Using Performance Profiles to Analyze the Results of the 2006 CEC Constrained Optimization Competition

Helio J.C. Barbosa, *Member, IEEE*, Heder S. Bernardino and André M.S. Barreto

***Abstract*— Performance profiles are an analytical tool for the visualization and interpretation of the results of benchmark experiments. In this paper we discuss their explanatory power, and argue that they should be more widely used by the evolutionary computation community. We also introduce some novel performance measures which can be extracted from the performance profiles. In order to illustrate their potential, we apply the referred profiles to the analysis of the results of the CEC 2006 constrained optimization competition. While some of the results are corroborated, some new facts are pointed out and additional conclusions are drawn.**

I. Introduction

Optimization problems appear naturally in many areas as one is always interested in minimizing or maximizing quantities such as cost or profit, respectively. Furthermore, other problems, such as system identification, are often formulated as the minimization of a conveniently defined objective function subject to a set of constraints.

Here we are particularly interested in constrained optimization problems that are stated as the minimization (or maximization) of a given objective function $f(x)$, where $x \in R^n$ is the vector of design/decision variables, subject to inequality constraints $g_p(x) \leq 0$, $p = 1, 2, \ldots, \bar{p}$ as well as equality constraints $h_q(x) = 0$, $q = 1, 2, \ldots, \bar{q}$. Additionally, the variables are usually subject to bounds $x_i^L \leq x_i \leq x_i^U$.

Although many real problems can be straightforwardly cast in such a form, this formulation is usually not possible in more complex real world situations where there is no such a thing as explicit mathematical expressions for f, g_p or h_q as functions of the vector of decision variables x. As an example, consider the situation where one needs to find the values of the decision variables x that maximize the lowest natural vibration frequency of a framed structure. Such frequency can be found by solving an eigenvalue problem involving the so called mass and stiffness matrices (which depend on x) of the approximate discrete model of the structure, built via the finite element model, for example. As an example of constraint, it is mandatory that the stresses in each bar of a structure remain below the maximum (material dependent) admissible level. Such stresses can only be computed after a complete simulation of the structural behavior is performed.

It should then be kept in mind that for many real-world optimization problems, the constraints are in fact a complex implicit function of the design variables, and the check for feasibility requires an expensive computational simulation. Furthermore, derivatives of the objective function and/or constraints with respect to the design variables may be undefined, noisy, expensive or unavailable. Constraint handling techniques which do not require the explicit form of the constraints and do not require additional objective function evaluations are thus preferable. This so called "black-box" optimization context, where only "zero-order" information is available, is perhaps the most well suited setting for the application of stochastic population-based nature-inspired techniques.

Nature-inspired techniques, which can be readily applied to unconstrained optimization problems, must usually be equipped with a constraint handling procedure whenever the constraints cannot be automatically satisfied by the candidate solutions generated. Such constraint handling techniques can be direct (feasible or interior), when only feasible elements are considered, or indirect (exterior), when both feasible and infeasible elements are used during the search process. Direct techniques such as special closed genetic operators [1], special decoders [2], and repair techniques [3] are problem dependent and actually of reduced practical applicability. Several indirect techniques are available such as the use of Lagrange multipliers [4] combining fitness and constraint violation in a multi-objective optimization setting [5], the use of special selection techniques [6], and penalty techniques [7]–[10]. For other constraint handling methods in evolutionary computation see [11]–[15], and the on-line bibliography [16].

The need to assess the relative performance of nature-inspired metaheuristics (and their hybrid forms) augmented with a myriad of constraint handling techniques is thus obvious. Although in some cases a worse- and/or average case analysis can be made, the difficulties involved make it often impossible to obtain useful results for more realistic problems and algorithms. As a result, one must resort to an empirical evaluation of the candidate algorithms.

The definition of a set of problems which is representative of the domain of interest is not an easy task, as one would like it to span the target problem-space and at the same time to be as small as possible, in order to alleviate the computational burden associated with the experiments. It is also clear that as the number of test-problems grows the amount of data produced grows to the point of making its visualization, interpretation and analysis a very hard task.

The introduction of constraints brings in another dimension to the optimization problem, which is now more complicated to solve using nature-inspired techniques. The performance analysis is also more delicate now since it is important to also measure the ability of the algorithm in

The authors are with the Laboratório Nacional de Computação Científica, 25651-075 Petrópolis, RJ, Brazil (email: {hcbm,hedersb,amsb}@lncc.br).

producing feasible candidate solutions.

Although many performance measures have been suggested in the literature [17], in this paper we discuss performance profiles [18], an analytical tool that makes it easier to visualize and to interpret the results of benchmark experiments. Some possible useful extensions are also considered. It is argued that those tools should be more often used in the performance comparison of nature inspired metaheuristics applied to optimization, and, as a practical example, they are used here to analyze the results of the 2006 CEC real-parameter optimization competition. As a result of this new look, part of those results are of course corroborated, while some perhaps new facts are pointed out and some additional conclusions are drawn.

II. Benchmarking Optimization Techniques

The use of a set of test problems in order to assess and compare the performance of different nature-inspired techniques has a long history. Perhaps the first test suite that was later widely used was that proposed by De Jong in his Ph.D Thesis in 1975 [19] . It was composed by 5 functions with different characteristics in an effort to provide a sample of the various features and difficulties one might find in solving real-world problems. Such 5-function suite was used for quite some time before being gradually replaced by larger (order of 20) test-function suites. A similar situation occurs within the realm of constrained optimization problems in R^n. The first suite was composed by 11 test-problems [2], then enlarged to 13 [6], and later augmented to a 24-function suite [20]. Additionally, constrained optimization problems from specific areas, such as mechanical engineering [21], have also been often used to compare the performance of the different nature-inspired techniques.

The most common way of assessing the relative performance of a set S of solvers $s_i, i \in \{1, \ldots, n_s\}$ is to define a set P of "representative" problems $p_j, j \in \{1, \ldots, n_p\}$ and then test all solvers against all problems measuring the performance $t_{p,s}$ of solver $s \in S$ when applied to problem $p \in P$.

The first step in order to evaluate $t_{p,s}$ is to define a meaningful goal and then measure the amount of resources required by the solver to achieve that goal.

Examples of goals include (i) to reach a given objective function level $\bar{f}$: that means finding an $\bar{x}$ such that $f(\bar{x}) \leq \bar{f}$ $\left(f(\bar{x}) \geq \bar{f}\right)$ for a minimization (maximization) problem, and (ii) to find a feasible solution: for a constrained optimization problem, that means finding $\bar{x}$ such that $g_p(\bar{x}) \geq 0$, $p = 1, 2, \ldots, \bar{p}$ and $h_q(\bar{x}) = 0$, $q = 1, 2, \ldots, \bar{q}$.

Examples of resources include (i) CPU time and (ii) number of objective function evaluations (useful for black-box optimization involving expensive simulations).

A. Performance Profiles

In order to introduce the performance profiles, consider that $t_{p,s}$ is the CPU time spent by solver s to reach the objective function value $\bar{f}_p$ in problem p. If a solver fails to reach the goal for a given problem, the corresponding $t_{p,s}$ is set to a conveniently defined large number. Given the definition of $t_{p,s}$, we introduce the performance ratio as

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : s \in S\}}. \quad (1)$$

Although each $t_{p,s}$ or $r_{p,s}$ is worth considering by itself, we would like to be able to assess the performance of the solvers in S on a large set of problems P in a graphical form suitable for human inspection. This has been done by Dolan & More [18] and here we extend that performance analysis tool with some useful numerical performance indexes easily derived from such graph. Denoting the cardinality of a set A by $|A|$ and defining

$$\rho_s(\tau) = \frac{1}{n_p} \left|\{p \in P : r_{p,s} \leq \tau\}\right|$$

then $\rho_s(\tau)$ is the probability that the performance ratio $r_{p,s}$ of solver $s \in S$ is within a factor $\tau \geq 1$ of the best possible ratio. If the set P is large and representative of problems yet to be tackled then solvers with larger $\rho_s(\tau)$ are to be preferred.

From the definitions above, several properties follow:

1) The performance profile $\rho_s : [1, \tau_{max}] \mapsto [0, 1]$ for solver s is a nondecreasing, piecewise constant function, continuous from the right at each discontinuity point.

2) The performance profile is insensitive to the results on a small set of problems. This results from the fact that if the performances of solvers A and B differ only in problem q, then

$$\|\rho_A - \rho_B\|_\infty = \max_\tau |\rho_A(\tau) - \rho_B(\tau)| \leq \frac{1}{n_p}.$$

As n_p grows, the performance in a particular problem q does not substantially affect the performance profile ρ_s.

3) The performance profile is also insensitive to small variations in several problems. In fact, if $r_{p,A}$ and $r_{p,B}$ are the performance ratios for solvers A and B on problem p and one has

$$|r_{p,A} - r_{p,B}| \leq \varepsilon \quad \text{for } p = 1, \ldots, n_p$$

for some small $\varepsilon > 0$, then

$$\|\rho_A - \rho_B\|_1 = \int_1^\infty |\rho_A(t) - \rho_B(t)| \, dt \leq \varepsilon$$

This is Theorem 1 proved by Dolan & More [18].

4) $\rho_s(1)$ is the probability that solver s will provide the best performance in P among all solvers in S. If $\rho_A(1) > \rho_B(1)$ then solver A was the winner in a larger number of problems in P than solver B.

In addition to the properties above, which were noticed by Dolan & More [18], we also point out that:

5) Property 4 suggests that the area under the ρ_s curve ($AUC_s = \int \rho_s(t)\, dt$) is an overall performance indicator/measure for solver s in the problem set P: the larger the AUC the higher the solver efficiency.

6) Another desirable feature of a solver is its reliability. A measure of the reliability of solver s is its performance ratio in the problem where it performed worst

$$R_s = \sup\{\tau : \rho_s(\tau) < 1\}.$$

As a result, the most reliable solver is the one that minimizes R_s; that is, it presents the best worst performance in the set P. This is in fact a minsup criterion:

$$s^* = \arg\min_{s \in S} R_s = \arg\min_{s \in S} \sup\{\tau : \rho_s(\tau) < 1\}$$

7) An underlying hypothesis present in [18] is that all problems in P have the same importance/influence in the performance profiles. That may not always be the case and it could be previously agreed that some problems should have a higher influence in the performance profiles. Another extension proposed here is to define a weight for each test-problem. For instance, it could be considered that solving a larger problem (one with a larger number of variables) should be more rewarding than solving a smaller one. Another possibility is to define the weights "a posteriori". After all the testing is done, each problem could be weighted by the computer resources required by its solution, thus assigning more importance to those problems that proved to be harder to solve.

To illustrate the ideas above, we provide a simple example. Given the Table I below with performance ratios, the corresponding performance profiles for solvers A, B, and C are shown in Figure 1 .

TABLE I

EXAMPLE OF PERFORMANCE RATIOS FOR SOLVERS A, B, AND C.

	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9
A	1.5	2.0	2.0	2.2	2.4	2.5	2.6	2.8	2.9	3.0
B	1.0	1.0	1.0	1.0	1.6	1.8	2.2	2.4	2.5	4.0
C	4.5	4.2	1.5	1.1	1.0	1.0	1.0	1.0	1.0	1.0

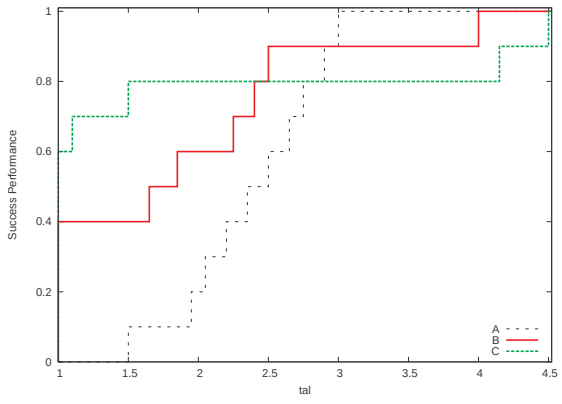

Fig. 1. Example considering the performance ratios for solvers A, B, and C from Table I.

In Figure 1 it is easy to see that:

a) As $\rho_C(1) > \rho_B(1) > \rho_A(1)$ solver C is the one that comes out first more often in the set P (actually in 60% of the problems.)

b) As $\rho_A(1) = 0$ solver A is never the best for the problems in P.

c) From Figure 1 one has $AUC_A = 21.15$, $AUC_B = 26.35$, and $AUC_C = 27.75$. As $AUC_C > AUC_B > AUC_A$, solver C is the most efficient one, followed by B and A.

d) However, under the minisup reliability criterion, solver A is the most reliable one, since it solves all problems in P within a factor of three from the best performance. Solver B comes in second (a factor of 4), followed by solver C (a factor of 4.5).

III. THE 2006 CEC COMPETITION

Along with the Special Session on Constrained Real-Parameter Optimization [22] organized by P. N. Suganthan, Carlos A. C. Coello, Kalyanmoy Deb, and Efren Mezura-Montes during the IEEE Congress on Evolutionary Computation in 2006, a competition to evaluate nature-inspired techniques applied to real-parameter constrained optimization problems was held.

A set of 24 minimization problems in R^n with equality as well as inequality constraints was used. The equality constraints were transformed into inequalities of the form

$$|h_q(x)| - \varepsilon \leq 0 \quad q = 1, 2, \ldots, \bar{q}$$

where ε was set to 0.0001.

A. The Benchmark

Some details of the 24 test-problems which compose the benchmark of the CEC 2006 competition are given in Table II, in which the column "Prob." indicates the problem, "n" is the number of decision variables, "Type" is the type of the objective function, $\frac{|F|}{|S|}$ is the estimated ratio between the feasible region and the search space, LI is the number of linear inequality constraints, NI is the number of nonlinear inequality constraints, LE is the number of linear equality constraints, NE is the number of nonlinear equality constraints, and AC is the number of active constraints at the best solution known. The best solutions known for the benchmark functions can be found in Table III.

B. Performance Evaluation Criteria

The contestants were required to perform 25 independent runs for each problem with a maximum of 500000 objective function evaluations and record the function error value $f(x) - f(x^*)$ (x^* is the best solution known) for the best solution achieved after 5×10^3, 5×10^4, and 5×10^5 function evaluations. For each problem, the best, median and worst results were presented along with the average and standard deviation in the 25 runs.

The number of function evaluations required in each run to achieve a feasible solution x such that

$$f(x) - f(x^*) \leq 0.0001 \tag{2}$$

was also recorded. Again, the best, median and worst results were presented along with the average and standard deviation in the 25 runs. Additional data from the experiments were

TABLE II

DETAILS OF THE CEC 2006 COMPETITION PROBLEMS.

Prob.	n	Type	$\frac{\lvert F\rvert}{\lvert S\rvert}$ (%)	LI	NI	LE	NE	AC
g01	13	quadratic	0.0111	9	0	0	0	6
g02	20	nonlinear	99.9971	0	2	0	0	1
g03	10	polynomial	0.0000	0	0	0	1	1
g04	5	quadratic	52.1230	0	6	0	0	2
g05	4	cubic	0.0000	2	0	0	3	3
g06	2	cubic	0.0066	0	2	0	0	2
g07	10	quadratic	0.0003	3	5	0	0	6
g08	2	nonlinear	0.8560	0	2	0	0	0
g09	7	polynomial	0.5121	0	4	0	0	2
g10	8	linear	0.0010	3	3	0	0	6
g11	2	quadratic	0.0000	0	0	0	1	1
g12	3	quadratic	4.7713	0	1	0	0	0
g13	5	nonlinear	0.0000	0	0	0	3	3
g14	10	nonlinear	0.0000	0	0	3	0	3
g15	3	quadratic	0.0000	0	0	1	1	2
g16	5	nonlinear	0.0204	4	34	0	0	4
g17	6	nonlinear	0.0000	0	0	0	4	4
g18	9	quadratic	0.0000	0	13	0	0	6
g19	15	nonlinear	33.4761	0	5	0	0	0
g20	24	linear	0.0000	0	6	2	12	16
g21	7	linear	0.0000	0	1	0	5	6
g22	22	linear	0.0000	0	1	8	11	19
g23	9	linear	0.0000	0	2	3	1	6
g24	2	linear	79.6556	0	2	0	0	2

TABLE III

BEST SOLUTION KNOWN FOR THE 24 TEST-FUNCTIONS.

Prob.	Value	Prob.	Value
g01	-15.0000000000	g13	0.0539415140
g02	-0.8036191042	g14	-47.7648884595
g03	-1.0005001000	g15	961.7150222899
g04	-30665.5386717834	g16	-1.9051552586
g05	5126.4967140071	g17	8853.5396748064
g06	-6961.8138755802	g18	-0.8660254038
g07	24.3062090681	g19	32.6555929502
g08	-0.0958250415	g20	0.2049794002
g09	680.6300573745	g21	193.7245100700
g10	7049.2480205286	g22	236.4309755040
g11	0.7499000000	g23	-400.0551000000
g12	-1.0000000000	g24	-5.5080132716

required; however, they were not used in the assessment of each algorithm performance.

It is important to note that (i) a *feasible run* is one in which at least one feasible solution is found in 500000 function evaluations, and (ii) a *successful run* is one in which a feasible solution is found such that condition (2) is satisfied.

The result of the competition was defined based on the computation, for each problem, of the three following quantities

$$\text{Feasible rate} = \frac{\text{number of feasible runs}}{\text{total of runs}} \quad (3)$$

$$\text{Success rate} = \frac{\text{number of successful runs}}{\text{total of runs}} \quad (4)$$

$$\text{Success performance} = \frac{M \times (\text{total of runs})}{\text{number of successful runs}} \quad (5)$$

where M is the average number of fitness function evaluations executed in the successful runs. While in the case of (5) the goal was to minimize the corresponding quantity, equations (3) and (4) should be maximized. Since the performance measure $t_{p,s}$ in equation (1) is to be minimized, we used the inverse of (3) and (4) to generate the performance profiles.

C. The Contestants

Twelve papers were published in the 2006 special session while ten of these participated in the competition. They were:

- jDE-2 – a Differential Evolution (DE) algorithm whose control parameters are self-adaptive. Any feasible solution is considered better than an infeasible one; the latter are ranked according to the sum over all the constraint violations;
- DE – this is the standard differential evolution algorithm, with the same constraint-handling method adopted by jDE-2.
- SaDE – this DE algorithm is an extension of the original SaDE. The constraint-handling method is similar to that of jDE-2 but the constraint violations are weighted;
- GDE – this algorithm extends DE for constrained multi-objective optimization. The constraint-handling method is similar to that of jDE-2;
- DMS-PSO – a dynamic and multiple Particle Swarm Optimization (PSO) algorithm. The constraint-handling method is similar to SaDE's;
- MDE – a DE-based approach modified to solve constrained optimization problems. Its constraint-handling method is similar to jDE-2's;
- PESO+ – a PSO-based approach with topological organization and constraint handling similar to that of jDE-2;
- PCX – this approach is derived from the population-based algorithm-generator and uses the parent-centric recombination (PCX) operator and a stochastic remainder selection over three different constraint handling principles;
- εDE – this DE uses the ε constraint-handling method and employs a gradient-based mutation/repair operator.
- MPDE – this is a multi-populated differential-evolution algorithm in which an adaptive penalty method is used to handle the constraint violations;

D. The Results

The algorithms were ranked considering equations (3), (4), (5), and a fourth measure denoted by $\overline{f^r(x)}$.[1] The final rank was computed based on the sum of the individual ranks induced by each measure (see Table IV).

IV. ANOTHER VIEW OF THE RESULTS

Before analyzing the results, two comments are in order.

REMARK 1 – It concerns the use of an absolute error (or distance) measure with respect to the best solution known given by equation (2). It is felt that a *relative* measure should have been adopted instead, such as

$$\frac{f(x) - f(x^*)}{f(x^*)} \leq \varepsilon,$$

[1]We were not able to reproduce the $\overline{f^r(x)}$ rank from the information available in [22].

TABLE IV
RANKS OF THE SPECIAL SESSION ON CONSTRAINED REAL-PARAMETER OPTIMIZATION COMPETITION.

Algorithm	$\overline{f^r(x)}$	Feasible Rate	Success Rate	Success Performance	Final Rank
εDE	1	1	1	4	1
DMS-PSO	3	1	3	3	2
SaDE	6	1	6	1	3
MDE	5	4	4	1	3
PCX	2	4	2	7	5
MPDE	4	9	5	5	6
DE	7	4	8	6	7
jDE-2	8	4	7	9	8
PESO+	9	8	10	10	9
GDE	10	10	9	8	9

since $f(x^*)$ is never zero for this set of problems. The consequence of using condition (2) is that while test-function g04 is treated with a relative error of

$$\frac{f(x) - f(x^*)}{f(x^*)} \leq \frac{0.0001}{30665.5386717834} \approx 3.2610^{-9}$$

(a variation in the 9-th significant figure), test-function g13 is treated with a relative error of

$$\frac{f(x) - f(x^*)}{f(x^*)} \leq \frac{0.0001}{0.0539415140} \approx 1.8510^{-3},$$

a variation in the third significant figure. This may have introduced an avoidable bias, as different test-problems were solved with different accuracy requirements.

REMARK 2 – It concerns the fact that the algorithm εDE [23] actually makes use of the explicit mathematical expressions for the constraints ($g_p(x) \geq 0$ and $h_q(x) = 0$), available for the test-problems considered. The gradient-based mutation used by εDE acts as a repair method that tries to bring an infeasible point to the feasible domain using the derivatives of the constraint functions.

As mentioned before, the availability of explicit expressions for f, g and h is just a convenience for the construction of the suite of test-problems. The results concerning the algorithm εDE [23] cannot thus be extended to real-world black-box optimization problems, where such explicit knowledge of constraint functions and their derivatives is often not available.

A. Analysis of the results

In this section we analyze the results of the CEC competition under different perspectives. The idea is to show how the overall conclusion regarding the performance of the algorithms changes dramatically depending on the criterion adopted to evaluate them. We use the performance profiles as the main tool for our analysis. In particular, for each metric presented in Table IV we show the performance-profile curves and the corresponding AUCs. In order to give a broader view of the competitions results, we also plot, for each algorithm, each one of its performance metrics along the corresponding vertical parallel-coordinate axis (see Figure 9). Each line joining the values marked along the parallel-coordinates represents one of the algorithms considered. The metrics are such that better performance corresponds to higher values along the vertical. As a result, the curve of the ideal algorithm should always appear "above" all the other curves. In fact, every time two of those curves cross each other the relative ranking of the algorithms involved changes from the current performance measure to the next one in the plot. For convenience, the values marked along each vertical axis are displayed in Table V.

In Figure 2 performance profiles for the measure defined by equation (5) are displayed. Each curve corresponds to one algorithm, and, at the left, $\rho(1)$ indicates the fraction of test-problems where the algorithm was the best performer. This fraction is also plotted in the last column ("%Wins") of the parallel coordinates plot of Figure 9. Figure 3 displays the area under each performance profile curve (in the interval $[1, \tau_{max}]$) which is also plotted in column 2 ("Succ. Perf. AUC") of the parallel coordinates plot of Figure 9. From Figures 2, 3 and 9 it is easy to see that PESO+ is the least performant algorithm, and that both jDE-2 and DE are also somewhat "behind the pack". Also, the rank resulting from Figure 3 turns out to be substantially different from that shown in the success performance column in Table IV. Specifically, the two top ranking algorithms, SaDE and MDE, are now surpassed by εDE, PCX, MPDE and GDE.

At the extreme right of the graphic shown in Figure 3, $\rho(\tau_{max})$, one can see the final fraction of problems that were eventually solved by each algorithm within the computational resources that were alloted for the contestants. Those values are plotted in the fourth column of the parallel coordinates plot of Figure 9. Only PCX, εDE, MPDE, GDE, and SaDE were 100% reliable, solving all problems in the suite.

Another very important measure that can be observed from the performance profiles in Figure 2 is the value of τ^* where an algorithm finally solves the whole suite, i.e. when its profile reaches $\rho(\tau^*) = 1$. To take this measure into account, the ratio τ_{min}/τ^* is plotted in the fifth column of the parallel coordinates plot of Figure 9. Here τ_{min} is the minimum value of τ^* among the algorithms. From the 100% reliable algorithms mentioned above, the best performers are PCX, εDE, MPDE, GDE, and SaDE.

Figures 4 and 5 show the feasible-rate performance profiles and the corresponding areas under the curves. Due to the type of problems considered here, it is important to verify this performance measure. Comparing Figure 5 and Tables IV and VI, it can be seen that the rank induced by the feasible rate AUC criterion coincides with CEC's competition final rank.

Figures 6 and 7 show the performance profiles and the corresponding areas under the curves, respectively, for the success rate. It can be seen that, while εDE presents the largest area, PCX and MPDE have very similar values. Differently to what happens when the feasible rate is considered, the rank induced by the success rate AUC criterion does not coincide with CEC's competition final rank. Notice

that, under CEC's criteria, DMS-PSO is the third algorithm while MPDE is ranked fifth. However, according to the AUC criterion, MPDE (the third) is considerably better than DMS-PSO (the fourth).

Also, it is important to notice that all problems, except g20, were considered when building the plots in Figures 4, 5, 6, and 7. However, the performance profiles for the success performance measure use only the results of problems where at least one algorithm was successful at least once (i.e., all problems except g20 and g22). The CEC 2006 results used this same criterion.

To summarize, Table VI displays the algorithm ranks that would be induced by all the performance measures discussed and, in parenthesis, the ranks from Table IV.

TABLE V

VALUES USED TO GENERATE THE PARALLEL COORDINATES PLOT.

Algorithm	Feas. Rate AUC	Succ. Perf. AUC	Succ. Rate AUC	$\rho(\tau_{max})$	$\tau_{min}/\tau*$	%Wins
εDE	1.00	1.0000	1.000	1.00	0.79	0.00
SaDE	1.00	0.9562	0.952	1.00	0.03	0.14
DMS-PSO	1.00	0.9559	0.954	0.95	0.00	0.18
MDE	0.96	0.9565	0.945	0.95	0.00	0.55
PCX	0.96	0.9992	0.999	1.00	1.00	0.05
MPDE	0.93	0.9942	0.992	1.00	0.38	0.05
DE	0.96	0.9020	0.896	0.91	0.00	0.00
jDE-2	0.96	0.8624	0.862	0.91	0.00	0.00
PESO+	0.95	0.7355	0.761	0.77	0.00	0.00
GDE	0.82	0.9722	0.936	1.00	0.09	0.05

TABLE VI

RANKS INDUCED BY THE PROPOSED MEASURES.

Algorithm	Feas. Rate AUC	Succ. Perf. AUC	Succ. Rate AUC	$\rho(\tau_{max})$	$\frac{\tau_{min}}{\tau*}$	%Wins
εDE	1 (1)	1 (4)	1 (1)	2	2	7
DMS-PSO	1 (1)	7 (3)	4 (3)	6	5	2
SaDE	1 (1)	6 (1)	5 (6)	5	6	3
MDE	4 (4)	5 (1)	6 (4)	6	6	1
PCX	4 (4)	2 (7)	2 (2)	1	1	4
MPDE	9 (9)	3 (5)	3 (5)	3	3	4
DE	4 (4)	8 (6)	8 (8)	8	6	7
jDE-2	4 (4)	9 (9)	9 (7)	8	6	7
PESO+	8 (8)	10 (10)	10 (10)	10	6	7
GDE	10 (10)	4 (8)	7 (9)	4	4	4

As different performance measures have been defined, one can think of the evaluation of an algorithm as a multi-objective optimization problem, where, ideally, one would like to optimize all performance measures simultaneously. However, that is often impossible, and it can be seen in the fourth, fifth, and sixth columns of Figure 9 that $\rho(\tau_{max})$ and $\tau_{min}/\tau*$ are conflicting with "%Wins", indicating that the more efficient algorithms tend to be less reliable. To verify the relation between these features Figure 8 shows the Pareto front considering: (i) the success performance when $\tau = 1$, versus $\tau = \tau_{max}$, in the first plot, and (ii) the success performance when $\tau = 1$ versus the τ value when the success

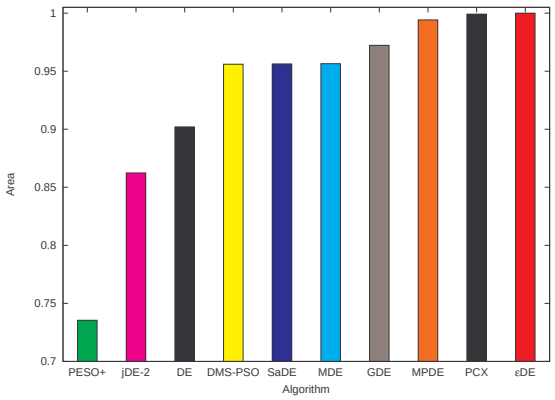

Fig. 3. Areas under the success performance profiles (normalized).

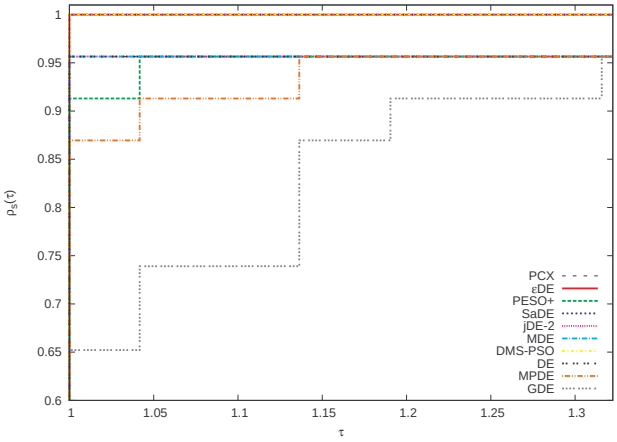

Fig. 4. Feasible rate performance profiles.

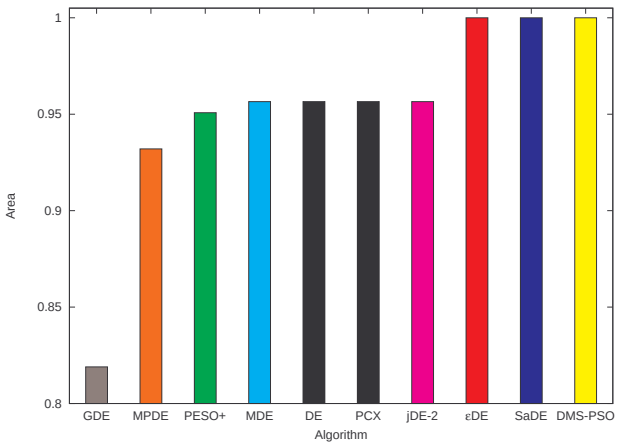

Fig. 5. Areas under the feasible rate performance profiles (normalized).

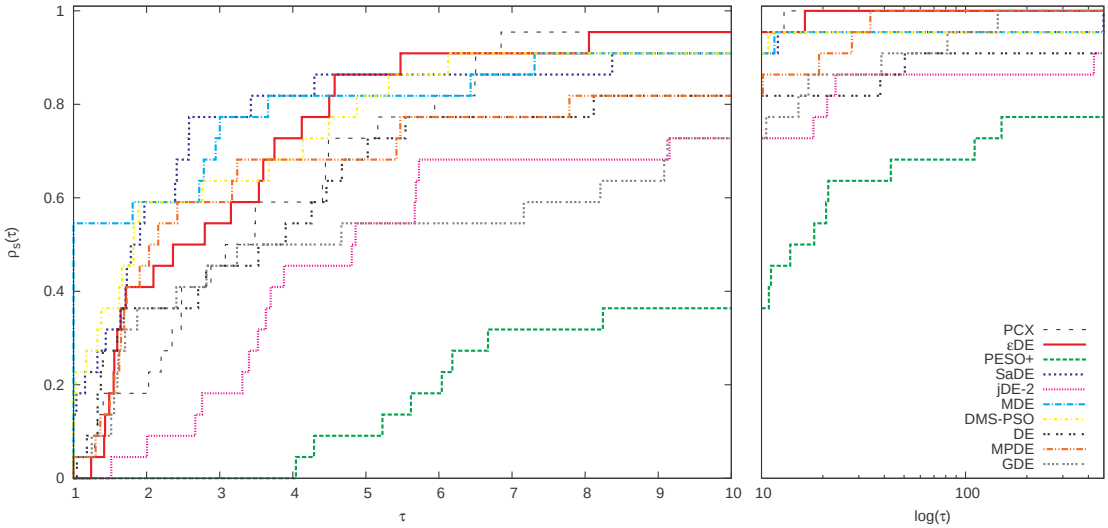

Fig. 2. Standard performance profiles for $\tau \in [1, 10]$ and, in logarithmic scale, for $\tau \in [10, \tau_{max}]$.

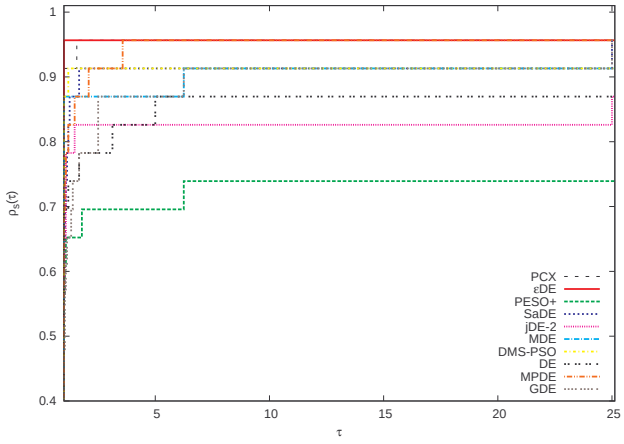

Fig. 6. Success rate performance profiles.

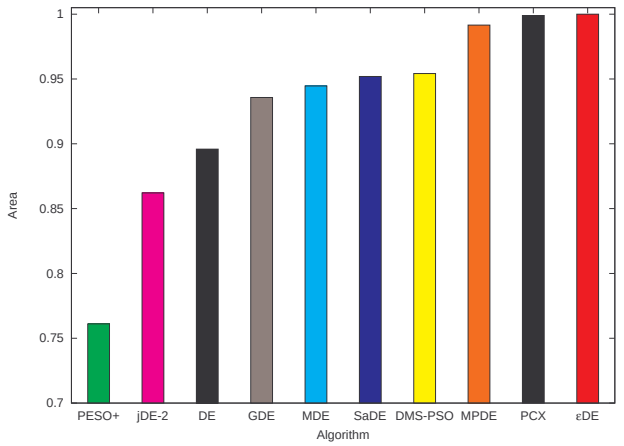

Fig. 7. Areas under the success rate performance profiles (normalized).

performance is equal to 1 ($\rho(\tau^*) = 1$), in the second one. All terms must be maximized, except the τ value when the success performance is equal to 1, in which case lower values are preferred.

It can be observed that two algorithms are non-dominated in the first plot of Figure 8. MDE presents better success performance when $\tau = 1$ (%Wins) while SaDE presents better reliability, i.e., success performance when $\tau = \tau_{max}$. The second plot of Figure 8 considers only the algorithms with $\rho = 1$ in $\tau = \tau_{max}$. Again, two algorithms are non-dominated: SaDE and PCX. SaDE presents better success performance when $\tau = 1$ while PCX has the lowest τ when the success performance is equal to 1.

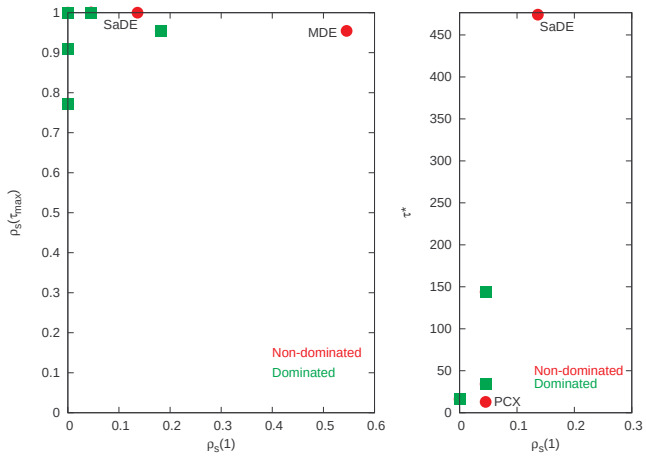

Fig. 8. Graphic with the success performance results.

V. Conclusions

Benchmark experiments play a fundamental role in the comparative evaluation of algorithms. In this paper we discuss performance profiles, which are tools for analyzing the

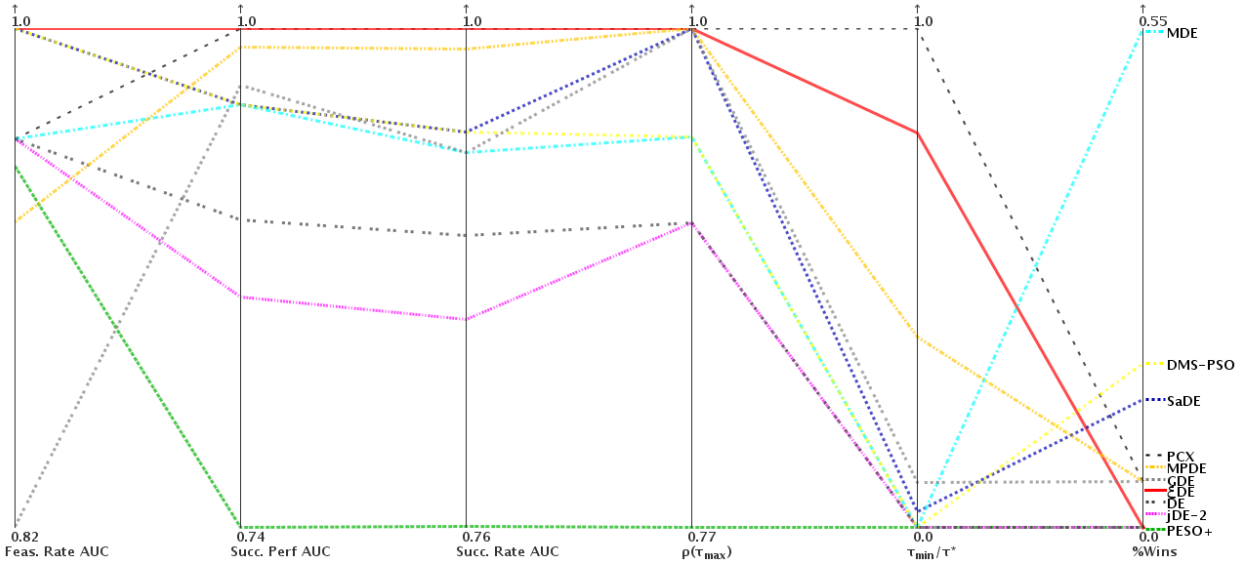

Fig. 9. Parallel coordinates.

results of such experiments. We argue that, due to their explanatory power, such profiles should be more widely used by the evolutionary computation community. We also introduce some novel performance measures which can be extracted from the performance profiles. In order to illustrate their potential, we use the referred profiles to analyze the results of the CEC 2006 constrained optimization competition. While some of the results were corroborated, some new facts were pointed out and additional conclusions were drawn.

Acknowledgment

The authors would like to thank CNPq (grants 311651/2006-2 and 140551/2008-5), FAPERJ (grant E-26/ 102.825/2008) and CAPES for the support.

References

[1] M. Shoenauer and Z. Michalewicz, "Evolutionary computation at the edge of feasibility," in *Parallel Problem Solving from Nature - PPSN IV*, H. M. Voigt *et al.*, Eds. Springer-Verlag, 1996, pp. 245–254.

[2] S. Koziel and Z. Michalewicz, "Evolutionary Algorithms, Homomorphous Mappings, and Constrained Parameter Optimization," *Evolutionary Computation*, vol. 7, no. 1, pp. 19–44, 1999.

[3] D. Orvosh and L. Davis, "Using a Genetic Algorithm to Optimize Problems with Feasibility Constraints," in *Proc. of the First IEEE Conference on Evolutionary Computation*, 1994, pp. 548–553.

[4] H. J. Barbosa, "A coevolutionary genetic algorithm for constrained optimization problems," in *Proc. of the 1999 Congress on Evolutionary Computation*, Washington, DC, USA,, pp. 1605–1611.

[5] P. Surry and N. Radcliffe, "The COMOGA Method: Constrained Optimisation by Multiobjective Genetic Algorithms," *Control and Cybernetics*, vol. 26, no. 3, pp. 391–412, 1997.

[6] T. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 3, pp. 284–294, September 2000.

[7] H. H. S.-Y. Lai and X. Qi, "Constrained optimization via genetic algorithms," *Simulation*, vol. 62, no. 4, pp. 242–254, 1994.

[8] J. Joines and C. Houck, "On the use of non-stationary penalty methods to solve nonlinear constrained optimization problems with GAs," in *Proc. of 1994 IEEE Conf. on Evolutionary Computation*, D. Fogel and Z. Michalewicz, Eds., 1994, pp. 579–585.

[9] D. C. A. Smith and D. Tate, "Adaptive penalty methods for genetic optimization of constrained combinatorial problems," *INFORMS Journal on Computing*, vol. 6, no. 2, pp. 173–182, 1996.

[10] H. J. C. Barbosa and A. C. C. Lemonge, "An adaptive penalty scheme in genetic algorithms for constrained optimization problems," in *Proc. of the Genetic and Evolutionary Computation Conference*, 2002, pp. 287–294.

[11] Z. Michalewicz and M. Shoenauer, "Evolutionary algorithms for constrained parameter optimization problems," *Evolutionary Computation*, vol. 4, no. 1, pp. 1–32, 1996.

[12] J.-H. Kim and H. Myung, "Evolutionary programming techniques for constrained optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 2, no. 1, pp. 129–140, 1997.

[13] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2-4, pp. 311–338, June 2000.

[14] S. Hamida and M. Shoenauer, "An adaptive algorithm for constrained optimization problems," in *Parallel Problem Solving from Nature - PPSN VI*, vol. LNCS 1917. Springer-Verlag, 2000, pp. 529–538.

[15] J. Wright and R. Farmani, "Genetic algorithms: A fitness formulation for constrained minimization." in *Proc. of the 2001 Genetic and Evolutionary Computation Conference*, pp. 725–732.

[16] C. A. C. Coello, "List of references on constraint-handling techniques used with evolutionary algorithms," http://www.cs.cinvestav.mx/ constraint/.

[17] T. Bartz-Beielstein, *Experimental Research in Evolutionary Computation – The New Experimentalism*. Springer-Verlag, 2006.

[18] E. Dolan and J. J. Moré, "Benchmarcking optimization software with performance profiles," *Math. Programming*, vol. 91, no. 2, pp. 201–213, 2002.

[19] K. De Jong, "Analysis of behavior of a class of genetic adaptive systems," Ph.D. dissertation, The University of Michigan, 1975.

[20] E. Mezura-Montes and C. A. C. Coello, "Identifying on-line behavior and some sources of difficulty in two competitive approaches for constrained optimization," in *Proc. of the 2005 IEEE Congress on Evolutionary Computation*, vol. 2. IEEE Press, pp. 1477–1484.

[21] A. C. Lemonge and H. J. Barbosa, "An Adaptive Penalty Scheme for Genetic Algorithms in Structural Optimization," *Intl. J. for Numerical Methods in Engineering*, vol. 59, no. 5, pp. 703–736, February 2004.

[22] P. N. Suganthan, "Special session on constrained real-parameter optimization," http://www3.ntu.edu.sg/home/epnsugan/index_files/CEC-06/CEC06.htm.

[23] T. Takahama and S. Sakai, "Constrained optimization by the ε constrained differential evolution with gradient-based mutation and feasible elites," in *Proc. of the IEEE Congress on Evolutionary Computation*, 2006, pp. 1–8.