An Enhanced Statistical Approach for Evolutionary Algorithm Comparison

Eduardo G. Carrano
Research Group for Intelligent Systems - GPSI
Centro Federal de Educação Tecnológica de Minas Gerais
Av. Amazonas, 5253 - Nova Suiça - 30480-000
Belo Horizonte - MG, Brazil
egcarrano@deii.cefetmg.br

Ricardo H. C. Takahashi
Dep. Mathematics
Universidade Federal de Minas Gerais
Av. Antônio Carlos, 6627 - Pampulha - 31270-010
Belo Horizonte - MG, Brazil
taka@mat.ufmg.br

Elizabeth F. Wanner
Dep. Mathematics
Universidade Federal de Ouro Preto
Campus Universitário, Morro do Cruzeiro - 35400-000
Ouro Preto - MG, Brazil
efwanner@iceb.ufop.br

ABSTRACT

This paper presents an enhanced approach for comparing evolutionary algorithm. This approach is based on three statistical techniques: (a) Principal Component Analysis, which is used to make the data uncorrelated; (b) Bootstrapping, which is employed to build the probability distribution function of the merit functions; and (c) Stochastic Dominance Analysis, that is employed to make possible the comparison between two or more probability distribution functions. Since the approach proposed here is not based on parametric properties, it can be applied to compare any kind of quantity, regardless the probability distribution function. The results achieved by the proposed approach have provided more supported decisions than former approaches, when applied to the same problems.

Categories and Subject Descriptors

G.1.6 [**Mathematic of Computing**]: Numerical Analysis—*Optimization*

General Terms

Algorithms

Keywords

evolutionary algorithms, algorithm comparison, evolutionary encoding schemes, tree network design

1. INTRODUCTION

When deterministic optimization algorithms are compared, their performances are characterized by their computational complexity only. Since those algorithms always perform the same sequence of deterministic steps, it is guaranteed under some assumptions that, starting from a given initial point, an algorithm converges (i.e., reaches a stop criterion) in a fixed number of algorithm iterations (the solution that is reached is not necessarily the global optimum of the problem) [5].

The performance analysis of evolutionary algorithms cannot follow the same methodology. The stochastic nature of those algorithms introduces another issue that must be considered: it is not guaranteed, in any single run, the achievement of the same solution that has been found in another run and, even when the same solution is reached, the computational effort spent to obtain this solution varies in different runs of the same algorithm [5].

The flexible structure of the evolutionary algorithms makes possible to build them in several different ways, what leads to different algorithms which usually present different performance. This combinatorial scenario of possible algorithms motivates the efforts for developing evaluation/comparison methods for evolutionary algorithms, such as the ones presented in [5, 4, 18, 19, 1].

Some of the approaches proposed in literature state the assumption that, in some cases, the convergence is almost ensured, and therefore could be assumed as true. Therefore, it is possible to compare the algorithms based on a single criterion: the computational effort required to reach the optimum. In this approach, the computational cost of the algorithm is modeled as a random variable, and some statistical analysis are employed to infer on this variable [5]. Although this approach can be useful in some particular situations, the assumption of convergence is too strong. Notice that in large $\mathcal{NP}$-hard problems, for instance, the evolutionary algorithms are expected to find "good" sub-optimal solutions only, and the assumption of finding the exact optimum is infeasible, in practice. There are similar approaches which fix the computational cost, establishing a pre-determined number of algorithm generations. This approach also can carry problems, since the computational cost becomes an arbitrary parameter, that can hardly affect the algorithm analysis.

It is reasonable to consider the trade-off between faster algorithms that lead to rough solutions, and slower algorithms that deliver more accurate solutions. A feasible way for comparing algorithms in which the convergence is not ensured is the employment of multicriteria comparison schemes, such as discussed in [19] and presented in [18]. In this kind of approach, the convergence ability and the computational cost of the algorithms are modeled as random variables, and a statistical estimator, such as the mean, is employed to allow comparisons. The reference [18] for example, uses the number of function evaluations to estimate the computational cost of the algorithm and the convergence rate to estimate its convergence capacity. These criteria are evaluated through their mean,

GECCO'08, July 12–16, 2008, Atlanta, Georgia, USA.

and a multiobjective analysis [9] is employed to find which algorithms can be considered efficient. Since a multiobjective analysis is adopted, the result of this approach is a set of efficient algorithms, instead of a single "best one".

Although the approach described above is more well suited than a single-criterion one, it still presents an weakness: the comparisons are made based only in mean values, what often implies the loss of the information about how the points are spread around that mean. This paper proposes a multiobjective comparison approach of evolutionary algorithms that is based on the concept of stochastic dominance [16], instead of using comparisons of the mean values only. It makes possible to consider the deviation of the criteria around mean, what consequently provides more supported decisions. The approximated probability distribution functions (PDF) are built using a Bootstrapping procedure [6]. Since this approach is non-parametric, it can be employed for comparing any data set, regardless the distribution. The results achieved have shown that the approach has a high capacity of detecting significant statistical differences between algorithms.

The paper is structured as follows:

- Section 2 introduces the statistical concepts which are used in the comparison approach;
- Section 3 presents the methodology of the comparison proposed in this work;
- Section 4 presents the results achieved by the comparison approach in three instances of a discrete problem. The results obtained by the approach are compared with previous results for the same problem.

2. CONCEPTUAL BACKGROUND

The concepts which are employed in the comparison approach proposed in this paper are briefly introduced in this section.

2.1 Principal Component Analysis

[1]Principal Component Analysis (PCA) is a mathematical procedure that is employed to transform a set of correlated variables into a new set (sometimes smaller) of uncorrelated variables. The new variables are arranged in such a way that they are sorted by variability, with the first component accounting for as much of the variability in the data as possible, and each succeeding component accounts for as much of the remaining variability as possible. The variables which represent a significant part of variability are called principal components. Sometimes, this procedure is called Proper Orthogonal Decomposition (POD) or Hotelling Transform.

The PCA is usually employed for some specific tasks:

- Reduce the dimensionality of a data set;
- Identify new meaningful underlying variables;
- Find the uncorrelated data, to analyze the variables independently.

Let X be a set of M variables with N observations each. The PCA of X can be performed in 13 steps:

1. Arrange the data in N data vectors, $X_1, \ldots, X_N$. Each vector X_i is a column vector ($M \times 1$) with a single observation of the M variables.
2. Build a matrix X ($M \times N$) with the column vectors.
3. Find the empirical mean of each variable $m = 1, \ldots, M$:
   $u[m,1] = \frac{1}{N} \sum_{n=1}^{N} X[m,n]$.
4. Subtract the empirical mean u from each column of matrix X:
   $B = X - u \cdot h$
   where $h[1,m] = 1 \ \forall \ m = [1, \ldots, M]$.
5. Find the empirical covariance matrix C ($M \times M$):
   $C = \frac{1}{N} B \cdot B^*$
   where B^* is the conjugate transpose of matrix B.
6. Find the eigenvalues of matrix C and build a column vector E ($M \times 1$).
7. Find the matrix of eigenvectors V in which:
   $V^{-1} \cdot C \cdot V = D$
   where D is a diagonal matrix, such that:
   $D[m,m] = E(m,1) \ \forall \ m = [1, \ldots, M]$.
8. Sort the columns of V and D in descendant order of eigenvalues.
9. Find the cumulative energy content for each eigenvector:
   $g[m] = \sum_{i=1}^{M} D[i,i] \ \forall \ m = [1, \ldots, M]$
   - Eigenvectors with higher cumulative energy represent most part of variance than eigenvectors with lower cumulative energy.
10. Select a subset of the eigenvectors as basis vectors:
    $W[p,q] = V[p.q] \ \forall \ p = [1, \ldots, M] \ \ q = [1, \ldots, L]$
    where $1 \leq L \leq M$.
    - The vector g can be used as a guide for choosing L. For instance, L can be chosen such that: $g[m = L] \geq 0.9$.
11. Find the empirical standard deviation vector s ($M \times 1$):
    $s[m,1] = \sqrt{C(m,m)} \ \forall \ m = [1, \ldots, M]$.
12. Calculate the z-score matrix Z ($M \times N$):
    $Z = \frac{B}{s \cdot h}$ (divide element by element).
13. Project the z-scores of the data onto the new basis:
    $Y = W^* \cdot Z$.

The result of this procedure is a set of data Y, which is uncorrelated, and is fitted to a lower dimension ($L \leq M$). This procedure is useful in the analysis of high dimension data.

In the specific case of this work, the PCA is used only to make the data uncorrelated, regardless the dimension reduction that could be exploited. Therefore, only the steps 1 to 8 are required. The uncorrelated data, is obtained through (1).

$$Y = V \cdot X \tag{1}$$

In evolutionary algorithm comparison, at least two merit criteria should be considered: a criterion which estimates the convergence capacity of the algorithm and a criterion which estimates its computational cost [18]. Therefore, each algorithm run results in a point in a two-dimensional space (or in higher dimension, if more criteria are considered), which is often correlated. The decoupling of data can be useful to make meaningful the independent analysis of each merit criterion considered.

2.2 Bootstrapping

[2]Bootstrapping is a statistical procedure employed to infer the properties of an estimator (such as mean or variance) by sampling from an approximating distribution. The empirical distribution obtained in some observations is generally used as approximating distribution.

[1]The description of Principal Component Analysis presented in this section has been adapted from the references [10, 11, 13].

[2]The description of Bootstrapping presented in this section has been adapted from the references [2, 6, 7, 8].

The bootstrapping procedure can be useful in the following situations:

- build hypothesis tests;
- make possible inferences based on parametric assumptions;
- make possible inferences in non-parametric data sets, using distribution comparison methods, such as non-parametric tests or stochastic dominance.

Let $P = \{x_1, x_2, x_3, \ldots, x_N\}$ be a population and $S = \{X_1, X_2, X_3, \ldots, x_n\}$ be an independent sample extracted from P (where n is much smaller than N). The properties of an estimator T for the population, $\Theta = T(P)$ can be inferred from the sample, $\theta = T(S)$, by the following steps:

1. Extract one sample S_i of size n from S with replacement.
2. Calculate the value of the estimator T for the sample S_i.
3. Repeat the steps 1 and 2 for a pre-determined number of times.
4. Build the probability density function of the estimator T for P using the values obtained in step 2.

From the distribution probability function obtained in bootstrapping, it is possible to find confidence intervals, quantiles distribution, etc, which can be useful for further analysis.

The main advantage of bootstrapping over analytical methods is its simplicity – it is easy to employ the bootstrapping in order to estimate standard errors and confidence intervals for estimators of complex parameters of the distribution, such as percentile points, proportions and correlation coefficients.

In this work, the bootstrapping is employed for estimating the independent probability density function (PDF) of the mean of each merit criterion. Since data is uncorrelated (which is guaranteed by the employment of PCA) it is possible to perform analysis over the independent PDFs instead of using a joint PDF. The comparison of evolutionary algorithms using the PDF can provide more supported decisions than approaches which are based only in mean values, which is commonly adopted for this task.

2.3 First Order Stochastic Dominance

Stochastic dominance is a statistical concept employed to compare two (or more) distribution functions. Since it is not dependent on parametric distributions it can be employed for comparing any entity, provided that the PDFs are known. The stochastic dominance can be defined as follows:

DEFINITION 2.1. *Stochastic Dominance: Consider a problem of minimization. A random variable X_1 presents stochastic dominance of first order over another random variable X_2 if:*

$$cdf(X_1) \geq cdf(X_2) \quad \forall\, x_i \in X_1 \cup X_2 \tag{2}$$

where cdf(X) is the cumulative distribution function of X. ■

The stochastic dominance is illustrated in Figure 1.

Once the stochastic dominance is going to be evaluated computationally, a discrete approximation of the concept becomes necessary, since it is not possible to process an infinite-dimensional variable (represented by a continuous function over a non-compact interval). This approximation can be calculated considering a finite number of values of the cumulative distribution function of the variables, what leads to an interval comparison, as follows.

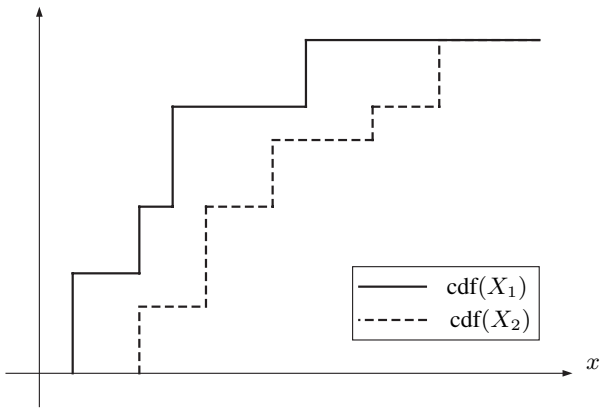

Figure 1: 1^{st} order stochastic dominance example: X_1 dominates X_2, since the quantiles of X_1 always occur before the quantiles of X_2 (minimization problem).

Let X_1 be a random variable, and $x_1^{q1}, x_1^{q2}, \ldots, x_1^{qi}, \ldots, x_1^{qn}$ be the points of X_1 which define the quantiles, $q_0, q_{\frac{1}{n-1}}, \ldots, q_{\frac{i-1}{n-1}}, \ldots, q_1$, as shown in Figure 2[3].

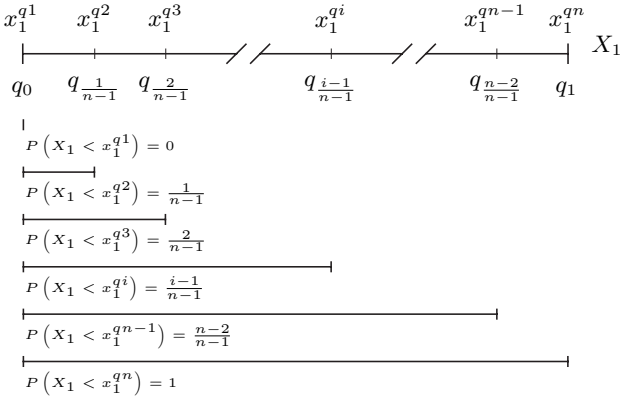

Figure 2: Random variable X_1

Let X_2 be another random variable, and $x_2^{q1}, \ldots, x_2^{qi}, \ldots, x_2^{qn}$ be the points of X_2 which define the same quantiles, $q_0, q_{\frac{1}{n-1}}, \ldots, q_{\frac{i-1}{n-1}}, \ldots, q_1$.

It is said that X_1 "is better than" X_2 if, and only if:

$$\begin{array}{lcll} x_1^{qi} & \leq & x_2^{qi} & \forall\, i \in [1, n] \\ x_1^{qi} & < & x_2^{qi} & \text{for some } i \in [1, n] \end{array} \tag{3}$$

One should note that, when $n \to \infty$, (3) is equivalent to First Order Stochastic Dominance [16].

3. COMPARISON METHODOLOGY

In the proposed approach, two merit criteria have been established for comparison:

- Number of function evaluations performed by the algorithm (n_{fe}), which is employed for estimating the computational

[3]A point x^k of X defines the quantile q_α if $P(X < x^k) = \alpha$.

cost required by the algorithm. Optionally, the computation time of the algorithm could be used as an indicator of the algorithm cost [19].

- Objective function value of the best solution achieved by the algorithm (f_{bs}), which is employed for estimating the convergence ability of the algorithm.

It is straightforward to note that each algorithm run results in a point in the 2 dimension space $[n_{fe}\ ,\ f_{bs}]^T$.

The algorithm comparison methodology proposed in this paper is performed through the following steps:

1. Each algorithm is executed n_r times, in a given test problem;
2. In each run, the number of objective function evaluations up to the stopping criterion being reached, n_{fe}, and the best reached value of objective function, f_{bs}, are recorded;
3. The whole data achieved (all runs performed by all algorithms) is then submitted to a PCA, in order to obtain uncorrelated data with regard to the merit functions;
4. For each merit criterion:
   (a) A bootstrapping procedure is performed over the uncorrelated data:
      - Bootstrapping is employed to estimate the probability distribution function of the estimator **mean**, for all algorithms analyzed.
   (b) The stochastic dominance analysis is used to rank the algorithms, based on the PDFs found in bootstrapping. The algorithms which are not dominated by any other one receive rank 1, the algorithms which are dominated only by the algorithms of rank 1 receive rank 2, and so forth.

This procedure provides a rank of the algorithms in each criterion under which they are being analyzed. A multiobjective analysis can be easily derived from this rank:

- If an algorithm A is better than an algorithm B in one criterion without being worse in the other, it is possible to say that A dominates B, taking into account all the merit functions considered.

It is possible to note that this approach is more powerful than former methods that are based on mean values, since it considers the whole distribution instead of considering only the mean. It makes possible to take into account the spread of the merit function values for each one of the algorithms.

It is important to make clear that the accuracy of this approach is hardly dependant on the quality of the estimated PDFs. When the PDFs present high variability over different bootstrapping runs, it is possible to take conclusions which does not reflect exactly what is present in data. Higher samples tends to provide better PDF estimations, and should be used when possible. However, as can be seen in results section, the approach proposed in this paper has leaded to reliable results, even for small samples (15 and 30 points), showing that this approach can be useful in complex problems (where each algorithm run requires high computation time). Finally, the use of unidimensional PDFs instead of using a joint PDF (which considers all merit functions jointly) does not introduces analysis errors, since the data is uncorrelated by PCA.

4. NUMERICAL RESULTS

The comparison methodology proposed has been tested in the same instances proposed in [1], for the Optimal Communication Spanning Tree problem (OCST). The comparison approach proposed in that paper has been used as a benchmark for testing the approach proposed in this work. The comparison method proposed in [1] is based on Kruskal-Wallis Nonparametric Tests and Multiple Comparisons [3]. Although this method is general, and consequently can be applied to any kind of distribution, it is less powerful than parametric approaches, due to the characteristics of non-parametric tests: the lower power of those tests increases the chance of do not find significant difference between algorithms that could be considered different in practical situations. It is expected that the approach proposed in this paper presents higher capacity of discriminating between algorithms that could be significantly different for practical purposes.

4.1 Optimal Communication Spanning Tree

In *Optimal Communication Spanning Tree* problem [12], the algorithm looks for the spanning tree which presents minimum cost and complies with the communication requirements between the nodes [17]. This problem has been proved to be MAX SNP-hard [14, 17].

The problem can be stated as follows:

$$\min \sum_{i,j \in V} R_{i,j} \cdot C^X_{i,j} \tag{4}$$

where:
$C^X_{i,j}$ is the sum of weights of edges in path $i - j$.
$R_{i,j}$ is the communication requirement between i and j.
V is the set of vertices.

The only constraint considered here is that the network must be a tree. Additionally, constraints which limit the maximum degree of each node could be considered, for modeling equipments used in real cases, such as hubs or switches (which present a limited number of ports).

Such as in the reference [1], the comparison approach has been employed to compare 12 genetic algorithms, when they are applied to solve 10, 25 and 50 node instances of the OCST problem.

In order to make possible the comparison of the approaches, the best function value and the computation time have been considered here as merit criteria. The algorithms have been tested on a Pentium IV (Prescott) at 3.2GHz with 1GB of RAM, using the Matlab 7 environment. Although the computation times are not comparable with other approaches, since they are strictly dependent of the hardware and software used, the time ratio between the methods can provide useful information about the computational cost of the methods. They have been labeled as follows:

Label	Decoding	Crossover	Mutation
A	Characteristic vector	single point	swap
B	Prüfer numbers	single point	point
C	Prüfer numbers	single point	swap
D	Network random keys	single point	point
E	Network random keys	single point	swap
F	Edge sets	single point	point
G	Edge sets	single point	swap
H	Node biased	single point	point
I	Node biased	single point	swap
J	Link and node biased	single point	point
K	Link and node biased	single point	swap
L	-	kruskal	kruskal

Finally, the following parameters have been adopted in all simulations:

- Number of runs: 30 runs per method;
- Population size: 50 individuals;
- Crossover probability: 0.80;
- Mutation probability: 0.45 (per individual);
- Linear ranking and roulette-wheel selection;
- Generational replacement with elitism;
- Stop criterion: 100 generations without improvement.

4.2 Comparison Results - Full Data

The results achieved by the former approach [1] (which will be referred here as ***Kruskal+MC***) have been reproduced here to make easier the evaluation of the comparison methods. The tables 1 and 2 show the results (convergence time and best function value) obtained on three instances of the OCST problem. The ordering of the algorithm performances provided by both comparison approaches is presented together with the tables. The following notation has been adopted for exposing the results: the methods are listed in descending order of performance; in this list, underlining and overlining are used to indicate which methods did not exhibit statistically significant differences from one to the other one.

Table 1: OCST (30 runs) - Best function value

	10 nodes		25 nodes		50 nodes	
Lab.	avg	sd	avg	sd	avg	sd
A	3.65e5	7.25e3	2.58e6	1.09e4	1.19e7	3.52e5
B	3.67e5	1.24e4	2.90e6	2.01e5	1.32e7	7.53e5
C	3.74e5	1.62e4	2.85e6	2.00e5	1.30e7	8.96e5
D	3.75e5	1.59e4	2.93e6	2.81e5	1.42e7	1.36e6
E	3.74e5	1.57e4	2.80e6	1.59e5	1.42e7	1.64e6
F	3.64e5	-	2.57e6	9.89e3	1.16e7	2.28e5
G	3.64e5	-	2.60e6	2.13e4	1.25e7	7.30e5
H	3.64e5	-	2.56e6	6.24e3	1.11e7	2.53e4
I	3.64e5	1.65e3	2.58e6	1.02e4	1.12e7	3.97e4
J	3.64e5	1.44e3	2.64e6	3.37e4	1.17e7	2.98e5
K	3.64e5	9.32e2	2.62e6	3.16e4	1.17e7	3.07e5
L	3.64e5	-	2.57e6	5.60e3	1.20e7	4.99e5

Kruskal+MC:

10 nodes:

$F\ G\ H\ L\ K\ I\ J\ A\ B\ E\ C\ D$

25 nodes:

$H\ L\ F\ I\ A\ G\ K\ J\ E\ C\ B\ D$

50 nodes:

$H\ I\ F\ K\ J\ A\ L\ G\ C\ B\ E\ D$

Proposed approach:

10 nodes:

$L\ F\ G\ H\ K\ I\ J\ A\ B\ E\ C\ D$

25 nodes:

$H\ L\ A\ I\ F\ G\ K\ J\ E\ C\ B\ D$

50 nodes:

$H\ I\ F\ K\ J\ A\ L\ G\ C\ B\ E\ D$

The Tables 1 and 2 show that the comparison methods have presented coherent results: algorithms which have been classified as efficient by one method have also received similar classification by the other one. However, it is important to see that the approach proposed in this paper has shown the capacity of distinguishing between some algorithms that have been considered equivalent by the *Kruskal+MC* method. This was expected, and can be credited to the higher power of the statistical tests employed. The *Kruskal+MC* method presents some intransitivity in classification that can make the decision process harder: for instance, in Table 1 (for 25 nodes), the *Kruskal+MC* determined that the algorithm H is equivalent to L and L is equivalent to G, however H is considered statistically better than G. This phenomenon has not occurred in the approach proposed in this work.

From Table 1 and the results obtained by the bootstrapping-based approach, it is possible to conclude that the algorithm H has presented the best convergence performance. This conclusion could not be safely taken with the *Kruskal+MC* approach, since it has not detected any significant difference between H and I in any instance considered. With respect to the convergence time, the comparison approach proposed in this work has stated that the algorithm C is significantly faster than other ones. Once more, this conclusion could not be drawn by the *Kruskal+MC* approach, since it could not distinguish between B and C.

As discussed in section 3 it is possible to compare the algorithms considering both criteria jointly. It is assumed that one algorithm is better than other if it is better in one criterion without being worse in the other one. This methodology leads to the following classification:

Table 2: OCST (30 runs) - Computation time (ms)

	10 nodes		25 nodes		50 nodes	
Lab.	avg	sd	avg	sd	avg	sd
A	1.956e4	2.628e3	1.325e5	2.235e4	7.317e5	2.262e5
B	8.984e3	7.426e2	3.009e4	7.329e3	1.917e5	4.363e4
C	8.578e3	1.655e3	2.246e4	4.737e3	1.272e5	4.821e4
D	1.492e4	2.255e3	7.437e4	1.828e4	3.376e5	1.046e5
E	1.648e4	3.692e3	7.306e4	2.400e4	3.194e5	1.132e5
F	1.516e4	1.225e3	6.471e4	9.189e3	4.280e5	9.074e4
G	1.638e4	2.266e3	7.306e4	2.147e4	3.412e5	1.241e5
H	1.648e4	8.892e2	7.474e4	1.331e4	5.401e5	1.531e5
I	1.682e4	4.310e2	5.902e4	1.546e4	3.409e5	9.116e4
J	1.773e4	1.085e3	1.103e5	2.824e4	7.909e5	2.169e5
K	1.913e4	1.785e3	1.321e5	4.066e4	8.369e5	2.968e5
L	2.001e4	1.745e3	9.654e4	1.227e4	6.972e5	1.792e5

Kruskal+MC:

10 nodes:

$C\ B\ D\ F\ G\ E\ H\ I\ J\ K\ A\ L$

25 nodes:

$C\ B\ F\ E\ G\ D\ I\ H\ L\ J\ A\ K$

50 nodes:

$C\ B\ E\ D\ G\ F\ I\ H\ L\ A\ J\ K$

Proposed approach:

10 nodes:

$C\ B\ F\ D\ G\ H\ I\ E\ J\ K\ A\ L$

25 nodes:

$C\ B\ F\ G\ I\ E\ H\ D\ L\ A\ J\ K$

50 nodes:

$C\ B\ I\ G\ F\ H\ E\ D\ L\ A\ J\ K$

10 nodes:

$F \quad B \quad G \quad H \quad C \quad L \quad I \quad J \quad D \quad K \quad E \quad A$

25 nodes:

$C \quad F \quad H \quad I \quad B \quad G \quad L \quad A \quad E \quad D \quad J \quad K$

50 nodes:

$C \quad H \quad I \quad B \quad F \quad A \quad G \quad J \quad K \quad D \quad E \quad L$

This classification suggests that the algorithms H and C are the best ones with regard to convergence capacity and time required respectively. The algorithms F and I also represent intermediate choices, presenting good convergence (however lower than H) and requiring a computation time smaller than H. The choice between these algorithms should be taken considering the time available to achieve a good solution. In the authors' opinion the algorithms H and I are the most suitable ones in this case. The algorithm C should not be taken as a reasonable choice, since it presents very poor convergence performance, and could lead to expensive solutions.

4.3 Comparison Results - Reduced Data

A lower dimension set of 15 algorithm runs has been built by sampling (without replacement) 15 runs of the 30 previously performed. The intention of this test is to evaluate the effect of the reduction of the number of algorithm runs in the results achieved by the comparison approaches. If one comparison method requires a smaller set of runs to efficiently compare two or more algorithms it can be considered more suitable, since the acquisition of additional data to perform the comparison carries a high computational cost (it is required a whole algorithm run to find a single point for comparison).

From Tables 3 and 4, it is noticeable that the smaller set of algorithm runs has reduced the capacity of *Kruskal+MC* detects significant difference between the algorithms. When this comparison method is applied over the reduced set, it provides a sorting similar to one achieved in the higher set (with 30 runs). However the uncertainty about the extent to which each method is better than other ones increases significantly, what makes harder the decision of which algorithm should be used.

On the other hand, the smaller number of algorithm runs does not carry too many losses to the bootstrapping based approach. It is obvious that some information is lost, since a small sample often implies in a less reliable approximating function. However, at least in the particular case discussed in this paper, this loss has not caused significant changes in algorithm sorting. This is an interesting property of the proposed method, since it seems to provide efficient comparisons from small sets of algorithm runs.

5. CONCLUSION AND FUTURE WORK

5.1 Concluding Remarks

This paper has proposed a new methodology for comparison of evolutionary algorithms that is based on the techniques of (a) Principal Component Analysis, which is used to make the data uncorrelated; (b) Bootstrapping, which is employed to build the probability distribution function of the merit functions; and (c) Stochastic Dominance Analysis, which performs a multicriteria ordering between algorithms, considering their ability to reach solution accuracy and the related computational effort.

The proposed technique has presented a high capability to discriminate between different algorithms, allowing to detect statistically significant differences between performances which were not detected by former methodologies.

Table 3: OCST (15 runs) - Best function value

	10 nodes		25 nodes		50 nodes	
Lab.	avg	sd	avg	sd	avg	sd
A	3.64e5	-	2.57e6	8.76e3	1.19e7	3.03e5
B	3.67e5	1.16e4	2.93e6	2.29e5	1.32e7	9.19e5
C	3.73e5	1.58e4	2.84e6	1.63e5	1.28e7	6.72e5
D	3.71e5	1.27e4	2.92e6	2.19e5	1.42e7	1.43e6
E	3.69e5	9.57e4	2.85e6	1.91e5	1.42e7	1.59e6
F	3.64e5	-	2.57e6	9.26e3	1.16e7	1.98e5
G	3.64e5	-	2.60e6	2.33e4	1.25e7	7.64e5
H	3.64e5	-	2.57e6	8.40e3	1.11e7	8.60e3
I	3.64e5	2.13e3	2.58e6	1.22e4	1.12e7	3.51e4
J	3.64e5	1.33e3	2.65e6	3.92e4	1.17e7	3.35e5
K	3.64e5	9.76e2	2.63e6	2.70e4	1.17e7	3.78e5
L	3.64e5	-	2.57e6	5.10e3	1.19e7	3.78e5

Kruskal+MC:

10 nodes:

$A \quad F \quad G \quad H \quad L \quad K \quad J \quad I \quad B \quad E \quad D \quad C$

25 nodes:

$H \quad L \quad F \quad A \quad I \quad G \quad K \quad J \quad C \quad E \quad D \quad B$

50 nodes:

$H \quad I \quad F \quad J \quad K \quad A \quad L \quad G \quad C \quad B \quad E \quad D$

Proposed approach:

10 nodes:

$A \quad F \quad G \quad H \quad L \quad K \quad J \quad I \quad B \quad E \quad D \quad C$

25 nodes:

$H \quad L \quad A \quad F \quad I \quad G \quad K \quad J \quad C \quad E \quad D \quad B$

50 nodes:

$H \quad I \quad F \quad J \quad K \quad A \quad L \quad G \quad C \quad B \quad E \quad D$

5.2 Future Work

There are some points which are being considered as future extensions of this work:

- To exploit a parametric version of bootstrapping: when applied to estimate the characteristics of the operator mean the bootstrapping often returns normal (or normal-like) distributions, what can be proved by the Central Limit Theorem [15]. This property can be exploited in order to try to differentiate algorithms which have been considered similar in the stochastic dominance analysis.

- To extend the proposed comparison approach for multiobjective problems: the approach proposed in this paper can be extended to multiobjective problems by replacing the convergence criteria by some Pareto quality indicator. Scalar Pareto-set quality metrics, such as the S-Metric and the Integrated Sphere Counting can be used for this task.

6. ACKNOWLEDGMENTS

The authors would like to thank Brazilian agencies Capes, CNPq and Fapemig by the financial support.

7. REFERENCES

[1] E. G. Carrano, C. M. Fonseca, R. H. C. Takahashi, L. C. A. Pimenta, and O. M. Neto. A preliminary comparison of tree

Table 4: OCST (15 runs) - Computation time (ms)

	10 nodes		25 nodes		50 nodes	
Lab.	avg	sd	avg	sd	avg	sd
A	1.986e4	2.664e3	1.478e5	2.311e4	7.842e5	2.213e5
B	8.974e3	8.852e2	3.965e4	7.581e3	2.075e5	4.319e4
C	8.730e3	1.601e3	3.072e4	3.284e3	1.466e5	4.711e4
D	1.570e4	2.495e3	9.002e4	1.823e4	3.515e5	1.040e5
E	1.519e4	2.476e3	8.940e4	2.319e4	3.823e5	1.292e5
F	1.531e4	1.452e3	8.035e4	9.831e3	4.737e5	1.006e5
G	1.692e4	2.438e3	8.970e4	2.388e4	3.823e5	1.312e5
H	1.645e4	4.474e2	1.165e5	1.438e4	7.136e5	1.634e5
I	1.673e4	3.829e2	9.570e4	1.413e4	4.742e5	8.480e5
J	1.745e4	1.116e3	1.520e5	2.759e4	9.112e5	2.364e5
K	1.912e4	1.563e3	1.637e5	4.161e4	8.590e5	1.875e5
L	2.009e4	1.218e3	1.221e5	1.415e4	7.615e5	1.623e5

Kruskal+MC:

10 nodes:

$C\ B\ E\ F\ D\ H\ I\ G\ J\ K\ A\ L$

25 nodes:

$C\ B\ F\ E\ G\ D\ I\ H\ L\ A\ J\ K$

50 nodes:

$C\ B\ D\ E\ G\ F\ I\ H\ L\ A\ K\ J$

Proposed approach:

10 nodes:

$C\ B\ E\ F\ H\ I\ G\ D\ J\ K\ A\ L$

25 nodes:

$C\ B\ F\ G\ I\ E\ H\ D\ L\ A\ J\ K$

50 nodes:

$C\ B\ I\ G\ F\ D\ H\ E\ L\ A\ K\ J$

encoding schemes for evolutionary algorithms. In *Proc. IEEE International Conference on Systems Man and Cybernetics*, Vancouver, Canada, 2007.

[2] M. R. Chernick. *Bootstrap Methods, A practitioner's guide*, volume Wiley Series in Probability and Statistics. John Wiley and Sons, 1999.

[3] W. J. Connover. *Practical Nonparametric Statistics*. Wiley, 3rd edition, 1999.

[4] B. G. W. Craenen, A. E. Eiben, and J. I. van Hemert. Comparing evolutionary algorithms on binary constraint satisfaction problems. *IEEE Transactions on Evolutionary Computation*, 7:424–444, 2003.

[5] P. Dutta and D. DuttalMajumder. Performance comparison of two evolutionary schemes. In *Proc. International Conference on Pattern Recognition*, pages 659–663, Viena, Austria, 1996.

[6] B. Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7:1–26, 1979.

[7] B. Efron. Nonparametric estimates of standard error: The jackknife, the bootstrap and other methods. *Biometrika*, 68:589–599, 1981.

[8] B. Efron. The jackknife, the bootstrap, and other resampling plans. Technical report, Society of Industrial and Applied Mathematics CBMS-NSF Monographs, 1982.

[9] M. Ehrgott. *Multicriteria Optimization*. Springer, 2000.

[10] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Elsevier, 2 edition, 1990.

[11] R. C. Gonzalez and R. E. Woods. *Digital image processing*. Addison Wesley, 2 edition, 1992.

[12] T. C. Hu. Optimum communication spanning trees. *SIAM Journal of Computing*, 3:188–195, 1974.

[13] E. Oja. Neural networks, principal component, and subspaces. *International Journal of Neural Systems*, 1:61–68, 1989.

[14] C. H. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer System Science*, 43:425–440, 1991.

[15] A. Papoulis. *Probability, Random Variables and Stochastic Processes*. McGraw Hill, 3rd edition, 1991.

[16] S. Pemmaraju and S. Skiena. *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*. Cambridge University Press, 2003.

[17] S. Soak, D. W. Corne, and B. Ahn. The edge-window-decoder representation for tree-based problems. *IEEE Transactions on Evolutionary Computation*, 10(124–144), 2006.

[18] R. H. C. Takahashi, J. A. Vasconcelos, J. A. Ramirez, and L. Krahenbuhl. A multiobjective methodology for evaluating genetic operators. *IEEE Transactions on Magnetics*, 39:1321–1324, 2003.

[19] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, 7:117–132, 2003.