets and R as the reference set as follows.

$$C1_R(A) = \frac{|A \cap R|}{|R|}. \tag{76}$$

3.2.3 C_{2R}-Metric (Hansen and Jaskiewicz 1998)

The metric defines the ratio of non-dominated points by reference set R. It is quite similar to C-metric but based on R, where the metric estimates the number of solutions that are non-dominated by R. The formulation of C_{2R} measure is as follows:

$$C2_R(A, R) = \frac{\left|x \in A \nexists r \in R : r \prec x\right|}{|A|}, \tag{77}$$

with x as individual points that has elements of A and r elements of R. This metric suffers similar drawbacks as C-metric (Audet et al. 2018).

3.2.4 Error ratio, ER (Van Veldhuizen and Lamont 1999)

This metric is quite similar to C_{1R} but measures the proportions of non-true Pareto points or the solution intersections between approximation set A with PF as the following formulation:

$$ER(A, PF) = 1 - \frac{|A \cap PF|}{|PF|} = \frac{\sum_{i=1}^{n} e(x_i)}{n}, \tag{78}$$

where A ∩ PF is the solutions in both A and PF, x_i as the individual in the approximation set A and n as the number of individuals in A. $ER(A, F) = 0$ if $x_i \in PF$ and $ER(A, F) \approx 1$ if the non-dominated solutions are further from PF. Audet et al. (2018) proposed a threshold that quantifies elements belonging to the PF. The metric typically depends on the cardinality of Pareto set approximation that possible to misguide the interpretation as highlighted by Knowles and Corne (2002).

3.2.5 ONVG, ONVGR (Van Veldhuizen and Lamont 1999)

The Overall non-dominated vector generation, ONVG simply represents the counts of non-dominated solutions found in an approximation front as the following expression:

$$ONVG(A) = |\mathrm{A}|, \tag{79}$$

where A is the number of non-dominated solutions in the optimal solution set (A) and $|\cdot|$ is the number of components in the set. The other variant is denoted as Overall non-dominated vector generation ratio, ONVGR, and mathematically defined as the division of the number of points of approximation set to the cardinality of Pareto optimal solution set:

$$ONVGR(A, PF) = \frac{|\mathrm{A}|}{|\mathrm{PF}|}. \tag{80}$$

This formulation describes the cardinality of the optimal solution set (A) with respect to the PF. However, both measures are not reliable as both do not necessarily imply that an algorithm is better than the other as demonstrated by Knowles and Corne (2002), Van Veldhuizen and Lamont (2000) and Audet et al. (2018).

3.2.6 GNVG, GNVGR and NVA (Van Veldhuizen and Lamont 2000)

These metrics are proposed to capture the cardinality measures by search stages of the algorithm. The first metric is the Generational Non-dominated Vector Generation $GNVG = |A(t)|$ and Generational Non-dominated Vector Generation, $GNVGR(A, PF) = |A(t)|/|PF|$ that similar to ONVG and ONVGR respectively but with consideration of the search progress. The third metric is defined as Non-dominated Vector Additional, $NVA(A, t) = GNVG(A, t) - GNVG(A, t-1)$, which measures the cardinality change of solution set A during the algorithm search within t many generations.

3.2.7 Pareto dominance indicator, NR (Goh and Tan 2009)

This n-ary metric measure the ratio of non-dominated solutions contributed by solution A to the non-dominated solutions found by all compared algorithms with the following formulation:

$$NR\left(A_1, A_2, \ldots, A_n\right) = \frac{\left|\mathrm{A}_1 \cap \mathrm{B}\right|}{|\mathrm{B}|}, \tag{81}$$

with $B = \left\{b_i | \forall b_i, \exists a_j \in \left(A_1 \cup A_2 \cup \ldots \cup A_n\right) \prec b_i\right\}$ and $a_j \prec b_i$ denotes a_j dominates b_i and A_1 as the evaluated set.

3.2.8 Mutual domination rate, MDR (Martí et al. 2016)

This measure was initially proposed as stopping criteria for the evolutionary algorithm. The metric is then listed as a cardinality based measure for monitoring the algorithm progress during the iteration search (Audet et al. 2018). The formulation of MDR is shown as follows:

$$MDR(A, k) = \frac{|\Delta(\mathrm{A(k-1)}, \mathrm{A(k)})|}{|\mathrm{A(k-1)}|_{10}} - \frac{|\Delta(\mathrm{A(k)}, \mathrm{A(k-1)})|}{|\mathrm{A(k)}|}, \tag{82}$$

with A(k) as the Pareto set approximation generated at kth iteration. In general, MDR describes the number of non-dominated solutions at $k-1$th iteration being dominated by non-dominated points at kth iteration. The set of non-dominated solutions at kth iteration completely dominates the solutions at $k-1$th iteration by $MDR(A, k) = 1$. Otherwise if $MDR(A, k) = 0$, No significant progress is occurred and even more worst, by $MDR(A, k) = -1$ resembled the total loss of domination at the current iteration.

3.3 Convergence measure

The indicator of this metric is related to the distance of solutions set to PF. For problems with unknown PF, a reference set R needs to be considered. This usually took place for real-world problems, when it is complicated and difficult to determine PF. The reference set is an approximation of PF that contains all known non-dominated solutions (Riquelme et al. 2015).

3.3.1 Seven points average distance, SPAD (Schott 1995)

This metric is designed especially for the bi-objective optimization problems that use a reference set composed of seven points. This metric did not require the knowledge of PF. Due to its low resolution of seven points estimation, there is a possibility of points in the reference set that fails to capture the whole form of PF, and the limitation only for bi-objective is also inconvenient.

3.3.2 Progress metric, Pg (Back 1996)

This metric measures the progression of algorithm approximation towards PF with respect to the number of iterations, defined by the following equation:

$$P_g = \ln \sqrt{\frac{f_i^{best}(0)}{f_i^{best}(k)}}, \tag{83}$$

with $f_i^{best}(k)$ as the best value of objective i in iteration k. Pg metric estimate the speed of convergence. However, this metric is not defined if $f_i^{best}(0)$ or $f_i^{best}(k)$ has a negative or zero value (Audet et al. 2018).

3.3.3 Distance metric, D_{1R} and D_{2R} (Czyzak and Jaskiewicz 1998)

It is a distance measure based on reference set R. Czyzak and Jaskiewicz (1998) proposed two types of distance measures, the first measure is defined as D_{1R} that measures average distance from a reference set and the second measure is D_{2R} that measure the worst distance from the reference set. D_{1R} is slightly similar to IGD, but based on the weighted average over the points of Pareto optimal set or reference set. This metric is also used in numerous MOP comparisons (Riquelme et al. 2015). On the other hand, D_{2R} gives the information of the biggest distance from $r \in R$ to the closest solution in A as shown in the formulation below.

$$D_{1R}(A, R) = \frac{1}{|R|} \sum_{i=1}^{|R|} \left\{ \min_{a \in A} \left\{ c\left(r_i, a\right) \right\} \right\}, \tag{84}$$

$$D_{2R}(A, R) = \max_{r \in R} \left\{ \min_{a \in A} \left\{ c(a, r) \right\} \right\}, \tag{85}$$

where for D_{1R}, $c\left(r_i, a = \max_{j=1,2,\ldots,m} \left\{0, w_j\left(f_j(a) - f_j\left(r_i\right)\right)\right\}\right)$ with w_j as the reciprocal of f_j in the reference set R. However the measures are weakly compatible with the outperformance relation (Hansen and Jaskiewicz 1998).

3.3.4 Generational distance, GD (Van Veldhuizen and Lamont 1999)

GD is a unary metric that measures the average distance that obtained by metaheuristic algorithm to the true PF of the problem with the following formulation:

$$GD(A, \mathrm{PF}) = \frac{\left(\sum_{i=1}^{|A|} d_i^P\right)^{1/p}}{|A|}, \tag{86}$$

with PF as the Pareto front, A as the approximation set obtained by metaheuristic and d as the Euclidian distance in the objective space between solution $i \in A$ and the nearest PF. GD measure determines the accuracy of the solution (Riquelme et al. 2015). The method is easy to compute but very sensitive to the number of points found by an algorithm if the algorithm misses a big portion of PF without being penalized by this metric. This metric requires normalization and replace the quadratic mean with the arithmetic mean. Applicable when the compared sets are non-dominated to each other and no PF range can be properly estimated.

3.3.5 Standard deviation from the generational distance, STDGD (Van Veldhuizen and Lamont 1999)

This metric measures the deformation of the Pareto set according to a Pareto optimal set. With the following definition:

$$STGD = \frac{\sum_{i=1}^{n} \left(d_i - GD\right)^2}{n}, \tag{87}$$

with d_i as the Euclidean distance and GD as the Generational Distance measure. This metric is sensitive to the number of points found by an algorithm.

3.3.6 M_1 Metric (Zitzler et al. 2000)

It is referred to as the average distance to the Pareto optimal set with the formulation as shown in the following equation:

$$M_1 = \frac{1}{|A|} \sum_{p \in A} min\left\{ \|p - \bar{p}\|; \bar{p} \in \bar{A} \right\}, \tag{88}$$

with $|A|$ as the number of non-dominated solutions in front A and $\|p - \bar{p}\|$ represents the distance metric. The formulation of this metric is almost similar to GD with $p = 1$. Computing this metric alone is insufficient to provide the overall performance evaluation since extremely distributed fronts may have the same distance to PF.

3.3.7 Hausdorff distance, d_H (Heinonen 2001)

The metric is used to measure the proximity of different sets. The d_H measure between sets A and B is defined as follows:

$$d_H(A, B) = max\{d(A, B), d(B, A)\}, \tag{89}$$

with A and B being the solution sets. This metric is however not practical for metaheuristic algorithms since it penalizes single outliers of the candidate set. (Bogoya et al. 2018; Schutze et al. 2012).

3.3.8 Distance metric, Υ (Deb et al. 2002)

This metric measures the extent of convergence to a known Pareto optimal solution. The method is firstly evaluated by defining a set of H uniformly spaced solutions that lie on PF and N number of Pareto optimal obtained. Then the minimum Euclidean distance between each generated solution of an algorithm with H chosen points is evaluated. The Υ-metric is then determined by averaging these distances as the following equation:

$$\Upsilon = \frac{\sum_{i=1}^{N} \sum_{j=1}^{H} d_{i,j}}{NH}, \tag{90}$$

1. $i = 1$;
2. Compute PF using solution set points. Remove PF points from solution set
3.**if** result set = empty, wave = i; **else** $i = i+1$ and go to step 2.

Fig. 19 Computation of wave metric

with $d_{i,j}$ as the Euclidean distance from the ith solution obtained to the jth PF or reference point. The spread of each minimum distance can also be used as a performance indicator by evaluating standard deviation σ_{Υ}. The formulation of this metric is almost similar to GD with $p = 1$. Smaller value denotes better convergence. This metric is appropriate to present in terms of average and variance for comparison between algorithms.

3.3.9 ϵ-Indicator, I_{ε} (Zitzler et al. 2003)

This metric measure the minimum factor to scale the optimal solution set such that A dominates B as follows:

$$I_{\varepsilon}(A,B) = min\{\varepsilon \in \mathbb{R} | \forall b \in B \exists a \in A : a \succ b\}. \tag{91}$$

If $I_{\varepsilon}(A,B) < 1$, all solutions in B are dominated by solution in A. Otherwise if $I_{\varepsilon}(A,B) > 1$ and $I_{\varepsilon}(B,A) > 1$, then A and B are incomparable. However if both $I_{\varepsilon}(A,B) = I_{\varepsilon}(B,A) = 1$, then A and B hold similar Pareto front approximation. The metric is suitable for both continuous and discontinuous approximation of PF. The major drawback is the metric only considers one objective, which may lead to information loss (Audet et al. 2018).

3.3.10 Wave metric (Collette and Siarry 2005)

The wave metric is used to compute the depth of solutions in the numbers of PF. The calculation of wave is based on the condition as in the following code (Fig. 19).

A good algorithm may result in a lower wave, wave = 1 denotes that all solutions set are equal to PF. Some drawbacks of wave metrics are: unable to differentiate between two solutions sets and it is impossible to compare the same result of wave metric on two different solution sets (Collette and Siarry 2005).

3.3.11 Pareto ratio, PR (Collette and Siarry 2005)

The PR metric is the ratio between numbers of solution points of set A in PF with the total number of points of set A at a given iteration as the following equation:

$$PR = \frac{|PF(A)|}{|A|}. \tag{92}$$

3.3.12 Speed metric, SM (Collette and Siarry 2005)

This metric compares the iterations or function evaluations required of an algorithm to evaluate solutions that lie within a threshold of PF. This method applies to all MOPs with defined PF. The comparison is carried out by counting the number of points that fall below the threshold of PF.

3.3.13 Run-time metric, $\Lambda(t)$ (Zhou et al. 2006)

The run-time metric measures the convergence dynamic of the algorithm with the following formulation:

$$\Lambda(t) = \frac{1}{2}\left[\Upsilon(A(t), A^*) + \Upsilon(A^*, A(t))\right], \tag{93}$$

where $A(t)$ as the non-dominated solution in generation t and A^* represents the Pareto optimal solutions. The term Υ is referred to as the distance metric (Deb et al. 2002). The metric can be plotted over t number of generations and can be compared against different algorithms. The metric only represents convergence if $\Upsilon(A(t), A^*) >> \bar{d}(A^*, A^*)$ such that $\bar{d}(A^*, A^*)$ equal to the average distance between the Pareto optimal set. The metric $\Lambda(t)$ represents both convergence and diversity if $\Upsilon(A(t), A^*) = .\bar{d}(A^*, A^*)$ (Zhou et al. 2006).

3.3.14 Average Hausdorff distance, Δ_p (Schutze et al. 2012)

This metric is referred to as the Hausdorff distance of the modified version of GD and IGD:

$$\Delta_p(A, B) = max\left\{GD_p(A, B), IGD_p(A, B)\right\}, \tag{94}$$

with $GD_p(A, B)$ and $IGD_p(A, B)$ as the modified version of GD and IGD respectively with the formulation as below. Notice that the Eq. (94) is similar to the d_H measure as in Eq. (89).

$$GD_p(A, B) = \left(\frac{1}{|A|}\sum_{a \in A} d(a, B)^p\right)^{\frac{1}{p}}, \tag{95}$$

$$IGD_p(A, B) = \left(\frac{1}{|B|}\sum_{b \in B} d(b, A)^p\right)^{\frac{1}{p}}. \tag{96}$$

The Δ_p metric requires the knowledge of PF and can be used to compare continuous and discontinuous approximations of PF (Audet et al. 2018). Disadvantage: only defines an *infra-metric* instead of metric and applicable for finite approximations of Pareto set (Bogoya et al. 2018).

3.3.15 Degree of approximation, DOA (Dilettoso et al. 2017)

The indicator encompasses the distribution, extension, and the cardinality of a Pareto front approximation with the formulation as follows:

$$DOA(A,B) = \frac{1}{|B|}\sum_{y\in B} min\{d(y,A), r(y,A)\}, \tag{97}$$

with $d(y,A)$ as the Euclidean distance between $y \in A$ that belong to $D_{y,s}$, where:

$$d(y,A) = \begin{cases} \min\limits_{s\in D_{y,s}} df(y,s) & if \ \left|D_{y,s}\right| > 0, \\ \infty & if \ \left|D_{y,s}\right| = 0. \end{cases} \tag{98}$$

Similarly,$r(y,s)$ is referred to as values for $y \in A$ that not belong to $D_{y,s}$ as follows:

$$r(y,A) = \begin{cases} \min\limits_{x\in s/D_{y,s}} rf(y,x) & if \ \left|s/D_{y,s}\right| > 0, \\ \infty & if \ \left|s/D_{y,s}\right| = 0. \end{cases} \tag{99}$$

Next, $rf(y,x)$ is defined as in the following equation:

$$rf(y,x) = \sqrt{\sum_{i=1}^{m} max\left\{0, f_i(x) - f_i(y)\right\}^2}. \tag{100}$$

The DOA metric can be used to compare algorithms if the PFs are known. The value of this metric is not dependent on the number of points and the metric partitions into the subset element. DOA is computationally low and this method can be applied in the continuous and discontinuous approximation of PF.

3.3.16 (p,q)-Averaged Hausdorff distance, $\Delta_{p,q}$ (Vargas and Bogoya 2018)

This metric is a generalization of Δ_p and d_H. It is a modification of Δ_p with p describes the closeness to PF and q reflects the dispersion of the solution set. The formulation of this metric is as follows:

$$\Delta_{p,p}(A,B) = max\left\{GD_{p,q}(A, B\backslash A), GD_{p,q}(B, A\backslash B)\right\}, \tag{101}$$

where $GD_{p,q}$ represents the (p,q)-average Hausdorff distance between A and B and

$$GD_{p,q}(A,B) = \left(\frac{1}{|A|}\sum_{a\in A}\left(\frac{1}{|B|}\sum_{b\in B} d(a,b)^q\right)^{\frac{p}{q}}\right)^{\frac{1}{p}}. \tag{102}$$

The parameters p and q can be modified independently, which is to evaluate a customized spread that depending on q in customized closeness location that depends on p to the PF. This metric is limited to finite sets.

3.3.17 $\mathcal{H}$-Indicator (Santos and Xavier 2018)

The $\mathcal{H}$-indicator is a convergence measure inspired by Shannon entropy formulation. The metric is defined as in the following equation:

$$\mathcal{H}(A) = \frac{1}{2n} \sum_{i=1}^{n} -q_i \log_2 (q_i), \tag{103}$$

with $A = \{x_1, x_2, \ldots x_n\}$ and $q_i = min\left\{ 1/exp(1), \left\| q(x_i) \right\|^2 \right\}$. The metric did not require the knowledge of PF. $\mathcal{H} \approx 0$ for good convergence towards Pareto set and $\mathcal{H} \approx 0.26537$ otherwise.

3.3.18 Variants of convergence speed measures

In the measurement of MOP convergence speed, some authors such as Durillo et al. (2010) defined three criteria to obtain this metric, which is by determining the number of non-dominated solutions generated by each algorithm, convergence of approximation to Pareto front using epsilon indicator and both convergence plus diversity of Pareto front based on hypervolume indicator. Other literature, such as Liu and Zhang (2019) proposed a convergence index that is equivalent to the approaching degree of algorithm based on the minimum distance from the solution set to reference or Pareto front. Smaller distance showing a higher approaching degree of the solution set. The convergence index is calculated as $CI = \sum_{i=1}^{NDset^t} Pd/NDset^t$ with Pd as the shortest distance from i to the reference set and $NDset^t$ as the non-dominated set of the tth generation. Other literature (Nebro et al. 2009) described convergence speed with the median and IQR (inter-quartile range) of the number of FEVs required by the algorithm to reach 98% of HV value, which is a good measure for determining dispersions within third and first quartiles. The performance related to the number of function evaluations, FEVs required by the algorithm for solving MOP is also another variant of speed measurement. Sun et al. (2018) plotted the number of FEVs required by three algorithms on various sizes of objectives. The criteria on counting the number of FEVs are based on either the maximum FEV (100,000) is met or upon reaching the metric $E \leq 0.01$ with E formulated as follows:

$$E = \sqrt{\sum_{i=1}^{m} \frac{\left(z_i^{nadir} - z_i\right)^2}{\left(z_i^{nadir} - z_i^*\right)^2}}, \tag{104}$$

with z_i as the ith element of the estimated nadir point derived from the extreme points, z_i^{nadir} as the nadir point and z_i^* as the ideal point.

3.4 Diversity measures

The diversity measure is referred to as the distribution and the spread of the computed solutions in PF approximation. This group of metrics is not suitable for measuring the convergence criteria, rather it demonstrates the scatterings of points along with PF approximation. Thus, this metric is sensible for the Pareto set that consists of several solutions. Some literature distinguishes this section into distribution, spread, and distribution-spread characteristics (Jiang et al. 2014, Requilme et al. 2015). For general understanding, this paper summarizes each sub-characteristic into diversity-related metrics.

3.4.1 Spacing metric, *SP* (Schott 1995)

SP or also denoted as Diversity Spacing is designed to measure the distribution evenness of the members of the approximation set or the diversity of Pareto front gained by algorithms. The method is a unary metric that is determined by calculating the relative distance between consecutive solutions *A*. The equation, however, depends on the scale of the objective functions with the following formulation:

$$SP(A) = \sqrt{\frac{1}{|A|-1}\sum_{i=1}^{|A|}\left(\bar{d}-d_i\right)^2}, \tag{105}$$

with d_i as the l_1 distance between each point $A_i \in A$ with the closest point of PF approximation executed by an algorithm, $d_i = \min_{k\in\Lambda\neq i}\sum_{m=1}^{M}\left|f_m^i - f_m^k\right|$ and $\bar{d}$ as the average of d_i. $SP=0$ denotes that all members of the approximation set are equidistantly spaced. The computation is straightforward, however, the metric provides limited information if the points are separated into multiple groups. Furthermore, the results of *SP* depends on the scale of objective functions.

3.4.2 Maximum spread, MS (Zitzler 1999)

This metric defines the maximum distance between a_i as the maximum value in the *i*th objective with b_i as the minimum value of *i*th objective with *m* number of objectives as shown below:

$$MS = \sqrt{\sum_{i=1}^{m} max\left(d\left(a_i, b_i\right)\right)}. \tag{106}$$

3.4.3 M_2^* and M_3^* metrics (Zitzler et al. 2000)

The M_2^* the metric is designed with user-specified parameters that equipped with a niche radius, σ with the following formulation:

$$M_2^*(A) = \frac{1}{|A-1|}\sum_{p\in A}|\{q\in A; \|p-q\| > \sigma\}|, \tag{107}$$

with σ as the niche radius. This metric measures how many solutions of $q \in A$ are in the local vicinity $\|p-q\| > \sigma$ for a solution of $p \in A$. This metric considers both distribution and the number of non-dominated solutions. Another diversity measure proposed by Zitzler et al. (2000) is the M_3^* metric that considers the maximum extent in dimension space to estimate the range of fronts spread out. For two-dimensional problems, this metric is equivalent to the distance of outer solutions between both objectives. The formulation of this metric is calculated as follows:

$$M_3^*(A) = \sqrt{\sum_{i=1}^{n} max\left\{\left\|p_i - q_i\right\|; p, q \in A\right\}}, \tag{108}$$

3.4.4 Laumann's metric, I_L (Laumanns et al. 2000)

The I_L metric is defined as a ratio of the Lebesgue measure between the intersection of dominated space by a set A (as the PF approximation A) and hypercube, $H(A)$ as follows:

$$I_L(A) = \frac{\lambda(D(A) \cap H(A))}{\lambda(H(A))}, \tag{109}$$

with λ as the Lebesgue measure,$D(A)$ as the dominated set A. This metric has high complexity and increased by a higher number of dimensions.

3.4.5 Overall Pareto spread, OS and the kth objective Pareto spread, OS_K (Wu and Azarm 2001)

The OS metric is conceptually similar to M_3^* metric that quantifies the extended spread of Pareto solution set over the objective space. The metric is defined as the volume ratio of two hyper-rectangles between extreme points concerning the good and bad points, simplified by Okabe et al. (2003) as follows:

$$OS\left(A, P_g, P_b\right) = \prod_{i=1}^{m} OS_k\left(A, P_g, P_b\right), \tag{110}$$

with P_g and P_b as the ideal point and nadir point respectively,$OS_k\left(A, P_g, P_b\right)$ is the kth objective Pareto Spread with the following formulation:

$$OS_K\left(A, P_g, P_b\right) = \frac{\left|max_{s \in A} f_k(s) - min_{s \in A} f_k(s)\right|}{\left|f_K\left(P_b\right) - f_K\left(P_g\right)\right|}, \tag{111}$$

Both OS_K and OS method has similarity except that the OS_K metric able to quantify the solution range concerning the individual objective.

3.4.6 Accuracy of Pareto frontier, AC (Wu and Azarm 2001)

The goodness of observed Pareto set can be measured with this metric. The metric is equivalent to the reciprocal of $AP(P)$ value; the $AP(P)$ is defined as the front approximation of the observed Pareto solution set P. Higher value of AC is preferable by comparing two observed Pareto solution sets. Detail formulation can be observed in Wu and Azarm (2001).

3.4.7 Number of distinct choices, NDC_μ and cluster, CL_μ (Wu and Azarm 2001)

The NDC metric represents the number of distinct choices for a pre-specified value of μ is defined as the following equation:

$$NDC_\mu(A) = \sum_{l_m=0}^{v-1} \cdots \sum_{l_2=0}^{v-1} \sum_{l_1=0}^{v-1} NT_\mu(q, A), \tag{112}$$

with $q = (q_1, q_2, \ldots, q_m)$ where $q_i = l_i/v$ and $v = 1/\mu$. The metric divides the objective space into $(1/\mu)^m$ grids with $\mu \in [0, 1]$. The Pareto solution set with higher $NDC_\mu(A)$ value is preferred over the lower counterpart. Another measure proposed by Wu and Azarm (2001) is the CL_μ metric. The metric evaluates the average number of indistinct solutions on a grid-scale specified by $1/\mu$. The formulation for $CL\mu$ is equal to the ratio of the number of observed Pareto solutions,$N(A)$ to the $NDC_\mu(A)$ value as follows.

$$CL_\mu(A) = \frac{N(A)}{NDC_\mu(A)}. \tag{113}$$

Ideally, $CL_\mu(A) = 1$, which shows that each obtained Pareto solution is distinct. Higher value of CL_μ denotes a more clustered of the solution set, which is less preferred.

3.4.8 Entropy-based metric (Farhang-Mehr and Azarm 2002)

The Entropy metric measures the uniformity and coverage of solution sets by employing influence functions to estimate the solution densities. The formulation is based on Shannon entropy of discrete domain as follows:

$$H = -\sum_{k_1=1}^{a_1}\sum_{k_2=1}^{a_2}\cdots\sum_{k_m=1}^{a_m}\rho_{k_1,k_2,\ldots k_m}\ln\left(\rho_{k_1,k_2,\ldots k_m}\right), \tag{114}$$

with $a_1, a_2, \ldots, a_m$ as the number of grids that represents the size of cells and its corresponding normalized density function between $k_1, k_2, \ldots, k_m$. A solution set with a higher entropy metric is referred to as more evenly spread throughout the feasible region and thus provides better coverage of the space. Some of the main disadvantages for this metric are (Deb and Jain 2002): (1) the variance of normal entropy function affects the distribution being either peaky or flat, (2) the method resulted in erroneous by disconnected Pareto optimal fronts due to the characteristic of continuous entropy function.

3.4.9 Diversity metric, Δ (Deb et al. 2002)

This metric measures the extent of spread attained by the obtained solutions. It compares all of the solution's consecutive distances with the average distance as follows:

$$\Delta(A, PF) = \frac{d_f + d_l + \sum_{i=1}^{|A|-1}\left|d_i - \bar{d}\right|}{d_f + d_l + (|A| - 1)\bar{d}}, \tag{115}$$

with d_i as the Euclidean distance between consecutive solutions, $\bar{d}$ as the average distance,d_f and d_l as the minimum Euclidean distances from solution A to the extreme solutions of PF. This metric is however limited to only two objectives. The metric is further improved by Zhou et al. (2006) by calculating the distance from a point to its nearest neighbor instead of only two consecutive solutions.

3.4.10 Integrated preference function, IPF (Carlyle et al. 2003)

This metric measures the volume of polytopes resolved by non-dominated solutions in a set with utility function over the corresponding optimal weights. It represents the expected

utility of a diversity measure. The IPF is firstly derived by finding the optimal weight interval for each non-dominated solutions. Then the utility functions are integrated over the optimal weights interval as follows:

$$IPF(A) = \int h(w)u^*(A, w)dw, \tag{116}$$

with $h(w)$ as the weight density function, $u^*(A, w)$ as the best utility function value of the solution in A and weight w. A lower IPF value denotes a better metric. The main drawback of this metric is its computation complexity that increases by the number of objectives (Bozkurt et al. 2010).

3.4.11 U-measure (Leung and Wang 2003)

The U-measure is denoted as a uniformity measure given by the following formulation:

$$U(A) = \frac{1}{A}\sum_{i\in A}\frac{d_i^{'}}{d_{ideal}} - 1, \tag{117}$$

with $d_i^{'}$ as the distance between ith point to the closest neighbor in the objective space and translated into the extreme points of PF to the nearest neighbor.$d_{ideal} = 1/|A|\sum_{i\in A} d_i^{'}$. This method quantifies the uniformity of the found PF approximation, smaller U reflects a better uniformity of solutions.

3.4.12 Evenness ξ (Messac and Mattson 2004)

The measure denotes the uniformity of the solution set and also considered as the coefficient of variation, COV since the formulation is similar to COV as expressed in the following equation:

$$\xi(A) = \frac{\sigma_D}{\bar{D}}, \tag{118}$$

with σ_D as the standard deviation and $\bar{D}$ as the mean of set D, where $D = \left\{d_a^u, d_a^l : a \in A\right\}$ of a given point $F(a), a \in A$ in PF approximation. The parameters d_a^u and d_a^l represent the largest sphere of diameter and closest neighbor distance respectively. The uniformity of solutions is better by $\xi \approx 0$. For continuous PF, the method did not consider holes in the PF approximation as it only considers the closest distance between two points of objective space.

3.4.13 Modified spacing metric, MSP (Collette and Siarry 2005)

This metric is a further improvement of the Spacing metric, SP by removing the objective function scale and computing the distance,d_i by sorting PF in ascending order. The modified formulation is given as follows:

$$MSP(A) = \sqrt{\frac{1}{|A|-1}\sum_{i=1}^{|A|}\left(1 - \frac{d_i}{\bar{d}}\right)^2}. \tag{119}$$

3.4.14 Extension measure, EX (Meng et al. 2005)

This measure defines the extent of PF approximation given by the following formulation:

$$\mathrm{EX}(A) = \frac{1}{m}\sqrt{\sum_{i=1}^{m} d\left(f_i^*, A\right)^2}, \tag{120}$$

with $d\left(f_i^*, A\right)^2$ as the minimal distance between the solution to the ith single objective problem of total m single objectives. The method is straightforward to compute and penalize well distributed PF approximation that neglecting the extreme values.

3.4.15 Generalized spread Δ^* (Zhou et al. 2006)

The metric is a further improvement of diversity metric, Δ that extends for more than two objectives as follows:

$$\Delta^*(A, PF) = \frac{\sum_{k=1}^{m} d\left(e_k, A\right) + \sum_{i=1}^{|A|} \left|d_i - \bar{d}\right|}{\sum_{k=1}^{m} d\left(e_k, A\right) + |A|\bar{d}}, \tag{121}$$

where $d\left(e_k, A\right) = \min_{x \in A} \left\| F\left(e_k\right) - F(x) \right\|$ with $e_k \in PF$ as the solution of extreme solutions on the kth objective. The parameter $d_i = \min_{(A_i, A_j) \in A, A_i \neq A_j} \left\| F\left(A_i\right) - F\left(A_j\right) \right\|$ referred to as the minimal Euclidean distance between two points of PF approximation. This metric suffers similar drawbacks as the Spacing metric due to the scope of the shortest distance between elements of PF approximation. Also, this measure requires the information of extreme solutions of PF.

3.4.16 Sphere counting, SC (Wanner et al. 2006)

The metric compares the spread uniformity of Pareto set A and B with the following seven steps:

Steps	Activity
1	Define a radius, r on the objective space
2	Locate a sphere with radius r centered at any point of A
3	Define a sphere counter equal to one
4	Eliminate all points located in the sphere
5	Place another sphere at the remaining points that closest to the previous center and increment the sphere count
6	Go to step 2 until no remaining points observed in the estimate set
7	Sphere counting for set A is completed. Do the same procedure (step 1–6) for set B

1. Normalize f_i for $i = 1, \ldots, m$;
2. **for** $i = 1$ to n_r
   $n_b(i) =$ sphere counting$Counting\left(r_d(i)\right)$
   end for
3. **Do** Numerical integration of (n_b, r_d)t

Fig. 20 Computation of the ISC metric

This set with more spheres is identified as the front with greater spread, thus indicating a better description of Pareto set. The advantage of this method is due to its simple and straightforward steps; however, the number of spheres is heavily dependent on the radius value.

3.4.17 Integrated sphere counting, ISC (Silva et al. 2007)

This metric is a further development of SC with the interpretation of signal processing. In contrast to SC, The ISC metric defines an interval of radius variation to cope with the scale difference that is appropriate for the quality measure by removing the effect of under- and oversampling. The range of r is set with a maximum 10% of maximum distance R and minimum with 1% of R, with $R = \sqrt{m}$ and m is the number of objectives. Then the sphere counting is integrated within the minimum–maximum interval. A higher integral value denotes a better spread of PF. The procedure of ISC metric is summarized in the ISC algorithm as shown in the following figure (Fig. 20)

3.4.18 Radial coverage metric, ψ (Lewis et al. 2009)

This metric divides the objective space into radial sectors originating from the Utopia point. The value of this measure is given as the ratio of the sectors that contain at least one member of non-dominated solutions to the total number of sectors. The idea behind this metric is to emphasize that the solutions from the metaheuristic algorithm may not adequately or evenly cover the entire range of objective values, which may give a misleading estimate of the quality of the approximate PF. The mathematical formulation is shown as follows:

$$\Psi = \frac{1}{N} \sum_{i=1}^{m} \Psi_i, \tag{122}$$

with $\Psi = 1$ if $if P_i \in PF$ and $\alpha_{i-1} \leq \tan \frac{f_1(x)}{f_2(x)} \leq \alpha_m$;$\Psi = 0$ otherwise.

3.4.19 Diversity comparison indicator, DCI (Li et al. 2014)

This metric is a k-ary spread indicator that divides the zone in the objective space into several hyper-boxes. The coordinate of grids environment is constructed based on Nadir point, Ideal point as well as lower and upper bounds as follows:

$$ub_k = np_k + \frac{np_k - ip_k}{2 * div}, \tag{123}$$

with ub_k as the lower bound,np_k is the Nadir point,ip_k is the Ideal point and div as a constant referred to as the number of divisions with k as the number of objectives. The hyper-box size d_k of the kth objective is formalized as in (124).

$$d_k = \frac{ub_k - lb_k}{div}. \tag{124}$$

Further description of grids calculation can be observed in Li et al. (2014). For each hyper-box, a contribution coefficient is computed that represents the number of non-dominated solutions in each PF approximation as below:

$$CD(P,h) = \begin{cases} 1 - D(P,h)^2/(m+1), & if\ D(P,h) < \sqrt{m+1}, \\ 0, & if\ D(P,h) \geq \sqrt{m+1}, \end{cases} \tag{125}$$

with $D(P,h)$ as the shortest Euclidean distance from the hyper-box in the grid, h to the hyper-boxes in the approximation set P. The contribution factor is computed for all other hyper-boxes with respect to the reference set. The DCI metric is then evaluated by calculating the average of contribution coefficient relatively to all hyper-boxes as in the following equation:

$$DCI = \frac{1}{|h|} \sum_{i=1}^{|h|} CD\big(P, h_i\big). \tag{126}$$

3.4.20 Modified diversity indicator, M-DI (Asafuddoula et al. 2015)

As the name implies, the metric is a modification of DCI that computes the diversity with respect to the reference PF. In this method, the number of reference-points on reference PF is associated with the population size used during the optimization. The evaluation of the contributing factor remains the same as DCI method in Eq. (125) with h as the set of hyper-boxes from reference PF. The formulation of M-DI metric is as follows:

$$M - DI = \frac{1}{|h_{rPF}|} \sum_{i=1}^{|h_{rPF}|} CD\big(P, h_{rPFi}\big), \tag{127}$$

where h_{rPF} is the hyper-boxes from reference PF. A Higher M-DI value is preferable as it shows a better diversity of solution concerning PF. The unique improvement of M-DI over DCI is the modified version can demonstrate the uniformity of solutions concerning the reference PF point of view.

3.4.21 d_2 and d_{hyp} Metric (Asafuddoula et al. 2015)

The d_2 metric is referred to as the perpendicular distance of solution to the reference directions with a set of solution $P = \{p_1, p_2, \ldots, p_n\}$. The metric is defined by firstly evaluating the distance d_1 that represents the reference kth direction of W number of reference points and d_2 is evaluated based on the trigonometric formulation as shown in Eqs. (128) and (129) respectively.

$$d_{1,i}^k = p_i \frac{W^k}{\|W^k\|}, \tag{128}$$

$$d_2\big(W^k, P\big) = \min_{i=1} \left(\sqrt{\|p_i\|^2 - \left(d_{1,i}^k\right)^2} \right). \tag{129}$$

The d_{hyp} metric is referred to as the normal (or the shortest) distance of solutions to the hyperplane. This metric is useful to identify the nature of reference Pareto front, which may either convex, concave, or lie on the hyperplane itself with hyperplane, L as in the following equation:

$$d_{hyp}(p_1, L) = \frac{a_1 p_{11} + a_2 p_{12} + \cdots + a_n p_{1n} - 1}{\sqrt{a_1^2 + a_2^2 + \cdots + a_n^2}}, \tag{130}$$

with $a_1 f_1 + a_2 f_2 + \cdots + a_n f_n = 1$ as the generic formulation of the plane in objective space and $f_1, f_2, \ldots, f_n$ as well as $a_1, a_2, \ldots, a_n$ as the objectives and normal vector to the plane respectively. A positive value of d_{hyp} denotes a non-convex region, whereas negative d_{hyp} indicates a convex region and $d_{hyp} = 0$ shows that the point lies on the hyperplane.

3.4.22 Distribution metric, DM (Zheng et al. 2017)

This metric was proposed to improve the Spacing metric with additional information on the extent of PF. The Spacing metric did not normalize the measured distance, which may lead to a biased conclusion. Furthermore, the method also considers the closest neighbors and did not capture the holes in the PF. These drawbacks are improved by DM metric with the following formulation:

$$\mathrm{DM}(A) = \frac{1}{|A|} \sum_{i=1}^{m} \left(\frac{\sigma_i}{\mu_i} \right) \left(\frac{\left| f_i(P_G) - f_i(P_B) \right|}{R_i} \right), \tag{131}$$

with $\sigma_i = 1/(|A| - 2) \sum_{e=1}^{|A|-1} \left(d_e^i - \bar{d}^i \right)^2$;$\mu_i = 1/(|A| - 1) \sum_{e=1}^{|A|-1} d_e^i$; and $R_i = \max_{a \in A} f_i(a) - \min_{a \in A} f_i(a)$. The parameter $|A|$ is the number of non-dominated solutions,$f_i(P_G)$ and $f_i(P_B)$ are the function values of ideal and nadir points respectively,d_e^i as the distance of eth interval between two adjacent solutions that correspond to the ith objective, σ_i and μ_i represent the standard deviation and mean of the distances of the ith objective. Better distribution is denoted with a smaller DM value. This metric requires high computation time and it is more relevant for the continuous PF approximation.

3.4.23 Diversity vector based on reference vectors, DIR (Cai et al. 2018)

As the name indicated, this diversity metric is measured based on reference vectors with $V = \left\{ \lambda^1, \lambda^2, \ldots, \lambda^m \right\}$ that uniformly generated. Then for each approximation set $s \in S$, the distance between s and reference vector λ^i for $i = 1,2,\ldots,m$ is defined as in Eq. (132):

$$\mathrm{angle}\left(\lambda^i, F(s) \right) = \cos^{-1} \frac{\left(\lambda^i \right)^T \left(F(s) - F^I \right)}{\lambda^i F(s) - F^I}. \tag{132}$$

The closest reference vector λ^i to an element s of S is denoted as "s covers the reference vector λ^i" (Cai et al. 2018). Then a coverage vector c is defined by the number of reference vector that s covers for each $s \in S$ and the normalized standard deviation of the coverage vector, c is defined as follows:

$$\text{DIR} = \frac{\sqrt{\frac{1}{|S|}\sum_{i=1}^{|S|}\left(c_i - \bar{c}\right)^2}}{\frac{M}{|S|}\sqrt{|S|-1}}, \tag{133}$$

with $\bar{c}$ as the average of coverage vectors c_i with $i = 1, 2, \ldots, |S|$. This measure captures the distribution and spread of solutions, thus lower value denotes better performance. DIR metric is computationally cheap, but it requires the information of ideal points and the number of reference vectors needs to be properly selected. The method might be biased if PF is continuous (Audet et al. 2018).

3.5 Convergence and distribution measure

3.5.1 R-Metrics (Hansen and Jaskiewicz 1998)

The R-metrics is a binary metric. There are three R indicators (R_1, R_2, R_3) used in this measure based on a set of utility functions u. R_1 calculates the probability of approximation A that is better than B over the set of utility functions. The measurement of the R_1 metric is calculated as follows:

$$R_1(A, B, U, p) = \int_{u \in U} C(A, R, u)p(u)du, \tag{134}$$

with U as the utility functions, A and B as two approximations of Pareto set. u is a value that maps each point in the objective space into the measure of utility. $p(u)$ is an intensity function that shows the probability density of the utility $u \in U$. C is determined as in Eq. (135).

$$C(A, R, u) = \begin{cases} 1, & \text{if } u*(A) > u*(B), \\ 1/2, & \text{if } u*(A) = u*(B), \\ 0, & \text{if } u*(A) < u*(B). \end{cases} \tag{135}$$

Several advantages of R_1 is lower computation cost and is independent of scaling (Knowles and Corne 2002). Other R-metrics: R_2 and R_3 are defined as the following equations:

$$R_2(A, B) = \frac{\sum_{\lambda \in \Lambda} u(\lambda, A) - u(\lambda, B)}{|\Lambda|}, \tag{136}$$

$$R_3(A, B) = \frac{\sum_{\lambda \in \Lambda} [u(\lambda, B) - u(\lambda, A)]/u(\lambda, B)}{|\Lambda|}. \tag{137}$$

R_2 metric includes the expected values of the utility function by calculating the expected difference in the utility of an approximation A with B. The R_2 metric is compatible with all outperformance relations and able to differentiate between different levels of complete outperformance (refer to Table 5). The third R-metric,R_3 compares the ratio of the best utility values instead of difference as calculated in R_2. Knowles and Corne (2002) recommend R-metrics for evolutionary multi-objective researchers since it is compatible with outperformance relations and able to differentiate between levels of complete outperformance.

Furthermore, the R-metrics have less computational power even by increasing the number of objectives (Audet et al. 2018).

3.5.2 Hypervolume, HV (Zitzler and Thiele 1998), and hyperarea ratio, HR (Van Veldhuizen and Lamont 1999)

The hypervolume indicator is among the most used measures for MOP. It is defined as a volume in the objective function space that covered by $p_i(1 = 1, \ldots, N)$ of non-dominated set solutions. HV is a unary metric and is a union of hypercuboids that bounded by PF in the Lebesgue measure with the following notation:

$$HV(A, r) = \lambda_m \left(\bigcup_{z \in A} [z; r] \right), \tag{138}$$

with λ_m as the Lebesgue measure in m-dimension, r as reference point such that for all $z \in A, z \prec r$. A larger hypervolume shows a wider range of Pareto optimal solutions. Therefore maximization of HV is preferred. The notation of HV is also known as S-metric, Lebesgue measure (Riquelme et al. 2015), and Hyperarea. HV covers both accuracy and diversity of performance (Coello et al. 2010). Some of the drawbacks of this metric are the computation complexity that exponentially increased by the number of objectives (Bringmann and Friedrich 2010) and inconsistent of HV value by choosing a different reference point (Li and Yao 2019). For a known PF, a ratio of HV with respect to the known PF can be calculated and denoted as Hyperarea Ratio (Van Veldhuizen and Lamont 1999).

$$HR(A, PF, r) = \frac{HV(A, r)}{HV(P, r)}. \tag{139}$$

This ratio demonstrates the approximation quality, a lower ratio denotes a better approximation. The characteristic of HV that guarantees strict compliance of Pareto dominance made this indicator preferable compared to other metrics and is widely been used for measuring the performance of algorithms in MOPs. Yet, numerous improvements for HV measures were proposed. One of the improvements is by using a weight distribution function that serves to emphasize certain regions of the objective space (Zitzler et al. 2007). The author generalized the HV indicator as the integration of attainment function, α_A as $HV(A) = \int_{(0,\ldots,0)}^{(1,\ldots,1)} \alpha_A(z)dz$; with A as objective vector and A is weakly Pareto dominance,$A \succcurlyeq \{z\}$. The attainment function described here is a binary function that defines all weakly dominated objective vectors as 1 and other remaining objective vectors as 0. The proposed metric is then defined as the integral over the product of weight distribution function and the attainment function with $HV_w = \int_{(0,\ldots,0)}^{(1,\ldots,1)} w(z) \cdot \alpha_A(z)dz$ with w as the weight function. Other HV variant is proposed by Friedrich et al. (2011) that introduced a logarithmic version of HV as follows:

$$\log HV(A, r) = \lambda_m \left(\bigcup_{z \in A} \left[\log A; \log r \right] \right), \tag{140}$$

with $\log P := \{(\log x, \log y)(x, y) \in A\}$ and $\log r := \left(\log r_x, \log r_y\right)$. The reason for the logarithmic modification is to get a good multiplicative approximation, refer to Friedrich et al. (2011) for further readings. Another generalization of HV measure is defined as cone-based HV, defined as CHI (Emmerich et al., 2013). The method is formulated as follows:

$$CHI(A) = \lambda_m((A \oplus C) \cap (\{r\} \ominus C)), \tag{141}$$

with A as finite set, r as a reference point, and C as the pointed convex cone. The operators $\oplus$ and $\ominus$ are referred to as the Minkowski sum that defined as in Eq. (142).

$$\begin{aligned} a \oplus b &= \{a + b | a \in A \text{ and } b \in B\}, \\ a \ominus b &= \{a - b | a \in A \text{ and } b \in B\}. \end{aligned} \tag{142}$$

Another improvement is by projecting the solutions of an approximation onto a linear PF defined by a reference point and computes the HV metric to assess the diversity of the projections (Jiang et al. 2016). The proposed metric is defined as HV_d since it measures a better estimate of the diversity of the points along with the PF.

3.5.3 The Hyperarea difference, HD (Wu and Azarm 2001)

This metric evaluates the difference between the sizes of objective space dominated by an observed Pareto solution set with the size of the true Pareto set. The true Pareto set dominates the entire solution set, whereas the observed Pareto set might only dominate a portion of solution space. In other words, HD is a normalization of the dominated space of an approximated PF over a given rectangle.

3.5.4 Inverted generational distance, IGD (Coello and Sierra 2004)

IGD is an inverted variation of GD and exhibits a significantly different measure compared to GD. This metric has been used in numerous MOPs due to its lower computation cost and ability to show convergence, spread, and distribution of solutions. Thus, it reflects both efficiency and effectiveness. Among the important feature of IGD are: (1) distance measured are the minimum Euclidian distance, whereas GD measures the average distance, (2) distance calculation between two sets are based on reference solution in Pareto front and not the solution in A, (3) IGD can measure diversity and convergence of the algorithm. The formulation of IGD is shown as follows:

$$IGD = \frac{\sum_{p \in p^*} dist(p, A)}{|p^*|}, \tag{143}$$

where p^* is a set of reference points or uniformly distributed solutions in Pareto front, A is the non-dominated solutions from an algorithm, $dist(p, A)$ is the nearest distance from p to the solutions in A that is calculated with $dist(p, y) = \sqrt{\sum_{j=1}^{m} \left(p_j - y_j\right)^2}$. Nonetheless, IGD is not able to differentiate the quality of generated solutions when it is not dominated to the solutions in p^*. Furthermore, IGD requires a large number of reference points for reliable performance comparison and the number of required reference points exponentially increases with the number of objectives (Ishibuchi et al. 2015). This metric is also unable to detect poor distribution that depends on the value of p (Bezerra et al. 2017).

3.5.5 Modified inverted generational distance, IGD$^+$ (Ishibuchi et al. 2015)

The IGD$^+$ is proposed as to counter the problem of IGD by changing the calculation of $d(p, y)$ as shown in the following equation.

$$d(p,y) = \sqrt{\sum_{j=1}^{m} max\left(y_j - p_j, 0\right)^2}. \tag{144}$$

The modified distance is then used in the formulation of IGD^+ as follows:

$$IGD^+ = \frac{\sum_{z \in P} \min_{s \in S} d(p,y)}{|P|}, \tag{145}$$

with P as the dominated solution. This metric is applicable for both continuous and discrete domains.

3.5.6 Domination move, DoM (Li and Yao 2017)

This metric quantifies the minimum sum of move distance required such that the set B is weakly dominated. The metric is Pareto compliant and does not require additional problem knowledge and parameters. The formulation of dominance move for A to B is as follows:

$$DoM(A,B) = \min_{A' \preccurlyeq B} \sum_{i=1}^{n} d\left(a_i, a_i^{'}\right), \tag{146}$$

$$d\left(a_i, a_i^{'}\right) = \sum_{j=1}^{m} \left| a_i^j - a_i^{'j} \right|, \tag{147}$$

where $A = \left\{a_1, \ldots, a_n\right\}, A' = \left\{a_1^{'}, \ldots, a_n^{'}\right\}, a_i^j$ as the value of the solution a_i in the jth objective and m as the number of objectives. The DoM measure is applicable for bi-objective problems and more cases of more than two objectives are yet to be explored.

3.6 Knee points

Knee points is another criteria for the optimal solution of MOPs. The knee points are mainly divided into three categories: knees in convex regions, knees in concave regions, and edge knees. Further readings on knee points can be accessed in Bhattacharjee et al. (2016), Deb and Gupta (2011) and Yu et al. (2019). Some measures for knee points Yu et al. (2019) are the knee-driven GD (KGD), knee-driven IGD (KIGD), and knee-driven dissimilarity (KD). The KGD metric evaluates the convergence of obtained solutions to the reference points in the knee regions as shown in Eq. (148):

$$KGD = \frac{1}{|A|} \sum_{i=1}^{|A|} d\left(p_i, Z\right), \tag{148}$$

with $d\left(p_i, Z\right)$ as the Euclidian distance between the reference point p_i in A to closest reference point in Z. A smaller value of KGD denotes a better convergence to the knee region. This metric demonstrates the algorithm's capability of identifying solutions within the knee regions since outside of the knee regions will degrade the performance with increased KGD value The KIGD measure evaluates the diversity of achieved solutions in the knee region as in (149).

$$KIGD = \frac{1}{|Z|}\sum_{i=1}^{|Z|} d\left(p_i, A\right). \tag{149}$$

The formulation of KIGD is similar to KGD but with swapping the reference set and A. The term $d\left(p_i, A\right)$ refers to the Euclidean distance between the reference point p_i in Z and the solution closest to this reference point in A. Smaller KIGD shows the knee region is covered more evenly by the solutions. The third metric, KD evaluates the algorithm's ability to finding all knee points as follows:

$$KD = \frac{1}{|K|}\sum_{i=1}^{|K|} d\left(p_i, A\right), \tag{150}$$

with $d\left(p_i, A\right)$ as the Euclidian distance between true knee point p_i from K to its closest solution from A. KD indicate whether the solution set contains at least one solution close to the knee point.

More and more measures for multi-objective problems being proposed to improve the measurement criteria for these problems. Some examples are from Mirjalili and Lewis (2015) proposed three metrics (robust convergence, robust coverage metric, and robust success ratio) for robust multi-objective algorithms. Abouhawwash and Jameel (2019) proposed Benson's Karush–Kuhn–Tucker proximity measure (B-KKTPM) that determines a distance of solution from Pareto optimal in MOPs, However, this method has high computation cost. Another recently proposed metric is defined as the Domination Measure, DM (Hale et al. 2020). The method is integrated into an algorithmic search to measure the quality of MOP solutions by transforming the original problem into a stochastic single-objective problem with the goal of optimal solutions of $DM = 0$. DM is a unary performance indicator that measures the region in a solution space that dominates that solution with respect to a predefined probability measure. The formulation of DM is depicted as follows:

$$D(x) = \frac{v(Dx)}{v(X)} = \int \left\{y \prec_d x\right\} u(dy), \tag{151}$$

with $v(\cdot)$ as a Radon measure,Dx as the set of solutions that dominates x and $x \in X$, u is the uniform probability measure on X induced by $v(\cdot)$ and $y \prec_d x$ indicates that y dominates x with $\prec_d$ is referred to as the dominance relationship.

3.7 Data representation for MOPs metrics

All the data analyzed using the above metrics can be presented with statistical analysis as shown in Fig. 14. These indeed depend on the data distribution among the algorithms and metrics. Obviously, non-parametric analysis is appropriate for MOP (Coello et al. 2008) that is constructed with sets of solutions and unknown distribution of population metric. García and Herrera (2008) highlighted that for single-problem analysis, the parametric test usually obtains similar findings as a non-parametric test without proper fulfillment of parametric conditions. However, for multiple-problem analysis, a parametric test may reach an erroneous conclusion!

For comparison of a single metric with a defined number of algorithms, the Mann–Whitney rank-sum test can be adopted, whereas a comparison on a set of metrics is suitable using the Wilcoxon test (Coello et al. 2010). A recent performance indicator development with statistical analysis is the DSCTool (Eftimov et al. 2020). The tool provides Deep Statistical comparison, DSC for single and multiple problems by ranking the algorithms based on the

distribution of obtained solutions. One of the insights of this application is the accessibility with any programming language using REST web services

Another method to compare the evaluated metrics among algorithms is the performance score (Bader and Zitzler 2011) with the formulation: $P(Alg_i) = \sum_{j=1,j\neq i}^{k} \delta_{i,j}$. $\delta_{i,j} = 1$ if Alg_j is significantly better than Alg_i in the evaluated metric. $\delta_{i,j} = 1$ otherwise. The value $P(Alg_i)$ represents the number of algorithms that significantly better than the corresponding algorithm. Thus, a smaller index shows better performance and if $P(Alg_i) = 0$, it means that no other algorithms are significantly better than Alg_i. An example of this ranking method is shown in Gong et al. (2017) on the optimization of the communication system and Wang et al. (2019) on the performance comparison of the new proposed algorithm. Another unique approach is by designing the Pareto front with a defined confidence interval (Bassi et al. 2018), which is appropriate for the metaheuristic algorithms due to its stochastic nature, thus a Pareto front with $a\%$ confidence level and median Pareto front (drawn in the middle) may give good judgments on the metaheuristic algorithm quality.

3.8 Other specific metrics

Other performance metrics (as discussed in Sect. 2.2) can be evaluated statistically with either parametric or non-parametric depending on the analyzed data. For the Multi-objective Knapsack Problem (MKP), works of literature such as Chih et al. (2014) presented several MOPs measures such as Mean Absolute Deviation (MAD), Mean Average Percentage Error (MAPE), the average error of profit and Success Ratio (SR). Each of the instances is defined in the following equations:

$$MAD = \frac{1}{n} \sum_{n}^{i=1} |p_i - opt|, \tag{152}$$

with n as the number of test problems,p_i as the final solution of each run, and opt is the optimum solution of each test problem. The mean average percentage error is then calculated by deviating MAD with the optimum solution as follows:

$$MAPE = \frac{MAD}{opt}. \tag{153}$$

The average error of profit used to evaluate the algorithms are as per the following equation:

$$\overline{MKP}_{error} = \frac{1}{n} \sum_{i=1}^{n} \frac{z_i - p_i}{z_i} \times 100, \tag{154}$$

with z_i as the optimal profit and p_i as the profit calculated from the algorithm's best or mean. Another measure; success ratio (SR) is determined based on the optimal solution through the experimental runs.

3.9 Ensemble of metrics

As described in the previous sections, numerous performance indicators cover various characteristics of algorithm behavior such as cardinality, diversity, and convergence.

Assigning each performance indicator often may not result in a straightforward conclusion on which algorithm performs the best among the compared ones. Some researchers provide an ensemble technique or multicriteria analysis to consider a set of performance indicators, which then summarizes multiple quality of solutions together by the compared algorithms. There are many ensemble techniques proposed in the literature such as statistical-based ensembling, regression-based and voting-based methods. An example of the metrics ensembling method is from Yen and He (2014) that proposed an algorithm of ensembling method over n algorithms with a combination of five metrics: NR, IGD, SP, MS, and HV indicators. The ensemble algorithm ranks the n algorithms comprehensively through the collection of quality indicators and identifies an algorithm that resulted in the best approximation front through a double-elimination tournament selection. The winning algorithm is then removed from the list and the remaining approximation fronts are then compared through another round of double-elimination tournament with the $n-1$ algorithms. Another example of the ensemble method is the multi-criteria analysis based on cardinality, HV, SP, and set coverage metric by Janssens and Pangilinan (2010). Some recent approaches are such as from Yu et al. (2018) that designed a framework to compare 6 multi-objective algorithms with 5 performance measures and two variants of decision methods denoted as Multiple Criteria Decision Making, MCDM. The performance indicators for the framework include two convergence indicators: GD and MPFE, one diversity indicator: SP and two convergence and distribution indicators: IGD and HV. The MCDM or ensemble methods applied for framework decision making denoted as The Technique for Order of Preference by Similarity to Ideal Solution, TOPSIS (Hwang and Yoon 1981), and VIKOR (Opricovic 1998). As a side note, the name VIKOR was defined in Serbian and the English equivalent is defined as Multicriteria Optimization and Compromise Solution. The ranking system of the compared algorithm is based on the requirement of steps for each ensemble mode. Detailed information on both ensemble modes can be referred to in Yu et al. (2018). Other ensembles or MCDM schemes proposed in the works of literature are such as VTOPES, an abbreviation for 'VIKOR, TOPSIS Entropy, Standard deviation' (Deepa et al. 2019), AHP (Analytical Hierarchy Process) by Saaty (2004), ELECTRE (ELimmination and Choice Expressing REality) by Roy (1991) and Delphi method (Khorramshahgol and Moustakis 1988). Another recent ensemble method applied for performance criteria analysis is based on Deep Statistical Comparison, DSC (Eftimov and Kovec 2019, Eftimov et al. 2020). With this method, each algorithm obtains its ranking based on DSC analysis of the quality indicator by each problem. Then the ranking of algorithms is calculated using the ensemble combiner of the acquired rankings of each performance indicator. The ranking system is based on a standard competition ranking scheme as the following equation:

$$Rank_T = \text{Standard competition ranking}(Rank), \tag{155}$$

with $Rank$ as the $1 \times m$ vector of DSC rankings based on the quality indicator of a given problem, and $Rank_T$ as the $1 \times m$ vector of transformed DSC rankings using standard competition scheme (Eftimov and Kovec 2019). Eftimov et al. (2020) put forth the DSC ranking scheme is based on non-parametric distribution comparisons such as Anderson–Darling and Kolmogorov–Smirnov test. The author proposed two ensemble combiners. The first ensemble method based on the average of transformed ranking by each quality indicator of a problem. The lowest ranking is selected as the best performer. For data that contain outliers, the median can be used instead of average. The second ensemble method is based on the hierarchy of the majority vote. This method checks which algorithm wins in the

most quality indicator with the highest number of transformed DSC rankings. For the quality indicators, the author ensemble based on HV, I_ϵ, r_2 and GD indicators.

4 Future challenges

The complexity of either single- or multi-objective problems in both continuous and discrete domains increases as the advances of science and technology. Furthermore, the real-world problems often deal with unknown cost functions and uncertainty of constraints. A similar issue is also faced by a large number of feasible solutions or problems with a large scale, where the global optimum can't be found within a reasonable time due to the combinatorial explosion of the number of possible solutions. As a consequence, the findings for the correct solution of an algorithm with respect to the others are also challenging and require a proper justification as well as decisive performance assessments. To this extent, there are already numerous performance measures proposed in each problem type and yet, it is important for the researchers to understand the type of pertinent measures that may be selected as their analysis and to conclude on their findings. As discussed in the previous section, the scope of the performance evaluation must comprehend the efficiency and effectiveness aspect of the algorithms. Thus, regardless of problem types, both features of assessments need to be considered in the study. For the unknown global optimum problems, the best method is to compare the obtained solutions with the best-known solution found during the search by the respective algorithm or by other compared algorithms, even though it is unsure whether the best-known solution is globally optimum (Chopard and Tomassini 2018). The best known optimum or pseudo-optimal solution can be identified only in a relative measure of the best solution found by the current number of trials or FEVs concerning the previous solution. In a single objective problem, some of the relevant efficiency measures that suitable for the pseudo-optimal solution cover the convergence rate, and diversity of the algorithm solution. These are summarized as in the following points:

- The convergence rate as proposed by Senning (2015) in Eq. (2) is a typical application for unknown optimum as it measures the rate of convergence between current and previous FEVs. This method applies to both continuous and discrete domains.
- The relative difference of convergence between algorithms that measures the fraction of average value found in algorithm A to the best solution found in algorithm A and B as indicated in Eq. (6). Another metric for convergence difference between algorithm for discrete problems such as TSP is defined as the tour improvement as demonstrated by Tsai et al. (2004) in Eq. (28).
- The convergence speed for unknown global optimum can also be measured as a relationship with the number of steps such as FEVs, number of generations, and iteration. By calculating the fraction of the number of steps or FEVs that correspond to the best-known solution and divided by the total number of FEVs as shown in Eq. (7). The method applies to both continuous and discrete problems. Lower value denotes faster convergence.
- Another convergence related measure is the relative convergence between best values during the algorithm search. This measure was originally proposed for the known optimum solution as Eq. (4). Instead of measuring the difference between current best to

the known global optimum, the relative measures can also be defined between the current best of the current generation or FEVs concerning the best optimum found so far as follows: $E_{unknown\ optimum}(i) = 1 - \frac{F_{besti}}{f_{best}}$ with F_{besti} as the best solution by ith FEVs and f_{best} as the best-found solution so far. This may indicate the sensitivity of the algorithm towards converging to a better solution.

- The diversity measures that are usually applied to continuous problems can be used for unknown global optimum since it did not require the information of the predetermined optimum value, rather it concerns the spread of search agents in the problem space, and a good diversity may avoid premature convergence. Variants of the diversity measures from Eq. (14), (15), (18)–(25) are adaptable depending on the objective of the study. For discrete problems, such as TSP, diversity measures from Eq. (16), (17)–(26) are also applicable for unknown global optimum.
- The dynamic optimization problems, DOP are more or less suitable for the problem of unknown global optimum since the environment of objective space changes dynamically over time. This is contributed mainly by the frequency and the magnitude of environment changes (Mavrovouniotis et al. 2017; Herring et al. 2020) and measuring the algorithm performance concerning these factors are related to the speed of the algorithm to adapt the changes and how the algorithm behaves by the degree of changes that relates the diversity, accuracy and the distance of solution with respect to the changing objectives. Several metrics for DOP have been proposed in the literature such as *offline error* (Yang and Li 2010), *modified offline error* and *modified offline performance* (Branke 2002), *staged accuracy* and *adaptability* (Trojanowski and Michalewicz 1999), recovery rate, and absolute recover rate (Nguyen and Yao 2012) as discussed in the last segment of Sect. 2.1.1.

In the effectiveness measures of single optimization problems, the main question to solve is how close the converged solutions to the best value are.

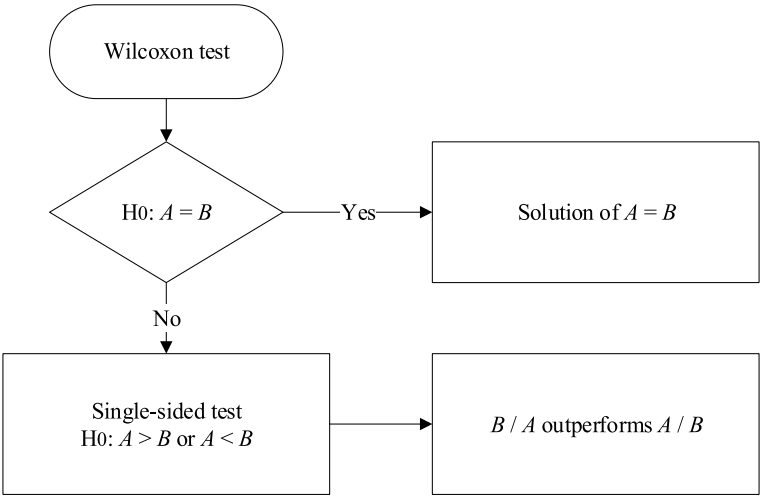


Fig. 21 Flow chart for comparing the effectiveness of two algorithms

- The effectiveness measures usually concentrated on the algorithm's ability to find the known global optimum. To the best of our knowledge, there are not many direct measures on the algorithm's effectiveness of locating global optimum for the case of the unknown cost function. However, the relative effectiveness between algorithms within a defined time frame or budget of evaluations can be compared using non-parametric statistical analyses such as Sign test, Wilcoxon test, or other multiple sample tests as summarized in Fig. 14. An example is a comparison using Wilcoxon's test between algorithms A and B as shown in Fig. 21. There are generally two tests to carry out: the first test is to check whether the median of A and B is the same with $H_0 : A = B$. The second test is a single-sided test used to find out which algorithm is more effective under a condition that if the first test is not met. Based on the hypothesis test, a conclusion can be made either B outperformed A or A outperformed B depending on the H_0 formulation.
- In some problems, the comparison of the current solution to the best-known solution can be approximated based on the theoretical lower bounds of the solution quality by using Lagrangian relaxation or integer programming relaxation. The converged solutions found by metaheuristic algorithms can be compared with these bounds (Chopard and Tomassini 2018). Generally, the Lagrangian relaxation is popularly been applied for solving numerous optimization problems such as in mixed-integer linear programming and combinatorial problems. The main idea behind this concept is to relax the hard constraints and unknown global optimum to solve the relaxed problems easily. The goal is to bound the unknown optimum value with lower and upper limit such that $f_{LB} \leq f_{opt} \leq f_{UB}$ with f_{LB} and f_{UB} as the lower and upper bounds respectively and f_{opt} as the unknown global optimum value. The distance between these bounds is defined as the optimality or duality gap that can be defined in percentage as $\textit{Optimality gap} = \left(f_{UB} - f_{LB}\right)/f_{UB} \times 100$. Therefore, the desired condition is to find the lowest optimality gap to find an optimal or near-optimal solution f_{opt}. One of the best methods is by applying a Lagrangian heuristic at each iteration to determine the feasible solution. Some examples of this concept are by Araya-Sassi et al. (2018) in optimizing the inventory location problem as well as Yu and Zhang (2014) in the NP-hard unit commitment problem.
- Another measurement is based on CDF such as relative stochastic dominance (Chicco and Mazza 2019). As a brief description, the method compares the CDF of the algorithm with respect to a reference CDF within H number of solutions and defined time frame. Then the area between algorithm CDF and reference CDF is determined and the metric OPISD is calculated as in Eq. (58). The OPISD value is then ranked in descending order that corresponds to the most effective to the least effective algorithm. The metric formulation is discussed in Sect. 2.2.2 and Eq. (58). A higher value of this metric denotes a better performance metric is inversely proportional to the area between the algorithm's CDF and the reference CDF. Other CDF based measures such as performance profile and data profile require the information of tolerance from the global optimum. The profile then displays a graphical chart of the number of solved problems with respect to the number of FEVs. Slight modification for the unknown optimum problem is by defining a tolerance using the best-known solution found by any compared algorithms. This best-known solution is then defined as superior to the ones found by other algorithms under study. This method is somehow applicable but it may bias towards the defined global optimum value.
- A comparison between the pseudo-optimal and the worst gained solution is another option for effectiveness measure. Lee et al. (2019b) proposed a ratio of optimum cost

obtained by an algorithm to the known worst solution defined as the improvement ratio in the constrained optimization problem. The average and standard deviation of this measure over n number of trials are used to compare with other algorithms. This metric is generally applicable to other optimization problems and higher average value reflects a better performance as the algorithm improves its search towards a better solution.

In MOP cases, the Pareto optimal front or PF usually cannot be calculated for real-world problems. For this reason, a reference set R is considered as an approximation of PF containing all non-dominated solutions. Not all MOP performance indicators are applicable when the knowledge of PF is not known. Some of the cardinality based measures require the knowledge of PF such as ONVG, ONVGR, and NVA metrics, whereas metrics such as C1R and C2R are applicable for unknown PF. The convergence based metrics usually also require the knowledge of PF but also can be based on the reference in the case of unknown PF (Riquelme et al. 2015). Several metrics that necessary to obtain the knowledge of PF are such as Δ_p, Pg, and Δ^*. Besides, Δ^* requires the knowledge of extreme solutions of PF (Audet et al. 2018).

5 Conclusion

It is well understood in the field of metaheuristic algorithms that a single run is not sufficient for a performance measure, rather it is necessary to analyze the algorithm' solution in n trials of runs. The performance of the algorithm is then analyzed in the scope of efficiency and effectiveness and it applies to numerous problem types that include single- and multiple-objectives as well as continuous and discrete problems. For single-objective problems, the main concern in the efficiency measure relates the rate of problem-solving that includes the convergence rate, search agents diversity, computational cost, complexity, and statistical measures such as cumulative distribution and ordered alternatives. Similar scope applies to the discrete single-objective problems. Some additional measures especially for combinatorial problems include runtime measures using FHT, sub-solution optimum rate, and tour improvement. In the effectiveness perspective, the overall measures for both single objective domains can be divided mainly into effectiveness rate, profile measures, scalability, 100-digit accuracy, and statistical measures that include descriptive and inferential. The option for statistical inference is further divided into frequentist and Bayesian tests.

Among the significant metrics implemented in various works of literature are the percentages of successful convergence either by the best solution as in Eq. (39) or the average of solutions in Eq. (40). It is also highlighted that using only the best solution as a performance indicator is not sufficient. The author should also include the average measures as well as the variability of the solution to describe the algorithm's effectiveness concerning the overall converged solutions. Comparing to other benchmarked algorithm as in Eq. (41) is also another supporting metric to understand the quality of the proposed algorithm with respect to other established algorithms. For the profiling measure, it is sufficient to select either one of the proposed methods (depending on the problem type) as discussed in Sect. 2. The profile measures enable the researcher to understand and compare the cumulative distribution of performance metric from a set of algorithms. For continuous problems with multiple dimensions, the scalability metric can be used to describe the relationship of the algorithm's performance concerning the number of dimensions. With this measure, the algorithm's ability (such as FEVs) on solving each dimension can be plotted against

the number of dimensions and can be compared between algorithms. On the other hand, the accuracy of converged solutions can be compared using 100-digits accuracy. One of the most crucial verification for the effectiveness measure is the statistical analysis and it is important to select the correct measurement for both descriptive and inferential analysis. In multi-objective optimization problems, the scope of efficiency and effectiveness are blended together in mainly five clusters of metrics: cumulative distribution based metric, cardinality, convergence, diversity, and convergence plus distribution. All of the metrics describe the solution distance, distribution, and accuracy with respect to the Pareto front or reference set. This measure is valid for both continuous and discrete problems and numerous metrics were introduced to measure the quality of solutions against Pareto front. However for multi-objective problems, one metric is not sufficient to resemble the algorithm performance. The comparison for multi-objective shall cover at least all five clusters of metrics to gain distinctive differences between algorithms. Some papers proposed an ensemble of metrics to combine the set of performance indicators, which then summarizes multiple quality of solutions together by the compared algorithms. Based on the review from each problem types, the analysis and reporting the results of algorithm solutions and comparisons are solely focused on tabular methods (such as statistical indices and percentage of improvements), trajectory plots (such as convergence and diversity curves) and ratio-based plots (such as performance, data, and other suitable profiles). Besides the comprehensive assessments of single- and multi-objective for continuous and discrete problems, this paper also reviews and proposed suitable measures for problems with unknown optimum or quasi-optimal solutions, which is practically significant in recent optimization problems with the advance of science and technology.

References

Abouhawwash M, Jameel MA (2019) In: Proceeding on 10th international conference EMO 2019. Springer, East Lansing, Mi, USA, pp 27–38

Adekanmbi O, Green P (2015) Conceptual comparison of population-based metaheuristics for engineering problems. Sci World J 936106:1–9

Agrawal AP, Kaur A (2016) An empirical evaluation of three popular meta-heuristics for solving travelling salesman problem. In: 6th International conference—cloud system and big data engineering (Confluence). Noida, India, pp 16–21

Akhand MAH, Ayon SI, Shahriyar SA, Siddique N, Adeli H (2020) Discrete spider monkey optimization for travelling salesman problem. Appl Soft Comput J 86:105887

Ali IM, Essam D, Kasmarik K (2020) A novel design of differential evolution for solving discrete travelling salesman problems. Swarm Evol Comput 52:100607

Ambati BK, Ambati J, Mokhtar MM (1991) Heuristic combinatorial optimization by simulated Darwinian evolution: a polynomial-time algorithm for the traveling salesman problem. Biol Cybern 65(1):31–35

Araya-Sassi C, Miranda PA, Paredes-Belmar G (2018) Lagrangian relaxation for an inventory location problem with periodic inventory control and stochastic capacity constraints. Math Probl Eng 8237925:1–27

Asafuddoula M, Ray T, Singh H (2015) Characterizing Pareto front approximations in many-objective optimization. In: Proceedings of the 2015 annual conference on genetic and evolutionary computation. ACM, pp 607–614

Askarzadeh A (2016) A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. Comput Struct 169:1–12

Aspnes J (2017) Notes on computational complexity theory CPSC 468/568 ch. 9. pp 58–59

Attiya I, Elaziz MA, Xiong S (2020) Job scheduling in cloud computing using a modified harris hawks optimization and simulated annealing algorithm. Comput Intell Neurosci 3504642:1–17

Audet C, Bigeon J, Cartier D, Le Digabel S, Salomon L (2018) Performance indicators in multiobjective optimization. Optimization Online. Retrieved from http://www.optimization-online.org/DB_HTML/2018/10/6887.html

Augusto OB, Rabeau S, Dépince Ph, Bennis F (2006) Multi-objective genetic algorithms: a way to improve the convergence rate. Eng Appl Artif Intell 19:501–510

Awad NH, Ali MZ, Suganthan PN, Liang JJ, Qu BY (2016) Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective real-parameter numerical optimization. Technical report

Aziz NAA, Mubin M, Ibrahim Z, Nawawi SW (2014) Performance and diversity of gravitational search algorithm. Adv Appl Converg Lett 3(1):232–235

Back T (1996) Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms. Oxford University Press, Oxford

Bader J, Zitzler E (2011) HypE: an algorithm for fast hypervolume-based many-objective optimization. Evol Comput 19(1):45–76

Bahameish HA (2014) Finding a cost effective LNG annual delivery program (ADP) using genetic algorithms. Master's thesis, Qatar University

Balogh J, Békési J, Dósa G, Sgall J, Stee R (2015) The optimal absolute ratio for online bin packing. In: SODA'15 Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms. Philadelphia, USA, pp 1425–1438

Barr RS, Golden BL, Kelly JP, Resende MG, Steward WR (1995) Designing and reporting on computational experiments with heuristic methods. J Heuristics 1:9–32

Bartz-Beielstein T (2005) New experimentalism applied to evolutionary computation. Ph.D. dissertation, Universitaet Dortmund, Fachbereich Informatik

Bassi M, Cursi ES, Pagnacco E, Ellaia R (2018) Statistics of the Pareto front in multi-objective optimization under uncertainties. Latin Am J Solids Struct 15(11):1–18

Beiranvand V, Hare W, Lucet Y (2017) Best practices for comparing optimization algorithms. Optim Eng 18(4):815–848

Benavoli A, Mangili F, Corani G, Zaffalon M, Ruggeri F (2014) A Bayesian Wilcoxon signed-rank test based on the Dirichlet process. In: Proceedings of the 30th international conference on machine learning (ICML 2014). pp 1–9

Benson HY, Shanno DF, Vanberei RJ (2000) Interior-point methods for nonconvex nonlinear programming: jamming and comparative numerical testing, technical report ORFE-00-02. Princeton University, Princeton

Bezerra LC, Ibánez ML, Stuetzel T (2017) An empirical assesment of the properties of inverted generational distance on multi- and many-objective optimization. In: EMO international conferecne on evolutionary multicriterion optimization. Springer, Muenster, Germany, pp 31–45

Bhattacharjee KS, Singh HK, Ray T (2016) A study on performance metrics to identify of interest from a trade-off set. In: Proceedings of the second Australasian conference on artificial life and computational intelligence, vol 9592. Springer, pp 66–77

Billups SC, Dirkse SP, Ferris MC (1997) A comparison of large scale mixed complementary problem solvers. Comput Optim Appl 7(1):3–25

Blanca MJ, Alarcón R, Arnau J, Bono R, Bendayan R (2017) Non-normal data: Is ANOVA still a valid option? Psicothema 29(4):552–557

Bogoya JM, Vargas A, Cuate O, Schütze O (2018) A (p, q)-averaged Hausdorf distance for arbitrary measurable sets. Math Comput Appl 23:51

Bolufé AR, González SF, Chen S (2015) A minimum population search hybrid for large scale global optimization. In: IEEE congress on evolutionary computation (CEC). Sendai, pp 1958–1965

Bongartz I, Conn AR, Gould NIM, Saunders MA, Toint PL (1997) A numerical comparison between the LANCELOT and MINOS packages for large-scale numerical optimization. Technical report 97/13, Namur University

Bonyandi MR, Michalewicz Z, Barone L (2013) The travelling thief problem: the first step in the transition from theoretical problems to realistic problems. In: IEEE congress on evolutionary computation. Cancun, pp 1037–1044

Bonyandi MR, Michalewicz Z, Przybylek MR, Wierbicki A (2014) Socially inspired algorithms for the travelling thief problem. In: GECCO'14 proceedings of the 2014 annual conference on genetic and evolutionary computation. pp 421–428

Boryczka U, Szwarc K (2019) The harmony search algorithm with additional improvement of harmony memory for asymmetric travelling salesman problem. Expert Syst Appl 122:43–53

Bozkurt B, Fowler JW, Gel ES, Kim B, Köksalan M, Wallenius J (2010) Quantitative comparison of approximate solution sets for multicriteria optimization problems with weighted Tchebycheff preference function. Oper Res 58(3):650–659
Branke J (2002) Evolutionary optimization in dynamic environments. Springer, New York, pp 13–29
Bringmann K, Friedrich T (2010) Approximating the volume of unions and intersections of high-dimensional geometric objects. Comput Geom 43(6–7):601–610
Brockhoff D, Auger A, Hansen N, Tušar T (2017) Quantitative performance assesment of multiobjective optimizers: the average runtime attainment function. In: Trautman H et al (eds) Evolutionary multicriterion optimization. EMO 2017 Lecture notes in computer science, 10173. Springer, Cham
Brownlee J (2011) Clever algorithms: nature-inspired programming recipes. Lulu, pp 404–405
Cai X, Sun H, Fan Z (2018) A diversity indicator based on reference vectors for many-objective optimization. Inf Sci 430:467–486
Calvo B, Shir OM, Ceberio J, Doerr C, Wang H, Bäck T, Lozano JA (2019) Bayesian performance analysis for black-box optimization benchmarking. In: proceedings of the genetic and evolutionary computation conference companion, GECCO 19, ACM, New York, NY, USA pp 1789–1797
Campuzano G, Obreque C, Aguayo MM (2020) Accelerating the Miller–Tucker–Zemlin model for the asymmetric travelling salesman problem. Expert Syst Appl 148:113229
Cardona JGF, Coello CAC (2020) Indicator-based multi-objective evolutionary algorithms: a comprehensive survey. ACM Comput Surv 53(2):1–35
Carlyle WM, Fowler JW, Gel ES, Kim B (2003) Quantitative comparison of approximation solution sets for bi-criteria optimization problems. Decis Sci 34(1):63–82
Carrasco J, García S, del Mar Rueda M, Herrera F (2017) rNPBST: an R package covering non-parametric and Bayesian statistical tests. In: International conferecne hybrid artificial intelligent systems. pp 281–292
Carrasco J, García S, Rueda M, Das S, Herrera F (2020) Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: practical guidelines and a critical review. Swarm Evol Comput 54:100665
Chang KH (2015) Design theory and methods using CAD/CAE. The computer aided engineering design series, Ch. 5—multiobjective optimization and advanced topics. pp 325–406
Chen WN, Zhang J, Chung HSH, Zhong WL, Wu WG, Shi YH (2010) A novel set-based particle swarm optimization method for discrete optimization problems. IEEE Trans Evol Comput 14(2):278–300
Cheng S, Shi Y, Qin Q, Zhang Q, Bai R (2014) Population diversity maintenance in brain storm optimization algorithm. J Artif Intell Soft Comput Res 4(2):83–97
Chi Y, Sun F, Jiang L, Yu C (2012) An efficient population diversity measure for improved particle swarm optimization algorithm. In: 6th IEEE international conference intelligent systems. Sofia, Bulgaria, pp 361–367
Chiarandini M, Paquete L, Preuss M, Ridge E (2007) Experiments on metaheuristics: methodological overview and open issues. Technical report DMF-2007-03-003, the Danish mathematical society, Denmark
Chicco G, Mazza A (2019) Heuristic optimization of electrical energy systems: refined metrics to compare the solutions. Sustain Energy Grids Netw 17:10097
Chih M, Lin CJ, Chern MS, Ou TY (2014) Particle swarm optimization with time-varying acceleration coefficients for the multimodal knapsack problem. Appl Math Mod. 38:1338–1350
Chopard B, Tomassini M (2018) Performance and limitations of metaheuristics. In: An introduction to metaheuristics for optimization. Natural computing series. Springer, Cham. https://doi.org/10.1007/978-3-319-93073-2_11
Coello CA, Sierra MR (2004) A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm. In: MICAI 2004: advances in artificial intelligence (LNCS 2972). Springer, Mexico City, pp 688–697
Coello CA, Lamont GB, Veldhuizen DAV (2008) Evolutionary algorithms for solving multi-objective problems, 2nd edn. Springer, New York
Coello CA, Dhaenens C, Jourdan L (2010) Multi-objective combinatorial optimization: problematic and context. In: Advances in multi-objective nature inspired computing, SCI272. Springer, pp 1–21
Collette Y, Siarry P (2005) Three new metrics to measure the convergence of metaheuristics towards the Pareto frontier and the aesthetic of a set of solutions in biobjective optimization. Comput Oper Res 32(4):773–792
Conn AR, Gould NIM, Toint PL (1996) Numerical experiments with the LANCELOT package (release a) for large-scale nonlinear optimization. Math Program 73:73–110
Conover J (1980) Practical nonparametric statistics. Wiley, Hoboken

Corani G, Benavoli A (2015) A Bayesian approach for comparing cross-validated algorithms on multiple data sets. Mach Learn 100(2–3):285–304
Cormen TH, Leiserson CE, Rivest RL, Stein C (2001) Introduction to algorithms. MIT Press, Cambridge, p 47
Crainic TG, Mancini S, Perboli G, Tadei R (2011) Multi-start heuristics for the two-echelon vehicle routing problem. In: Merz P, Hao JK (eds) Evolutionary computation in combinatorial optimization. EvoCOP 2011. Lecture notes in computer science, vol 6622. Springer, Berlin, Heidelberg
Cubukcuoglu C, Tasgetiren MF, Sariyildiz IS, Gao L, Kucukvar M (2019) A memetic algorithm for the bi-objective quadratic assignment problem. Procedia Manuf 39:1215–1222
Czyzak P, Jaskiewicz A (1998) Pareto simulated annealing—a metaheuristic technique for multiple-objective combinatorial optimization. J Multi-Criteria Decis Anal 7:34–47
Dahmani I, Hifi M, Saadi T, Yousef L (2020) A swarm optimization-based search algorithm for the quadratic knapsack problem with conflict graphs. Expert Syst Appl 148:113224
Das S (2018) Evaluating the swarm intelligence algorithms for continuous optimization—a (non-parametric) statistical perspective. In: ICSI, Shanghai, China
Das S, Abraham A, Chakraborty UK, Konar A (2009) Differential evolution using a neighborhood-based mutation operator. IEEE Trans Evol Comput 13(3):526–553
Dawoud SD, Peplow R (2010) Digital system design-use of microcontroller. River Publishers series in signal, image and speech processing. River Publishers, Gistrup
De Sousa Santos L, Secchi AR, Prata DM, Biscaica EC Jr (2019) An adaptive sequential wavelet-based algorithm developed for dynamic optimization problems. Comput Chem Eng 121:465–482
Deb K, Gupta S (2011) Understanding knee points in bicriteria problems and their implications as preferred solutions principles. Eng Optim 43(11):1175–1204
Deb K, Jain S (2002) Running performance metrics for evolutionary multi-objective optimizations. In: Proceedings of the fourth Asia-pacific conference on simulated evolution and learning (SEAL'02). Singapore, pp 13–20
Deb K, Pratap A, Agrawal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197
Deepa N, Vincent DR, Kumar SM, Srinivasan K, Chang CY, Bashir AK (2019) An efficient ensemble VTOPES multi-criteria decision-making model for sustainable sugarcane farms. Sustainability 11:4288
Degertekin SO, Lamberti L, Hayalioglu MS (2016) Heat transfer search algorithm for sizing optimization of truss structures. Latin Am J Solids Struct 14(3):373–397
Derrac J, García S, Hui S, Suganthan PN, Herrera F (2014) Analysing convergence performance of evolutionary algorithms: a statistical approach. Inf Sci 289:41–58
Dhivyaprabha TT, Subashini P, Krishnaveni M (2018) Synergistic fibroblast optimization: a novel nature-inspired computing algorithm. Front Inform Technol Electron Eng 19(7):825–833
Dilettoso E, Rizzo SA, Salerno N (2017) A weakly pareto compliant quality indicator. Math Comput Appl 22(1):25
Dixon WJ, Mood AM (1946) The statistical sign test. J Am Stat Assoc 41(236):557–566
Dolan E, Moré JJ (2002) Benchmarking optimization software with performance profiles. Math Program 91:201–213
Durillo JJ, Nebro AJ, Luna F, Coello CSC, Alba E (2010) Convergence speed in multi-objective metaheuristics: efficiency criteria and empirical study. Int J Numer Methods Eng 84:1344–1375
Eftimov T, Kovec D (2019) Performance measures fusion for experimental comparison of methods for multi-label classification. In: AAAI 2019 Spring symposium on combining machine learning with knowledge engineering. Palo Alto, California, USA
Eftimov T, Petelin G, Korošec P (2020) DSCTool: a web-service-based framework for statistical comparison of stochastic optimization algorithms. Appl Soft Comput J 87:105977
Ehrgott M (2005) Multicriteria optimization. Springer, Berlin, Heidelberg, pp 7–8
El-Ghandour HA, Elbeltagi E (2018) Comparison of five evolutionary algorithms for optimization of water distribution networks. J Comput Civ Eng 32(1):04017066. https://doi.org/10.1007/s00366-019-00718-z
Ellison SLR, Barwick VJ, Farrant TJD (2009) Practical statistics for the analytical scientist a bench guide, 2nd edn. Chapter 5 outliers in analytical data. RSC Publishing, Cambridge, pp 48–58
Embedded Staff (2011) CoreMark: A realistic way to benchmark CPU performance. Embedded. https://www.embedded.com/coremark-a-realistic-way-to-benchmark-cpu-performance/
Emmerich and Deutz (2018) A tutorial on multiobjective optimization: fundamentals and evolutionary methods. Nat Comput 17:585–609

Emmerich M, Deutz A, Kruisselbrink J, Shukla PK (2013) Cone-based hypervolume indicators: construction, properties, and efficient computation. In: Purshouse RC, Flemming PJ, Fonseca CM, Grego S, Shaw J (eds) Evolutionary multi-criterion optimization EMO 2013, lecture notes in computer science, vol 7811. Springer, Berlin, Heidelberg

Epitropakis MG, Tasoulis DK, Pavlidis NG, Plagionakos VP, Vrahatis MN (2011) Enhancing differential evolution utilizing proximity-based mutation operators. IEEE Trans Evol Comput 15(1):99–119

Fahoome GF (2002) Twenty nonparametric statistics and their large sample approximations. J Mod Appl Stat Methods 1(2):248–268

Farhang-Mehr A, Azarm S (2002) Diversity assesment of pareto optimal solution sets: an entropy approach. In: Proceedings of IEEE congress on evolutionary computation CEC 2002. pp 723–728

Fenet S, Solnon C (2003) Searching for maximum cliques with ant colony optimization. In: Cagnoni S et al (eds) Applications of evolutionary computing. EvoWorkshops 2003. Lecture notes in computer science, vol 2611. Springer, Berlin, Heidelberg

Ferrer J, Chicano F, Alba E (2012) Evolutionary algorithms for the multi-objective test data generation problem. Softw Pract Exp 42(11):1–31

Filipič B, Tušar T (2018) A taxonomy of methods for visualizing pareto front approximations. In: Proceedings of the genetic and evolutionary computation conference on—GECCO'18. p 649

Fomeni FD, Kaparis K, Letchford AN (2020) A cut-and-branch algorithm for the quadratic knapsack problem. Discrete Optim. https://doi.org/10.1016/j.disopt.2020.100579

Fonseca CM, Fleming PJ (1996) On the performance assessment and comparison of stochastic multiobjective optimizers. In: Proceeding of the 4th international conference on parallel problem solving from nature PPSN IV. Springer, London, pp 584–593

Friedrich T, Bringmann K, Voβ T (2011) The logarithmic hypervolume indicator. In: Proceedings of the 11th workshop on foundations of genetic algorithms. pp 81–92

García S, Herrera F (2008) Design of experiments in computational intelligence: on the use of statistical inference. In: Corchado E, Abraham A, Pedrycz W (eds) Hybrid artificial intelligence systems HAIS 2008. Lecture notes in computer science, vol 5271. Springer, Berlin, Heidelberg

García S, Molina D, Lozano M, Herrera F (2009) A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behavior: a case study on the CEC'2005 special session on real parameter optimization. J Heuristics 15:617–644

Garcia S, Fernández A, Luengo J, Herrera F (2010) Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power. Inf Sci 180(10):2044–2064

García J, Altimiras F, Pena A, Astorga G, Peredo O (2018) A binary cuckoo search big data algorithm applied to large-scale crew scheduling problems. Complexity 8395193:1–15

Goh CK, Tan KC (2009) A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization. IEEE Trans Evol Comput 13(1):103–127

Gong M, Wang Z, Zhu Z, Jiao L (2017) A Similarity-based multiobjective evolutionary algorithm for deployment optimization of near space communication system. IEEE Trans Evol Comput 21(6):878–897

Grange A, Kacem I, Martin S (2018) Algorithms for the bin packing problem with overlapping items. Comput Ind Eng 115:331–341

Grishagin VA (1978) Operation characteristics of some global optimization algorithms. Prob Stoch Search 7:198–206 **(in Russian)**

Gunantara N (2018) A review of multi-objective optimization: methods and its applications. Congent Eng 5(1):1502242

Gunantara N, Hendrantoro G (2013) Multi-objective cross layer optimization for selection of cooperative path pairs in multihop wireless ad hoc networks. J Commun Softw Syst 9(3):170

Hale JG, Zhu H, Zhou E (2020) Domination measure: a new metric for solving multiobjective optimization. INFORMS J Comput 32(3):1–17

Halim AH, Ismail I (2019) Combinatorial optimization: comparison of heuristic algorithms in travelling salesman problem. Arch Comput Methods Eng 26(2):367–380

Hamacher K (2007) Adaptive extremal optimization by detrended fluctuation analysis. J Comput Phys 227(2):1500–1509

Hamacher K (2014) Online performance measures for metaheuristic optimization, hybrid metaheuristics, vol 8457. Lecture notes in computer science. Springer, Cham

Hampel FR (1974) The influence curve and its role in robust estimation. J Am Stat Assoc 69:383–393

Hansen MP, Jaskiewicz A (1998) Evaluating the quality of approximations to the non-dominated set. Technical report IMM-REP-1998-7. Institute of Mathematical Modelling, Technical University of Denmark, Lyngby

Hansen N, Auger A, Finck S, Ros R (2012) Real-parameter blackbox optimization benchmarking: experimental setup. Technical report INRIA Futurs, Équipe TAO, Univ. Paris Sud, Orsay, France
Hansen N, Auger A, Brockhoff D, Tusar D, Tusar T (2016) COCO: performance assessment. arXiv :1605.03560 Retrieved from https://arxiv.org/abs/1605.03560
Hare WL, Sagastizábal C (2006) Benchmark of some nonsmooth optimization solvers for computing nonconvex proximal points. Pac J Optim 2(3):545–573
He J (2016) An analytic expression of relative approximation error for a class of evolutionary algorithms. In: Proceedings of IEEE congress on evolutionary computation. pp 4366–4373
He J, Lin G (2016) Average convergence rate of evolutionary algorithms. IEEE Trans Evol Comput 20(2):316–321
Heinonen J (2001) Lectures on Analysis on Metric Spaces. Springer, New York, USA
Heliodore F, Nakib A, Ismail B, Ouchraa S, Schmitt L (2017) Metaheuristic for intelligent electrical networks, ch: performance evaluation of metaheuristics. Wiley, Hoboken, pp 43–58
Hellwig M, Beyer HG (2019) Benchmarking evolutionary algorithms for single-objective real-valued constrained optimization–a critical review. Swarm Evol Comput 44:927–944
Hendtlass T (2004) An introduction to collective intelligence. In: Fulcher J (ed) Applied intelligent systems: new direction, studies in fuzziness and soft computing, vol 153. Springer, Berlin, Heidelberg
Hendtlass T, Randall M (2001) A survey of ant colony and particle swarm meta-heuristics and their applications to discrete optimization problems. In: Proceedings of the inaugural workshop on artificial life. pp 15–25
Herring D, Kirley M, Yao X (2020) Dynamic multi-objective optimization of the travelling thief problem. arXiv preprint: https://arxiv.org/abs/2002.02636
Hoos HH (1998) Stochastic local search—methods, models, applications. Ph.D. dissertation. Technical University Darmstadt, Germany
Hwang C, Yoon K (1981) Multiple attribute decision making: methods and applications, a state of the art survey. Springer, New York
Ibánez ML, Paquete L, Stuetzle T (2010) Exploratory analysis of stochastic local search algorithms in biobjective optimization. In: Bielstein TB, Chiarandini M, Paquete L, Preus M (eds) Experimental methods for the analysis of optimization algorithms. Springer, Berlin, pp 209–222
Ibrahim D. 2019. Arm-based microcontroller projects using Mbed, Chapter 3—the arm microcontrollers. Newnes, pp 25–41
Ishibuchi H, Masuda H, Nojima Y (2015) A study on performance evaluation ability of a modified inverted generational distance indicator. In: Proceedings of the 2015 annual conference on genetic and evolutionary computation. ACM, Madrid, Spain, pp 695–702
Ivkovic N, Jakobovic D, Golub M (2016) Measuring performance of optimization algorithms in evolutionary computation. Int J Mach Learn Comput 6(3):167–171
Jackson RHF, Boggs PT, Nash SG, Powell S (1991) Guidelines and reporting results of computational experiments: report of the ad hoc committee. Math Program 49:413–425
Jalili S, Husseinzadeh Kashan A, Hosseinzadeh Y (2016) League championship algorithms for optimum design of pin-jointed structures. J Comput Civ Eng. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000617
Janssens GK, Pangilinan JM (2010) Multiple criteria performance analysis of non-dominated sets obtained by multi-objective evolutionary algorithm for optimisation. In: 6th IFIP WG 12.5 international conference on artificial intelligence applications and innovations (AIAI). Larnaca, Cyprus, pp 94–103
Jena UK, Das PK, Kabat MR (2020) Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment. J King Saud Univ Comput Inf Sci. https://doi.org/10.1016/j.jksuci.2020.01.012
Jiang S, Ong YS, Zhang J, Feng L (2014) Consistencies and contradictions of performance metrics in multiobjective optimization. IEEE Trans Cybern 44(12):2391–2404
Jiang S, Yang S, Li M (2016) On the use of hypervolume for diversity measurement of Pareto front approximations. In: 2016 IEEE symposium series on computational intelligence (SSCI). Athens, pp 1–8
Jonckheere AR (1954) A distribution-free k-sample test against ordered alternatives. Biometrika 41:133–145
Kaur D, Murugappan M (2008) Performance enhancement in solving travelling salesman problem using hybrid genetic algorithm. In: Annual meeting of the North American fuzzy information processing society. New York City, NY, pp 1–6
Khorramshahgol R, Moustakis VS (1988) Delphi hierarchy process (DHP): a methodology for priority setting derived from the Delphi method and analytical hierarchy process. Eur J Oper Res 37:347–354
Kiliç H, Yüzgeç U (2019) Tournament selection based antlion optimization algorithm for solving quadratic assignment problem. Eng Sci Technol Int J 22:673–691

Kim JH, Lee HM, Jung D, Sadollah A (2016) Performance measures of metaheuristic algorithms. In: Kim J, Geem Z (eds) Harmony search algorithm. Advances in intelligent systems and computing, vol 382. Springer, Berlin, Heidelberg

Kiran MS (2017) Particle swarm optimization with a new update mechanism. Appl Soft Comput 60:670–678

Knowles JD, Corne D (2002) On metrics for comparing nondominated sets. In: Proceeding of the 2002 congress on evolutionary computation (CEC 2002). pp 711–716

Krink T, Vesterstrøm JS, Riget J (2002) Particle swarm optimisation with spatial particle extension. In: Proceedings of the congress on evolutionary computation, CEC'02. Honolulu, USA, pp 1474–1479

Laumanns M, Zitzler E, Thiele L (2000) A unified model for multi-objective evolutionary algorithms with elitism. In: Evolutionary computation proceedings of the 2000 congress on, vol 1. pp 46–53

Lee HM, Jung D, Sadollah A, Yoo DG, Kim JH (2019a) Generation of benchmark problems for optimal design of water distribution systems. Water 11:1637

Lee HM, Jung D, Sadollah A, Lee EH, Kim JH (2019b) Performance comparison of metaheuristic optimization algorithms using water distribution system design benchmarks. In: Yadav N, Yadav A, Bansal J, Deep K, Kim J (eds) Harmony search and nature inspired optimization algorithms. Advances in intelligent systems and computing, vol 741. Springer, Singapore

Lehre PK, Witt C (2011) Finite first hitting time versus stochastic convergence in particle swarm optimisation. arXiv:1105.5540v1. Retrieved from https://arxiv.org/pdf/1105.5540v1.pdf

Leung Y, Wang Y (2003) U-measure: a quality measure for multiobjective programming. IEEE Trans Syst Man Cybern Part A Syst Hum 33(3):337–343

Lewis A, Mostaghim S, Scriven I (2009) Asynchronous multi-objective optimisation in unreliable distributed environments. In: Lewis A, Mostaghim S, Randall M (eds) Biologigally-Inspired optimisation methods, vol 210. Springer, Berlin, Heidelberg, pp 51–78

Li Z, Hu X (2011) The orienteering problem with compulsory nodes and time window. ICSSSM11, Tianjin, pp 1–4

Li M, Yao X (2017) Dominance move: a measure of comparing solution sets in multiobjective optimization. arXiv preprint arXiv:1702.00477

Li M, Yao X (2019) Quality evaluation of solution sets in multiobjective optimisation: a survey. ACM Comput Surv 52(2):26

Li X, Tang K, Omidvar MN, Yang Z and Qin K (2013) Benchmark functions for the CEC'2013 special session and competition on large-scale global optimization. Technical report

Li M, Yang S, Liu X (2014) Diversity comparison of Pareto front approximations in many-objective optimization. IEEE Trans Cybern 44(12):2568–2584

Liang J, Runarsson TP, Mezura-Montes E, Clerc M, Suganthan P, Coello CC, Deb K (2006) Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization. J Appl Mech 41(8):1–24

Liang JJ, Qu BY, Suganthan PN (2013) Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization. Technical report 201311, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Technical Report, Nanyang Technological University, Singapore

Liu X, Zhang D (2019) An improved SPEA2 algorithm with local search for multi-objective investment decision-making. Appl Sci 9(8):1675. https://doi.org/10.3390/app9081675

Liu S, Qiu Z, Xie L (2017a) Convergence rate analysis of distributed optimization with projected subgradient problem. Automatica 83:162–169

Liu Q, Chen WN, Deng JD, Gu T, Zhang H, Yu Z, Zhang J (2017b) Benchmarking stochastic algorithms for global optimization problems by visualizing confidence intervals. IEEE Trans Cybern 47(9):2924–2937

Lo M, Liang Y, Hsieh J (2010) A modified variable neighborhood search algorithm for orienteering problems. In: 40th International conferecne on computers and industrial engineering. Awaji, pp 1–6

MacFarland TW, Yates JM (2016) Mann–Whitney U test. In: Introduction to nonparametric statistics for the biological sciences using R. Springer, Cham

Martí L, García J, Berlanga A, Molina JM (2016) A stopping criterion for multi-objective optimization evolutionary algorithms. Inf Sci 367:700–718

Mavrovouniotis M, Li C, Yang S (2017) A survey of swarm intelligence for dynamic optimization: algorithms and applications. Swarm Evol Comput 33:1–17

McGeoh CC (1996) Toward an experimental method for algorithm simulation INFORMS. J Comput 8(1):1–11

Memari A, Ahmad R, Rahim Abdul AR (2017) Metaheuristic algorithms: guidelines for implementation. J Soft Comput Decis Support Syst 4(6):1–6

Meng H, Zhang X, Liu S (2005) New quality measures for multiobjective programming. In: Wang L, Chen K, Ong YS (eds) Advances in natural computation. ICNC 2005, Lecture notes in computer science, vol 3611. Springer, Berlin, Heidelberg
Merz P, Freisleben B (2000) Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. IEEE Trans Evol Comput 4(4):337–352
Messac A, Mattson CA (2004) Normal constraint method with guarantee of even representation of complete pareto frontier. AIAA J 42(10):2101–2111
Mezura-Montes E, Miranda-Varela ME, Gómez-Ramón C (2010) Differential evolution in constrained numerical optimization: an empirical study. Inf Sci 180:4223–4262
Minella G, Ruiz R, Ciavotta M (2011) Restarted Iterated Pareto greedy algorithm for multi-objective flowshop scheduling problems. Comput Oper Res 38:1521–1533
Ming L, Wang Y, Cheung YM (2006) On convergence rate of class of genetic algorithms. In: Proceedings of 2006 world automation congress IEEE. Budapest, Hungary, pp 1–6
Mirjalili S, Lewis A (2015) Novel performance metrics for robust multi-objective optimization algorithms. Swarm Evol Comput 21:1–23
Monteiro RDC, Ortiz C, Svaiter BF (2016) An adaptive accelerated first-order method for convex optimization. Comput Optim Appl 64:31–73. https://doi.org/10.1007/s10589-015-9802-0
Mora-Melia D, Iglesias-Rey PL, Martinez-Solano FJ, Ballesteros-Pérez P (2015) Efficiency of evolutionary algorithms in water network pipe sizing. Water Resour Manag 29:4817–4831
Moré JJ, Wild SM (2009) Benchmarking derivative-free optimization algorithms. SIAM J Optim 20(1):172–191
Mortazavi A, Toğan V, Moloodpoor M (2019) Solution of structural and mathematical optimization problems using a new hybrid swarm intelligence optimization algorithm. Adv Eng Softw 127:106–123
Murata T, Ishibuchi H, Tanaka H (1996) Multi-objective genetic algorithm and its application to flow-shop scheduling. Int J Comput Eng 30(4):957–968
Nagata Y (2006) Fast EAX algorithm considering population diversity for traveling salesman problems. In: Gottlieb J, Raidl GR (eds) Evolutionary computation in combinatorial optimization. EvoCOP 2006. Lecture notes in computer science, vol 3906. Springer, Berlin, Heidelberg
Nakib A, Siarry P (2013) Performance analysis of dynamic optimization algorithms. In: Alba E, Nakib A, Siarry P (eds) Metaheuristic for dynamic optimization, recent research on metaheuristics for dynamic optimization, studies in computational intelligence. Springer, Berlin
Neave HR, Worthington PL (1988) Distribution free tests. Unwin Hyman Inc, Boston
Nebro AJ, Durillo JJ, Nietp JG, Coello CA, Luna F, Alba E (2009) SMPSO: A new PSO-based metaheuristic for multi-objective optimization
Nesmachnow S (2014) An overview of metaheuristics: accurate and efficient methods for optimization. Int J Metaheuristics 3(4):320–347
Ngamtawee R, Wardkein P (2014) Simplified genetic algorithm: simplify and improve RGA for parameter optimizations. Adv Electr Comput Eng 14(4):55–64
Nguyen T, Yao X (2012) Continuous dynamic constrained optimization—the challenges. IEEE Trans Evol Comput 16(6):769–786
Nguyen T, Yang S, Branke J (2012) Evolutionary dynamic optimization: a survey of the state of the art. Swarm Evol Comput 6:1–24
Nishida K, Aguirre HE, Saito S, Shirakawa S, Akimoto Y (2018) Parameterless stochastic natural gradient method for discrete optimization and its application to hyper-parameter optimization for neural network. arXiv:1809.06517. Retrieved from https://arxiv.org/pdf/1809.06517.pdf
Noergaard T (2013) Chapter 4—embedded processors, embedded systems architecture, 2nd edn. Elsevier, Waltham, pp 137–229
Obagbuwa IC, Adewumi AO (2014) An improved cockroach swarm optimization. Sci World J 2014:375358. https://doi.org/10.1155/2014/375358
Okabe T, Jin Y, Sendhoff B (2003) A critical survey of performance indices for multi-objective optimisation. In: 2003 Congress on evolutionary computation, CEC'03. Canberra, ACT, Australia
Olewuezi NP (2011) Note on the comparison of some outlier labeling techniques. J Math Stat 7(4):353–355
Olorunda O, Engelbrecht AP (2008) Measuring exploration/exploitation in particle swarm using swarm diversity. In: IEEE congress on evolutionary computationm. Hong Kong, pp 1128–1134
Opricovic S (1998) Multicriteria optimization of civil engineering systems. Fac Civ Eng Belgrade 2:5–21
Page EB (1963) Ordered hypotheses for multiple treatments: a significance test for linear ranks. J Am Stat Assoc 58:216–230

Pan QK, Gao L, Wang L, Liang J, Li XY (2019) Effective heuristics and metaheuristics to minimize total flowtime for the distributed permutation flowshop problem. Expert Syst Appl 124:309–324
Paul PV, Moganarangan N, Sampath KS, Raju R, Vengattaraman T, Dhavachelvan P (2015) Performance analyses over population seeding techniques of the permutation-coded genetic algorithm: an empirical study based on travelling salesman problems. Appl Soft Comput 32:383–402
Peitz S, Dellnitz M (2018) A survey of recent trends in multiobjective optimal control—surrogate models, feedback control and objective reduction. Math Comput Appl 23:30
Pierezan J, Maidl G, Yamao EM, Coelho LDS, Mariani VC (2019) Cultural coyote optimization algorithm applied to a heavy duty gas turbine operation. Energy Convers Manag 199:111932
Polap D, Kesik K, Woźniak M, Damaševičius R (2018) Parallel technique for metaheuristic algorithms using devoted local search and manipulating the solutions space. Appl Sci. 8(2):1–25
Price KV, Awad NH, Ali MZ, Suganthan PN (2018) The 100-digit challenge: problem definitions and evaluation criteria for the 100-digit shallenge special session and competition on single objective numerical optimization. Tech. rep. Nanyang Technological University, Singapore
Pyzdek T, Keller P (2010) The six sigma handbook, 3rd edn. Tata Mcgraw Hill, New York, pp 389–391
Rakesh K, Suganthan PN (2017) An ensemble of kernel ridge regression for multi-class classification. Procedia Comput Sci 108:375–383
Rardin RL, Uzsoy R (2018) Experimental evaluation of heuristic optimization algorithm: a tutorial. J Heuristics 7(3):261–304
Rashedi E, Nezamabadi-pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179:2232–2248
Ray SS, Bandyopadhyay S, Pal SK (2007) Genetic operators for combinatorial optimization in TSP and microarray gene ordering. J Appl Intell 26(3):183–195
Rey D, Neuhäuser M (2011) Wilcoxon-signed-rank test. In: Lovric M (ed) International encyclopedia of statistical science. Springer, Berlin, Heidelberg
Ribeiro CC, Rosseti I, Vallejos R (2009) On the use of run time distributions to evaluate and compare stochastic local search algorithms. In: SLS '09 Proceedings of the second international workshop on engineering stochastic local search algorithms. Designing, implementing and analyzing effective heuristics. Springer, Brussels, Belgium, pp 16–30
Riget J, Vesterstrøm JS (2002) A diversity-guided particle swarm optimizer—the arpso. EVALife Project Group, technical report 2002-02, Department of Computer Science, Aarhus Universitet
Riquelme N, Lücken CV, Barán B (2015) Performance metrics in multi-objective optimization. In: 2015 XLI Latin American computing conference
Rouky N, Abourraja MN, Boukachour J, Boudebous D, Alaoui AH, Khoukhi FE (2019) Simulation optimization based ant colony algorithm for the uncertain quay crane scheduling problem. Int J Ind Eng Comput 10:111–132
Roy B (1991) The outranking approach and the foundations of electre methods. Theor Decis 31:49–73. https://doi.org/10.1007/BF00134132
Saad AEH, Dong Z, Karimi M (2017) A comparative study on recently-introduced nature-based global optimization methods in complex mechanical system design. Algorithms 10(4):1–30
Saaty TL (2004) Decision making-The analytic hierarchy and network processes (AHP, ANP). J Syst Sci Syst Eng 13:1–35
Sadollah A, Sayyandi H, Yoo DG, Lee HM, Kim JH (2018) Mine blast harmony search: a new hybrid method for improving exploration and exploitaqtion capabilities. Appl Soft Comput 68:548–564
Salomon R (1998) Evolutionary algorithms and gradient search: similarities and differences. IEEE Trans Evol Comput 2(2):45–55
Santos G, Chagas JBC (2018) The thief orienteering problem: formulation and heuriatic approaches. In: IEEE congress on evolutionary computation (CEC). Rio de Jeneiro, pp 1–9
Santos T, Xavier S (2018) A convergence indiator for multi-objective optimisation algorithms. Tend Math Apli Comput 19(3):437–448
Santos LFOM, Iwayama RS, Cavalcanti LB, Turi LM, Morais FEdS, Mormilho G, Cunha CB (2019) A variable neighborhood search algorithm for the bin packing problem with compatible categories. Expert Syst Appl 124:209–225
Schott JR (1995) Fault tolerant design using single and multicretiria genetic algorithm optimization. Master's thesis, Massachusetts Institute of Technology
Schutze O, Esquivel X, Lara A, Coello CAC (2012) Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization. IEEE Trans Evol Comput 16(4):504–522
Senning JR (2015) Computing and estimating the rate of convergence. Dept. Math. Comput. Sci, Gordon Colledge, Wenham

Sergeyev YD, Kvasov DE, Mukhametzhanov MS (2017) Operational zones for comparing metaheuristic and deterministic one-dimensional global optimization algorithms. Math Comput Simul 141:96–109
Sergeyev YD, Kvasov DE, Mukhametzhanov MS (2018) On the efficiency of nature-inspired metaheuristics in expensive global optimization with limited budget. Sci Rep 8(453):1–9
Sheskin D (2006) Handbook of parametric and nonparametric statistical procedures. Chapman & Hall/CRC, Boca Raton
Shukor SA, Shaheed IM, Abdullah S (2018) Population initialisation methods for fuzzy job-shop scheduling problems: issues and future trends. Int J Adv Sci Eng Inf Tech 8(4–2):1820–1828
Silva VL, Wanner EF, Cerqueira SA, Takahashi RH (2007) A new performance metric for multiobjective optimization: the integrated sphere counting. In: Proceedings of the 2007 IEEE congress on evolutionary computation. IEEE Press, pp 3625–3630
Silva BCH, Fernandes IFC, Goldbarg MC, Goldbarg EFG (2020) Quota travelling salesman problem with passengers, incomplete ride and collection time optimization by ant-based algorithms. Comput Oper Res. https://doi.org/10.1016/j.cor.2020.104950
Skidmore GS (2006) Metaheuristics and combinatorial optimization problems. Master's thesis. Rochester Institute of Technology
Smith-Miles, Van Hemert J (2011) Discovering the suitability of optimisation algorithms by learning from evolved instances. Ann Math Artif Intell 61:87–104
Smith-Miles K, Van Hemert J, Lim XY (2010) Understanding TSP Difficulty by Learning from Evolved Instances. In: Blum C, Battiti R (eds) Learning and intelligent optimization. LION 2010. Lecture notes in computer science, vol 6073. Springer, Berlin¸Heidelberg
Snedecor GW, Cochran WG (1989) Statistical methods, 8th edn. Iowa State University Press, Iowa
Suganthan PN, Hansen N, Liang JJ, Deb K, Chen YP, Auger A, Tiwari S (2005) Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Technical report
Sun Y, Yen GG, Yi Z (2018) IGD indicator-based evolutionary algorithm for many-objective optimization problems. IEEE Trans Evol Comput 23(2):173–187
Suzuki J (1995) A Markov chain analysis on simple genetic algorithm. IEEE Trans Syst Man Cybern 25(4):655–659
Tang L, Li Z, Luo L, Liu B (2015) Multi-strategy adaptive particle swarm optimization for numerical optimization. Eng Appl Artif Intell 37:9–19
Terpstra TJ (1952) The asymptotic normality and consistency of Kendall's test against trend, when ties are present in one ranking. Indag Math 14:327–333
Trefethen N (2002) A hundred-dollar, hundred-digit challenge. SIAM News 35(1):65
Trojanowski K, Michalewicz Z (1999) Searching for optima in non-stationary environments, In: Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406), vol 3. pp 1843–1850
Tsai HK, Yang JM, Tsai YF, Kao CY (2003) Heterogeneous selection genetic algorithms for travelling salesman problems. Eng Optim 35(3):297–311
Tsai HK, Yang JM, Tsai YF, Kao CY (2004) An evolutionary algorithm for large travelling salesman problems. IEEE Trans Syst Man Cybern Part B Cybern 34(4):1718–1729
Tsai CW, Tseng SP, Chiang MC, Yang CS, Hong TP (2014) A high-performance genetic algorithms: using traveling salesman problem as a case. Sci World J 2014:1–14
Tseng LY, Yang SB (2001) A genetic approach to the automatic clustering problem. Pattern Recognit 34(2):415–424
Tukey JW (1977) Exploratory data analysis. Addison-Wesley, Philippines
Tušar T, Filipič B (2014) Visualizing exact and approximated 3d empirical attainment functions. Math Probl Eng 569346:1–18
Upton G, Cook I (2014) Dictionary of statistics. Oxford University Press, Oxford
Van Hemert J (2005) Property analysis of symmetric travelling salesman problem instances acquired through evolution. In: Raidl GR, Gottlieb J (eds) Evolutionary computation in combinatorial optimization. EvoCOP 2005. Lecture notes in computer science, vol 3448. Springer, Berlin, Heidelberg
Van Veldhuizen DA, Lamont GB (1999) Multiobjective evolutionary algorithm test suites. In Proceedings of the ACM symposium on applied computing. pp 351–357
Van Veldhuizen DA, Lamont GB (2000) On measuring multiobjective evolutionary algorithm performance. In: Proceedings of the IEEE congress on evolutionary computation. pp 204–211
Vargas A, Bogoya J (2018) A generalization of the averaged hausdorff distance. Comput Sist 22(2):331–345
Vaz AIF, Vicente LN (2009) PSwarm: a hybrid solver for linearly constrained global derivative-free optimization. Optim Methods Softw 24(4–5):669–685

Wagner M, Lindauer M, Misir M, Nallaperuma S, Hutter F (2018) A case study of algorithm selection for the travelling thief problem. J Heuristics 24(3):295–320
Wang WL, Li WK, Wang Z, Li L (2019) Opposition-based multi-objective whale optimization algorithm with global grid ranking. Neurocomputing 341:41–59
Wanner EF, Guimarães FG, Takahashi RH, Fleming PJ (2006) A quadratic approximation-based local search procedure for multiobjective genetic algorithms. In: Procedings of the IEEE congress on evolutionary computation. pp 938–945
Weck OLD (2004) Multiobjective optimization: history and promise. In: Proceedings of 3rd China–Japan–Korea joint symposium on optimization of structural and mechanical systems. Kanazawa, Japan
Whitley D, Rana S, Dzubera J, Mathias KE (1996) Evaluating evolutionary algorithms. Artif Intell 85:245–276
Woolway M, Majozi T (2019) On the application of a metaheuristic suite with parallel implementations for the scheduling of multipurpose batch plants. Comput Chem Eng 126:371–390
Wu J, Azarm S (2001) Metrics for quality assessment of a multiobjective optimization solution set, transactions of the ASME. J Mech Des 123:18–25
Wu W, Yagiura M, Ibaraki T (2018) Generalized assignment problem. In: Gonzalez TF (ed) Handbook of approximation algorithms and metaheuristics, methodologies and traditional applications, vol 1, 2nd edn. Chapman and Hall/CRC, New York
Wu Z, Jiang B, Karimi HR (2020) A logarithmic descent direction algorithm for the quadratic knapsack problem. Appl Math Comput 369:124854
Yaghini M, Momeni M, Sarmadi M (2011) DIMMA-implemented metaheuristics for finding shortest hamiltonian path between Iranian cities using sequential DOE approach for parameters tuning. Int J Appl Metaheuristic Comput 2(2):74–92
Yang XS (2010) Nature-inspired metaheuristic algorithms, 2nd edn. Luniver Press, Bristol
Yang XS (2011) Metaheuristics optimization: algorithm analysis and open problems. Lect Notes Comput Sci 6630:21–32
Yang S, Li C (2010) A Clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments. IEEE Trans Evol Comput 14(6):959–974
Yang S, Yao X (2013) Evolutionary computation for dynamic optimization problems, ch. 8. Springer, Berlin, pp 203–205
Yang XS, Deb S, Fong S (2011) Accelerated particle swarm optimization and support vector machine for business optimization and applications. In: Networked digital technologies (NDT2011), communications in computer and information science, vol 136. Springer, pp 53–66
Yen GG, He Z (2014) Performance metric ensemble for multiobjective evolutionary algorithms. IEEE Trans Evol Comput 18(1):131–144
Yu X, Zhang X (2014) Unit commitment using Lagrangian relaxation and particle swarm optimization. Electr Power Energy Syst 61:510–522
Yu HJ, Zhang LP, Chen DZ, Hu SX (2005) Adaptive particle swarm optimization algorithm based on feedback mechanism. J Zhejiang Univ (Eng Sci Ed) 39(9):1286–1291
Yu X, Lu Y, Yu X (2018) Evaluating multiobjective evolutionary algorithms using MCDM methods. Math Probl Eng 9751783:1–13
Yu G, Jin Y, Olhofer M (2019) Benchmark problems and performance indicators for search of knee points in multiobjective optimization. IEEE Trans Cybern 99:1–14
Zehmakan AN (2015) Bin packing problem: two approximation algorithms. arXiv:1508.01376. Retrieved from https://arxiv.org/pdf/1508.01376v1.pdf
Zhan B, Haslbeck MPL (2018) Verifying asymptotic time complexity of imperative programs in Isabelle. In: Galmiche D, Schulz S, Sebastiani R (eds) Automatic reasoning. IJCAR. Lecture notes in computer science, vol 10900. Springer, Cham
Zhang X, Hu H, Wang L, Sun Z, Zhang Y, Han K, Xu Y (2018) A novel bin design problem and high performance algorithm for e-commerce logistics system. arXiv:1812.02565v1. Retrieved from: https://arxiv.org/pdf/1812.02565v1.pdf
Zhang HG, Liang ZH, Liu HJ, Wang R, Liu YA (2020) Ensemble framework by using nature inspired algorithms for the early-stage forest fire rescue—a case study of dynamic optimization problems. Eng Appl Artif Intell 90:103517
Zheng K, Yang R, Xu H, Hu J (2017) A new distribution metric for comparing pareto optimal solutions. Struct Multidiscip Optim 55(1):53–62
Zhong Y, Wang L, Lin M, Zhang H (2019) Discrete pigeon-inspired optimization algorithm with metropolis acceptance criterion for large-scale travelling salesman problem. Swarm Evol Comput 48:134–144

Zhou A, Jin Y, Zhang Q, Sendhoff B, Tsang E (2006) Combining model-based and genetics-based offspring generation for multi-objective optimization using a convergence criterion. In: Evolutionary computation CEC 2006. IEEE congress on evolutionary computation. pp 892–899

Zhou AH, Zhu LP, Hu B, Deng S, Song Y, Qiu H, Pan S (2019) Traveling-salesman-problem algorithm based on simulated annealing and gene-expression programming. Information 10(7):1–15

Zitzler E (1999) Evolutionary algorithms for multiobjective optimization: methods and appliations. Doctor of technical sciences dissertation, Swiss Federal Institute of Technology Zurich

Zitzler E, Thiele L (1998) Multiobjective optimization using evolutionary algorithms—a comparative case study. In: Proceedings of the international conference on parallel problem solving from nature. Armsterdam, Netherlands, pp 292–301

Zitzler E, Thiele L (1999) Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Trans Evol Comput 3(4):257–271

Zitzler E, Deb K, Thiele L (2000) Comparison of multiobjective evolutionary algorithms: empirical results. Evol Comput 8(2):173–195

Zitzler E, Thiele L, Laumanns M, Foncesca CM, Foncesca VG (2003) Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans Evol Comput 7(2):117–132

Zitzler E, Brockhoff D, Thiele L (2007) The hypervolume indicator revisited: on the design of pareto-compliant indicators via weighted integration. In: Proceedings of the 4th international conference on evolutionary multi-criterion optimization (EMO'07). pp 862–876