Journal of
the Brazilian Computer Society
a SpringerOpen Journal

RESEARCH Open Access

Factorial design analysis applied to the performance of parallel evolutionary algorithms

Mônica S Pais[1*], Igor S Peretta[2], Keiji Yamanaka[2] and Edmilson R Pinto[3]

Abstract

Background: Parallel computing is a powerful way to reduce computation time and to improve the quality of solutions of evolutionary algorithms (EAs). At first, parallel EAs (PEAs) ran on very expensive and not easily available parallel machines. As multicore processors become ubiquitous, the improved performance available to parallel programs is a great motivation to computationally demanding EAs to turn into parallel programs and exploit the power of multicores. The parallel implementation brings more factors to influence performance and consequently adds more complexity on PEA evaluations. Statistics can help in this task and can guarantee the significance and correct conclusions with minimum tests, provided that the correct design of experiments is applied.

Methods: We show how to guarantee the correct estimation of speedups and how to apply a factorial design on the analysis of PEA performance.

Results: The performance and the factor effects were not the same for the two benchmark functions studied in this work. The Rastrigin function presented a higher coefficient of variation than the Rosenbrock function, and the factor and interaction effects on the speedup of the parallel genetic algorithm I (PGA-I) were different in both.

Conclusions: As a case study, we evaluate the influence of migration related to parameters on the performance of the parallel evolutionary algorithm solving two benchmark problems executed on a multicore processor. We made a particular effort in carefully applying the statistical concepts in the development of our analysis.

Keywords: Parallel evolutionary algorithms; Design of experiments; Factorial design

Background

Evolutionary algorithms (EA) are highly effective solvers of optimization problems for which no efficient methods are known [1]. Very often, EAs require computational power, and there have been efforts to improve their performance through parallel implementation [2,3]. In fact, for parallel EAs (PEAs), it is possible to achieve superlinear speedups [4].

Parallel computers have been synonymous with supercomputers: large and expensive machines that are restricted to few research centers. Since the microprocessor industry turned to multicore processors, multicore architectures have quickly spread to all computing domains, from embedded systems to personal computers, making parallel computing affordable to all [5]. This is a good incentive to convert EAs into PEAs. However, the complexity of PEA evaluations is a drawback. This complexity basically comes from the inherent complexity of parallelism and the additional parameters of a PEA.

Evaluations of parallel algorithms adopt a widely used performance measure called speedup. Speedup corresponds to the ratio of two independent random variables with positive means, and it does not, in general, have a well-defined mean [6,7]. As PEAs are randomized algorithms, measures are usually averages. In this work, we show how to guarantee the correct estimation of the average speedups.

*Correspondence: monica.pais@ifgoiano.edu.br
[1]Department of Informatics, Goiano Federal Institute of Education, Science and Technology (IFGoiano), Campus Urutaí, Rodv. Geraldo Silva km 2,5, Urutaí, Goiás 75790-000, Brazil
Full list of author information is available at the end of the article

PEA performances vary due to many factors, where factor is an independent variable - a manipulated variable in an experiment whose presence or degree determines the change in the output. These factors can be classified as EA factors, migration factors, computer platform factors, and factors related to the problem to be solved.

If one is interested in finding out how some of these factors can influence the PEA performance, it is necessary to make purposeful changes in these factors so that we may observe and identify the reasons for changes in their performances. A common method to evaluate performances is known as 'one-factor-at-a-time'. This method evaluates the influence of each factor by varying one factor at a time, keeping all other factors constant. It cannot capture the interaction between factors. Other common method is the exhaustive test of all factors and all combinations at all levels. This is a complete experiment but very expensive approach. Those two methods are used when the number of factors and the number of levels are small.

A factorial design is a strategy in which factors are simultaneously varied, instead of one at a time. It is recommended to use a 2^k factorial design when there are many factors to be investigated, and we want to find out which factors and which interactions between factors are the most influential on the response of the experiment. In this work, our response is the speedup of a PEA. The 2^k factorial designs are indicated when the main effects and interactions are supposed to be approximately linear in the interval of interest. Quadratic effects, for instance, would require another design, such as a central composite design, for an appropriate evaluation.

The objective of this paper is to introduce the factorial design methodology applied to the experimental performance evaluation of PEAs executed on a multicore platform. Our methodology addresses the planning of experiments and the correct speedup estimation. The main contributions of this paper are summarized as follows: (1) the measurement of factor effects on the performance of a PEA by varying two levels of each factor, (2) a method to guarantee the correct estimation of speedup, and (3) a case study of the influence of migration related to parameters on the performance of a PEA executed on a multicore processor.

This paper is structured as follows. In the 'Related work' subsection of the 'Methods' section, we summarize some recent works that present proposals of algorithm evaluation techniques based on factorial design. The 'Conceptual background' subsection introduces the theoretical concepts that are applied in our experiments. The implementation, test functions, and computer platform are described in the 'Implementation' subsection of the 'Results and discussion' section. The design of our experiments and analyses are described in the 'Case studies' subsection. The 'Conclusions' section presents the conclusion and further work.

Methods

Related work

There is a distinction among the purposes of the performance evaluation of evolutionary algorithm. In some works, the goal is to compare different algorithms and find out which one has the best performance. Others compare the same algorithm with different configurations, and the goal is to find out which configuration brings improvements to the algorithm performance. Our work is related to the latter, specifically to methods that make use of design of experiments and other statistical methods. Some of the works presented in Eiben and Smit's survey of tuning methods [8] are of the same kind as ours.

We also relate our work to the evaluation of program speedups. Touati et al. [9] proposed a performance evaluation methodology applying statistical tools. They argued that the variation of execution times of a program should be kept under control and inside the statistics. To do so, they proposed the use of the median instead of the mean as a better performance metric because it is more robust to outliers. The central limit theorem does not apply to the median but to the mean. In our work, we use the mean statistic with the awareness that it is not a robust statistic.

On the evaluation of parallel evolutionary algorithms, Alba et al. [10] presented some parallel metrics and illustrated how they can be used with parallel metaheuristics. Their definition of *orthodox* speedup is used in this work (see the 'Speedup' subsection).

Coy et al. [11] applied linear regression, the analysis of variance (ANOVA) test, fractional factorial design, and response surface methodology to adjust the parameters of two heuristics based on a local search, both deterministic procedures. In our work, we deal with nondeterministic procedures.

Czarn et al. [12] studied the influence of two genetic algorithm parameters: mutation and crossover rates. They applied the ANOVA test and multiple comparisons to find out the significance of the effects of the two parameters and their interactions on four benchmark functions. The authors advocate that the seed of the pseudorandom generator (PRNG) is a factor that influences the variability and that its influence should be blocked. Bartz [13] calls seeds antithetic: they are used to start up a sequence of pseudorandom numbers and can be used to reproduce the same sequence of numbers. Here, we did not block the seed factor, and we followed Rardin and Uzsoy [14] who recommend to execute several runs with different PRNG seeds controlling the evolution of computation to get a sense of the robustness of the procedure.

Factorial design has been applied to tune the genetic algorithm parameters in the works of Shahsavar et al. [15], Pinho et al. [16], and Petrovski et al. [17]. None of them addresses a PEA. The parallelism and the randomness of the algorithm bring additional issues to the application of statistical methods for the evaluation of performance, such as the choice of performance measures, the data distribution of these measures, and the variability brought by parallel executions. Those issues are addressed in our work.

Conceptual background

The concepts employed in this work are briefly introduced in this section.

Parallel evolutionary algorithms

EAs are naturally prone to parallelism since most of their operators can be easily undertaken in parallel. As Darwin realized long ago, populations may have a spatial structure, and this spatial structure may have influence on population dynamics [18]. The use of a structured population - a spatial distribution of individuals in the form of either a set of islands or a diffusion grid - determines the dynamical processes that can take place in complex systems. In a *panmict* population, all individuals are a potential partner for mating. A *multideme* population is constituted of isolated populations, called *demes*. Each *deme* can evolve differently than the ones not isolated. Even if the diversity in the demes is low, the diversity of the entire population is high [3].

There are different models to exploit the parallelism of EAs: master-slave model, fine-grained (or cellular) model, and coarse-grained (or island) model. There are also hybrid models which are a combination of these three basic models [10].

The coarse-grained (or island) model is easy to implement but complex to tune. Each subpopulation evolves in isolation, and periodically they exchange individuals with other subpopulations. This migration mechanism is adjusted by a set of parameters, such as migration frequency, migration topology, selection strategy of individuals to migrate, and placing strategy of immigrants. This model is implemented in our case study.

Experiments with algorithms

Since algorithms are mathematical abstractions, some researchers in computer science maintain a purely formal approach on the study of the algorithm behavior. At some point, the algorithm will be written into a programming language and run on a computer. This transforms a mathematical abstraction into a real-world matter which calls for a natural science approach: an experimental approach. Hooker [19,20] was one of the first authors to advocate that the theoretical approach alone is not able to explain how algorithms work on the solution of real problems. He showed the need of statistical thinking and principles into the experimental approach on the study of algorithms, the same that has been done by nature-related science experimentations.

McGeoch [21], Johnson [22], and Rardin and Uzsoy [14] are also pioneering authors who brought important contributions on the formation of an algorithm experimentation science and reinforced the use of statistics as a systematic way of analysis. Eiben and Jelasity [23] addressed the necessity of a sound research methodology supporting results of experiments with EAs.

EAs are nondeterministic algorithms. Their stochastic nature introduces some random variability in the answer provided by the algorithm: the solution obtained by the algorithm can vary considerably from one run to another, and even when the same solution is reached, the computational time required for achieving such a solution is usually different for different runs of the same algorithm.

These characteristics make the theoretical analysis of EAs more difficult, and most of the studies with EAs are done with an empirical approach [1]. Also, the present diversity of computer architectures cannot fit in one model; the experimentation on current computers is relevant and necessary to gain more precise prognostics about EA performance and robustness.

Experimental design

Experimentation has been used in diverse areas of knowledge. Statistical design of experiments has the pioneering work of Sir R.A. Fisher in the 1920s and early 1930s [24]. His work had profound influence on the use of statistics in agricultural and related life sciences. In the 1930s, applications of statistical design in industrial settings began with the recognition that many industrial experiments are fundamentally different from their agricultural counterparts: the response variable can usually be observed in shorter time than in agricultural experiments, and the experimenter can quickly learn crucial information from a small group of runs that can be used to plan the next experiment. Over the next 30 years, design techniques had spread through the chemical and the process industries. There has been a considerable utilization of designed experiments in areas like the service sector of business, financial services, and government operations [25].

Montgomery [25] defines an experiment as a test or a series of tests in which purposeful changes are made to the input variables - factors - of a process or system so that we may examine and identify the reasons for changes that may be observed in the output response. Statistical design of experiments refers to planning the experiment in a way that proper data will be collected and analyzed by statistical methods, resulting in valid and objective conclusions.

Experimental design has three principles: randomization, replication, and blocking. The order of the runs in the experimental design is randomly determined. Randomization helps in avoiding violations of independence caused by extraneous factors, and the assumption of independence should always be tested. Replication is an independent repeat of each combination of factors. It allows the experimenter to obtain an estimate of the experimental error. Blocking is used to account for the variability caused by controllable nuisance factors, to reduce and eliminate the effect of this factor on the estimation of the effects of interest. Blocking does not eliminate the variability; it only isolates its effects. A nuisance factor is a factor that may influence the experimental response but in which we are not interested.

The experimental planning phases are as follows: (1) defining of objectives of the experiment, (2) choosing measures of performance, factors to explore, and factors to be held constant (3) designing and executing the experiment (gather data), (4) analyzing the data and drawing conclusions (performing follow-up runs and confirmation testing to validate the conclusions), and (5) reporting the experiment's results [26].

An EA experiment is a set of algorithm's implementations that run under controlled conditions to check the validity of a hypothesis. These controlled conditions are a set of parameters, a set of execution platforms, a set of problem instances, and a set of performance measures.

Experimental goals

Computational experiments with algorithms are usually undertaken for (1) comparing the performance of different algorithms for the same class of problems or (2) characterizing or describing an algorithm's performance in isolation. The former motivation, comparing algorithms, is related to algorithm effectiveness in solving specific classes of problems. It often involves the comparison of a new approach to established techniques. On the latter motivation, experiments are created to study a given algorithm rather than compare it with others [26]. In this work, we are interested in the latter motivation. Once the goal of the experiment is defined, it will guide the choice of performance measures, as we describe next.

Performance measures

An algorithm experiment deals with a set of dependent variables called performance measures that are affected by a set of independent variables called factors; there are the problem factors, the algorithm factors, and the test environment factors. Since the goals of the experiment are achieved by analyzing observations of these factors and measures, they must be chosen with that aim in mind.

The stochastic nature of EAs introduces random variability in the answer provided by the algorithm: the solution obtained by the algorithm can vary from one run to another, and even when the same solution is reached, the computational time required for achieving such a solution is usually different for different runs of the same algorithm. In this case, there are two possible performance measures: solution quality and computational effort.

In some cases, when the convergence can be ensured, it would be possible to consider the computational effort required to reach the optimal solution as the only relevant performance indicator for the algorithm. The scope of this work is related to such cases.

On traditional computer performance evaluation, Hennessy and Patterson [27] consider the execution time of real programs as the only consistent and reliable measure of performance. Execution times have been continuously hampered by the variability of computer performance, specially for parallel programs which are affected by data races, thread scheduling, synchronization order, and contention of shared resources. In [28], it is shown that multicore processors bring even more variability to execution times.

Execution time can be defined in different ways depending on what we count. The most straightforward definition of time is called wall clock time, response time, or elapsed time, which is the latency to complete a task, including disk access, memory access, input/output activities, and operating system overhead.

In parallelism, the wall clock execution time is applied to a formula called speedup, described in the next section. The speedup is the most commonly used parallel performance measure. Other performance measures for parallel evolutionary algorithms, such as efficiency and incremental efficiency, are shown in [10,29].

Speedup

For parallel deterministic algorithms, speedup refers to how much a parallel algorithm is faster than the corresponding best known sequential algorithm. Speedup is defined by the ratio of T_1/T_p, where p is the number of processors, T_1 is the execution time of the sequential algorithm, and T_p is the execution time of the parallel algorithm with p processors.

For randomized algorithms such as EAs, the previous definition of speedup cannot be applied directly. As the execution times of EAs can vary from one run to another, the algorithm must be replicated and the average of execution times must be used. Thus, the speedup S_p for PEAs is the ratio between the average execution time on one processor $\overline{T}_1$ and the average execution time on p processors $\overline{T}_p$, as shown in the following equation:

$$S_p = \frac{\overline{T}_1}{\overline{T}_p} = \frac{\left(\sum_{i=1}^{k} T_{1i}\right)/k}{\left(\sum_{j=1}^{m} T_{pj}\right)/m} \tag{1}$$

where the metrics T_{1i} and T_{pj} correspond to wall clock times for the k sequential executions and the m parallel executions on p processors, respectively. This definition of speedup is the one adopted in this work, and it coincides to the weighted ratio definition in Equation (4).

In [10], the authors also recommend that the PEA should compute solutions having a similar accuracy as the sequential ones. This accuracy could be the optimal solution, if known, or an approximation solution, as though both algorithms produce the same value at the end. The stopping criterion of compared algorithms should be to find the same solution. The authors also advise to execute the parallel algorithm on one processor to obtain the sequential times. Thus, we have a sound speedup, both practical, i.e., no best known algorithm needed, and orthodox, i.e., same codes, same accuracy.

The speedup S_p is classified as *superlinear* when we have $S_p > p$, *sublinear* when we have $S_p < p$, and *linear* when S_p is approximately p.

Central limit theorem

Let $X_1, X_2, \ldots, X_n$ be n-independent random variables with finite mean μ and variance σ^2. The central limit theorem (CLT) states that the random variable

$$\mathrm{Sum}_n = \sum_{i=1}^{n} X_i \tag{2}$$

converges toward a normal distribution $N(n\mu, n\sigma^2)$, as n approaches ∞.

The CLT says that, even if a distribution of performance measurements is not normal, the distribution of the sample mean tends to a normal distribution as the sample size increases. For practical purposes, it is usually accepted that the resulting distribution is normally distributed when $n \geq 30$ [30]. Experimenters often mistake distribution of performance measurements and distribution of sample means.

An important application of CLT in this work arises in the estimation of speedup as the ratio of two means, $\overline{T}_p$ and $\overline{T}_1$. If the number of algorithm runs is large enough, the distribution of $\overline{T}_p$ and $\overline{T}_1$ will be nearly normally distributed. This makes the speedup a ratio of two normally distributed random variables, and it has important implications as we describe in the next section.

Ratio of two independent normal random variables

The distribution F_z of the ratio $Z = X/Y$ of two normal random variables X and Y is not necessarily normal. On the performance evaluation of parallel genetic algorithms (PGAs), if we adopt the speedup as the performance measure, we will need to ensure that the speedup density distribution approximates to a normal density distribution.

In this section, we describe two works that address the mean estimation of the ratio of two random variables which are normally distributed: Qiao et al. [7] and Díaz-Francés and Rubio [6].

Consider a sample of n observations (X, Y) from a bivariate normal population $N(\mu_X, \mu_Y, \sigma_X, \sigma_Y, \rho)$, $\mu_x, \mu_y \neq 0$ and X and Y are uncorrelated. In [7], the arithmetic ratio $\overline{R}_A$ is given by

$$\overline{R}_A = \frac{\sum X_i/Y_i}{n}, \tag{3}$$

and the weighted ratio $\overline{R}_W$ is given by

$$\overline{R}_W = \frac{\overline{X}}{\overline{Y}} = \frac{\sum X_i/n}{\sum Y_i/n} = \frac{\sum X_i}{\sum Y_i}. \tag{4}$$

Since $X \sim N(\mu_X, \sigma_X)$ and $Y \sim N(\mu_Y, \sigma_Y)$, it follows that $\overline{X} \sim N(\mu_X, \sigma_X/\sqrt{n})$ and $\overline{Y} \sim N(\mu_Y, \sigma_Y/\sqrt{n})$. The coefficient of variation of Y is $\delta_Y = \sigma_Y/\mu_Y$ and the coefficient of variation of $\overline{Y}$ is $\delta_{\overline{Y}} = \sigma_Y/\mu_Y\sqrt{n}$.

The simulations in [7] demonstrated that as long as $\delta_Y < 0.2$, both $\overline{R}_W$ and $\overline{R}_A$ are sound estimators of μ_X/μ_Y. Otherwise, if $\delta_{\overline{Y}} < 0.2$, $\overline{R}_W$ is an acceptable estimator of μ_X/μ_Y.

In practical situations, the population mean μ and standard deviation σ, if unknown, can be estimated by the sample mean and the sample standard deviation. In [7], an estimator of a sufficiently large sample size n_s given by

$$n_s > 25(s_Y^2/\overline{Y}^2), \tag{5}$$

where s_Y^2 is the sample variance and $\overline{Y}$ is the sample mean.

Another approach is presented by Díaz-Francés and Rubio in [6]. They demonstrate the existence of a normal approximation to the distribution of $Z = X/Y$, in an interval I centered at $\beta = E(X)/E(Y)$, which is given for the case where both X and Y are independent, have positive means and their coefficients of variation fulfill the conditions stated by the Theorem 1.

Theorem 1. (Díaz-Francés and Rubio [6]) *Let X be a normal random variable with positive mean μ_X, variance σ_X^2,* *and coefficient of variation* $\delta_X = \sigma_X/\mu_X$ *such that* $0 < \delta_X < \lambda \leq 1$*, where* λ *is a known constant. For every* $\varepsilon > 0$*, there exists* $\gamma(\varepsilon) \in \left(0, \sqrt{\lambda^2 - \delta_X^2}\right)$ *and also a normal random variable Y independent of X, with positive mean* μ_Y*, variance* σ_Y^2 *and coefficient of variation* $\delta_Y = \sigma_Y/\mu_Y$ *that satisfy the conditions,*

$$0 < \delta_Y \leq \gamma(\varepsilon) \leq \sqrt{\lambda^2 - \delta_X^2} < \lambda \leq 1, \tag{6}$$

for which the following result holds.

Any z that belongs to the interval

$$I = \left[\beta - \frac{\sigma_Z}{\lambda}, \beta + \frac{\sigma_Z}{\lambda}\right], \tag{7}$$

where $\beta = \mu_X/\mu_Y$*,* $\sigma_Z = \beta\sqrt{\delta_X^2 + \delta_Y^2}$*, satisfies that*

$$|G(z) - F_Z(z))| < \varepsilon, \tag{8}$$

where $G(z)$ *is the distribution function of a normal random variable with mean* β*, variance* σ_Z^2*, and* F_Z *is the distribution function of* $Z = X/Y$*.*

Theorem 1 states that for any normal random variable X with positive mean and coefficient of variation $\delta_X \leq 1$, there exists another independent normal variable Y with a positive mean and a small coefficient of variation, fulfilling some conditions, such that their ratio Z can be well approximated within a given interval to a normal distribution.

The circumstances established by [7] and [6] under which the ratio of two independent normal values can be used to safely estimate the ratio of the means are checked in our estimation of the speedup ratio.

Factors

In general, experiments often involve several factors which can have some influence on the output response. In [26], the factors that affect the performance of algorithms are categorized in the problem, algorithm, and test environment factors.

Problem factors comprise a variety of problem characteristics, such as dimensions and structure. Algorithm factors, specially for EAs, include multiple parameters related to its strategy to solve the problem, such as type of selection, mutation, and crossover, and in the case of PEAs, parameters related to parallel strategies, such as migration parameters.

It is necessary to select which factors to study, which to fix, and which to ignore and to hope that they will not influence the experimental results. The choice of experimentation factors and their values is central to the experimental design.

2^k Factorial design

A 2^k factorial design involves k factors, each at two levels. These levels can be quantitative or qualitative. The level of a quantitative factor can be associated with points on a numerical scale, as the size of population or the number of islands. For qualitative factors, their levels cannot be arranged in order of magnitude, such as topologies, or strategies of selection. The two levels are referred as 'low' and 'high', and denoted by '−' and '+', respectively. It does not matter which of the factor values is associated with the '+' and which with the '−' sign, as long as the labeling is consistent.

At the beginning of a 2^k factorial design, factors and levels are specified. When we combine them all, we get a design matrix. Table 1 shows the design matrix of the 2^3 factorial design.

For each combination of levels, also called treatment, the studied process is executed and the response variable y is collected.

After the data collection, the effects of factors can be calculated, and with appropriate statistical tests, we can determine whether the output depends in a significant way on the values of inputs. There are several excellent statistic software packages that are useful for setting up and analyzing 2^k designs. In our experiments, we use the R [31] open source software environment for statistical computing and graphics.

Basically, the average effect of a factor is defined as the change in response produced by a change in the level of that factor averaged over the levels of the other factors. The two-way interaction AB effect is defined as the average difference between the effect of factor A at the high level of factor B and the effect of A at the low level of B. The three-way interaction ABC occurs when there is any significant difference in two-way interaction plots corresponding to different levels of the third factor. For more details on this and other experimental designs, the reader is referred to [25,32].

Table 1 Notation of factors and level in a standard or Yates' order for the 2^3 factorial design

Run	A	B	C	Treatment	Coded factors			y
					A	B	C	
1	−	−	−	(1)	−1	−1	−1	y_1
2	+	−	−	*a*	+1	−1	−1	y_2
3	−	+	−	*b*	−1	+1	−1	y_3
4	+	+	−	*ab*	+1	+1	−1	y_4
5	−	−	+	*c*	−1	−1	+1	y_5
6	+	−	+	*ac*	+1	−1	+1	y_6
7	−	+	+	*bc*	−1	+1	+1	y_7
8	+	+	+	*abc*	+1	+1	+1	y_8

Regression model of the 2^k design

The results of the 2^k factorial design can be expressed in terms of a regression model. For the 2^2 factorial design, the effect full model is

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{12} x_1 x_2 + \epsilon \tag{9}$$

where y is the response, β are regression coefficients whose values are to be determined, x_i are predictor variables that represent coded factors levels: -1 and $+1$, and ϵ is the random error term. The method used to estimate the unknown coefficients β is the method of least squares [33].

Regression coefficients are related to effect estimates. The β_0 coefficient is estimated by the average of all responses. The estimates of β_1 and β_2 are one half the value of the corresponding main effect. In the same way, the interaction coefficients, as β_{12}, are one half the value of the corresponding interaction effect.

In regression, the R^2 coefficient of determination is a statistical measure of how well the regression line approximates the real data points. An R^2 of 1 indicates that the regression line perfectly fits the data. The *adjusted* R^2 is almost the same as R^2, but it penalizes the statistic as extra variables are included in the model.

Other measures have been developed to assess the model quality. The Akaike information criterion (AIC), the Chi-square test, the cross validation criterion, and others are methods used to compare models with different numbers of predictor variables [34]. These methods are useful in the model selection, one of the steps of the analysis of the 2^k factorial design (the 'Analysis of the 2^k design' subsection) where it is verified whether all the potential predictor variables are needed or a subset of them is adequate. The number of possible models grows with the number of predictors, which makes the selection a difficult task. Presently, there are a variety of automatic computer search procedures to simplify this task [33].

This model requires some assumptions to be satisfied: the errors are *normally* and *independently* distributed with *constant* variance σ^2. Violations of the basic assumptions and model adequacy can be easily investigated by the examination of residuals. Residuals are the difference between observed values and estimated values. If the model is adequate, the residuals should be structureless. Any suggestion of a pattern may indicate other problems such as a model misspecification due to either nonlinearity or the omission of important predictor variables, presence of outliers, and nonindependence of residuals.

The usual procedure for checking the normality assumption is to construct a normal probability plot of the residuals. If the underlying error distribution is normal, this plot will resemble a straight line. The constant variance of error assumption is easily verified if we plot the residuals versus the fitted values. This plot should not reveal any obvious pattern. To check for independence, in a plot of the residuals against time, if known, residuals should fluctuate in a more or less random pattern around the baseline zero.

Graphical analysis of residuals is inherently subjective, but frequently residual plots may reveal problems with the model more clearly than formal statistical tests. Formal tests like the Durbin-Watson test (independence), Breusch-Pagan test (constant variance), and Shapiro-Wilk test (normality) can check the model assumptions [33]. More about formal tests in the 'Implementation' subsection.

If a violation of the model assumptions occurs, there are two basic alternatives: (1) abandon the linear regression model and develop and use a more appropriate model, or (2) employ some transformation to the data. The first alternative may result in a more complex model than the second one. Sometimes, a nonlinear function can be expressed as a straight line using a suitable transformation. More on data transformation can be found in [25,33].

Analysis of variance

The analysis of variance (ANOVA) test is a statistical test which is used to compare means of two or more independent normal samples. It produces an F statistic that calculates the ratio of the variance among the means to the variance within the samples [33].

The ANOVA assumptions are the same as the regression model assumptions described in the 'Regression model of the 2^k design' subsection. The ANOVA is robust to the normality assumption. If the assumption of homogeneity of variances is violated, the ANOVA test is only slightly affected in the balanced (equal sample sizes in all treatments) fixed effect model. Lack of independence of error terms can have serious effects on the inferences in the analysis of variance [33].

Analysis of the 2^k design

The statistical analysis of the 2^k design follows the sequence of steps described:

1. *Estimate factor effects.* The factor effects are estimated, and their signs and dimensions are examined for a preliminary information regarding which factors and interactions may be important and in which directions these factors should be adjusted to improve the response.
2. *Perform statistical testing.* Many implementations of the 2^k factorial design rely on replications where each replicate represents a set of 2^k runs. When the design is replicated, for a full model as in Equation (9), the ANOVA can be applied to indicate whether one factor is more influential than another.

There are other methods to determine which effects are nonzero. The standard error of the effects can be calculated, and then confidence intervals on the effects are established. Box et al. [32] state a rough rule: effects greater than two or three times their standard error are not easily explained by chance alone.

For an unreplicated design, there is a method due to Cuthbert Daniel [35], in which effects are plotted on a normal probability plot. The effects that are negligible are normally distributed with mean zero and variance σ^2 and will tend to fall along a straight line on this plot. The significant effects will have nonzero means and will not lie along the straight line. An estimate of error can be obtained with the combined negligible effects. The formal tests of statistical significance are important to find out effects that are due to sampling error.

We should not confuse statistical significance with *practical significance* - whether an observed effect is large enough to matter. The statistical significance does not prove practical importance, but a practically significant effect should not be claimed unless it is statistically significant [14].

3. *Refine the model.* The model is adjusted as any nonsignificant factor can be removed from the model.
4. *Check the model adequacy.* The residual analysis is performed to check for model adequacy and assumptions. If it is found that the model is inadequate or if assumptions are badly violated, it is necessary to refine the model (step 3).
5. *Interpret results.* When examining the magnitude and sign of the factor effects, we can determine which factors are likely to be important. The main effect of a factor should be individually interpreted only if there is no evidence that the factor interacts with other factors. If the interaction is present, the interacting factors must be considered jointly. The negative sign on an effect indicates that a change from low level $-$ to high level $+$ will reduce the response variable.

Results and discussion

Implementation

The technical choices for the programming language and libraries are described in this section. Genetic algorithms (GA) are a widely used subfamily of EAs, which are stochastic search methods designed for exploring complex problem spaces in order to find optimal solutions using minimal information on the problem to guide the search.

We implemented a PGA with multiple populations following the island model (coarse-grained model). We named this implementation PGA-I. The population was divided into subpopulations, or islands, that evolve their local population in parallel. One of them was called central process, and it controlled the synchronization of all other subpopulations.

The parameters used in canonical GAs were kept fixed at the values shown in the Table 2 throughout the experimentation. The parameters related to the migration procedure were varied. They are shown in Table 3.

The efficiency of C/C^{++} compilers and the large number of available libraries led us to choose C/C^{++} for coding. The known sensitiveness of GAs to the choice of the PRNG motivated the option to a high-quality PRNG [36]. We chose a combination of the *SIMD-oriented Fast Mersenne Twister* (SFMT) and the *The Mother-of-All Random Generators* (MoA) available in the A. Frog library [37].

For parallelization, we chose a Message Passing Interface (MPI) library called MPICH2 [38]. The MPICH2 is a message passing interface library with tested and proved scalability on multicores [39] and proved performance of message passing on shared memory [40].

The PGA-I was executed on a multicore Intel Xeon E5504, with two CPUs, four cores each at 2 GHz, 256-KB cache L2, 4-MB cache L3, FSB at 1,333 MHz, 4 GB, Ubuntu 10.04 LTS (kernel 2.6.32).

The data analysis was performed using R, the free software environment for statistical computing and visualization [31]. The following formal statistical tests available in R were employed: the Shapiro-Wilk test of normality, `shapiro.test` from the `stats` package [31]; the Breusch-Pagan test of constant variance, `ncvTest` from the `car` package [41]; and the Durbin-Watson test of independence, `dwtest` from the `lmtest` package [42].

Case studies

The goal of our experimentation was to find out which of the factors that affects the speedup of the PGA-I the most when solving the selected test functions (see the 'Test functions' subsection). We selected seven factors

Table 2 Fixed canonical parameter settings

GA parameter	Value
Encoding	Real numbers
Selection	Tournament
Crossover rate	0.9
Mutation rate	0.001
Crossover operator	SBX
Stop criterion	f_{Ras}: optimal solution and
	f_{Ros}: near-optimal solution (10^{-10}),
	or 6×10^5 maximum generations for both
Problem size	30 dimensions

SBX, simulated binary crossover operator.

Table 3 Factors and levels

Id	Factors	Coded levels	
		High level (+)	Low level (−)
Proc	Number of islands	4	16
Top	Topology	Single ring	All-to-all
Rate	Migration frequency	10	100
Pop	Population size	1,600	3,200
Nindv	Amount of migrants	1	5
Sel	Selection strategy	Random	Best fitted
Rep	Replacement strategy	Random	Worst fitted

related to the migration parameters to investigate, as described in Table 3.

The 2^7 factorial design matrix was built as described in the '2^k Factorial design' subsection, with two additional lines related to sequential execution times, each with the population size of 1,600 and 3,200 individuals[a]. It adds up to 130 different configurations to be tested. A PRNG was used to determine the execution order and, for each treatment, the wall clock execution time was captured. As the speedup requires an estimate of sequential and parallel times, we replicated the experiment n times. We started with $n = 40$, and it was increased when necessary.

The conditions of the speedup ratio distribution to approximate to a normal distribution, as established in the 'Ratio of two independent normal random variables' subsection, were checked. If the conditions were not satisfied, the number of replications was increased and the conditions were verified again. Then, the analysis of the 2^7 factorial design, as described in the 'Analysis of the 2^k design' subsection, was performed. The statistical significance level of 0.05 and the practical significance of effects larger than 10% of the average speedup were considered in our analysis.

The following notation is used in this work: we denoted the two sets of sequential execution times by X_{pj}, where p is the population size and j is the replicate, and denoted the 128 sets of parallel execution times by Y_{ij}, where i is the treatment number and j identifies the replicate. Sample means and sample standard deviations were used on estimated coefficients of variation $\hat{\delta}$.

This procedure was performed for each test function, as described in the 'Rosenbrock function - 2^7 factorial design' and 'Rastrigin function - 2^7 factorial design' subsections.

Test functions

The selected test functions are well known and have been used in benchmarks, such as the CEC 2010 [43]. They are the Rastrigin function (f_{Ras}) and the Rosenbrock function (f_{Ros}). The former is one of the De Jong test functions and is relatively easy for GAs to solve. It is a nonlinear multimodal and separable function with a large number of local minima. The latter is a nonseparable function, and its global minimum is inside a long, narrow, parabolic-shaped flat valley. Even though it is trivial to find the valley, to converge to the global minimum is difficult.These functions were chosen by their different characteristics, and the experiments yielded different results for both.

The test function formulas are given:

$$f_{\text{Ras}}(X) = 10q + \sum_{i=1}^{q} (x_i^2 - 10\cos(2\pi x_i)), \tag{10}$$

where q is the dimension, $-5.12 \le x_i \le 5.12$, and its global minimum is $f_{\text{Ras}}(X^*) = 0$, and $X^* = [0, 0, \ldots, 0]$. We tested it with 30 dimensions ($q = 30$).

$$f_{\text{Ros}}(\mathbf{x}) = \sum_{i=1}^{q-1} \left[(1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2\right], \tag{11}$$

where q is the dimension, $-2.0 \le x_i \le 2.0$, and its global minimum is $f_{\text{Ros}}(X^*) = 0$, and $X^* = [1, 1, \ldots, 1]$. We tested it with 30 dimensions ($q = 30$) and searched for a near-optimal solution equal or less than 10^{-10}.

Rosenbrock function - 2^7 factorial design

The 2^7 factorial design plus two treatments for the sequential executions were replicated 40 times. It constituted a matrix with 40 columns and 130 lines filled with execution times. The execution times added up to approximately 120.4 h.

Table 4 presents statistics of the sequential execution times for each population sizes. Both estimated coefficients of variation $\hat{\delta}_{X_{1600}}$ and $\hat{\delta}_{X_{3200}}$ were lower than 0.2.

For the Y_{ij} parallel execution times, the maximum values of $\hat{\delta}_{Y_i}$ was 0.297, and the maximum value of $\hat{\delta}_{\overline{Y}_i}$ was 0.047. All $\hat{\delta}_{\overline{Y}_i}$ were under 0.2, as recommended in the 'Ratio of two independent normal random variables'

Table 4 Sequential execution times summary - Rosenbrock function

Pop	Minimum	First quartile	Median	Mean	Third quartile	Maximum
1,600	216,867	218,285	219,202	219,153	219,752	223,377
3,200	426,218	430,814	431,824	431,861	433,025	435,554

Table 5 Speedup summary - Rosenbrock function

Min	First quartile	Median	Mean	Third quartile	Maximum
2.36	3.41	3.70	3.91	4.30	7.69

Table 6 Factor levels for the lowest and highest observed speedup - Rosenbrock function

Id	Factors	Lowest speedup	Highest speedup
Proc	Number of islands	16	16
Top	Topology	Single ring	All-to-all
Rate	Migration frequency	100	10
Pop	Population size	1,600	3,200
Nindv	Amount of migrants	1	1
Sel	Selection strategy	Best fitted	Best fitted
Rep	Replacement strategy	Worst fitted	Worst fitted

subsection. The conditions established by Theorem 1 were satisfied. Thus, the distribution of the speedups, calculated by the weighted ratio formula (4), had an approximation to a normal distribution.

The summary shown in Table 5 gives a mean speedup of 3.91. There were 14 superlinear speedups observed for the PGA-I running on four islands.

The lowest and the highest speedups were observed when factors Proc, Top, Rate, Pop, Nindv, Sel, and Rep were set at $+ - + - - + +$ and $+ + - + - + +$ levels, respectively, as shown in Table 6.

At first, there was one speedup estimate for each treatment, and we had an unreplicated factorial. We started by assuming that all four-factor and higher interaction terms are unimportant[b]. The estimated effects of the third-order model of the 2^7 unreplicated factorial design were displayed in a normal probability plot. The significant effects are identified in Figure 1.

The factor Rep and all interaction involving Rep were negligible. We dropped the factor Rep from the experiment. The design became a 2^6 factorial with two replications. The projection of an unreplicated factorial into a replicated factorial in fewer factors is called design projection [25].

The analysis of the residuals of the third-order linear model for the 2^6 factorial design revealed that the spread of the residuals was increasing as the predicted speedup values get larger, an indication of nonhomogeneity variance, shown in Figure 2a. The Breusch-Pagan test [44] confirmed that the error variance changes with the level of the predicted speedup.

A log transformation of the speedup was performed, where $y^* = \log(y)$. For the log-transformed speedup model, the residual versus predicted value plot is shown in Figure 2b.

Following the log data transformation, an automatic search procedure for model selection based on the AIC criterion was performed. Table 7 presents the adjusted

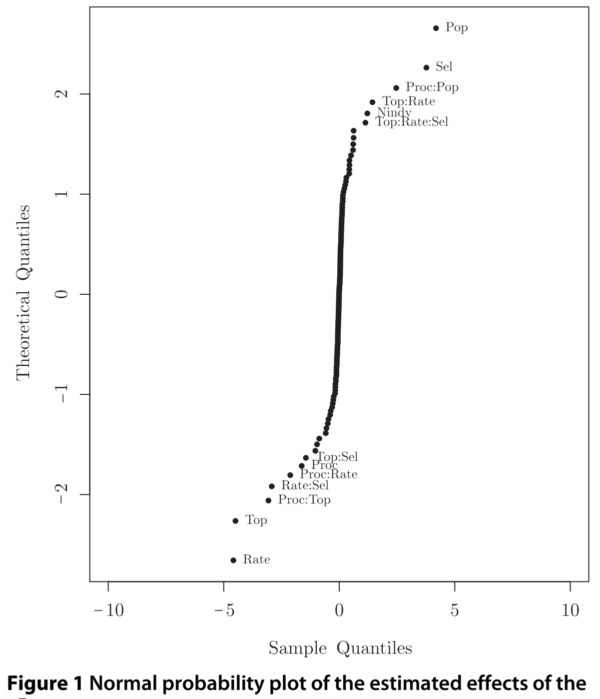

Figure 1 Normal probability plot of the estimated effects of the 2^7 factorial design - Rosenbrock function.

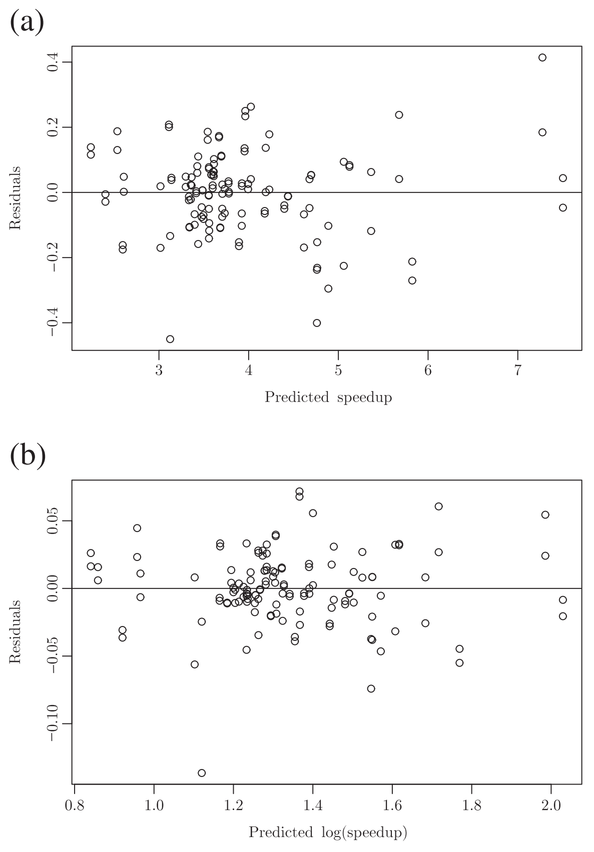

Figure 2 Residual versus predicted value plot for third-order model for the 2^6 factorial design - Rosenbrock function.
(a) Before data transformation and **(b)** after log data transformation.

Table 7 Adjusted linear regression model for the 2^6 factorial design following log data transformation - Rosenbrock function

	Estimate	Standard error	*t* value	Pr(>|*t*|)
(Intercept)	1.338	0.0035	385.842	<2e−16
Top	0.098	0.0035	28.141	<2e−16
Pop	0.096	0.0035	27.677	<2e−16
Rate	−0.095	0.0035	−27.481	<2e−16
Sel	0.075	0.0035	21.558	<2e−16
Nindv	0.029	0.0035	8.223	5.98e−13
Proc	0.018	0.0035	5.114	1.45e−06
Rate:Sel	−0.054	0.0035	−15.616	<2e−16
Top:Sel	0.021	0.0035	5.998	2.93e−08
Top:Proc	0.066	0.0035	19.125	<2e−16
Pop:Proc	0.055	0.0035	15.785	<2e−16
Rate:Proc	−0.041	0.0035	−11.726	<2e−16
Top:Rate	−0.017	0.0035	−5.039	1.98e−06
Top:Pop	−0.014	0.0035	−4.045	0.0001
Sel:Proc	0.014	0.0035	3.940	0.0001
Top:Nindv	−0.013	0.0035	−3.662	0.0004
Rate:Nindv	−0.012	0.0035	−3.511	0.0007
Sel:Nindv	0.010	0.0035	2.740	0.0072
Pop:Rate	0.008	0.0035	2.308	0.0230
Top:Rate:Sel	−0.015	0.0035	−4.305	3.80e−05
Rate:Sel:Proc	−0.012	0.0035	−3.522	0.0006
Top:Pop:Proc	−0.011	0.0035	−3.075	0.0027
Top:Sel:Nindv	−0.010	0.0035	−2.934	0.0041
Top:Rate:Proc	−0.008	0.0035	−2.183	0.0313

regression model for the 2^6 factorial design following the log data transformation. The adjusted model with all significant terms is given by

$$\begin{aligned}\widehat{y^*} &= 1.338 + 0.018x_1 + 0.098x_2 - 0.095x_3 + 0.096x_4\\ &+ 0.029x_5 + 0.075x_6 - 0.054x_3x_6 + 0.021x_2x_6\\ &+ 0.066x_2x_1 + 0.055x_4x_1 - 0.041x_3x_1\\ &- 0.017x_2x_3 - 0.014x_2x_4 + 0.014x_6x_1\\ &- 0.013x_2x_5 - 0.012x_3x_5 + 0.010x_6x_5\\ &+ 0.008x_4x_3 - 0.015x_2x_3x_6\\ &- 0.012x_3x_6x_1 - 0.011x_2x_4x_1\\ &- 0.010x_2x_6x_5 - 0.008x_2x_3x_1,\end{aligned} \tag{12}$$

where the coded variables x_1, x_2, x_3, x_4, x_5, and x_6 represent factors Proc, Top, Rate, Pop, Nindv, and Sel, respectively. This model presented a residual standard error equal to 0.0392 on 104 degrees of freedom, the coefficient standard error was 0.00347, and the R^2 was 0.975, which means the factors and interactions in the model explained approximately 98% of the variation in $\widehat{y^*}$. The adjusted R^2 was 0.969.

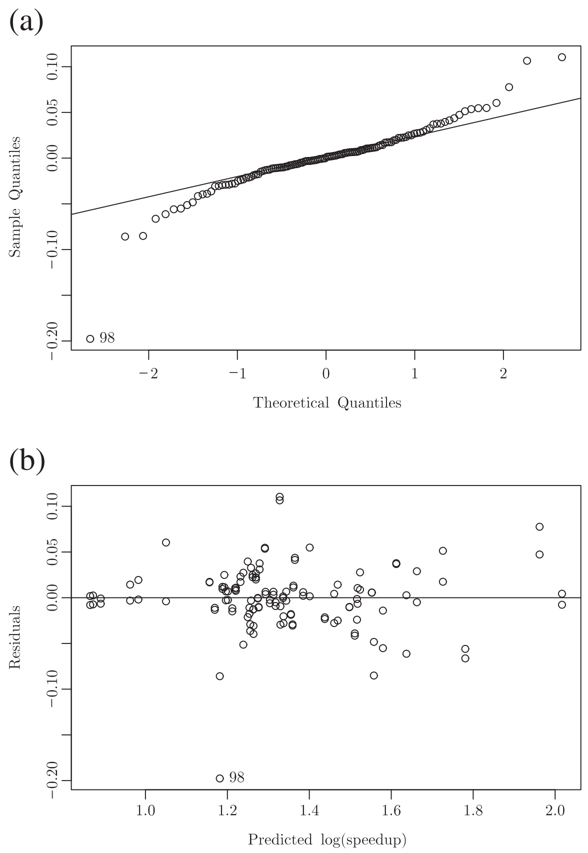

Figure 3 Residual analysis for the adjusted model (12) - Rosenbrock function. (a) Normal probability plot of residuals. **(b)** Plot of residuals versus predicted log (speedup).

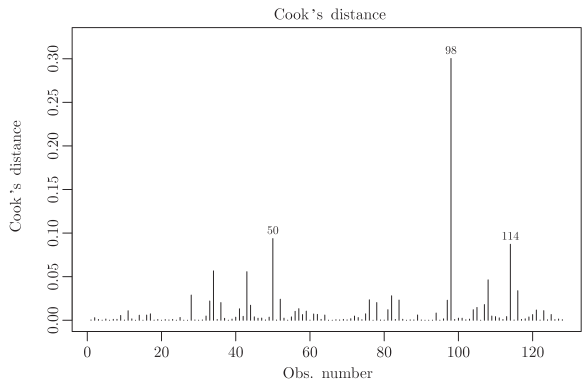

Figure 4 Cook's distance plot - Rosenbrock function.

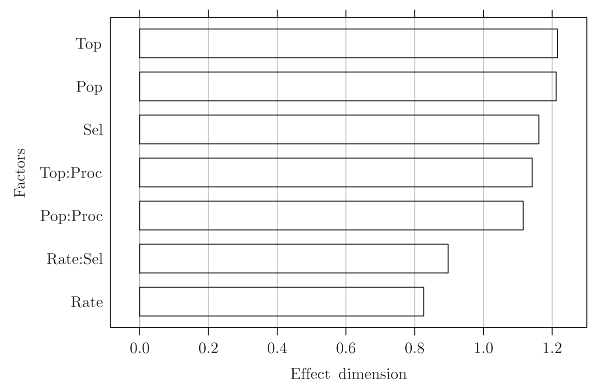

Figure 5 Bar plot of the back-transformed estimated effects for the adjusted model (12) - Rosenbrock function.

Figure 3 presents the normal probability plot of residuals and the plot of residuals versus predicted values. No unusual structure was apparent, and most of the residuals in the normal probability plot resembled a straight line, apart from residual 98.

Formal tests of equality of variances and normality and independence for the reduced model were executed. At a significance level of 0.05, the Breusch-Pagan test (p value $= 0.24$) could not reject the hypothesis that the residuals have a homogeneous variance, and the Durbin-Watson test (p value $= 0.26$) also could not reject the hypothesis that the residuals are independent. However, the Shapiro-Wilk test (p value $= 6.851e-08$) rejected the hypothesis that the errors are normally distributed.

Moderate departures from normality are of little concern in the fixed effect analysis of variance, though outlying values need to be investigated [25]. Formal tests to aid in evaluation of outlying cases have been developed, such as the Cook's distance, the studentized residuals and the Bonferroni test [33]. We run the R function `outlierTest` from the `car` package [41], and residual 98 had the Bonferroni adjusted p value < 0.05. Figure 4 shows the that residual 98 had the largest Cook's distance.

The adjusted model made without observation 98 showed that the inferences were not essentially changed. Observation 98 did not exercise undue influence so that no remedial measure was performed. We concluded that the model for $\widehat{y^*}$ given by Equation (12) was satisfactory, and we could estimate the speedup using back transformation given by $\widehat{y} = e^{\widehat{y^*}}$.

The log transformation has a multiplicative interpretation, e.g., adding 1 to log(y) multiplies y by e, where e is approximately 2.72. The bar plot of the statistically and practically significant back-transformed estimated effects for the adjusted model is displayed in Figure 5.

Discussion The effect of Nindv and its interactions were smaller than 10%, which is statistically but not practically significant. This could be due to the small difference between the high and low levels of Nindv. The high level was defined as five individuals based on the high level of number of islands, population size, and migration topology. When there were 16 islands and 1,600 total population size, each island had 100 individuals. If the topology was the all-to-all type, each island would receive a number of migrants equal to the number of islands versus the number of migrants. This calculation should be less than the island population.

The two-factor interactions, namely Proc:Top, Proc: Pop, and Rate:Sel, were statistically and practically significant. They are shown in Figure 6.

The effect of the Proc:Top interaction was 0.133 on log(speedup), i.e., 14.2% speedup increase. With Proc set at the low level with four islands, the speedup was increased by 6.5% when Top changed from the ring to all-to-all topology. With Proc set at the high level with 16

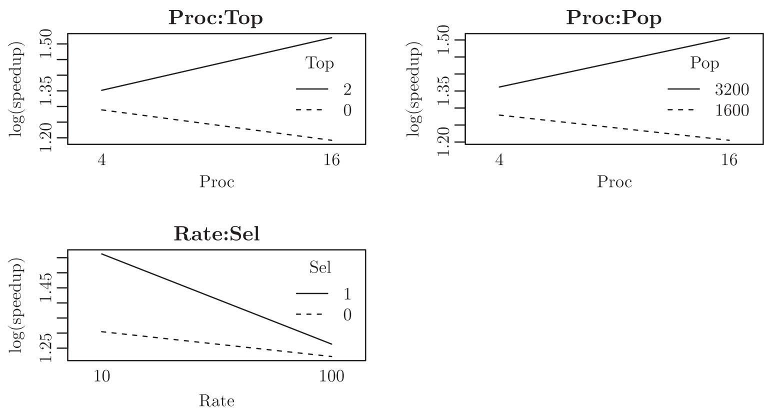

Figure 6 Interaction plot of the most significant two-factor interaction effects - Rosenbrock function. Proc:Top, Proc:Pop, and Rate:Sel, from left to right.

Table 8 Summary of sequential execution times for 1,000 replicates - Rastrigin function

Pop	Minimum	First quartile	Median	Mean	Third quartile	Maximum
1,600	2,544	2,672	2,699	6,770	2,729	551,307
3,200	5,112	5,309	5,351	5,358	5,397	6,516

islands, the change from ring to all-to-all topology had the effect of 38.8% increase on the speedup.

The factors Proc and Pop interacted with effect size of 11.6% speedup raise. With Proc set to four islands, speedup increased by 8.6% as Pop modified from 1,600 to 3,200 individuals. When Proc was 16 islands, speedup increased by 35.2% as Pop changed from 1,600 to 3,200 individuals.

The effect of Rate:Sel interaction was a 10.3% speedup decrease. At the low level of Rate, migration at every 10 generations, the speedup increased by 29.4% when Sel changed from random to best-fitted selection strategy. With Rate set to 100 generations, the speedup increased by 4.2% when factor Sel changed from low to high level.

Some of the three factor interactions were statistically significant, but their effects had a small influence on the speedup, which was less than 10%.

Rastrigin function - 2^7 factorial design

The 2^7 factorial design plus two treatments for the sequential executions were replicated 40 times. It constituted a matrix with 40 columns and 130 lines filled with execution times. The execution times added up to approximately 21.3 h. The conditions to the ratio distribution to approximate to a normal distribution were checked.

The estimated $\hat{\delta}_{X_{1600}}$ was >1. Thus, it did not satisfy Theorem 1. For the parallel execution times, the estimated $\hat{\delta}_{Y_i}$ ranged from 0.014 to 4.998 and the estimated $\hat{\delta}_{\overline{Y}_i}$ ranged from 0.020 to 0.790. Some $\hat{\delta}_{\overline{Y}_i}$ were higher than the recommended limit of 0.2 (see the 'Ratio of two independent normal random variables' subsection). There was no guarantee of the existence of a normal approximation to the distribution of the speedup.

A sample of a sufficiently large size reduces $\delta_{\overline{Y}_n}$. The estimator of the sample size is given by Equation (5). We applied this to the Y_{ij} sample and found $n_s > 624$. Although some of the Y_i have a small coefficient of variation, the full factorial was replicated 1,000 times and the data were kept balanced. The execution times added up to approximately 438 h.

The analysis of the 1,000 replicates of the 2^7 factorial design produced the X_p summary statistics presented in Table 8. X_{1600} showed extremely high values that were unusually far from other observations. Extreme values could also be seen among Y_i. The mean is not robust to extreme values. It is affected more by outliers than the median. According to [45], the mean exploits the sample because it gives equal weight to each observation. Contrarily, the median is resistant to several outlying observations since it ignores a lot of information.

The trimmed mean is a robust estimator of central tendency. To find a trimmed mean, the x% largest and smallest observations are deleted, and the mean is computed using the remaining observations. In fact, the median is just a trimmed mean with the percentage of trim equal to 50% [46].

We applied the trimmed mean with the percentage of trim equal to 10% to estimate $\overline{X}_p$ and $\overline{Y}_i$ in the speedup estimation. This procedure was equivalent to sort X_p and Y_i observations and discarded 10% of the smallest and 10% of the largest values from each one.

Table 9 presents the summary statistics for X_p after the trimming procedure. The estimated coefficients of variation $\hat{\delta}_{X_p}$ and $\hat{\delta}_{\overline{X}_p}$ were under 0.0107 and 0.0004, receptively. The estimated coefficients of variation $\hat{\delta}_{Y_i}$ and $\hat{\delta}_{\overline{Y}_i}$ were under 1.529 and 0.0541, respectively. These values did not satisfy the conditions stated in Theorem 1, but they satisfied the condition of $\hat{\delta}_{\overline{Y}_i}$ being under 0.2, described in the 'Ratio of two indepen- dent normal random variables' subsection. The weighted ratio given by Equation (4) was preferable to estimate the speedup. The mean speedup of all 128 observations was 2.32, as shown in Table 10.

Table 9 Summary of sequential execution times after the removal of extreme values - Rastrigin function

Pop	Minimum	First quartile	Median	Mean	Third quartile	Maximum
1,600	2,648	2,678	2,699	2,701	2,722	2,764
3,200	5,270	5,318	5,351	5,352	5,385	5,453

Table 10 Speedups summary - Rastrigin function

Minimum	First quartile	Median	Mean	Third quartile	Maximum
0.01	0.65	3.04	2.32	3.65	4.85

Table 11 Factor levels for the lowest and the highest observed speedup - Rastrigin function

Id	Factors	Lowest speedup	Highest speedup
Proc	Number of islands	16	16
Top	Topology	Single ring	All-to-all
Rate	Migration frequency	100	10
Pop	Population size	1,600	3,200
Nindv	Amount of migrants	1	5
Sel	Selection strategy	Random	Random
Rep	Replacement strategy	Random	Worst fitted

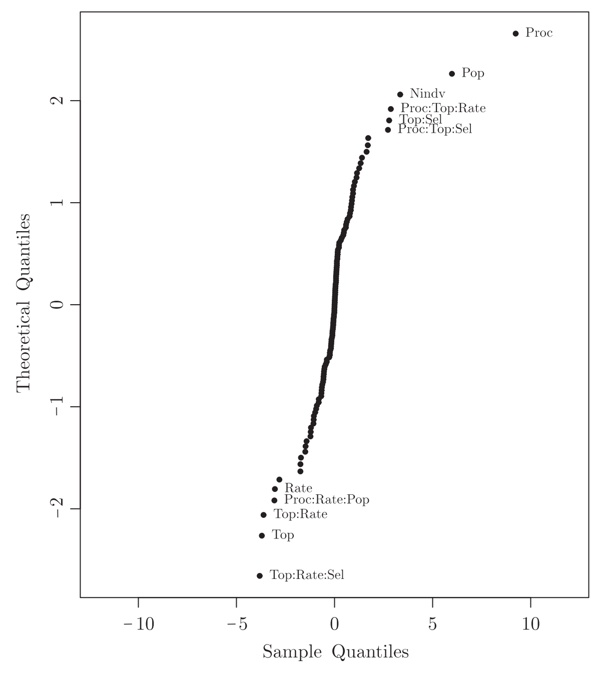

Figure 7 **Normal probability plot of the estimated effects of the 2^7 factorial design - Rastrigin function.**

There were 38 sublinear speedups, all of them occurred when running on 16 islands, and two superlinear speedups when running on four islands. The lowest and the highest speedups were observed when factors Proc, Top, Rate, Pop, Nindv, Sel, and Rep were set at $+ - + - - - -$ and $+ + - + + - +$ levels, respectively, as shown in Table 11.

Then, we proceeded to the analysis of the unreplicated 2^7 factorial design. Figure 7 presents the normal probability plot of the estimated effects.

The factor Rep and all interactions involving Rep were negligible. We dropped the factor Rep from the experiment. The design became a 2^6 factorial with two replicates.

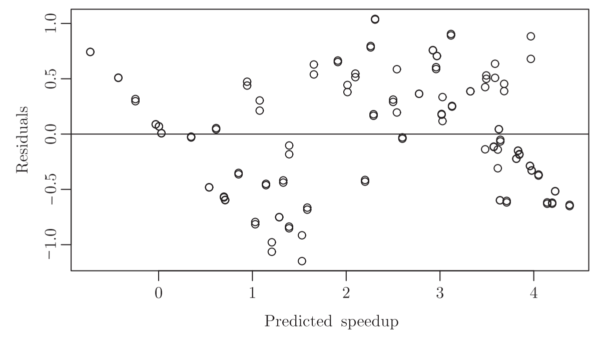

Figure 8 **Plot of residuals versus predicted speedups for the third-order model for 2^7 factorial design - Rastrigin function.**

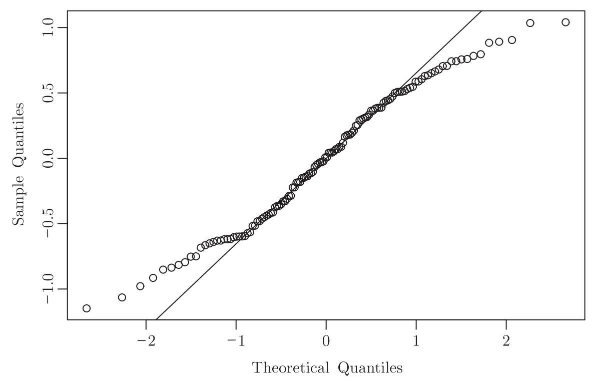

Figure 9 **Normal probability plot of residuals for the 2^6 factorial design - Rastrigin function.**

Figure 8 shows the plot of residuals of the third-order model 2^6 factorial design against the predicted values. The residuals looked structureless. The normal probability plot of the residuals resembled a straight line, as shown in Figure 9.

From the third-order linear regression model for the 2^6 factorial design, an automatic search procedure for model selection based on the AIC criterion resulted in the model described in Table 12. The adjusted model was built as follows:

$$\widehat{y} = 2.318 - 0.816x_1 + 0.327x_2 - 0.269x_3 + 0.529x_4 + 0.296x_5 + 0.320x_2x_3, \quad (13)$$

where the coded variables x_1, x_2, x_3, x_4, and x_5 represent factors Proc, Top, Rate, Pop, and Nindv, respectively. This model presented a residual standard error of 0.978 on 121 degrees of freedom, and R^2 was 0.593, which means that the model explained 59% of the variation of $\widehat{y}$. The adjusted R^2 was 0.573.

Formal tests of equality of variances, normality, and independence on the reduced model were executed. The Shapiro-Wilk test (p value $= 0.13$), the Breusch-Pagan test (p value $= 0.06$), and the Durbin-Watson test

Table 12 Third-order linear regression model for the 2^6 factorial design - Rastrigin function

	Estimate	Standard error	*t* value	Pr(>|*t*|)
(Intercept)	2.318	0.0864	26.821	< 2e−16
Proc	−0.816	0.0864	−9.445	3.38e−16
Pop	0.529	0.0864	6.121	1.19e−08
Top	0.327	0.0864	3.789	0.0002
Nindv	0.296	0.0864	3.423	0.0008
Rate	−0.269	0.0864	−3.110	0.0023
Top:Rate	0.320	0.0864	3.700	0.0003

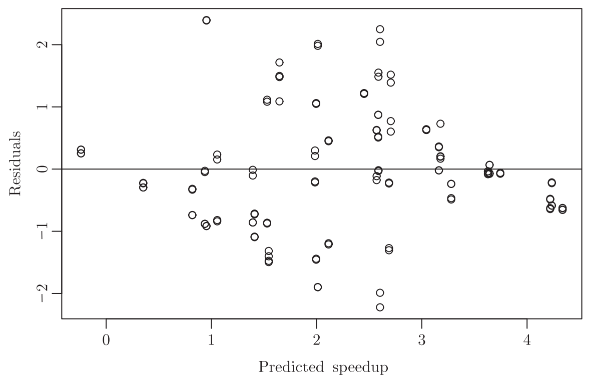

Figure 10 **Plot of residuals versus predicted speedups for the adjusted model (13) - Rastrigin function.**

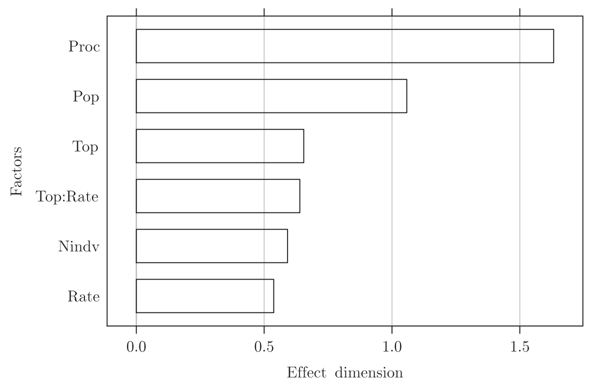

Figure 12 **Bar plot of the absolute estimated effects for the adjusted model (13) - Rastrigin function.**

(p value $= 0.20$) did not reject the null hypotheses of nonnormality, nonhomogeneity and nonindependence of residuals, respectively.

Figure 10 presents a plot of the residuals versus predicted speedup, and Figure 11 presents the normal probability plot of the residuals for this adjusted model. Though the formal tests did not indicate any problem, Figure 10 shows some nonlinear pattern. The small coefficient of determination R^2 also corroborated that the model might be improved if higher order terms were added to the model or other predictor variables were considered. The bar plot of the absolute value of the estimated effects is presented in Figure 12. All effects were practically significant because they were higher than 0.23, i.e., 10% of the mean speedup.

Discussion The factor Proc had the largest main effect of -1.63. Changing Proc from 4 to 16 islands dropped the mean speedup by 1.63. Factor Pop had the second largest main effect of 1.06. If the population size increased from 1,600 to 3,200, the speedup increased by 1.06. The main effect of Nindv was 0.6. The speedup increased when the number of migrants changed from one to five.

The effect of interaction Top:Rate was 0.64, and it is shown in Figure 13. With factor Top set to ring topology, when Rate changed from 10 to 100 generations, the mean speedup dropped by 1.18. With factor Top set to all-to-all topology, the mean speedup increased by 0.10 as the Rate changed from 10 to 100 generations.

Rastrigin is a multimodal function, and it presented execution times with higher coefficient of variation when compared to the Rosenbrock function. This variability affects the sharpness of the statistics, and a larger number of replicates was necessary. This was also reflected on the higher error estimates.

The execution times of PGA-I on solving the Rastrigin function presented some extreme execution times. We investigated these outlying values, and we could reproduce them using the same PRNG seed. They were not

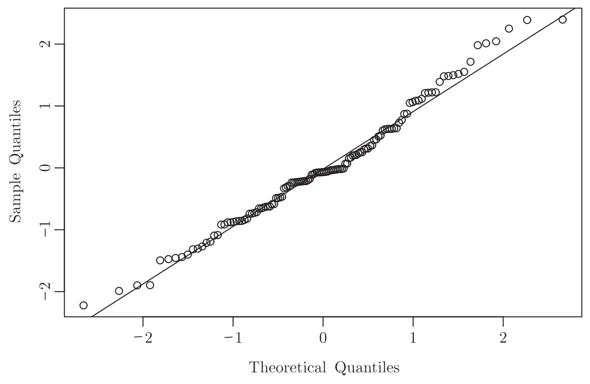

Figure 11 **Normal probability plot of residuals for the adjusted model (13) - Rastrigin function.**

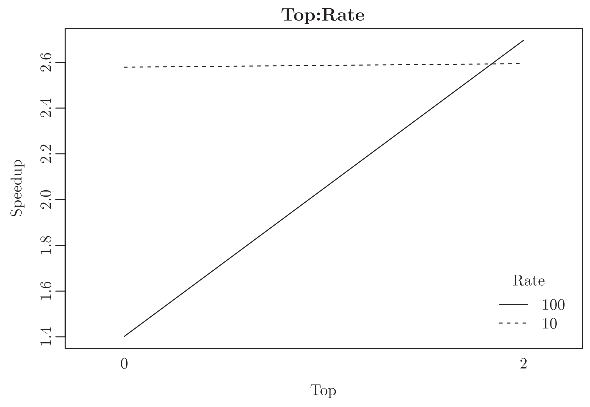

Figure 13 **Interaction plot of the two-factor interaction effect Top:Rate - Rastrigin function.**

due to erroneous measures, failures of registry, or external perturbations. By trimming them, we were effectively discarding information about the best and worst performances of the algorithm. The distribution of the execution times were also changed. This is a limitation of our method.

Conclusions

This paper introduced a method to guarantee the correct estimation of speedups and the application of a factorial design on the analysis of PEA performance. As a case study, we evaluated the influence of migration related to parameters on the performance of a parallel evolutionary algorithm solving two benchmark problems executed on a multicore processor. We made a particular effort in carefully applying the statistical concepts in the development of our analysis.

The performance and the factor effects were not the same for the two benchmark functions studied in this work. The Rastrigin function presented higher coefficient of variation than the Rosenbrock function, and the factor and interaction effects on the speedup of the PGA-I were different in both.

Further work will be done on the high variability of the results yielded by the algorithms tested. We observed extreme values in the execution times of PGA-I when solving the Rastrigin function, in the ‘Rastrigin function - 2^7 factorial design’ subsection. These extreme measures were investigated, and they could be reproduced by applying the same seed.

We intend to study alternative approaches to deal with extreme values. One approach is to perform a rank transformation [47]. Another option is to block the PRNG seed to isolate its influence on the variability of execution times. Blocking the seed is a controversial subject: it is a statistically sound approach, but it is not a common practice to specify the seed that is used in the execution of EAs. EAs are usually executed several times with different seeds to get a sense of the robustness of the procedure. The seed effects may be of little relevance to the EA community. Future works demand a proper investigation of this issue.

Endnotes

[a]The population size is one of the investigated factors with two levels: 1,600 and 3,200 individuals. It is important to ensure the same workload conditions for both sequential and parallel times. Thus, it is necessary to capture the sequential times for these two population sizes and to have the same population size for the sequential and the parallel times in the speedup ratio.

[b]The sparsity of the effect principle states that a system is usually dominated by main effects and low-order interactions, and most high-order interactions are negligible [25].

Competing interests

The authors declare that they have no competing interests.

Authors’ contributions

MSP carried out the background and literature review, implemented the parallel genetic algorithm, conceived the design of experiments, performed the statistical analyses and drafted the manuscript. ISP participated in the implementation of the parallel genetic algorithms and in the design of experiments, and helped to draft the manuscript. KY was involved in conceiving of the study and participated in its design. ERP participated in the design of the experiments and the statistical analyses, and helped to draft the manuscript. All authors read and approved the final manuscript.

Acknowledgements

We would like to thank the anonymous referees whose really careful reading, useful comments, and corrections have led to significant improvement of our work. Mônica S Pais and Igor S Peretta were partially supported by IFGoiano/FAPEG/CAPES and CNPq, respectively.

Author details

[1]Department of Informatics, Goiano Federal Institute of Education, Science and Technology (IFGoiano), Campus Urutaí, Rodv. Geraldo Silva km 2,5, Urutaí, Goiás 75790-000, Brazil. [2]Faculty of Electrical Engineering, FEELT, Federal University of Uberlândia (UFU), Uberlândia, Minas Gerais 38400-902, Brazil. [3]Faculty of Mathematics, FAMAT, Federal University of Uberlândia (UFU), Uberlândia, Minas Gerais 38408-100, Brazil.

Received: 29 November 2013 Accepted: 2 December 2013
Published: 3 February 2014

References

1. Chiarandini M, Paquete L, Preuss M, Ridge E (2007) Experiments on metaheuristics: methodological overview and open issues. Technical report DMF-2007-03-003 The Danish Mathematical Society
2. Cantú-Paz E (2000) Efficient and accurate parallel genetic algorithms. Kluwer Academic Publishers, Norwell
3. Tomassini M (2005) Spatially structured evolutionary algorithms: artificial evolution in space and time (natural computing series). 1st edn. Springer, Secaucus
4. Alba E (2002) Parallel evolutionary algorithms can achieve super-linear performance. Inf Process Lett 82(1): 7–13
5. Kim H, Bond R (2009) Multicore software technologies. IEEE Signal Process Mag 26(6): 80–89
6. Diaz-Frances E, Rubio F (2013) On the existence of a normal approximation to the distribution of the ratio of two independent normal random variables. Stat Pap 54(2): 309–323
7. Qiao CG, Wood GR, Lai CD, Luo DW (2006) Comparison of two common estimators of the ratio of the means of independent normal variables in agricultural research. J Appl Math Decis Sci 2006. doi:10.1155/JAMDS/2006/78375
8. Eiben AE, Smith SK (2011) Parameter tuning for configuring and analyzing evolutionary algorithms. Swarm and Evol Comput 1: 19–31
9. Touati S, Worms J, Briais S (2012) The speedup-test: a statistical methodology for programme speedup analysis and computation. Concurrency and Comput: Practice and Experience 25(10): 1410–1426
10. Alba E, Luque G, Nesmachnow S (2013) Parallel metaheuristics: recent advances and new trends. Int Trans Oper Res 20(1): 1–48. doi:10.1111/j.1475-3995.2012.00862.x.
11. Coy SP, Golden BL, Runger GC, Wasil EA (2001) Using experimental design to find effective parameter settings for heuristics. J Heuristics 7: 77–97. 10.1023/A:1026569813391
12. Czarn A, MacNish C, Vijayan K, Turlach BA, Gupta R (2004) Statistical exploratory analysis of genetic algorithms. Evol Comput, IEEE Trans 8(4): 405–421. doi:10.1109/TEVC.2004.831262
13. Bartz-Beielstein T (2006) Experimental research in evolutionary computation: the new experimentalism (natural computing series). 1st edn. Springer, Secaucus
14. Rardin RL, Uzsoy R (2001) Experimental evaluation of heuristic optimization algorithms: a tutorial. J Heuristics 7: 261–304. doi:10.1023/A:1011319115230

15. Shahsavar M, Najafi AA, Niaki STA (2011) Statistical design of genetic algorithms for combinatorial optimization problems. Math Probl Eng: 1–7. doi:10.1155/2011/872415
16. Pinho AFD, Montevechi JAB, Marins FAS (2007) Análise da aplicação de projeto de experimentos nos parâmetros dos algoritmos genéticos. Sistemas & Gestão 2: 319–331
17. Petrovski A, Brownlee AEI, McCall JAW (2005) Statistical optimisation and tuning of GA factors. The 2005 IEEE Congress on Evolutionary Computation (CEC 2005) 1: 758–764
18. Mühlenbein H (1991) Darwin's continent cycle theory and its simulation by the prisoner's dilemma. Complex Syst 5: 459–478
19. Hooker J (1994) Needed: an empirical science of algorithms. Oper Res 42: 201–212
20. Hooker J (1995) Testing heuristics: we have it all wrong. J Heuristics 1: 33–42. doi:10.1007/BF02430364
21. McGeoch CC (1996) Feature article—toward an experimental method for algorithm simulation. INFORMS J Comput 8(1): 1–15
22. Johnson D (2002) A theoretician's guide to the experimental analysis of algorithms. Data Structures, Near Neighbor Searches, and Methodology: Fifth and Sixth DIMACS Implementation Challenges 5: 215–250
23. Eiben AE, Jelasity M (2002) A critical note on experimental research methodology in experimental research methodology in EC. Proceedings of the 2002 Congress on Evolutionary Computation (CEC'2002) 1: 582–587
24. Steinberg DM, Hunter WG (1984) Experimental design: review and comment. Technometrics 26(2): 71–97
25. Montgomery DC (2009) Design and analysis of experiments. 7th edn. John Wiley and Sons, Hoboken
26. Barr RS, Golden BL, Kelly JP, Resende MGC, Stewart WR (1995) Designing and reporting on computational experiments with heuristic methods. J Heuristics 1(1): 9–32
27. Hennessy JL, Patterson DA (2006) Computer architecture: a quantitative approach. 4th edn. Morgan Kaufmann Publishers, San Francisco
28. Mazouz A, Touati SAA, Barthou D (2011) Analysing the variability of openMP programs performances on multicore architectures. In: Fourth workshop on programmability issues for heterogeneous multicores (MULTIPROG-2011), Heraklion, 23 Jan 2011
29. Barr RS, Hickman BL (1993) Reporting computational experiments with parallel algorithms: Issues, measures, and experts' opinions. ORSA J Comput Winter 5: 2–18
30. Montgomery DC, Runger GC (2003) Applied statistics and probability for engineers. 3rd edn. John Wiley & Sons, Danvers
31. R Development Core Team (2012) R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna. http://www.R-project.org/. ISBN 3-900051-07-0
32. Box GEP, Hunter JS, Hunter WG (2005) Statistics for experimenters: design, discovery, and innovation. 2nd edn. John Wiley & Sons, Hoboken
33. Kutner MH, Neter J, Nachtsheim CJ, Li W (2005) Applied linear statistical models. 5th edn. The McGraw-Hill/Irwin series operations and decision sciences, McGraw-Hill/Irwin, New York
34. Busemeyer JR, Wang YM (2000) Model comparisons and model selections based on generalization criterion methodology. J Math Psychol 44(1): 171–189
35. Daniel C (1959) Use of half-normal plots in interpreting factorial two-level experiments. Technometrics 1(4): 311–341
36. Cantú-Paz E (2002) On random numbers and the performance of genetic algorithms In: Proceedings of the genetic and evolutionary computation conference (GECCO 2002). Morgan Kaufmann Publishers, San Francisco, pp 311–318
37. Fog A (2010) Random number generator libraries. 2008-2010, Instructions for the random number generator libraries on www.agner.org. GNU General Public License. Version 2.01. 2010-08-03. Accessed 15 April 2011.
38. MPICH2 (2001) MPICH-2. Argonne National Laboratory, Lemont. http://www.mcs.anl.gov/mpi/mpich2. Accessed 20 May 2011.
39. Buntinas D, Mercier G, Gropp W (2007) Implementation and evaluation of shared-memory communication and synchronization operations in mpich2 using the nemesis communication subsystem. Parallel Comput 33(9): 634–644. doi:10.1016/j.parco.2007.06.003. Selected papers from EuroPVM/MPI 2006
40. Balaji P, Buntinas D, Goodell D, Gropp W, Hoefler T, Kumar S, Lusk EL, Thakur R, Träff JL (2011) MPI on millions of cores. Parallel Process Lett (PPL) 21(1): 45–60
41. Fox J, Weisberg S (2011) An R companion to applied regression. second edn. Sage, Thousand Oaks. http://socserv.socsci.mcmaster.ca/jfox/Books/Companion
42. Zeileis A, Hothorn T (2002) Diagnostic checking in regression relationships. R News 2(3): 7–10. http://CRAN.R-project.org/doc/Rnews/
43. Táng K, Lǐ X, Suganthan PN, Yáng Z, Weise T (2010) Benchmark functions for the CEC'2010 special session and competition on large-scale global optimization. Technical report, University of Science and Technology of China (USTC)
44. Breusch TS, Pagan AR (1979) A simple test for heteroscedasticity and random coefficient variation. Econometrica: J Econometric Soc: 1287–1294
45. Jain R (1991) The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling. 1st edn. John Wiley & Sons Inc, New York
46. Tukey JW (1977) Exploratory data analysis. Addison–Wesley Publishing Company, Boston
47. Conover WJ, Iman RL (1981) Rank transformations as a bridge between parametric and nonparametric statistics. The American Statistician 35(3): 124–129. doi:10.1080/00031305.1981.10479327. http://amstat.tandfonline.com/doi/abs/10.1080/00031305.1981.10479327

doi:10.1186/1678-4804-20-6
Cite this article as: Pais *et al.*: **Factorial design analysis applied to the performance of parallel evolutionary algorithms.** *Journal of the Brazilian Computer Society* 2014 **20**:6.