Reducing the Experiments Required to Assess the Performance of Metaheuristic

Reducción de los Experimentos Requeridos para Evaluar el Desempeño de Algoritmos Metaheurísticos

Héctor Joaquín Fraire Huacuja, Rodolfo Abraham Pazos Rangel, Juan Javier González Barbosa, Laura Cruz Reyes, Guadalupe Castilla Valdez, and José A. Martínez Flores
Instituto Tecnológico de Ciudad Madero,
hfraire@prodigy.net.mx; r_pazos_r@yahoo.com.mx;
jjgonzalezbarbosa@hotmail.com;
lcruzreyes@prodigy.net.mx; gpe_cas@yahoo.com.mx; jmtz05@yahoo.com.mx

Article received on July 31, 2008; accepted on February 17, 2009

Abstract. When assessing experimentally the performance of metaheuristic algorithms on a set of hard instances of an NP-complete problem, the required time to carry out the experimentation can be very large. A means to reduce the needed effort is to incorporate variance reduction techniques in the computational experiments. For the incorporartion of these techniques, the traditional approaches propose methods which depend on the technique, the problem and the metaheuristic algorithm used. In this work we develop general-purpose methods, which allow incorporating techniques of variance reduction, independently of the problem and of the metaheuristic algorithm used. To validate the feasibility of the approach, a general-purpose method is described which allows incorporating the antithetic variables technique in computational experiments with randomized metaheuristic algorithms. Experimental evidence shows that the proposed method yields a variance reduction of the random outputs in 78% and that the method has the capacity of simultaneously reducing the variance of several random outputs of the algorithms tested. The overall reduction levels reached on the instances used in the test cases lie in the range from 14% to 55%.

Keywords: Experimental algorithm analysis, variance reduction techniques and metaheuristic algorithms.

Resumen. Cuando se evalúa el desempeño de algoritmos metaheurísticos, con un conjunto de instancias difíciles de un problema NP-completo, el tiempo requerido para realizar la experimentación puede ser muy grande. Una forma de reducir el esfuerzo necesario es incorporar técnicas de reducción de la varianza en los experimentos computacionales. Para incorporar dichas técnicas, los enfoques tradicionales proponen métodos que dependen de la técnica, del problema y del algoritmo usado. En este trabajo se propone desarrollar métodos de propósito general, los cuales permitan incorporar técnicas de reducción de la varianza, independientemente del problema y del algoritmo metaheurístico usado. Para validar la factibilidad del enfoque, se describe un método de propósito general, el cual permite incorporar la técnica de variables antitéticas en experimentos computacionales con algoritmos metaheurísticos aleatorizados. La evidencia experimental muestra que el método propuesto produce una reducción de la varianza de las salidas aleatorias en un 78% de las instancias consideradas y que el método tiene la capacidad de reducir simultáneamente la varianza de varias salidas aleatorias de los algoritmos probados. Los niveles globales de reducción alcanzados con las instancias usadas en los casos de prueba van del 14% al 55%.

Palabras clave: Análisis experimental de algoritmos, técnicas de reducción de la varianza y algoritmos metaheurísticos.

1 Introduction

Many real-world applications require the solution of optimization problems which belong to a special class denominated NP-complete problems. Currently, efficient algorithms to solve large instances for this kind of problems are not known, and it is suspected that it is not possible to devise them. In [Garey and Johnson, 1979] and [Papadimitriou and Steiglitz, 1998] it is indicated that the solution of large instances of NP-complete problems can only be solved simplifying the problem or using an approximate solution method. Metaheuristic methods [Glover, 1986] or modern heuristic methods [Revees, 1993] are approximate algorithms that combine basic heuristics (constructive or local search), aiming at navigating effectively and efficiently the solution space [Blum and Roli, 2003]. Metaheuristic algorithms implement navigation strategies through the solution space that permit escaping from local optima and approach the global optimum. There currently exists a large diversity of available metaheuristcs such as: Genetic Algorithms, Simulated Annealing, Threshold Accepting, Ant Colony Optimization, Grasp and Scatter Search [Blum and Roli, 2003 and Brito et al., 2004]. Many metaheuristics incorporate random decisions in the implementation of several elements of their basic structure such as: generation of initial solutions, neighbor selection, operator selection, state transition rules, etc. We will refer to this type of algorithms as randomized metaheuristic (RM) algorithms.

Given the random nature of RM algorithmss, their performance is analyzed conducting a series of computational experiments. In these experiments the behavior of the averages of the solution quality and of the algorithm execution time are observed. When the variation of these averages is high, a large number of experiments is required to determine a clear trend in the measurements carried out. In this work the problem of reducing this variation, and in consequence, the number of experiments required to analyze the performance of RM algorithms is approached.

In the following sections the problem description, the related works, the proposed method and the experimental results are presented.

2 Problem description

It is indicated in [McGeoch, 1992 and 2002, and Johnson, 2002] that for reducing the variance in computational experiments, variance reduction techniques [Wilson, 1984] must be incorporated. When the performance of RM algorithms is analyzed experimentally on a group of hard instances of an NP-complete problem, the time required to carry out the experimentation is very long. A reduction in the number of experiments can be the only alternative to successfully obtain enough evidence to support the conclusions of the analysis of algorithm performance.

The goal of a simulation study is to determine the value of certain amount θ related to a stochastic process. A simulation produces a variate x_1 whose expected value is θ. A second execution of the simulation produces a new variate x_2, independent of the previous one, whose average is also θ. This process continues until completing a total of n executions and producing n independent variates $x_1, x_2, ..., x_n$, with $\theta = E(x_i)$ and $\sigma^2 = \mathrm{Var}(x_i)$. In order to estimate the mean (θ) and variance (σ^2) of the population, the sample mean $\overline{x} = \Sigma_i\, x_i / n$ and the sample variance $s^2 = \Sigma_i\, ((x_i - \overline{x})^2) / (n-1)$ are used. In [Ross, 1999] it is indicated that for determining the goodness of the estimator $\overline{x}$ with respect to θ, its quadratic error average $E[(\overline{x} - \theta)^2] = \mathrm{Var}(\overline{x}) = \mathrm{Var}(\Sigma_i\, x_i / n) = (\Sigma_i\, \mathrm{Var}(x_i)) / n^2 = \sigma^2/n = s^2/n$ must be used. This expression has two consequences: (a) that it is possible to increase the quality of the estimation of θ by increasing the number of experiments and/or reducing the sample variance, and (b) that it is possible to reduce the number of experiments without decreasing the quality of the estimation by reducing the sample variance.

RM algorithms perform a search through the space of solutions making random decisions in order to avoid to get stuck at some local optima and to advance quickly toward the global optima. Typically, in each computational experiment or execution of an algorithm, an initial solution is randomly generated using a uniform distribution and a local search is carried out in the neighborhood of this solution to improve it. Once the possibility of improving is exhausted, a new experiment begins and the process is repeated. In each experiment i, the target output x_i is obtained. When all the experiments are carried out, the mean and the variance of x_i are calculated. Given the randomness of RM algorithms, their outputs depend on a set of uniform random numbers $\{u_i\}$ generated during the algorithm execution such that $x_i = x_i(u_1, u_2, \ldots, u_l)$. Some of these numbers are related to the decisions associated to the construction of the initial solution, and others to the decisions in the local search process. Unlike simulation, in an RM algorithm the amount of random decisions carried out in each experiment is variable; this constitutes the most important difficulty to apply the method of antithetic variates for reducing the variance in experiments with RM algorithms.

3 Related work

Unfortunately, there exist few works that address the problem of applying variance reduction techniques to the experimental analysis of algorithms. This problem was first introduced by Catherine McGeoch in 1992 [McGeoch, 1992]. Some of the most recent and relevant works related to the problem of incorporating the techniques of variance reduction in computational experiments with RM algorithms are described next.

McGeoch is one of the pioneers in the development of variance reduction methods in experiments with randomized algorithms. In [McGeoch, 1992] and [McGeoch, 2002] the use of common random numbers, antithetic and control variates techniques is proposed to reduce the variance in the experimental analysis of the performance of RM algorithms applied to the solution of the self-organized search problem. She states that, in the experimental analysis of algorithm performance, there exist many opportunities to apply variance reduction techniques, because the algorithms are simpler and have a more precise definition than simulation problems. The main limitation of this approach is that the proposed methods depends on the technique, the problem and the algorithm used.

In [Fraire, 2005] and [Fraire, et al., 2006] a method to reduce the variance in experiments with RM algorithms based on the antithetic variates technique is presented. The method is described in the context of the solution of the SAT problem with the Threshold Accepting metaheuristic algorithm. The method requires that the solutions of the problem be represented using binary vectors and that the metaheuristic algorithm generate an initial random solution.

As we can see the traditional approach consists of developing variance reduction methods that are dependent on the technique, the problem and the RM algorithm used. In this work a new approach is proposed, it consists of developing general-purpose methods that allow incorporating techniques of variance reduction, which are independent of the problem and the metaheuristic approach used. In order to validate the proposed approach, a general-purpose method based on the technique of antithetic variates is described and it is applied to different problems and RM algorithms.

4 Proposed method

4.1 Main idea

In order to develop a general-purpose variance reduction method, the basic idea consists of identifying the first (conditional independent) random decision that always occurs in each computational experiment or algorithm execution. This decision, which will be denominated base decision, should be associated to the generation of just one random number. Unlike the rest of the random numbers generated in the process, the random number associated to the base decision must be stored in the global variable $a \in (0,1)$, so that it will be available in the following experiment. Thus, before generating each number for the base decision, we must check if the number of the experiment currently carried out is even or odd. If the number of the experiment is odd, a new random number must be generated and stored in a, otherwise the random number currently stored in a must be complemented. The idea is to use only one of the random numbers generated in the algorithm process, to negatively correlate all the outputs of all experiments conducted.

4.2 Method description

The proposed method consists of the following four steps:

Step 1. Identify the base decision.
Analyze the sequence of random decisions that occurs in each computational experiment. Determine all the decisions that do not necessarily occur in each experiment, which happens when a conditional expression of the algorithm evaluates to true. Identify those that always occur in the experiments, their occurrence is non-conditional dependent. From the set of decisions that consistently occur, select the first one and consider it as the base decision.

Step 2. Generate the random number associated to the base decision.
Let a and c be the global variables in which the random number associated to the base decision and the current experiment number will be respectively stored. Now, before carrying out the base decision, if c is odd a new random number must be generated and stored (a = random()), otherwise the complement of a must be

stored (a=1–a). Upon completing the previous process, variable a is used to carry out the base decision. Now the pair of random numbers associated to the base decision in successive experiments (odd and even) will be negatively correlated.

Step 3. Determine the outputs of each experiment.
Execute all the experiments and record the values of the specified output t_i, for each experiment i = 1, 2, ..., N_{exp}.

Step 4. Determine the sample mean and the sample variance.
The values of the variates x, y, and z are determined, using the following expressions:

$$x_j(a) = t_{2j-1}, \quad y_j(1-a) = t_{2j} \tag{1}$$

and

$$z_j = (x_j(a) + y_j(1-a))/2 \tag{2}$$

$$j = 1, 2, \ldots, N_{exp}/2.$$

Calculate the mean and the variance of z.

By definition, t_i, x_j, y_j, and z_j are estimators of the expected value of variate t, and since $z = (x+y)/2$, then $\mathrm{Var}(z) = \mathrm{Var}((x+y)/2) = ¼\ [\mathrm{Var}(x) + \mathrm{Var}(y) + 2\mathrm{Cov}(x, y)]$. Now, since x and y were generated from negatively correlated inputs, our main assumption is that $\mathrm{Cov}(x, y) < 0$. If this happens, the method should generate a variance reduction for estimator z_i, with respect to the same estimator when this is generated from non-negatively correlated inputs. If this method is not applied, the sequence of random numbers associated to the base decision of the even and odd experiments might or might not be negatively correlated. If they are not negatively correlated, the expected variance should be larger than the one obtained with the proposed method. Conversely, if they are negatively correlated, then the variance of the estimator z_i could be smaller or larger than the variance obtained when the proposed method is applied. Therefore, if the assumption is correct it is expected that variance reduction occurs at least in 50% of the cases considered.

In this work, experimental evidence is presented, which shows that the impact of the application of the proposed method on the variance reduction is significant and that can be used with any combination of a RM algorithm and an NP-complete problem. The method is applicable to any kind of randomized algorithm; nevertheless in this work only the results of experiments conducted with RM algorithms are shown.

4.3 Example

In this example the SAT problem instance f600 is used [SATLIB]. The instance was solved executing 30 times the Threshold Accepting metaheuristic algorithm reported in [Pérez, 1999]. This algorithm starts the navigation process from an initial feasible solution, which is randomly generated in each run of the algorithm. In this case the assignment of values to each of the components of the initial solution constitute random decisions that are carried out unconditionally in each experiment. Therefore, any of these decisions can be used as base decision. In this case the base decision was decided to be the one associated to the determination of the value of the first component of the initial solution. Once the base decision was identified, the remaining steps of the proposed approach were applied. As a result of step 2, in the odd experiments, the first component of the initial solution is randomly generated and stored; while in the even experiments the value of such component is determined by complementing the random number that was generated in the previous experiment. Step 3 consists of recording the random outputs of the experiment, in this case the output recorded was the execution time of each experiment t_i, for i = 1, 2, ..., N_{exp}. In step 4 the set of the random outputs of the experiments are divided into two sets: one for holding the outputs generated from the independent random numbers (odd experiments) and the other for holding those generated from the complements of those numbers (even experiments). The outputs from the first set correspond to variate x, and those from the second correspond to variate y. Finally, the values of variate z are calculated (average of variates x and y) and its variance is calculated.

Table 1 shows the execution time t_i observed for each experiment i = 1, 2, …, 30, when the proposed method is not applied (all the components of the initial solution are randomly generated in each experiment). In this case, both the mean and variance of variate t must be directly calculated. However, variates x, y, and z will be calculated to show the effects produced by the application of the method.

Table 2 shows the x_j, y_j, and z_j values obtained without applying the method. As we can see, the covariance of x and y is 0.0035 and the variance of z is 0.0142.

Table 3 shows the execution time t_i observed when the proposed method is applied. In this case, both the mean and variance of variate t are calculated using the proposed method.

Table 4 shows the x_j, y_j, and z_j values obtained when the proposed method is applied. In this case, the covariance of x and y is –0.00052 and the variance of z is 0.00088. In this case the method produces a variance reduction of 37%.

5 Experimental results

A set of experiments were conducted to evaluate the variance reduction level when the proposed method is applied in experiments with RM algorithms. A set of problem instances were used, and each one was solved with two implementations. For each algorithm an implementation incorporates the proposed method and the other not. Thirty experiments were carried out with each instance and the specified outputs were measured. Finally, the variances before and after the application of the method were used to determine the variance reduction level.

To describe a typical set of experimental results, three test cases will be used, which include three problems and three RM algorithms. A test case includes the SAT problem and the GRASP [González-Velarde, 1996] and the Tabu Search [Glover and Laguna, 1997] metaheuristic algorithms, another includes the Lennard-Jones problem and two variants of the genetic algorithm described in [Romero, et al., 1999], and the last test case includes the hot rolling scheduling problem and a genetic algorithm [Espriella, 2008].

Table 5 shows the variance reduction level of the average solution quality Avg(%E), obtained when the instances were solved with the GRASP metaheuristic algorithm. The first column indicates the instances used in the experiments. Columns VB and VA show the variance obtained before and after the method application. Column %R shows the variance reduction percentage. Also, the table includes the overall average of the reduction level reached.

Table 6 shows the variance reduction level of the average solution quality Avg(%E), obtained with the Tabu Search metaheuristic algorithm. The structure of Table 6 is similar to that of Table 5.

Table 7 shows the variance reduction level observed in three random outputs. In this test case 15 instances of the Lennard Jones problem were solved with a genetic algorithm (v1). The random outputs were the average error percentage Avg(%E), the average number of evaluations of the objective function Avg(NEOF) and the average number of generations Avg(NG). The first column indicates the instances used in the experiments, and the additional columns are grouped in three subtables, each one with three columns. The structure of the sub-tables is similar to that of Table 5.

Table 8 has the same structure as that of Table 7, and contains the reduction level observed when a variant of the genetic algorithm (v2) was used.

Table 9 shows the variance reduction level observed in two random outputs. In this test case 17 industrial instances of the hot rolling scheduling problem were solved with a genetic algorithm. The random outputs were the average rolling time Avg(%RT) and the average of the time required to obtain the best solution Avg(TB). The first column indicates the instances used in the experiments, and the additional columns are grouped in two sub-tables, each one with three columns. The structure of the sub-tables is similar to that of Table 5.

The experimental evidence shows that the proposed method reduces the variance for 29 out of 37 instances considered, independently of the problem and the solution method employed. This means that in 78% of the cases considered a variance reduction is observed, which is consistent with the forecast established as a consequence of the assumption expressed.

As can be seen, the proposed method can reduce the variance of the random outputs for most of the instances considered and the overall reduction level lies in the range from 14% to 55%. The investment of resources required to obtain this reduction level is minimum, since it only requires controlling one of the random numbers generated by the algorithm. An outstanding observed behavior of the method is that it simultaneously produces a reduction in the variance of all the random outputs of the algorithm. In Tables 3, 4 and 8 we can see that the method operates simultaneously on several outputs. As we can see the proposed method allows to incorporate a variance reduction technique in computational experiments, independently of the problem solved and of the RM algorithm used.

Table 1. Execution times obtained without applying the proposed method

i	t_i	***I***	t_i	***i***	t_i	***I***	t_i	***I***	t_i	***i***	t_i
1	0.704	6	0.906	11	0.828	16	1.125	21	1.015	26	1.110
2	0.718	7	0.657	12	0.907	17	0.906	22	0.953	27	0.625
3	1.188	8	0.937	13	0.890	18	1.063	23	0.719	28	0.781
4	0.890	9	0.656	14	0.688	19	0.797	24	0.890	29	0.844
5	0.641	10	0.797	15	0.734	20	1.156	25	0.922	30	0.687

Table 2. Values of *x, y,* and *z* obtained without applying the proposed method

j	x_i	y_i	z_i	***J***	x_i	y_i	z_i
1	0.704	0.718	0.711	9	0.906	1.063	0.984
2	1.188	0.890	1.039	10	0.797	1.156	0.976
3	0.641	0.906	0.773	11	1.015	0.953	0.984
4	0.657	0.937	0.797	12	0.719	0.890	0.804
5	0.656	0.797	0.726	13	0.922	1.110	1.016
6	0.828	0.907	0.867	14	0.625	0.781	0.703
7	0.890	0.688	0.789	15	0.844	0.687	0.765
8	0.734	1.125	0.929				

Table 3. Execution times obtained applying the proposed method

i	t_i	***i***	t_i	t_i	t_i	***I***	t_i	***I***	t_i	***i***	t_i
1	0.704	6	0.641	11	0.688	16	0.922	21	0.922	26	0.703
2	0.687	7	0.922	12	0.734	17	0.782	22	1.000	27	0.797
3	1.000	8	0.812	13	0.907	18	0.843	23	0.875	28	0.891
4	0.750	9	0.641	14	0.640	19	0.844	24	0.860	29	0.625
5	0.734	10	1.109	15	1.031	20	0.609	25	0.937	30	0.781

Table 4. Values of *x, y,* and *z* obtained when the proposed method is applied

j	x_j	y_j	z_j	***J***	x_j	y_j	z_j
1	0.704	0.687	0.695	9	0.782	0.843	0.812
2	1.000	0.750	0.875	10	0.844	0.609	0.726
3	0.734	0.641	0.687	11	0.922	1.000	0.961
4	0.922	0.812	0.867	12	0.875	0.860	0.867
5	0.641	1.109	0.875	13	0.937	0.703	0.820
6	0.688	0.734	0.711	14	0.797	0.891	0.844
7	0.907	0.640	0.773	15	0.625	0.781	0.703
8	1.031	0.922	0.9765				

Table 5. Variance reduction level observed with the SAT problem and a GRASP algorithm

	Avg(%E)		
Instance	VB	VA	%R
uf200-0100	7.40221	1.809563	76%
uf250-01	3.68832	2.1144068	43%
uf250-010	4.80004	2.8571141	40%
f600	9.0066	10.409657	-16%
f1000	24.0414	9.4095563	61%
Overall average reduction			41%

Table 6. Variance reduction level observed with the SAT problem and a tabu search algorithm

	Avg(%E)		
Instance	VB	VA	%R
uf200-0100	9.13128	4.12374	55%
uf250-01	15.0622	7.20976	52%
uf250-010	11.3515	3.35256	70%
F600	164.599	101.353	38%
F1000	256.397	101.923	60%
Overall average reduction			55%

Table 7. Variance reduction observed with the Lennard Jones problem and a genetic algorithm (v1)

	Avg(%E)			Avg(NEOF)			Avg(NG)		
I	VB	VA	%R	VB	VA	%R	VB	VA	%R
15	0.05	0.03	40%	61,407	15,761	74%	1.08	0.54	50%
16	0.05	0.07	-40%	139,689	42,153	70%	3.46	1.27	63%
17	0	0	0%	9,760	3,789	61%	0.03	0.03	0%
18	0.06	0.02	67%	250,824	142,806	43%	5.1	3.31	35%
19	0.78	0.24	69%	294,092	210,221	29%	6.16	2.99	51%
20	0.49	0.29	41%	280,123	208,500	26%	4.64	2.36	49%
21	0.45	0.2	56%	411,777	160,780	61%	6.88	2.55	63%
22	0.82	0.28	66%	462,982	156,176	66%	6.22	1.98	68%
23	0.97	0.51	47%	338,332	285,224	16%	4.31	3.94	9%
24	0.46	0.16	65%	462,886	378,918	18%	5.79	5.37	7%
25	0.26	0.15	42%	198,652	357,526	-80%	3.21	5.3	-65%
26	0.81	0.19	77%	688,233	228,416	67%	8.72	2.62	70%
27	0.81	0.23	72%	535,909	129,526	76%	6.38	2.43	62%
28	0.64	0.27	58%	403,537	194,353	52%	4.99	3.04	39%
29	0.4	0.42	-5%	495,611	344,752	30%	4.88	3.88	20%
30	0.41	0.24	41%	1,341,653	112,631	92%	12.74	1.14	91%
Overall reduction (Avg)			43%			38%			38%

Table 8. Variance reduction observed with the Lennard Jones problem and a genetic algorithm (v2)

I	Avg(%E)			Avg(NEOF)			Avg(NG)		
	VB	VA	%R	VB	VA	%R	VB	BA	%R
15	0.08	0	100%	175,755	71,702	59%	4.2	2.5	40%
16	0.03	0	100%	37,722	26,511	81%	4.2	0.83	80%
17	0	0	0%	4,647	2,253	52%	0	0	0%
18	0.11	0.04	64%	679,707	512,567	25%	14.7	10.8	27%
19	0.55	0.22	60%	953,268	397,323	58%	18.39	6.59	64%
20	0.68	0.21	69%	1,758,666	1,021,182	42%	28.48	16.7	41%

21	0.45	0.25	44%	916,672	997,527	-9%	15.93	13	19%
22	0.62	0.24	61%	1,547,998	750,757	52%	25.79	11.2	57%
23	0.33	0.25	24%	2,357,355	986,281	58%	29.86	13.1	56%
24	0.38	0.28	26%	873,663	1,758,849	-101%	11.43	26.8	-134%
25	0.29	0.1	66%	1,777,160	391,122	78%	21.61	6.85	68%
26	0.48	0.18	63%	2,334,290	1,020,252	56%	34.12	13.7	60%
27	0.54	0.39	28%	3,261,303	1,257,075	61%	44.78	14	69%
28	0.62	0.28	55%	2,197,110	597,272	73%	26.33	7.46	72%
29	0.49	0.22	55%	2,885,274	716,521	75%	31.06	8.7	72%
30	0.27	0.26	4%	4,187,946	722,313	83%	48.33	6.98	86%
Overall reduction (avg)			51%			42%			42%

Table 9. Variance reduction observed with the hot rolling scheduling problem and a genetic algorithm

	Avg(RT)			**Avg(TB)**		
Instance	VB	VA	%R	VB	VA	%R
hsm002.txt	0.000229	0.000188	17.90%	28.619	34.049	-18.97%
hsm003.txt	0.000186	0.000115	38.17%	3.591	2.812	21.69%
hsm004.txt	0.000180	0.000143	20.56%	4.901	2.275	53.58%
hsm005.txt	0.000183	0.000149	18.58%	5.089	10.467	-105.68%
hsm006.txt	0.000394	0.000312	20.81%	4.230	10.849	-156.48%
hsm007.txt	0.000395	0.000293	25.82%	4.807	3.265	32.08%
hsm008.txt	0.000472	0.000317	32.84%	4.840	3.031	37.38%
hsm009.txt	0.000302	0.000332	-9.93%	4.027	3.794	5.79%
hsm010.txt	0.000288	0.000196	31.94%	5.606	2.063	63.20%
hsm011.txt	0.000299	0.000161	46.15%	2.586	2.298	11.14%
hsm012.txt	0.000230	0.000154	33.04%	3.951	2.441	38.22%
hsm013.txt	0.000217	0.000129	40.55%	2.991	1.954	34.67%
hsm014.txt	0.000840	0.000626	25.48%	4.187	1.907	54.45%
hsm015.txt	0.000846	0.000624	26.24%	3.648	1.861	48.99%
hsm016.txt	0.000835	0.000661	20.84%	4.114	2.102	48.91%
hsm017.txt	0.000922	0.000494	46.42%	3.148	1.957	37.83%
hyl001.txt	0.001740	0.000503	71.09%	4.441	2.823	36.43%
Overall reduction (avg)			29.79%			14.31%

6 Conclusions and future work

In this paper the problem of how to reduce the number of required experiments to analyze the performance of RM algorithms was approached. Traditional approaches propose to apply variance reduction methods that depend on the technique, the problem and the randomized metaheuristic (RM) algorithm used. The solution approach proposed in this work consists of developing reduction methods which are independent of the problem and of the RM algorithm used. In order to validate the proposed approach, a general-purpose method based on the antithetic variates technique is introduced and it was applied to three NP-complete problems and three RM algorithms. The experimental evidence shows that the proposed method yields a variance reduction in 78% of the cases considered. Additionally, the method shows the capacity of simultaneously reducing the variance of several random outputs of the algorithms tested. The overall reduction levels on the instances used in the test cases lie in the range from 14% to 55%.

The proposed method is one of the first general-purpose methods that allows to incorporate a variance reduction technique in computational experiments, independently of the problem solved and of the RM algorithm used. The method is applicable without significant modifications to any type of randomized algorithm.

As a future work we are considering to develop new general purpose methods to apply other variance reduction techniques in the experimental algorithms analysis.

References

1 **Blum, C., & Roli, A. (2003)**. Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing, 35* (3), 268–308.

2 **Brito, J., Campos, C., García, F., García, M., Melián, B. Moreno, J. & Moreno, M. (2004)**, Metaheuristicas: una revisión actualizada (informe técnico, no. 2/2004. España: Universidad de la Laguna, Departamento de Estadística, Investigación Operativa y Computación.

3 **Espriella, F. K. (2008).** Optimización Mediante Algoritmos Genéticos: Aplicación a la Laminación en Caliente. *Tesis de maestría*, Instituto Politécnico Nacional, CICATA Unidad Altamira, Tamaulipas, México.

4 **Fraire, H. J. (2005).** Una Metodología Para el Diseño de la Fragmentación y Ubicación en Grandes Bases de Datos Distribuidas. *Tesis doctoral*, CENIDET, Cuernavaca, Morelos, México.

5 **Fraire, H. J., Castilla, V. G., Hernández, R. A., Aguilar, L. S., Castillo, V. G., and Camacho, A. C. (2006).** Reducción de la varianza en la evaluación experimental de algoritmos metaheurísticos. *Proceedings 13th International Congress on Computer Science Research CIICC06*. Tampico, Tamaulipas, México, 14-22.

6 **Garey, M.R., and Johnson, D.S. (1979).** Computer and intractability: a Guide to the Theory of NP-Completeness. New York: WH Freeman.

7 **Glover, F. (1986).** Future Paths for Integer Programming and Links to Artificial Intelligence. *Computers and Operations Research*, 13, 533-549.

8 **Glover, F., Laguna, M. (1997).** Tabu Search. Boston, MA, USA: Kluwer Academic Publishers.

9 **González-Velarde, J.L. (1996).** GRASP. In A. Díaz, (ed.) *Heuristic Optimization and Neural Networks in Operations Management and Engineering*. (143-161). Madrid: Editorial Paraninfo.

10 **Johnson, D.S. (2002).** Theoretician's Guide to the Experimental Analysis of Algorithms. In Goldwasser, M. H., Johnson, D. S., & McGeoch C. C. (eds.); *Data Structures, Near Neighbor Searches, and Methodology: Proceedings of the Fifth and Sixth DIMACS Implementation Challenges*, Providence, RI, USA, 59, 215-250.

11 **McGeoch, C.C. (1992).** Analyzing Algorithms by Simulation: Variance Reduction Techniques and Simulation Speedups. *ACM Computer Survey*. 24 (2), 195-212.

12 **McGeoch. C.C. (2002),** Experimental Analysis of Algorithms. In: Pardalos, P.M. & Romeijn, H.E. (eds.); Handbook of Global Optimization, 2 (489-513), Gainesville, FL, USA: University of Florida.

13 **Pérez, J. (1999).** Integración de la Fragmentación Vertical y Ubicación en el Diseño Adaptivo de Bases de Datos Distribuidas. *Tesis doctoral*, ITESM Cuernavaca, México.

14 **Papadimitriou, C., & Steiglitz, K. (1998).** Combinatorial Optimization: Algorithms and Complexity. New York, USA: Dover Publications.

15 **Reeves, C. R. (1993).** Modern Heuristic Techniques for Combinatorial Problems. Oxford: Blackwell Scientific Publishing.

16 **Romero, D., Barrón, & C., Gómez, S. (1999).** The Optimal Geometry of Lennard-Jones Clusters: 148-309. *Computational Physics Communications*. 123 (1-3), 87-96.

17 **Ross, S.M. (1999).** Simulación. (Segunda edición.) México: Prentice Hall.

18 **SATLIB.** Retrieved from:www.satlib.org

19 **Wilson, J.(1984).** Variance Reduction Techniques for Digital Simulation. *American Journal of Mathematical and Management Sciences*. 4 (3-4), 277-312.

Héctor Joaquín Fraire Huacuja*. He received the Ph.D. in computer science from the Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, México in 2005. His scientific interests include metaheuristic optimization and machine learning.*

Rodolfo A. Pazos Rangel*. He received the Ph.D. in computer science from the University of California at Los Angeles, USA in 1983. His scientific interests include distributed database design and natural language processing.*

Juan Javier González Barbosa*. He received the Ph.D. in computer science from the Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, México in 2006. His scientific interests include metaheuristic optimization and machine learning.*

Laura Cruz Reyes*. She received the Ph.D. in computer science from the Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, México in 2004. His scientific interests include metaheuristic optimization and machine learning.*

Guadalupe Castilla Valdez*. He received the M.D. in computer science from the Instituto Tecnológico de León, México in 2002. His scientific interests include metaheuristic optimization and machine learning.*

José Antonio Martínez Flores*. He received the Ph.D. in computer science from the Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, México in 2006. His scientific interests include distributed database design and natural language processing.*