Journal of Heuristics (2020) 26:851–883
https://doi.org/10.1007/s10732-020-09454-w

Sample size calculations for the experimental comparison of multiple algorithms on multiple problem instances

Felipe Campelo[1,2] · **Elizabeth F. Wanner**[1,3]

Received: 5 August 2019 / Revised: 13 May 2020 / Accepted: 23 July 2020 / Published online: 5 August 2020

Abstract

This work presents a statistically principled method for estimating the required number of instances in the experimental comparison of multiple algorithms on a given problem class of interest. This approach generalises earlier results by allowing researchers to design experiments based on the desired best, worst, mean or median-case statistical power to detect differences between algorithms larger than a certain threshold. Holm's step-down procedure is used to maintain the overall significance level controlled at desired levels, without resulting in overly conservative experiments. This paper also presents an approach for sampling each algorithm on each instance, based on optimal sample size ratios that minimise the total required number of runs subject to a desired accuracy in the estimation of paired differences. A case study investigating the effect of 21 variants of a custom-tailored Simulated Annealing for a class of scheduling problems is used to illustrate the application of the proposed methods for sample size calculations in the experimental comparison of algorithms.

Keywords Experimental comparison of algorithms · Statistical methods · Sample size estimation · Iterative sampling · Multiple hypotheses testing

F. Campelo worked under grants from Brazilian agencies FAPEMIG (APQ-01099-16) and CNPq (404988/2016-4). E. F. Wanner has been funded by The Leverhulme Trust through Research Fellowship RF-2018-527/9.

✉ Felipe Campelo
f.campelo@aston.ac.uk

Elizabeth F. Wanner
e.wanner@aston.ac.uk

1 College of Engineering and Physical Sciences, Aston University, Birmingham B4 7ET, UK

2 Department of Electrical Engineering, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil

3 Department of Computer Engineering, CEFET-MG, Belo Horizonte, Brazil

1 Introduction

Experimental comparison of algorithms has long been recognised as an essential aspect of research on meta-heuristics (Barr et al. 1995; Hooker 1996; McGeoch 1996; Coffin and Saltzman 2000). Despite well-earned criticisms of some disconnect between theory and experimentation (Hooker 1994; Chimani and Klein 2010) and a few methodological issues that still require adequate attention (Eiben and Jelasity 2002; Bartz-Beielstein 2015), experimental evaluation remains a central aspect and a major component of the development and understanding of meta-heuristics in general. Experimental comparison of algorithms involves the use of performance indicators, with the choice of indicator being closely related to the problem domain as well as to the question of interest being investigated by the experimenter. For optimisation algorithms, performance can be measured in terms, e.g., of final objective function value, *best-of-iteration* fitness, first hitting time, time-to-convergence, success rate and quality set measures of multi-objective problems, to name a few. Machine learning approaches to classification, which also rely heavily on experimental evaluation and comparison, commonly use accuracy, area under the ROC curve or F1-score to quantify different aspects of their performance.

This reliance on experimental assessment and comparison of algorithms is evidenced by the continuing effort of researchers in devising better experimental protocols for performance assessment and comparison of algorithms. While many of the most important points were presented as far back as the late 1990s (Barr et al. 1995; McGeoch 1996; Hooker 1996), research into adequate protocols and tools for comparing algorithms has continued in the past two decades, with several statistical approaches being proposed and employed for comparing the performance of algorithms (Coffin and Saltzman 2000; Johnson 2002; Yuan and Gallagher 2004; Demšar 2006; Yuan and Gallagher 2009; Birattari 2004; Birattari and Dorigo 2007; Bartz-Beielstein 2006; Bartz-Beielstein et al. 2010; García et al. 2008, 2010; Derrac et al. 2011; Carrano et al. 2011; Derrac et al. 2014; Benavoli et al. 2014; Krohling et al. 2015; Hansen et al. 2016; Campelo and Takahashi 2019; Calvo et al. 2019). This increased prevalence of more statistically sound experiments in the field of optimisation heuristics can be seen as part of the transition of the area into what has been called the *scientific* period of research on metaheuristics (Sörensen et al. 2018).

Despite of this area-wide effort, a few important aspects of experimental comparisons of algorithms remain largely unexplored in the literature. One of those topics is the question of how to adequately determine the relevant sample sizes for the comparisons of algorithms—number of problem instances to use, and number of runs to employ for each algorithm on each instance. The standard approach has been that of maximizing the number of instances, limited only by the computational budget available (Barr et al. 1995; García et al. 2008, 2009; Derrac et al. 2011; del Amo et al. 2012), and of running arbitrarily-set numbers of repeated runs, usually 30 or 50. While it is indeed true that the sensitivity of comparisons increases with the number of instances, this does not mean that a large sample size can substitute a well-designed experiment (Lenth 2001; Mathews 2010; Campelo and Takahashi 2019). Also, it is important to be aware that arbitrarily large sample sizes allow tests to detect even minuscule differences at arbitrarily strict significance levels, which may lead to the

wrongful interpretation that effects of no practical consequence are strongly significant (Mathews 2010; Bartz-Beielstein 2005) if certain methodological safeguards are not put into place when designing the experiment (Campelo and Takahashi 2019).

The dual issues of statistical power and sample size have been touched by a few authors in the past, albeit only superficially (Jain 1991; Coffin and Saltzman 2000; Czarn et al. 2004; Ridge 2007; Bartz-Beielstein 2005, 2006; Bartz-Beielstein et al. 2010). Until recently, the best works on the subject were those by Birattari (2004, 2009) and Chiarandini and Goegebeur (2010, Ch. 10). The former correctly advocated for a greater focus on the number of instances than on repetitions, showing that the optimal allocation of computational effort, in terms of accuracy in the estimation of mean performance indicator value for a given problem class, is to maximize the amount of test instances, running each algorithm a single time on each instance if needed. Birattari's approach, however, was focussed only on accuracy of parameter estimation for the mean performance indicator value across a problem class, yielding very little information on the specific behaviour of each algorithm on each instance, as well as not considering questions of desired statistical power or sample size calculations. The work by Chiarandini and Goegebeur (2010, Ch. 10) provided a good discussion on statistical power and sample size in the context of nested linear statistical models, as well as some guidelines on the choice of the number of instances and number of repeated runs based on the graphical analyses of power curves. It was, however, limited to nested models, and required manual inspection of power curves by the user, which may have precluded its broader adoption in the literature.

More recently, we have proposed a principled approach for calculating both the number of instances and number of repeated runs when comparing the performance of two algorithms for a problem class of interest (Campelo and Takahashi 2019). That approach calculates the number of instances based on the desired sensitivity to detect some *minimally relevant effect size* (MRES), i.e., the smallest difference between algorithms that is considered to have some degree of practical relevance. Alternatively, the number of instances can be fixed a priori (e.g., when using standard benchmark sets) and the sensitivity curves can be derived instead. As for the number of repeated runs per instance, the approach proposed in Campelo and Takahashi (2019) was based on minimising the number of algorithm runs required to achieved a the desired accuracy in the estimation of the paired differences in performance indicator values for each instance. While statistically sound, the fact that the results presented in that work applied only to comparisons of two algorithms limited their applicability to the general case of experimental comparisons of algorithms, which often aim to investigate the relative performances of multiple algorithms, or multiple variants of a given algorithmic framework.

In this paper we generalise the results from Campelo and Takahashi (2019) to enable the calculation of required sample sizes for the comparison of an arbitrary number of algorithms. More specifically, the main original contributions of this paper are as follows: (i) the development of formulas to calculate the number of required instances based on considerations regarding the test of multiple hypotheses, correcting the significance level of each test so as to maintain the familywise error rate controlled at a given desired level (Shaffer 1995); (ii) derivation of optimality-based ratios of sample sizes for sampling multiple algorithms on each individual instance, both for

the simple difference of means and for two types of percent differences. A simple sampling heuristic is also provided, which can easily generalise the sampling approach presented here for cases in which general performance statistics are of interest. An open source implementation of all methods presented in this paper is provided in the form of `R` package `CAISEr` (https://cran.r-project.org/package=CAISEr).

It is important to highlight at this point that we do not claim that the approach considered in this paper is the only way to test algorithms. Other methods of analysis, both analytic and graphical, can be useful for answering distinct questions related to the performance of algorithms on individual instances or problem classes. Bayesian approaches, in particular, have been gaining popularity for the comparison of machine learning and optimisation algorithms (Benavoli et al. 2017; Calvo et al. 2019). While the methods proposed here are placed within the framework of frequentist statistics, the general sampling approach can be easily adapted to Bayesian estimation and hypothesis testing, which also requires adequate sampling practices. In fact, the iterative heuristic derived in Sect. 3.2 of this manuscript has clear parallels with the Bayesian approach described by Kruschke (2010). Despite these possibilities, however, the current paper does not investigate Bayesian methods, but rather aims at providing researchers working within the classical hypothesis testing methods of the field (based on null hypotheses significance testing and confidence intervals) with tools for the application of these tests at the highest levels of methodological quality.

The remainder of this paper is organised as follows: Sect. 2 explicitly defines the algorithm comparison problem considered in this work, states the hypotheses to be tested, and provides the rationale for the choices that justify the proposed approach for sample size calculation. Section 3 describes the proposed approach for sampling an arbitrary number of algorithms on any given problem instance, and presents the derivation of optimal sample size ratios for three common cases. Section 4 describes the general concepts behind the estimation of the required number of instances for an experiment to achieve desired statistical properties. Section 5 presents an example of application of the proposed methodology for the evaluation of 21 algorithmic variants of a state-of-the-art, custom-tailored simulated annealing approach Santos et al. (2016) for a class of scheduling problems (Vallada and Ruiz 2011). Finally, Sect. 6 presents final considerations and conclusions.

2 Problem definition

Before proceeding to investigate the required sample sizes for the experimental comparison of algorithms, it is important to formally define the questions one is trying to answer when those comparisons are performed. While there are several different scientific questions that can be investigated by experimentation, arguably the most general case for experimental meta-heuristic comparisons is: *given a finite subset of problem instances and a finite number of runs that can be performed by each algorithm being tested, what can be said regarding the relative performance of those algorithms (according to a given set of performance indicators) for the problem class from which the test instances were drawn?* Notice that this question is broad enough to encompass the most common cases in the experimental comparison of algorithms, and even sev-

eral comparisons of scientific relevance that are not routinely performed (Eiben and Jelasity 2002).

Aiming to highlight the specific aspects that are considered in the present work, we refine this problem definition as follows: Let $\Gamma_S = \{\gamma_\ell : \ell \in \{1, \ldots, N\}\} \subset \Gamma$ represent a finite sample of problem instances drawn from a given problem class of interest, Γ; and let $\mathcal{A} = \{a_1, a_2, \ldots, a_A\}$ denote a set of algorithms[1] that we want to compare. We define the algorithm comparison problem in the context of this work as that of obtaining a (partial) ordering of the algorithms according to their mean performance indicator value on the problem class of interest, based on the observed performance indicator values across the set of test instances used. The algorithm performance is measured according to some of indicator choice. We assume here that (i) all algorithms can be run on the same sample of instances; (ii) any run of an algorithm returns some tentative solution, which can be used to estimate the performance of that algorithm for the problem instance; and (iii) the researcher is interested primarily in comparing the performance of the algorithms for the *problem class* Γ, instead of for individual instances. Each of these three experimental assumptions can be associated with a specific aspect when comparing algorithms:

- Assumption (i) indicates that the variability due to instance effects can be modelled out of our analysis by *pairing* or *blocking* (Campelo and Takahashi 2019; Montgomery and Runger 2013), which is the underlying approach of methods such as ANOVA with blocking, or Friedman's test (Demšar 2006).
- Assumption (ii) essentially prevents us from having to deal with the problem of missing values in our analysis, since every run will return some valid performance indicator observation. The question on how to deal with missing values in the comparative analysis of algorithms—e.g., if a given algorithm fails to converge in a study where performance is measured by time-until-convergence—is a very relevant one, but it falls outside the scope of this particular work.
- Assumption (iii) is associated with the fact that comparative testing of algorithms should, in the vast majority of cases, be focused on generalising the performance observed on limited test set Γ_S to the wider class of problems Γ. This contrasts with a somewhat common practice in the heuristics literature of performing individual Rank Sum tests on each instance and tallying up the wins/losses/ties, without any further inference being performed on these summary quantities. The problem with this approach for the comparison of algorithms on multiple problem instances has been recognised at least since 2006 (Demšar 2006), but despite of this the practice still persists in the literature.

Focusing on testing hypotheses regarding the (relative) performance of algorithms for a problem class of interest also determines two other aspects of the design and analysis of comparative experiments. The first is the issue of what is considered the *effective* sample size for these experiments. As argued in earlier works (Campelo and Takahashi 2019; Birattari 2004; Bartz-Beielstein et al. 2010), the effective sample size to be considered when testing the hypotheses of interest is the number of *instances*, rather than the amount of repeated runs (or even worse, the product of these two

[1] Each representing a complete instantiation of a given algorithmic framework, i.e., with both structure and parameter values fully specified.

quantities). Failure to account for this when performing inference results in pseudoreplication (Hurlbert 1984; Millar and Anderson 2004; Lazic 2010), that is, the (often implicit) use of artificially inflated degrees-of-freedom in statistical tests. This results in violation of the independence assumption underlying these tests, and results in actual confidence levels that can be arbitrarily inferior to the nominal ones.

The second aspect, which is a consequence of the first, is that the analysis of data generated by the experiments can be done using either a hierarchical model (Gelman and Hill 2006) or, alternatively, a block design applied on summarised observations, in which the performance of each algorithm on each instance is summarised as a single value, often the mean or median of multiple runs. This second approach is equivalent to the first under the assumption (iii) stated above, and results in the application of the well-known and widely-used methods for comparisons of the average performance indicator values of multiple algorithms on multiple problem instances: omnibus tests such as Complete Block Design (CBD) ANOVA or Friedman's test (Demšar 2006). Those tests are used to investigate whether at least one algorithm has an average performance indicator value different from at least one other. If that is the case, then multiple comparison procedures are employed to answer the more specific statistical questions of which algorithms differ from which in terms of their average performance (Sheskin 2011; Montgomery and Runger 2013).

In the following, we detail the hypotheses of interest that are tested by these statistical procedures when performing comparative experiments with algorithms, and suggest that a greater focus on the multiple comparison procedures can provide experimenters with better tools for designing and analysing their experiments.

2.1 Test hypotheses

Let $Y_{k|\ell}$ denote the performance indicator value of algorithm a_k on instance γ_ℓ. Based on the definitions from the preceding section, we can decompose the performance indicator value according to the usual model of the complete block design (Montgomery 2013):

$$Y_{k|\ell} = \mu_k + \theta_\ell + \epsilon_{k\ell} = \mu + \tau_k + \theta_\ell + \epsilon_{k\ell}, \tag{1}$$

in which μ_k denotes the mean performance indicator value of algorithm a_k for the problem class Γ after the (additive) effect of each instance, θ_ℓ, is blocked out; and $\epsilon_{k\ell}$ is the residual relative to that particular observation, i.e., the term that accounts for all other unmodelled effects that may be present (e.g., uncertainties in the estimation of $Y_{k|\ell}$). Notice that the algorithm mean μ_k can be further decomposed into a grand mean of all algorithms for the problem class of interest, μ, and the effect of algorithm a_k on this grand mean, represented by τ_k. By construction, $\sum_{k=1}^{A} \tau_k = 0$.

As mentioned earlier, an usual approach for comparing multiple optimisation algorithms on multiple problem instances [which is an analogous problem to that of testing multiple classifiers using multiple data sets (Demšar 2006)] employs the usual 2-step approach provided in most introductory texts in statistics. It starts by using omnibus tests, such as CBD ANOVA or Friedman's test, to investigate the existence of some

effect by testing hypotheses of the form:

$$\begin{aligned} H_0 &: \ \tau_k = 0, \quad \forall k \in \{1, \ldots, A\}; \\ H_1 &: \ \exists \tau_k \neq 0, \end{aligned} \tag{2}$$

which is equivalent to testing whether all algorithms have the same mean value *versus* the existence of at least one algorithm with a different mean value for the problem class of interest. If the omnibus test returns a statistically significant result at some particular confidence level, pairwise comparisons are then performed as a second step of inference, to pinpoint the significant difference(s). These are usually performed using common tests for the difference of means of two matched populations, such as variations of the paired t-test for post-ANOVA analysis, or Wilcoxon's Signed Ranks test (or the Binomial Sign Test) for post-Friedman analysis.

There are several different sets of pairwise comparisons that can be executed in this step, but the two most commonly used ones are *all versus all* and the *all versus one*. In the former, $A(A-1)/2$ tests are performed on the paired difference of means (again, with *instance* as the pairing factor) for all pairs of algorithms $a_i \neq a_j$:

$$\begin{aligned} H_0^{(ij)} &: \ \mu_{(ij)} = 0; \\ H_1^{(ij)} &: \ \mu_{(ij)} \neq 0, \end{aligned} \tag{3}$$

in which $\mu_{(ij)} = \tau_i - \tau_j$ represents the mean of the paired differences in performance indicator values between a_i and a_j. *All versus all* pairwise comparisons are the default approach in the meta-heuristics literature, but in practice are only required when one is really interested in obtaining a complete "ordering" of all algorithms in a given experiment.

The second type of pairwise comparisons is the *all versus one*, in which there is a reference algorithm (e.g. a new proposal one is interested in evaluating). Assuming that the first algorithm, a_1, is always set as the reference one, *all versus one* pairwise comparisons are defined as:

$$\begin{aligned} H_0^{(j)} &: \ \mu_{(1j)} = 0; \\ H_1^{(j)} &: \ \mu_{(1j)} \neq 0, \end{aligned} \tag{4}$$

for all $j \neq 1$. This approach results in only $(A-1)$ hypotheses to be tested and, as we will discuss later, results in inferential procedures with higher statistical power or, if sample sizes are being estimated, in experiments requiring fewer instances to achieve a given desired power (Mathews 2010).[2]

[2] Statistical power (or sample sizes) can be further improved if the alternative hypotheses in (4) are directional,

$$\begin{aligned} H_0^{(j)} &: \ \mu_{(1j)} = 0; \\ H_1^{(j)} &: \ \mu_{(1j)} > 0, \end{aligned}$$

It is important to highlight at this point that the ANOVA or Friedman tests are not strictly required when performing the comparison of multiple algorithms. From an inferential perspective, it is equally valid to perform only the pairwise comparisons, as long as the rejection threshold of each hypothesis tested is adequately adjusted to prevent the inflation of Type-I error rates resulting from multiple hypothesis testing (Shaffer 1995). While it is true that post-ANOVA pairwise tests can usually benefit from increased sensitivity (by using the more accurate estimate of the common residual variance that results from the ANOVA model), this improvement of statistical power is only achieved under the assumption of equality of variances, which is rarely true in algorithm comparisons. We argue that this potential (albeit occasional) disadvantage is more than compensated by a range of possible benefits that emerge when we focus on the pairwise comparisons:

- Tests for the mean of paired differences are generally much simpler than their corresponding omnibus counterparts, and generally require fewer assumptions. For instance, unlike the 'CBD ANOVA + post-hoc testing' approach, the paired t test does not require equality of variances between algorithms (or between vectors of paired differences); nor does it assume that the data from each algorithm is normally distributed—only the paired differences need be approximately normal, and not even that if the sample size is large enough. These tests also yield much more easily interpretable results, and are considerably simpler to design and analyse.
- In the context of experimental comparison of algorithms one is commonly interested in relatively few tests: the number of algorithms compared is usually small, and in several cases tests can be performed using the *all versus one* approach (e.g., when assessing the performance of a proposed method in comparison to several existing ones). This alleviates the main issue with testing multiple hypotheses, namely the loss of statistical power (or, alternatively, the increase in the sample size required to achieve a certain sensitivity) that results from adjusting the significance levels to control the Type-I error rate (Shaffer 1995).
- From the perspective of estimating the number of required instances for algorithm comparisons, one of the most challenging quantities that need to be defined in the design stage of these experiments is a *minimally relevant effect size* (MRES) (Campelo and Takahashi 2019). The definition of MRES in the context of omnibus tests is very counter-intuitive, since it must be defined in terms, e.g., of the ratio of between-groups to within-groups variances. While still challenging, the definition and interpretation of MRES in the context of comparing a pair of algorithms is incomparably simpler: it essentially involves defining the smallest difference in mean performance indicator values between two algorithms (either in terms of raw difference or normalised by the standard deviation) that would have some practical consequence in the context of the research being performed (Campelo and Takahashi 2019).

Footnote 2 continued
which can be employed if the researcher is specifically interested in knowing whether or not the reference algorithm is superior (in terms of mean performance value) to the others (i.e., if equality or inferiority are considered equally undesirable outcomes). Of course it makes no sense trying to use directional alternative hypotheses in the *all versus all* case, since in that case there is no reference algorithm common to all tests.

Given the considerations above we argue that, while it is common practice to perform the experimental analysis as the 2-step procedure outlined earlier, it is often more practical to focus on the pairwise tests, particularly from the perspective of sample size calculations. In the following sections we detail how the two sample sizes involved in the experimental comparison of algorithms—number of instances, number of runs of each algorithm on each instance—can be estimated by focusing on the pairwise comparisons. This of course does not preclude the use of omnibus tests as part of the analysis, or of any other type of analytical tool for investigating and characterising the behaviour and performance of the algorithms being tested. In fact, sample sizes estimated by focusing on the pairwise comparisons result in omnibus tests with at least the same statistical power for which the pairwise tests were planned, assuming that their assumptions, which are often more restrictive, are satisfied. However, the use of omnibus tests becomes optional, and the analyst can choose to proceed directly with the pairwise tests knowing that the statistical properties of their experiment are guaranteed.

Finally, based on the statistical model from (1) we can define the two kinds of paired differences in performance, which we will examine in this work. The first one is the simple paired difference in performance indicator values, $D_{(ij)|\ell} = Y_{i|\ell} - Y_{j|\ell}$, which can be estimated from the data as

$$\widehat{D}_{(ij)|\ell} = \bar{X}_{i|\ell} - \bar{X}_{j|\ell}, \tag{5}$$

in which $\bar{X}_{i|\ell}$ and $\bar{X}_{j|\ell}$ are the mean (or median) observed performance indicator values of algorithms a_i and a_j on instance γ_ℓ, computed from $n_{i\ell}$ and $n_{j\ell}$ runs, respectively. An alternative approach is to use percent differences, which can be estimated for the *all versus one* case as

$$\widehat{D}_{(1j)|\ell} = \frac{\bar{X}_{1|\ell} - \bar{X}_{j|\ell}}{\bar{X}_{1|\ell}} = 1 - \frac{\bar{X}_{j|\ell}}{\bar{X}_{1|\ell}}, \tag{6}$$

and for the *all versus all* case as

$$\widehat{D}_{(ij)|\ell} = \frac{\bar{X}_{i|\ell} - \bar{X}_{j|\ell}}{\bar{X}_{\bullet|\ell}}, \tag{7}$$

in which $\bar{X}_{\bullet|\ell}$ is the estimate of the grand mean of all algorithms on instance ℓ,

$$\bar{X}_{\bullet|\ell} = \frac{1}{A} \sum_{k=1}^{A} \bar{X}_{i|\ell}. \tag{8}$$

3 Calculating the number of repetitions for comparisons of multiple algorithms

3.1 Preliminary statistical concepts

Before we present the derivation of sample size formulas, there are a few statistical concepts that need to be clarified. More information and detailed discussions of these concepts are available in our previous work (Campelo and Takahashi 2019), and here we will only present a very brief review of these definitions.

The *minimally relevant effect size* (MRES), which is an essential concept for determining the smallest sample size in the comparison of algorithm, is defined in the context of this work as *the smallest difference between two algorithms that the researcher considers as having some practical effect*. There exists several different effect size estimators that are relevant for a variety of experimental questions (Ellis 2010), but in the context of the hypotheses considered in this work we focus on Cohen's d coefficient,

$$d = \frac{\delta}{\sigma} = \frac{|\mu_{ij}|}{\sigma}, \tag{9}$$

which is the (real) value of the mean of the paired differences[3] normalised by σ, the (real) standard deviation of the paired differences. In the context of this work, the value in the numerator of (9) is estimated as the mean of the $|\widehat{D}|$ values calculated using (5) (for the simple paired difference) or (6)–(7) (for the percent paired differences).

Based on this definition, the MRES is defined as

$$d^* = \frac{|\delta^*|}{\sigma} = \frac{|\mu^*_{ij}|}{\sigma} \tag{10}$$

i.e., as the smallest magnitude of the standardised mean of paired differences between two algorithms that would actually result in some effect of practical relevance.[4]

Let $X^t_{k|\ell}$ denote the observed performance indicator value of algorithm a_k on the tth run on a given instance γ_ℓ, and

$$\bar{X}_{k|\ell} = \frac{1}{n_{k|\ell}} \sum_{t=1}^{n_{k|\ell}} X^t_{k|\ell} \tag{11}$$

denote the sample estimator of the mean performance indicator value of a_k on that instance, based on $n_{k|\ell}$ independent runs. Under relatively mild assumptions, we know

[3] More generally, the denominator of Cohen's d is the magnitude of the deviation between the actual value of a parameter and the value suggested under the null hypothesis. However, in this work we always assume the null hypothesis to suggest a difference of zero, hence the simpler definition.

[4] Different areas have distinct standards of what constitutes small or large effect sizes in terms of d (Ellis 2010), and domain-specific knowledge needs to be considered when determining d^*. We provide some broad guidelines in our previous work (Campelo and Takahashi 2019), but a broader investigation of generally accepted MRES values in the experimental research on meta-heuristics is yet to be conducted.

that the sampling distribution of means converges to a Normal distribution even for reasonably small sample sizes, with

$$\bar{X}_{k|\ell} \sim \mathcal{N}\left(\mu_{k|\ell}, \frac{\sigma^2_{k|\ell}}{n_{k|\ell}}\right). \tag{12}$$

where the $\sigma^2_{k|\ell} \,/\, n_{k|\ell}$ term is the squared standard error of the sample mean.

Given a single problem instance γ_ℓ and the A algorithms one wishes to compare, the approach for calculating the number of repetitions is similar to the one outlined in Campelo and Takahashi (2019), namely finding the smallest total number of runs such that the standard errors of estimation—which can be interpreted as the *measurement errors* of the pairwise differences in performance—can be controlled at a predefined level. This can be expressed as an optimisation problem,

$$\begin{aligned} &\text{Find } \mathbf{n}^\star_\ell = \arg\min \sum_{k=1}^{A} n_{k|\ell} \\ &\text{Subject to: } se^2_{(ij)|\ell} \leq \left(se^*\right)^2, \ \forall \left(a_i, a_j\right) \text{ pairs of interest} \end{aligned} \tag{13}$$

in which $n_{k|\ell}$ is the number of runs of algorithm a_k on instance γ_ℓ, $se_{(ij)|\ell}$ is the standard error of estimation of the relevant statistic (e.g., simple or percent difference of means) for a pair of algorithms a_i, a_j. The $\left(a_i, a_j\right)$ pairs of interest depend on the types of comparisons planned. While there can be a multitude of comparison types that can be performed, as said before, two comparisons are of particular interest in experimental algorithm: *all versus all* and *one versus all*. All pairs $\left(a_i, a_j\right) : i \neq j$ are of interest in the *all versus all* case while, In the *all versus one* case, if a_1 is defined as the reference algorithm, then the pairs of interest are $\left(a_1, a_j\right) : j \in \{2, \ldots, A\}$.

It is worth noting that the most adequate formulation for this problem is an integer one, but in this work we employ a relaxed version of the formulation that accepts continuous values. Since this formulation is used only to derive reference values for the optimal *ratio* of sample sizes (which can be continuous even under the discrete sample size constraint) this relaxation does not result in any inconsistencies. Also notice that there is a second set of constraints that is omitted from formulation (13), namely $n_{k|\ell} \geq 2, \ \forall k \in \{1, \ldots, A\}$, which are needed to guarantee that standard deviations can be estimated for each algorithm. We did not explicitly include these constraints since the proposed method assumes that all algorithms will be initially run a number $n_0 > 2$ times before iterative sample size estimations start, so these constraints will always be satisfied.

The specific calculation of the standard errors se_{ij} depends on the type of differences being considered: simple differences or percent differences and, in the latter case, whether the planned comparisons are an *all versus one* or *all versus all* case. The results for the comparison of two algorithms were presented in our previous work (Campelo and Takahashi 2019), and in this section we generalise those results to the case of comparisons of multiple algorithms. We also introduce a heuristic that can be

used to estimate the numbers of repetitions in the case of more general statistics that may be of interest in the context of experimental comparison of algorithms.

3.2 Derivation of optimal ratios $n_{i|\ell}/n_{j|\ell}$

While it was possible to derive an analytic solution to the problem of determining the optimal ratio of sample sizes in the two algorithm case (Campelo and Takahashi 2019), a similar closed solution does not seem to be possible for the general problem (13) with $A \geq 3$. It is, however, possible to derive optimality-based ratios of sample sizes for any pair of algorithms, which allows us to propose a principled heuristic to generate adequate samples for a set of A algorithms on any individual instance. The proposed heuristic iteratively generates observations of performance indicator values for each of the algorithms considered, so that the constraints $se^2_{(ij)|\ell} \leq (se^*)^2$ can be satisfied with as few total algorithm runs as possible. This heuristic is described in Algorithm 1.

Algorithm 1 Sample algorithms on a single instance.

Require: Instance γ_ℓ; Algorithms $a_1, \ldots, a_A$; accuracy threshold se^*; initial sample size n_0; maximum sample size n_{max}.
1: Generate n_0 initial samples of each algorithm on instance γ_ℓ.
2: Estimate all $\widehat{se}_{ij}$ of interest
3: **while** $\left(\exists\, \widehat{se}_{(ij)|\ell} > se^*\right)$ and $\left(\sum_{k=1}^{A} n_{k|\ell} < n_{max}\right)$ **do**
4: $\quad \widehat{se}_{max} \leftarrow \max_{i,j} \widehat{se}_{(ij)|\ell}$
5: $\quad$ Perform **one** additional run of the algorithm that is contributing the most to $\widehat{se}_{max}$ on instance γ_ℓ
6: $\quad$ Estimate all $\widehat{se}_{(ij)|\ell}$ of interest.
7: **end while**
8: **return** $\mathbf{x}_{k|\ell},\ k = 1, \ldots, A$ ▷ Vectors of observations generated for all algorithms.

The procedure detailed in Algorithm 1 is general enough to accommodate any statistic one may wish to use for quantifying the pairwise differences of performance indicator values between algorithms, as long as the standard errors and their sensitivities to changes in $n_{k|\ell}$ can be estimated (which can be done using bootstrap, if analytical expressions are not available). In the case where analytical expressions for the optimal ratio of sample sizes between two algorithms are available (see below), the determination of the algorithm that is contributing the most to $\widehat{se}_{max}$ on a given instance γ_ℓ (line 5 of the algorithm) can be done based on these *optimal ratios*: if the observed ratio is smaller than the optimal one, run the algorithm in the numerator; otherwise, run the one in the denominator.

Another point worth mentioning is that the procedure outlined in Algorithm 1 is essentially a greedy approach to reduce the worst-case standard error, which means that even if it is interrupted by the computational budget constraint the resulting standard errors of estimation will be the smallest ones achievable. This, in turn, means that the residual variance due to these estimation uncertainties will be as small as possible, which in turn increases the sensitivity of the experiment to smaller differences in

performance indicator values by reducing, even if slightly, the denominator term in (9).

The derivation of the standard errors and optimal ratios of sample sizes, for the simple and percent differences of means, both for the *all versus one* and *all versus all* cases, are presented below. Notice that while we can derive optimal sample size ratios for all paired differences of interest, Algorithm 1 will result in ratios that can deviate (sometimes substantially so) from the optimal reference values. This is a consequence of the fact that ratios of sample sizes are not independent quantities when multiple algorithms are compared. In practice, pairs that present the worst-case standard error ($\widehat{se}_{max}$) more often will tend to have ratios of sample sizes closer to the optimal values derived below, with the other pairs deviating from optimality due to the algorithm focusing the sampling on the pairs that are associated with $\widehat{se}_{max}$. This, however, does not reduce the validity and usefulness of the optimality derivations (as they are needed to decide which algorithm from the selected pair should receive a new observation), nor does it affect the quality of the results provided by Algorithm 1.

Finally, for the sake of clarity we will omit the instance index ℓ from all derivations in the next section, but the reader is advised to keep in mind that the calculations in the remained of this section are related to a single given instance.

3.2.1 Simple differences of means

In the case of the simple differences of means, the statistic of interest for any pair of algorithms a_i, a_j is

$$\phi_{ij} = \mu_i - \mu_j = \tau_i - \tau_j, \tag{14}$$

which is estimated using the sample quantities

$$\widehat{\phi}_{ij} = \bar{X}_i - \bar{X}_j. \tag{15}$$

Given (12), the sampling distribution of $\widehat{\phi}_{ij}$ will be

$$\widehat{\phi}_{ij} \sim \mathcal{N}\left(\mu_i - \mu_j, \sigma_i^2 n_i^{-1} + \sigma_j^2 n_j^{-1}\right) \tag{16}$$

with squared standard error

$$se_{ij}^2 = \sigma_i^2 n_i^{-1} + \sigma_j^2 n_j^{-1}. \tag{17}$$

For any pair of algorithms a_i, a_j, the optimal sample size ratio n_i/n_j for a given instance γ can be found using the formulation in (13):

$$\begin{aligned} &\text{Find } \left[n_i, n_j\right]^\star = \arg\min f_{ij}\left(n_i, n_j\right) = n_i + n_j; \\ &\text{Subject to: } g_{ij}\left(n_i, n_j\right) = se_{ij}^2 - \left(se^*\right)^2 \leq 0. \end{aligned} \tag{18}$$

This formulation has a known analytical solution (Campelo and Takahashi 2019) in terms of the ratio of sample sizes:

$$r_{opt} = \frac{n_i^*}{n_j^*} = \frac{\sigma_i}{\sigma_j}, \tag{19}$$

which can be rewritten in terms of sample estimators as

$$\hat{r}_{opt} = \frac{S_i}{S_j}, \tag{20}$$

with S_i, S_j being the sample standard deviations of the observations of algorithms a_i, a_j, respectively. In this case the decision rule for which algorithm should receive a new observation (line 5 of Algorithm 1) can be expressed as: let a_i, a_j be the two algorithms that define $\widehat{se}_{max}$. Then:

$$\begin{cases} \text{If } n_i/n_j < \hat{r}_{opt} & \rightarrow \text{sample } a_i \\ \text{Else} & \rightarrow \text{sample } a_j. \end{cases} \tag{21}$$

It is worthwhile to notice that this result is independent on whether one is interested in *all versus all* or *all versus one* comparisons, the only difference being which pairs a_i, a_j generate se_{ij} constraints in (13).

3.2.2 Percent differences of means

For the percent differences of means[5] we need to distinguish between the two types of comparisons discussed in this work.

All versus one For *all versus one* comparisons, assuming a_1 is the reference algorithm, the statistic of interest is given as:

$$\phi_{1j} = \frac{\mu_1 - \mu_j}{\mu_1} = 1 - \frac{\mu_j}{\mu_1}, \tag{22}$$

which can be interpreted as the *percent mean loss in performance indicator value of* a_j *when compared to* a_1. This is estimated using

$$\widehat{\phi}_{1j} = 1 - \frac{\bar{X}_j}{\bar{X}_1}, \tag{23}$$

[5] Strictly speaking the difference discussed here is the proportional (per unit) difference, which must be multiplied by a factor of 100 to be actually expressed as *percent*. This, however, has no effect in the derivations that follow, and is merely a matter of linear scaling.

which is distributed according to one plus the ratio of two normal variables R/S, with

$$R \sim \mathcal{N}\left(-\mu_j, \sigma_j^2 n_j^{-1}\right);$$
$$S \sim \mathcal{N}\left(\mu_1, \sigma_1^2 n_1^{-1}\right).$$

If we can assume that all algorithms involved return observations that are strictly positive (which is quite common, e.g., in several families of problems in operations research as well as for several common performance indicators in multiobjective optimisation), the squared standard error of (23) can be calculated as (Campelo and Takahashi 2019; Fieller 1954; Franz 2007):

$$\begin{aligned} se_{1j}^2 &= \left(\frac{\mu_j}{\mu_1}\right)^2 \left[\frac{\sigma_1^2 n_1^{-1}}{\mu_1^2} + \frac{\sigma_j^2 n_j^{-1}}{\mu_j^2}\right] \\ &= \frac{\sigma_1^2 \mu_j^2}{\mu_1^4} n_1^{-1} + \frac{\sigma_j^2}{\mu_1^2} n_j^{-1} \\ &= C_1^{1j} n_1^{-1} + C_2^{1j} n_j^{-1}, \end{aligned} \tag{24}$$

with

$$C_1^{1j} = \frac{\sigma_1^2 \mu_j^2}{\mu_1^4} \quad \text{and} \quad C_2^{1j} = \frac{\sigma_j^2}{\mu_1^2}.$$

The KKT conditions for the problem formulation in (18) state that, at the optimal point $\left[n_1^*, n_j^*\right]$ the following should be true:

$$\begin{aligned} \nabla f_{1j} + \beta_{1j} \nabla g_{1j} &= \mathbf{0} \\ \beta_{1j} g_{1j} &= 0 \\ \beta_{1j} &\geq 0 \end{aligned} \tag{25}$$

The gradient of the objective function is trivially derived as $\nabla f_{1j} = [1, 1]$, and the partial derivatives of g_{1j} are:

$$\frac{\partial g_{1j}}{\partial n_1} = -C_1^{1j} n_i^{-2} \quad \text{and} \quad \frac{\partial g_{1j}}{\partial n_j} = -C_2^{1j} n_j^{-2}, \tag{26}$$

The first row of (25) can then be expanded as:

$$\begin{aligned} \beta_{1j} C_1^{1j} n_1^{-2} &= 1 \\ \beta_{1j} C_2^{1j} n_j^{-2} &= 1 \end{aligned} \tag{27}$$

which means that $C_1^{1j} n_1^{-2} = C_2^{1j} n_j^{-2}$ at the optimal point.[6] Isolating the ratio of sample sizes and simplifying finally yields:

$$r_{opt} = \frac{n_1^*}{n_j^*} = \frac{\sigma_1/\mu_1}{\sigma_j/\mu_j}, \tag{28}$$

which can be estimated from the sample using:

$$\hat{r}_{opt} = \frac{S_1}{S_j} \frac{\bar{X}_j}{\bar{X}_1}. \tag{29}$$

Given this expression for $\hat{r}_{opt}$, the same rule expressed in (21) can be used in Line 5 of Algorithm 1, simply replacing n_i by n_1.

All versus all For *all versus all* comparison using percent differences the question of which mean to use as the normalising term in the denominator must be decided. Assuming that no specific preference is given to any of the algorithms in this kind of comparison, we suggest using the grand mean μ instead of any specific mean μ_k, i.e.:

$$\phi_{ij} = \frac{\mu_i - \mu_j}{\mu} \tag{30}$$

which can now be interpreted as the *difference between the means of a_i and a_j, in proportion to the grand mean*. This is estimated using

$$\widehat{\phi}_{ij} = \frac{\bar{X}_i - \bar{X}_j}{\bar{X}_\bullet}, \tag{31}$$

where

$$\bar{X}_\bullet = \frac{1}{A} \sum_{k=1}^{A} \bar{X}_k$$

is an estimator of the grand mean μ. This estimator is unbiased regardless of unequal sample sizes,

$$E\left[\bar{X}_\bullet\right] = \frac{1}{A} \sum_{k=1}^{A} E\left[\bar{X}_k\right] = \frac{1}{A} \sum_{k=1}^{A} (\mu + \tau_k) = \mu + \frac{1}{A} \sum_{k=1}^{A} \tau_k = \mu; \tag{32}$$

and has a squared standard error of:

$$V\left[\bar{X}_\bullet\right] = \frac{1}{A^2} \sum_{k=1}^{A} V\left[\bar{X}_k\right] = \frac{1}{A^2} \sum_{k=1}^{A} \sigma_k^2 n_k^{-1}, \tag{33}$$

[6] Eqnarray (27) also mean that the Lagrange multiplier must be strictly greater than zero, which in turn means that g_{1j} is active at the optimal point, as expected from a sample size minimisation perspective.

Under the same mild assumptions under which $\bar{X}_k \sim \mathcal{N}\left(\mu_k, \sigma_k^2 n_k^{-1}\right)$, we have that the sampling distribution of this estimator is progressively closer to a normal distribution as the sample sizes are increased. These considerations allow us to calculate the standard error of (31) in the same manner as (24), using the simplified formulation of Fieller's standard error (Fieller 1954; Franz 2007), i.e.:

$$\begin{aligned} se_{ij}^2 &= \left(\frac{\mu_i - \mu_j}{\mu}\right)^2 \left[\frac{\sigma_i^2 n_i^{-1} + \sigma_j^2 n_j^{-1}}{\left(\mu_i - \mu_j\right)^2} + \frac{\sum_{k=1}^{A} \sigma_k^2 n_k^{-1}}{A^2 \mu^2}\right] \\ &= C_1^{ij}\left(\sigma_i^2 n_i^{-1} + \sigma_j^2 n_j^{-1}\right) + C_2^{ij}, \end{aligned} \tag{34}$$

with

$$\begin{aligned} C_1^{ij} &= \frac{1}{\mu^2} + \left(\frac{\mu_i - \mu_j}{A\mu^2}\right)^2 = \frac{1}{\mu^2}\left(1 + \frac{\phi_{ij}^2}{A^2}\right); \\ C_2^{ij} &= \left(\frac{\mu_i - \mu_j}{A\mu^2}\right)^2 \sum_{\substack{k=1 \\ k \neq i,j}}^{A} \sigma_k^2 n_k^{-1} = \frac{1}{\mu^2}\frac{\phi_{ij}^2}{A^2} \sum_{\substack{k=1 \\ k \neq i,j}}^{A} \sigma_k^2 n_k^{-1}, \end{aligned}$$

with ϕ_{ij} defined as in (30). This new expression for the standard error leads to (26) being replaced by:

$$\frac{\partial g_{ij}}{\partial n_1} = -C_1^{ij} {\sigma_i}^2 n_i^{-2}, \quad \frac{\partial g_{ij}}{\partial n_j} = -C_1^{ij} {\sigma_j}^2 n_j^{-2}, \tag{35}$$

which, when inserted into the KKT conditions (25) yields:

$$\sigma_i^2 n_1^{-2} = \sigma_j^2 n_j^{-2} \tag{36}$$

and, consequently,

$$r_{opt} = \frac{n_i^*}{n_j^*} = \frac{\sigma_i}{\sigma_j}, \tag{37}$$

which is identical to the optimal ratio for the simple differences of means, and can be similarly estimated using the sample standard deviations. Once again, the same decision rule from (21) can be used.

4 Calculating the number of required instances for comparisons of multiple algorithms

In this section we deal with the calculation of the number of instances, N, required to obtain an experiment with predefined statistical properties. In a previous work

(Campelo and Takahashi 2019) we developed sample size formulas for the comparison of two algorithms on a problem class of interest. Since we have reduced the multiple algorithm comparison problem to a series of pairwise comparisons (at least for the purposes of sampling), most of the theoretical justifications remain the same, and the only actual difference in the calculation of the required number of instances is the need to adjust the significance levels so as to maintain the overall probability of Type-I errors (false positives) controlled at the desired level α (Shaffer 1995).

The calculation of the number of instances for comparing two algorithms is based on the definition of a few desired statistical properties for the test (Campelo and Takahashi 2019). More specifically, we want the test to have a predefined statistical power, $\pi^* = (1 - \beta^*)$, to detect differences equal to or greater than a *minimally relevant effect size* d^*, at a predefined significance level α. This led to the definition of the smallest required number of instances for the comparison of two algorithms (based on the paired t-test, for a two-sided alternative hypothesis) as:

$$N^* = \min N \mid t_{1-\alpha/2}^{(N-1)} \leq t_{\beta^*;|ncp^*|}^{(N-1)} \tag{38}$$

in which $t_q^{(N-1)}$ is the q-quantile of the central t distribution with $N - 1$ degrees of freedom, and $t_{q;|ncp^*|}^{(N-1)}$ is the q-quantile of the non-central t distribution with $N - 1$ degrees of freedom and noncentrality parameter $|ncp^*| = |d^*|\sqrt{N}$. The required sample sizes can be further reduced in the *all versus one* comparisons if one-sided alternative hypotheses can be used, in which case we simply replace $t_{1-\alpha/2}^{(N-1)}$ by $t_{1-\alpha}^{(N-1)}$ in (38).[7]

Since our proposed protocol for sample size calculation is based on designing the experiments from the perspective of the pairwise tests that will need to be performed, the sample size formulas outlined above remain valid, and we only need to adjust the significance level to account for the testing of multiple hypotheses. As mentioned previously, we have two common cases in the experimental comparison of algorithms: *all versus all* comparisons (3), which result in $K = A(A-1)/2$ tests; and *all versus one* comparisons (4), in which case only $K = A - 1$ comparisons are performed.

As discussed by Juliet Shaffer in her review of multiple hypothesis testing (Shaffer 1995), "when many hypotheses are tested, and each test has a specified Type I error probability, the probability that at least some Type I errors are committed increases, often sharply, with the number of hypotheses". This is quite easy to illustrate if we consider that each test has a base probability of falsely rejecting a true null hypothesis given by its significance level α. If K comparisons are performed, and assuming that (i) the comparisons are independent, and (ii) all null hypotheses are true, the overall probability of observing one or more false positives (commonly called the *family-wise*

[7] The derivation of these results and a review of the statistical concepts associated with it are available in Campelo and Takahashi (2019). In that work we also showed how these results can be easily extended to the design of experiments based on nonparametric alternatives to the paired t-test, such as Wilcoxon's Signed Ranks test or the Binomial Sign test, by using the asymptotic relative efficiency of these tests (Montgomery and Runger 2013; Sheskin 2011). The same generalisations apply to the current work.

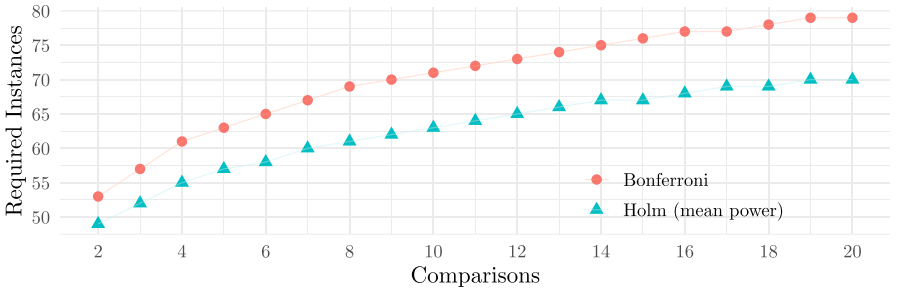

Fig. 1 Example of required sample sizes for Bonferroni-corrected tests in a common case with $\alpha_f = 0.05$, $\pi^* = 0.9$ and $d^* = 0.5$, considering a two-sided alternative, in comparison with the sample sizes required for obtaining the same mean power under Holm's correction. For reference, fifteen comparisons would represent *all versus all* comparisons involving six algorithms

error rate, or FWER) can be calculated as

$$\begin{aligned} FWER = P\left(fp \geq 1\right) &= 1 - P\left(fp = 0\right) \\ &= 1 - \left(1 - \alpha\right)^K . \end{aligned}$$

As an example, if we consider a somewhat typical case of *all versus all* comparisons of 5 algorithms (which would generate $K = 10$ hypotheses) with a per-comparison $\alpha = 0.05$, this would result in an $FWER = 1 - 0.95^{10} = 0.401$, i.e., a chance of around 40% of falsely rejecting at least one of the null hypotheses.

To prevent this inflation of the FWER, a wide variety of techniques for multiple hypothesis testing are available in the literature. Possibly the most widely known is the Bonferroni correction (Dunn 1961; Shaffer 1995), which can maintain the FWER strictly under a desired level α_f by using, for each comparison, a reduced significance level $\alpha' = \alpha_f / K$ (or, equivalently, by multiplying all p-values by K and comparing against the original α). This correction is known to be overly conservative, leading to substantial reductions of the statistical power of each comparison, or to substantially higher sample sizes being required to achieve the same power. However, its simplicity often makes it quite appealing, particularly when few hypotheses need to be tested, in which case the penalisation applied to α is not too extreme. If the Bonferroni correction is selected to control the FWER in a given experiment, the calculation of the number of instances N can be performed by simply dividing the value of α in (38) by the number of planned comparisons ($K = A - 1$ for the *all versus one* case, or $K = A(A-1)/2$ for *all versus all*). Figure 1 illustrates an example of the increase in the required sample size for Bonferroni-corrected tests as the number of comparisons is increased.

A uniformly more powerful alternative that also controls the FWER at a predefined level is Holm's step-down method (Holm 1979; Shaffer 1995). Instead of testing all hypotheses using the same corrected significance level, Holm's method employs a sequential correction approach. First, the p-values for all tests are computed. The hypotheses and their corresponding p-values are then ranked in increasing order of p-values, such that $p_1 \leq p_2 \ldots \leq p_K$. Each of these ordered hypotheses is then tested

at a different, increasingly stricter significance level,

$$\alpha'_r = \alpha_f / (K - r + 1) \tag{39}$$

in which r is the rank of the hypothesis being tested. If R is the largest value of r for which a hypothesis would be rejected at the corresponding α'_r level, then Holm's step-down method leads to the rejection of all null hypotheses with ranks $r \leq R$. Holm's method is also known to maintain the FWER under the desired value (Shaffer 1995), but it leads to significantly less conservative tests than the Bonferroni correction. However, while the calculation of the number of instances is quite straightforward for the Bonferroni correction, it requires some clarification when Holm's method is employed. This is because each test is performed using a different significance level, which leads to individual tests with heterogeneous power levels under a constant sample size.

One possible approach, which is found in the statistical literature, is to calculate the sample size based on the least favourable condition. We can calculate the sample size such that the test with the most strict significance level α'_r will have at least the desired power π^*, which guarantees that the statistical power for all other comparisons will be greater than π^*. A simple examination of (39) shows that the most strict test will occur for $r = 1$, in which case $\alpha'_r = \alpha_f / K$, i.e., the same corrected significance level generated by the Bonferroni correction. In this case, the same simple adjustment can be made for the formulas in (38), namely dividing the value of α by K. A related possibility, which generalises this one, is to calculate the sample size so that some predefined number of tests can have a statistical power of at least π^*. This can also be easily set up by determining a number K' of comparisons that should have power levels above the predefined threshold, and then divide α by $K - K' + 1$ in (38).

A second possibility, which is also relatively straightforward, is to design the experiment so that the mean (or median) power of the tests is maintained at the nominal level. The median case is roughly equivalent to setting $K' = \lceil K/2 \rceil$ and applying the method outlined above. Estimating sample size for achieving a mean power of π^* is more challenging from an analytic perspective, but can be done iteratively without much effort using Algorithm 2. In this pseudocode `SampleSize` (α, π, d^*, H_1) denotes the calculation of the required sample size for performing a hypothesis test (e.g., using a paired t-test) with significance level α, power β, MRES d^* and type of alternative hypothesis H_1, as defined in (38). Similarly, `Power` (α, n, d^*, H_1) calculates the power of a test procedure under the same parameters, assuming a sample size of n. Notice that since power increases monotonically with α, the final sample size returned by the procedure outlined in Algorithm 2 is guaranteed to be smaller than the result based on the least favourable condition.

Figure 2 illustrates the resulting power profile of an experiment designed for mean power using the procedure outlined in Algorithm 2. Two interesting aspects of this planning are immediately clear. First, the worst-case power appears to stabilize at about 0.85, which is not much lower than the desired 0.9, and would still be considered an acceptable power level in most applications. The second one is that the resulting sample sizes are, as expected, lower than those required if the Bonferroni correction were to be used.

Algorithm 2 Estimate sample size for mean power in Holm's procedure.

Require: Desired mean power (π^*); desired FWER (α_f); type of alternative hypotheses (one or two-sided) (H_1); number of comparisons (K)

$\bar{p} \leftarrow 0$
$N \leftarrow \texttt{SampleSize}(\alpha_f, \pi, d^*, H_1) - 1$ ▷ Using (38)
while $\bar{p} < \pi^*$ **do**
  $N \leftarrow N + 1$
  for $i \in \{1, \ldots, K\}$ **do**
    $p_i \leftarrow \texttt{Power}(\alpha/i, N, d^*, H_1)$ ▷ See Campelo and Takahashi (2019) for details.
  end for
  $\bar{p} = \sum_{i=1}^{K} p_i / K$ ▷ Calculate mean power
end while
return N

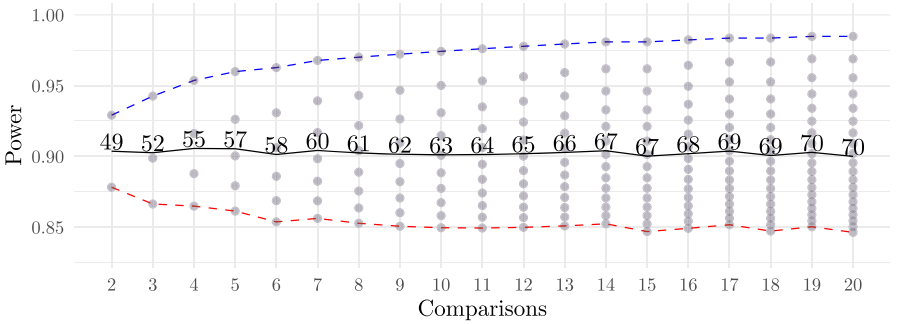

Fig. 2 Example of the resulting power of each comparison when designing experiments focused on mean power using Holm's method, in a common case with $\alpha_f = 0.05$, $\pi^* = 0.9$ and $d^* = 0.5$, considering a two-sided alternative. The lines illustrate the mean, best and worst-case power values, and the gray dots represent the power of each comparison. The labels following the mean-power line inform the sample sizes required for each value of K. Notice that the worst-case power seems to stabilize around 0.85 in this particular case

Based on these considerations, we can offer some suggestions as to how to proceed calculating the required number of instances (N) for the comparison of multiple algorithms using multiple problem instances:

- It is more efficient, often considerably so, to use Holm's correction instead of Bonferroni's, given that the former requires smaller sample sizes to achieve similar power characteristics (see Fig. 1).
- In general it seems reasonable to design experiments based on average power for Holm-corrected tests, given that the resulting worst-case power will not be much lower than the nominal value π^*, and the number of instances is substantially lower than what would be required for Bonferroni-corrected tests (see Fig. 2).
- If worst-case power must be guaranteed, the number of instances can be estimated using the formulas for the Bonferroni case, but the actual analysis should still be conducted using Holm's method, which in this case will result in better statistical power for all comparisons.

– Whenever possible, *all versus one* comparisons should be used instead of *all versus all*, since they result in fewer hypotheses being tested and, consequently, smaller sample sizes (or larger power if sample sizes are constant).
– If one-sided alternative hypotheses make sense in the experimental context of *all versus one* comparisons they should also be preferred, since this also improves efficiency (in terms of requiring fewer instances to achieve a predefined power).

It is worth repeating here an important point also highlighted in our previous work (Campelo and Takahashi 2019): the sample sizes calculated with the methodology presented in this section represent the *smallest* sample size required for a given experiment to present desired statistical properties. This can be particularly useful when, e.g., designing new benchmark sets, sampling a portion of an existing (possibly much larger) set of test functions as part of algorithm development, or designing experiments in computationally expensive contexts. If more instances are available in a particular experiment they can of course be used, which will result in experiments with even higher statistical power and can enable the execution of finer analyses, e.g., to investigate the effect of dimension or other instance characteristics on performance. However, even if that is the case, we argue that the proposed methodology can still be useful for a number of reasons:

– It can be used by researchers to obtain a more solid understanding of the statistical properties of their experiments (e.g., by examining power $\times$ effect size curves at a given sample size).
– By defining a MRES a priori, the proposed approach provides a sanity check for researchers in overpowered experiments, with can result in arbitrarily-low p-values even for minuscule differences of no practical consequence (Mathews 2010). That is, it provides a *practical relevance* aspect, to contrast with the *statistical significance* of the results.
– Even if the calculation of the number of instances is considered unnecessary in the context of a given experiment, the methodology for estimating the number of repeated runs of each algorithm on each instance can still be used to provide a statistically principled way to generate the data.

In the next section we provide an example of application of the proposed methodology for investigating the contribution of different neighbourhood structures for a scheduling problem. Please notice, however, that the main objective of the experiment is to illustrate (i) how the application of the proposed approach to the sampling of algorithms on each instance is capable of controlling the standard errors at the desired levels; (ii) how the sample size calculation routine can be applied to determining the smallest amount of instances that are required for an experiment to have the desired statistical properties; and (iii) how the proposed methodology as a whole can be applied in an iterative experimental investigation. The experiment was designed as a didactic example, not necessarily to answer very specific questions regarding the algorithms considered.

5 Application example

5.1 Problem description

In the *unrelated parallel machine scheduling problem (UPMSP) with sequence dependent setup times* (Graham et al. 1979; Lawler et al. 1993; Vallada and Ruiz 2011) a number of jobs, J, needs to be processed by a number of machines, M, minimising the completion time of the last job to leave the system, i.e., the makespan. This scheduling problem also presents a few other specificities which are not relevant for the current discussion, but are presented in detail in the relevant literature (Vallada and Ruiz 2011; Santos et al. 2016).

So far the best algorithm for the solution of this problem seems to be a finely-tuned Simulated Annealing (SA) proposed by Santos et al. (2016). Santos' SA was tested on a benchmark set originally introduced by Vallada and Ruiz (2011), which is composed of 200 tuning instances with $J \in \{50, 100, 150, 200, 250\}$ jobs and $M \in \{10, 15, 20, 25, 30\}$ machines. For each (M, J) pair, eight instances with different characteristics (e.g., distribution of setup times) are present. A larger set, composed of 1000 instances, is also available to test hypotheses derived using the tuning set.[8]

One of the core aspects of Santos' SA is the use of six neighbourhood structures to explore the performance landscape of the problem, namely *Shift*, *Two-shift*, *Task Move*, *Swap*, *Direct Swap*, and *Switch*. At each iteration one of these perturbation functions is selected randomly and generates a new candidate The authors do not provide any discussion related to how much each of these perturbation functions affects the performance of the algorithm. However, preliminary results (Maravilha et al. 2019; Pereira 2019) suggest that the six neighbourhood structures have considerably different effects on the algorithm, with *Task move* having a much more critical influence than the others.

5.2 Experimental questions

To further investigate the effect of the different neighbourhoods on the performance of the algorithm, we designed an experiment in which the *full* algorithm (i.e., equipped with all six neighbourhood structures) was compared against variants with perturbation functions suppressed from the pool of available movements. We focussed on removing at most two perturbations from the pool, resulting in 21 algorithm variants (6 without a single perturbation, and 15 without two perturbations) which enabled the quantification of contributions of neighbourhood structures to the algorithm's ability to navigate the performance landscape of this particular problem class.

Given that we wanted to investigate the effects of removing these neighbourhood functions from the *full* algorithm, an *all versus one* design was be the most appropriate, resulting in a total of 21 hypotheses to be tested. Also, the variability in instance

[8] Notice that the population of interest in this experiment is not the performance of the algorithms in the 200 tuning instances, or even in the 1000 test ones—rather, it is the (implicitly defined, possibly infinite) set of instances for which the available ones may be seen as a representative sample.

hardness suggested the use of percent differences (see Sect. 3.2.2), which quantify differences in a scale that is usually more consistent across heterogeneous instances.

5.3 Experimental parameters

We set the MRES at a (reasonably high) value of $d^* = 0.5$, i.e., we were interested in detecting variations in the mean of percent differences of at least half a standard deviation. We decided to design our experiment with a mean power $\pi^* = 0.8$, under a familywise significance level $\alpha = 0.05$ using Holm's correction. To prevent excess variability in the estimates of within-instance differences from inflating the overall residual variance of the experiment, we set the desired accuracy for the estimates as $se^* = 0.05$, i.e., a 5% error in the estimates of paired percent differences on each instance. This relatively good accuracy can also be useful if we later want to further investigate the differences on individual instances, or even instance subgroups.

The method described in Sect. 4 yielded $N = 57$ instances as the smallest amount necessary to obtain an experiment with the desired properties.[9] At this point we could have opted to use any number of instances greater than or equal to this value, up to the full available set of 200 (Vallada and Ruiz 2011). We opted to use only the suggested value of 57 not only because it was enough to guarantee the desired statistical properties for our experiment, but also to maintain a reserve of "untouched" tuning instances, which could be useful for further testing of eventual proposals of algorithmic improvement.[10]

The $A = 22$ algorithms (*full* algorithm plus the 21 variants) were sampled on each instance under a maximum budget of $50A = 50 \times 22 = 1100$ runs per instance, using an initial sampling of $n_0 = 10$ runs/algorithm/instance.

5.4 Experimental results

Figure 3 shows the standard errors obtained for all algorithm pairs, as well as the total sample sizes generated, for each instance used in this experiment. It is very clear that the proposed approach for calculating the number of runs was able to control the standard errors at the desired level of $se^* = 0.05$ using, in most cases, a fraction of the total available computational budget. In only three instances was the available budget insufficient to bring all standard errors under the desired threshold, but even in these cases the resulting standard errors were controlled at relatively low values. These three particular instances all had $J = 50$ jobs (one with $M = 15$ and two with $M = 25$ machines), which suggests further investigation into the behaviour of the methods tested on this particular subset of instance sizes.

[9] The full methodology for calculating the number of instances and algorithm runs is implemented in the `R` package `CAISEr` version 1.0.5, available at https://cran.r-project.org/package=CAISEr. The Java implementation of Santos' algorithm, as well as the instances used, can be retrieved online from https://github.com/fcampelo/upmsp.

[10] This is also a way to safeguard against overfitting our algorithms to the benchmark set, which is a common problem in algorithm design for problems with standard benchmark sets, such as the one being considered here (Bartz-Beielstein 2015; Hooker 1994, 1996).

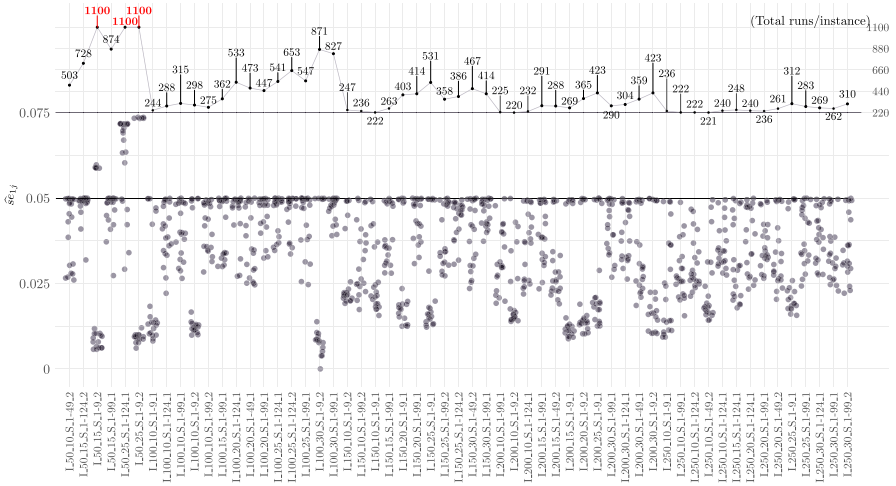

Fig. 3 Standard errors of estimation of the percent differences in performance indicator value between the *full* algorithm and each of the 21 variants, for all instances tested. The numbers at the top indicate the total number of runs performed on each instance, and the labels on the x-axis provide the instance identifiers. Notice that only in three cases the allocated budget was insufficient to reduce all standard errors below the preset limit of $se^* = 0.05$

Figure 4 shows the distribution of sample sizes for each algorithm involved in this experiment. As expected, the *full* algorithm received a reasonably large number of observations for the majority of instances, since it was involved in all comparisons. Another feature that becomes quite clear in this figure is that all versions in which the *Task Move* (TSK) neighbourhood function was suppressed also received much more observations, which suggests this neighbourhood as strongly influential and corroborates preliminary results obtained for Santos' SA on the UPMSP (Maravilha et al. 2019; Pereira 2019).

The results of the pairwise comparisons are shown in Fig. 5, which provides the confidence intervals (adjusted for a familywise error rate of $\alpha = 0.05$) for all comparisons between the *full* algorithm and the suppressed variants. As indicated by the relative importance attributed to the methods by the sampling method, the suppression of the TSK neighbourhood function led to strong degradations of performance, with mean percent losses around 50% or more. Also of interest in this table is the last column, which shows that the desired statistical properties of the experiment—particularly the ability to detect effects at or above approximately $d^* = 0.5$—were actually obtained in the designed experiment.

Of possibly equal interest in terms of understanding the contributing factors to algorithm performance is the fact that the algorithm did not experience significant performance losses after the removal of some of its neighbourhood structures. Table 1 presents the results of the t tests[11] on the mean of paired percent differences of performance. Examining the bottom rows of this table, it shows that the suppression of

[11] Graphical analysis of residuals indicated that the sampling distributions of means in all cases were sufficiently close to normality, allowing the use of t tests for inference.

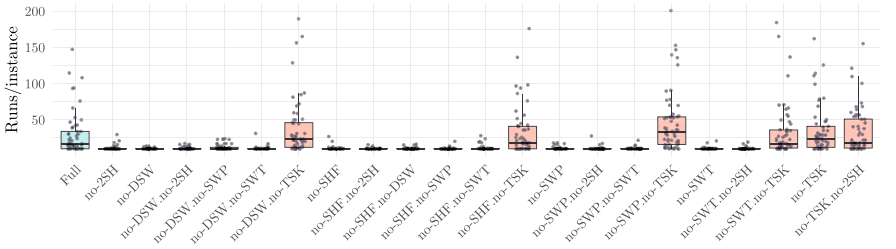

Fig. 4 Distribution of $n_{k|\ell}$, i.e., of the number of runs for each algorithm variant on each instance. The majority of runs were allocated either to the *full* algorithm—which was involved in all comparisons—or to variants that had the *Task Move* (TSK) suppressed, suggesting a strong effect of this perturbation function

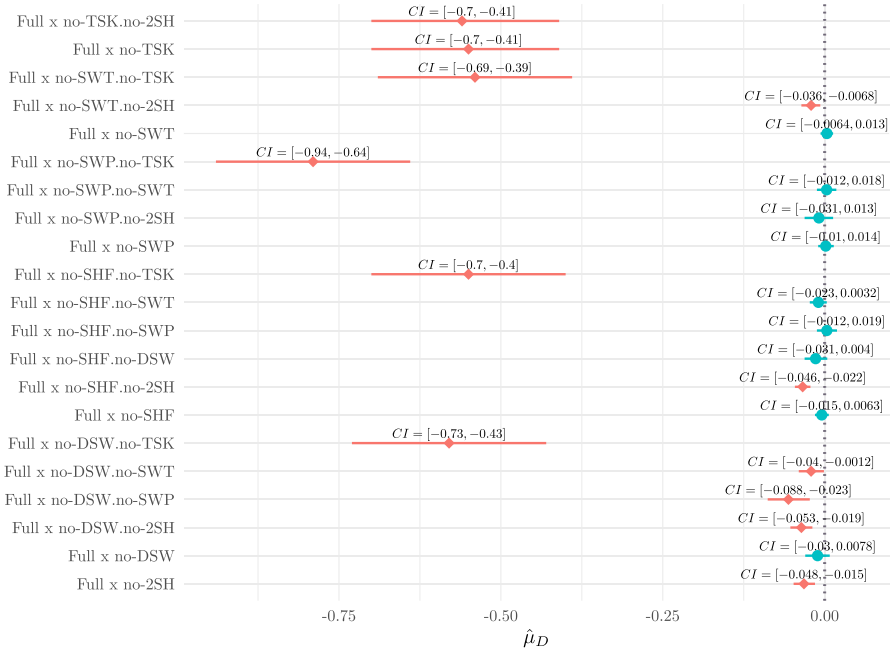

Fig. 5 Point estimates and confidence intervals (95% joint confidence level) for the mean of paired percent differences of performance for the 21 comparisons of algorithm variants against the *full* simulated annealing. Red intervals with a diamond marker indicate statistically significant results, while green intervals with circle markers indicate those where the null hypotheses were not rejected. Apart from the variants without the *Task Move* perturbation function (TSK), all others resulted only in minor performance degradation, if any

Swap (SWP), *Switch* (SWT) and *Shift* (SHF), individually as well as in pairs, resulted in no significant differences in performance. The width of the associated confidence intervals also suggests that any effects that may have gone undetected must be rather small.

To further explore these initial insights a follow-up experiment was performed. The objective of this second analysis was to obtain a more precise estimate of the effect (or lack thereof) of removing these three neighbourhood functions, individually as well

Table 1 Summary of the 21 comparisons against the *Full* algorithm, using Holm's step-down procedure

Comparison	α'_r	p value	Confidence interval	$\widehat{d}$
Full × no-SWP.no-TSK	0.0024	3.7×10^{-23}	-0.79 ± 0.15	-2.2
Full × no-TSK.no-2SH	0.0025	1.7×10^{-17}	-0.56 ± 0.14	-1.6
Full × no-DSW.no-TSK	0.0026	5.2×10^{-17}	-0.58 ± 0.15	-1.6
Full × no-TSK	0.0028	1.0×10^{-16}	-0.55 ± 0.15	1.6
Full × no-SWT.no-TSK	0.0029	4.4×10^{-16}	-0.54 ± 0.15	-1.5
Full × no-SHF.no-TSK	0.0031	6.3×10^{-16}	-0.55 ± 0.15	-1.5
Full × no-SHF.no-2SH	0.0033	3.6×10^{-12}	-0.034 ± 0.012	-1.2
Full × no-DSW.no-2SH	0.0036	3.5×10^{-8}	-0.036 ± 0.017	-0.85
Full × no-2SH	0.0038	3.3×10^{-7}	-0.032 ± 0.016	-0.77
Full × no-DSW.no-SWP	0.0042	4.5×10^{-6}	-0.056 ± 0.033	-0.67
Full × no-SWT.no-2SH	0.0045	6.0×10^{-5}	-0.021 ± 0.015	-0.57
Full × no-DSW.no-SWT	0.005	0.003	-0.021 ± 0.019	-0.41
Stop rejecting H_0				
Full × no-SHF.no-DSW	0.0056	0.03	-0.014 ± 0.018	0.29
Full × no-SHF.no-SWT	0.0062	0.038	-0.0097 ± 0.013	-0.28
Full × no-DSW	0.0071	0.1	-0.011 ± 0.019	-0.22
Full × no-SWP.no-2SH	0.0083	0.27	-0.0089 ± 0.022	-0.15
Full × no-SHF	0.01	0.27	-0.0045 ± 0.011	0.15
Full × no-SWT	0.012	0.37	0.0035 ± 0.0099	0.12
Full × no-SHF.no-SWP	0.017	0.6	0.0033 ± 0.015	0.071
Full × no-SWP.no-SWT	0.025	0.65	0.003 ± 0.015	0.06
Full × no-SWP	0.05	0.79	0.0016 ± 0.012	0.036

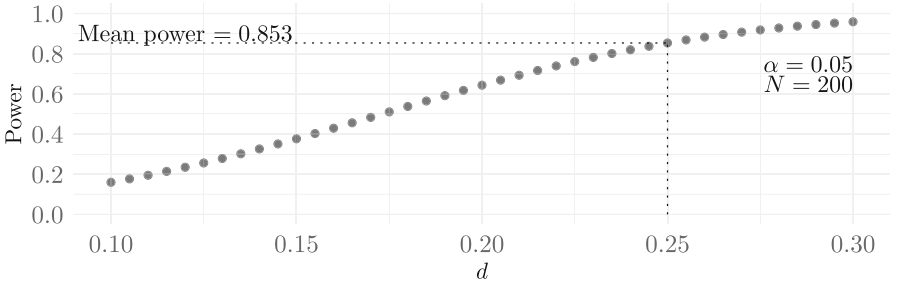

Fig. 6 Power curve for the follow-up experiment, obtained by setting $N = 200$ and iterating over effect sizes to calculate the corresponding mean power of the experiment to detect different values of d

as in pairs, from the pool of movements available to the algorithm. Further, we also added the variant generated by simultaneously removing these three functions, which was not present in the first experiment. This means that this follow-up experiment involved 7 variants being compared against the *Full* algorithm. The full 200-instance test set was employed for this follow-up experiment, which provides a mean power of

Table 2 Summary of the 7 follow-up comparisons against the *Full* algorithm

Comparison	α'_r	p value	Confidence interval	$\widehat{d}$
Full × no-SWP.no-SWT	0.0071	0.032	0.018 ± 0.022	0.15
Full × no-SWP	0.0083	0.043	0.017 ± 0.022	0.14
Full × no-SHF.no-SWP	0.01	0.072	0.015 ± 0.021	0.13
Full × no-SHF.no-SWP.no-SWT	0.012	0.12	0.013 ± 0.021	0.11
Full × no-SHF	0.017	0.28	0.0087 ± 0.019	0.077
Full × no-SHF.no-SWT	0.025	0.29	0.0084 ± 0.018	0.075
Full × no-SWT	0.05	0.98	$5.3 \times 10^{-5} \pm 0.0048$	0.0015

None of the results was statistically significant at the joint 95% confidence level

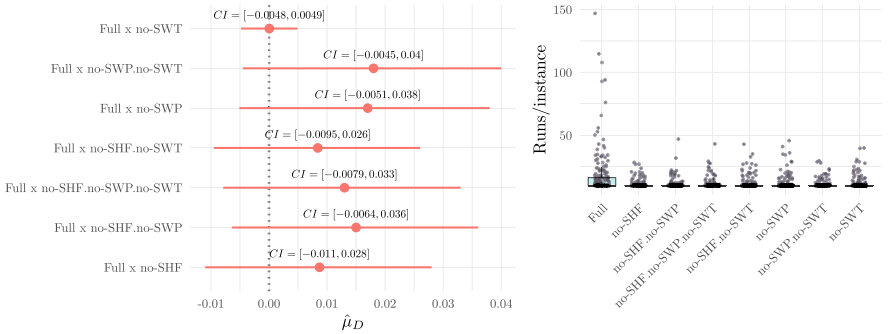

Fig. 7 Joint 95% confidence intervals and sample sizes in the follow-up experiment. Notice that no comparison yielded statistically significant results, and that the narrow confidence intervals suggest that even if the neighbourhoods tested have some effect on the algorithm performance it is likely to be of minor consequence

$\pi \approxeq 0.85$ to detect differences as small as $d^* = 0.25$. This was calculated by deriving the $d^* \times \pi^*$ curve shown in Fig. 6, using the same approach described in Sect. 4.

Table 2 and Fig. 7 summarise the results obtained for the follow-up experiment. Notice that even with the increased sensitivity of the tests to the mean of percent differences for the variants that had the SWP, SHF and SWT neighbourhood functions removed from the pool of possible movements the tests failed to detect any statistically significant differences. This result, coupled with the narrow confidence intervals, indicates that any effect that these neighbourhood functions may be having on the performance of the algorithm are probably very small, if they exist at all. Based on these results, algorithm designers and developers could expand this exploration even further by, for instance, (i) focussing on the structures that are being explored by the remaining neighbourhood functions, so as to gain better understanding of the problem's landscape, or (ii) perform further exploration into instance subsets, to investigate whether the systematic lack of effect observed in this experiment also occurs, e.g., when conditioning the results on problem size. These further explorations are, however, outside the scope of the current paper.

6 Conclusions

In this work we expanded methodology first introduced in Campelo and Takahashi (2019) to enable sample size calculations for comparative experiments involving an arbitrary number of algorithms for the solution of a problem class of interest. The two aspects of sample size estimation for the comparative performance experiments, namely the number of instances and the amount of runs allocated to each algorithm on each instance, were addressed in a statistically principled manner.

The number of instances is determined based on the desired sensitivity (in terms of statistical power) for the detection of effects larger than a predefined *minimally relevant effect size* (MRES), under a given family-wise confidence level. Holm's step-down procedure was employed to control the rate of false positives while enabling the design of experiments at desired levels of power for the best, worst, median or mean case. The proposed methodology is based on the assumptions that underlie the t-tests, but the results can be easily generalised to the most common non-parametric tests (e.g., Wilcoxon signed-ranks test) based on asymptotic relative efficiency, as discussed previously in (Campelo and Takahashi 2019).

The proposed iterative sampling method used to determine the number of runs of each algorithm on each instance is based on the interpretation of the standard error of estimation as a measure of precision, and generating samples aimed at minimising the total amount of runs necessary to obtain estimates with standard errors below predefined thresholds. Optimal sample size ratios were derived for simple and percent differences between the mean of all pairs of algorithms on any given instance, and a sampling heuristic was proposed that enables generalisation for other statistics that may be of eventual interest to researchers, such as the variance, algorithm rankings etc.

An example of application was provided in Sect. 5, using the publicly available `R` package `CAISEr` (Campelo 2019), which provides an easy to use implementation of the proposed methodology. The example focussed on the comparison of a state-of-the-art heuristic for the solution of a class of scheduling problems against 21 algorithm variants generated by suppressing different neighbourhood functions used to generate candidate solutions. The preliminary results indicated not only the most relevant neighbourhood function but also the apparent lack of effect of three of the six ones commonly employed. The first part of the experiment illustrated the ability of the proposed method to sample the algorithms so as to control the standard errors at each instance under the desired accuracy threshold. It also showcased the compliance between the MRES used for designing the experiment, and the effect sizes that the hypothesis tests were able to detect as statistically significant.

The results obtained in the first part of the experiment were used to design a follow-up investigation, in which the effects of the three neighbourhood structures flagged in the initial part as not statistically significant were further investigated. This follow-up experiment used the full available instance set, and was used to showcase the use of the proposed methodology in a fixed sample size situation. The results obtained not only corroborated those obtained in the first part, but further reinforced the tentative conclusion that the effect of using the three neighbourhood functions as part of the

algorithm must be minor at best, which suggests a few possibilities of exploration for algorithm designers and developers.

6.1 Limitations and possibilities

As in our previous work upon which the present paper was based, it is important to reinforce at that the proposed methodology is not the definitive way to compare algorithms. For instance, convergence analysis, algorithm reliability, and performance analysis conditional on problem characteristics are research questions that require different methodologies. The proposed methodology is simply an additional tool in the research arsenal, one that can provide answers to several common questions in the experimental comparison of algorithms.

One of the most straightforward extensions of the work presented in this paper is the incorporation of sequential analysis approaches (Botella et al. 2006; Bartroff et al. 2013) to enable further reductions in the required number of instances, which is possible in cases where the actual effect sizes are substantially larger than the MRES defined in the experimental design. Extending the sample size calculations performed here to Bayesian alternatives (Calvo et al. 2019) is also relatively straightforward. In the Bayesian case the number instances and repetitions on each instance are used to reduce the uncertainty in distinct levels of a hierarchical model of algorithm performance (Kruschke 2010), and iterative sampling approaches can be used without the corrections required by frequentist sequential analysis. This feature of Bayesian methods is starting to be explored for the comparison of algorithms in machine learning and optimisation (Benavoli et al. 2017; Calvo et al. 2019), but as far as we are aware there are still no specific quantitative discussions on the balance between the number of repetitions and of instances for these Bayesian approaches to algorithm comparisons. The issue of determining priors based on previously published results, which may eventually provide meta-analytical tools in algorithmic research, also remains an exciting possibility for these methods.

References

Barr, R.S., Golden, B.L., Kelly, J.P., Resende, M.G.C., Stewart, W.R.: Designing and reporting on computational experiments with heuristic methods. J. Heuristics **1**(1), 9–32 (1995)

Bartroff, J., Lai, T., Shih, M.C.: Sequential Experimentation in Clinical Trials: Design and Analysis. Springer, New York (2013)

Bartz-Beielstein, T.: New Experimentalism Applied to Evolutionary Computation. Ph.D. thesis, Universität Dortmund, Germany (2005)
Bartz-Beielstein, T.: Experimental Research in Evolutionary Computation. Springer, New York (2006)
Bartz-Beielstein, T.: How to create generalizable results. In: Kacprzyk, J., Pedrycz, W. (eds.) Handbook of Computational Intelligence. Springer, New York (2015)
Bartz-Beielstein, T., Chiarandini, M., Paquete, L., Preuss, M.: Experimental Methods for the Analysis of Optimization Algorithms. Springer, New York (2010)
Benavoli, A., Corani, G., Mangili, F., Zaffalon, M., Ruggeri, F.: A Bayesian Wilcoxon signed-rank test based on the Dirichlet process. In: 30th International conference on machine learning, pp. 1026–1034 (2014)
Benavoli, A., Corani, G., Demšar, J., Zaffalon, M.: Time for a change: a tutorial for comparing multiple classifiers through Bayesian analysis. J. Mach. Learn. Res. **18**(1), 2653–2688 (2017)
Birattari, M.: On the estimation of the expected performance of a metaheuristic on a class of instances: how many instances, how many runs? Technical report IRIDIA/2004-001, Université Libre de Bruxelles, Belgium (2004)
Birattari, M.: Tuning Metaheuristics—A Machine Learning Perspective. Springer, Berlin (2009)
Birattari, M., Dorigo, M.: How to assess and report the performance of a stochastic algorithm on a benchmark problem: mean or best result on a number of runs? Optim. Lett. **1**, 309–311 (2007)
Botella, J., Ximénez, C., Revuelta, J., Suero, M.: Optimization of sample size in controlled experiments: the CLAST rule. Behav. Res. Methods **38**(1), 65–76 (2006)
Calvo, B., Shir, O.M., Ceberio, J., Doerr, C., Wang, H., Bäck, T., Lozano, J.A.: Bayesian performance analysis for black-box optimization benchmarking. In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '19, pp. 1789–1797. ACM (2019)
Campelo, F.: CAISEr: Comparison of Algorithms with Iterative Sample Size Estimation (2019). https://CRAN.R-project.org/package=CAISEr. Package version 1.0.13
Campelo, F., Takahashi, F.: Sample size estimation for power and accuracy in the experimental comparison of algorithms. J. Heuristics **25**(2), 305–338 (2019)
Carrano, E.G., Wanner, E.F., Takahashi, R.H.C.: A multicriteria statistical based comparison methodology for evaluating evolutionary algorithms. IEEE Trans. Evol. Comput. **15**(6), 848–870 (2011)
Chimani, M., Klein, K.: Algorithm engineering: concepts and practice. In: Bartz-Beielstein, Th, Chiarandini, M., Paquete, L., Preuss, M. (eds.) Experimental Methods for the Analysis of Optimization Algorithms, pp. 131–158. Springer, Berlin (2010)
Coffin, M., Saltzman, M.J.: Statistical analysis of computational tests of algorithms and heuristics. INFORMS J. Comput. **12**(1), 24–44 (2000)
Czarn, A., MacNish, C., Vijayan, K., Turlach, B.: Statistical exploratory analysis of genetic algorithms: the importance of interaction. In: Proceedings of the 2004 IEEE Congress on Evolutionary Computation. Institute of Electrical & Electronics Engineers (IEEE) (2004)
del Amo, I.G., Pelta, D.A., González, J.R., Masegosa, A.D.: An algorithm comparison for dynamic optimization problems. Appl. Soft Comput. **12**(10), 3176–3192 (2012)
Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**, 1–30 (2006)
Derrac, J., García, S., Molina, D., Herrera, F.: A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol. Comput. **1**(1), 3–18 (2011)
Derrac, J., García, S., Hui, S., Suganthan, P.N., Herrera, F.: Analyzing convergence performance of evolutionary algorithms: a statistical approach. Inf. Sci. **289**, 41–58 (2014)
Dunn, O.J.: Multiple comparisons among means. J. Am. Stat. Assoc. **56**(293), 52–64 (1961)
Eiben, A., Jelasity, M.: A critical note on experimental research methodology in EC. In: Proceedings of the 2002 IEEE Congress on Evolutionary Computation. Institute of Electrical & Electronics Engineers (IEEE) (2002)
Ellis, P.D.: The Essential Guide to Effect Sizes, 1st edn. Cambridge University Press, Cambridge (2010)
Fieller, E.C.: Some problems in interval estimation. J. R. Stat. Soc. Ser. B (Methodol.) **16**(2), 175–185 (1954)
Franz, V.: Ratios: a short guide to confidence limits and proper use (2007). arXiv:0710.2024v1
García, S., Molina, D., Lozano, M., Herrera, F.: A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special session on real parameter optimization. J. Heuristics **15**(6), 617–644 (2008)

García, S., Fernández, A., Luengo, J., Herrera, F.: A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability. Soft Comput. **13**(10), 959–977 (2009)
García, S., Fernández, A., Luengo, J., Herrera, F.: 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 (2010)
Gelman, A., Hill, J.: Data Analysis Using Regression and Multilevel/Hierarchical Models. Cambridge University Press, Cambridge (2006)
Graham, R.L., Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G.: Optimization and approximation in deterministic sequencing and scheduling: a survey. Ann. Discrete Math. **5**, 287–326 (1979)
Hansen, N., Tusar, T., Mersmann, O., Auger, A., Brockoff, D.: COCO: the experimental procedure (2016). arXiv:1603.08776
Holm, S.: A simple sequentially rejective multiple test procedure. Scand. J. Stat. **6**(2), 65–70 (1979)
Hooker, J.N.: Needed: an empirical science of algorithms. Oper. Res. **42**(2), 201–212 (1994)
Hooker, J.N.: Testing heuristics: we have it all wrong. J. Heuristics **1**(1), 33–42 (1996)
Hurlbert, S.H.: Pseudoreplication and the design of ecological field experiments. Ecol. Monogr. **54**(2), 187–211 (1984)
Jain, R.K.: The Art of Computer Systems Performance Analysis. Wiley, New York (1991)
Johnson, D.: A theoretician's guide to the experimental analysis of algorithms. In: Goldwasser, M., Johnson, D., McGeoch, C. (eds.) Data Structures, Near Neighbor Searches, and Methodology: Fifth and Sixth DIMACS Implementation Challenges, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 59, pp. 215–250. American Mathematical Society, Providence (2002)
Krohling, R.A., Lourenzutti, R., Campos, M.: Ranking and comparing evolutionary algorithms with hellinger-TOPSIS. Appl. Soft Comput. **37**, 217–226 (2015)
Kruschke, J.K.: Doing Bayesian Data Analysis: A Tutorial with R and BUGS, 1st edn. Academic Press, Cambridge (2010)
Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H., Shmoys, D.B.: Sequencing and scheduling: algorithms and complexity. In: Handbooks in Operations Research and Management Science, chapter 9, vol. 4, pp. 445–522. Elsevier (1993)
Lazic, S.E.: The problem of pseudoreplication in neuroscientific studies: is it affecting your analysis? BMC Neurosci. **11**(5), 397–407 (2010)
Lenth, R.V.: Some practical guidelines for effective sample size determination. Am. Stat. **55**(3), 187–193 (2001)
Maravilha, A.L., Pereira, L.M., Campelo, F.: Statistical characterization of neighborhood structures for the unrelated parallel machine problem with sequence-dependent setup times **(in preparation)**
Mathews, P.: Sample Size Calculations: Practical Methods for Engineers and Scientists, 1st edn. Matthews Malnar & Bailey Inc., Painesville (2010)
McGeoch, C.C.: Feature article-toward an experimental method for algorithm simulation. INFORMS J. Comput. **8**(1), 1–15 (1996)
Millar, R., Anderson, M.: Remedies for pseudoreplication. Fish. Res. **70**, 397–407 (2004)
Montgomery, D.C.: Design and Analysis of Experiments, 8th edn. Wiley, New York (2013)
Montgomery, D.C., Runger, G.C.: Applied Statistics and Probability for Engineers, 6th edn. Wiley, New York (2013)
Pereira, L.M.: Análise de Estruturas de Vizinhança para o Problema de Sequenciamento de Máquinas Paralelas Não Relacionadas com Tempos de Preparação . Master's thesis, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil (2019). https://ppgee.ufmg.br/defesas/1615M.PDF**(in Portuguese)**
Ridge, E.: Design of Experiments for the Tuning of Optimisation Algorithms. Ph.D. thesis, The University of York, UK (2007)
Santos, H.G., Toffolo, T.A., Silva, C.L., Berghe, G.V.: Analysis of stochastic local search methods for the unrelated parallel machine scheduling problem. Int. Trans. Oper. Res. (2016). https://doi.org/10.1111/itor.12316
Shaffer, J.P.: Multiple hypothesis testing. Annu. Rev. Psychol. **46**(1), 561–584 (1995)
Sheskin, D.J.: Handbook of Parametric and Nonparametric Statistical Procedures. Taylor & Francis, Milton Park (2011)
Sörensen, K., Sevaux, M., Glover, F.: A history of metaheuristics. In: Martí, R., Pardalos, P.M., Resende, M.G. (eds.) Handbook of Heuristics, pp. 1–18. Springer, New York (2018)

Vallada, E., Ruiz, R.: A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times. Eur. J. Oper. Res. **211**(3), 612–622 (2011)
Yuan, B., Gallagher, M.: An improved small-sample statistical test for comparing the success rates of evolutionary algorithms. In: Proceedings of the 11th Annual conference on Genetic and evolutionary computation—GECCO09. Association for Computing Machinery (ACM) (2009)
Yuan, B., Gallagher, M.: Statistical racing techniques for improved empirical evaluation of evolutionary algorithms. Parallel Probl. Solving Nat. PPSN VIII **3242**, 172–181 (2004)