

# Combining Genetic Algorithm with Variable Neighbourhood Search for MAX-SAT

Noureddine Bouhmala<sup>1</sup>, Kjell I. Øvergård<sup>2</sup>  
{noureddine.bouhmala@usn.no<sup>1</sup>, koe@usn.no<sup>2</sup>}

University College of Southeast Norway, Department of Southeast Norway,  
Postboks 4, 3199 Borre, Norway<sup>1,2</sup>

**Abstract.** Metaheuristics used to tackle different optimization problems that arise in various fields aim at finding a tactical interplay between diversification and intensification to overcome local optimality. In this paper, a genetic algorithm (GA) combined with variable neighbourhood search (VNS) is proposed for solving the maximum satisfiability problem. VNS systematically changes different neighbourhoods within a local search. The idea is that a local optimum reached within one neighbourhood structure is not necessarily the same local optimum of another neighbourhood structure, thus the search can systematically explore different search areas which are defined by different neighbourhood structures. Results comparing the genetic with and without VNS are presented.

**Keywords:** genetic algorithm, variable neighbourhood search, maximum satisfiability problem.

## 1 Introduction

The MAX-SAT problem which is known to be NP-complete [1] is defined as follows. Given a positive constant  $k$ , a propositional formula  $\Phi = \bigwedge_{j=1}^m C_j$  with  $M$  clauses and  $N$  Boolean variables. Each Boolean variable,  $x_i$ ,  $i \in \{1, \dots, n\}$ , takes one of the two values, TRUE or FALSE. Each clause  $C_j$ , in turn, is a disjunction of Boolean variables and has the form as shown in formula (1) where  $I_j, \bar{I}_j \subseteq \{1, \dots, n\}$ ,  $I \cap \bar{I} = \emptyset$ , and  $\bar{x}$  denotes the negation of  $x$ .

$$C_j = \left( \bigvee_{l \in I_j} x_l \right) \vee \left( \bigvee_{l \in \bar{I}_j} \bar{x}_l \right). \quad (1)$$

As a simple example (see formula 2) let  $\phi(x)$  be the following formula containing 4 variables and 3 clauses. The task is to determine whether or not there exist a truth assignment for  $\phi$  that satisfies the maximum number  $k$  of clauses.:

$$\phi(x) = (x_1 \vee \neg x_4) \wedge (\neg x_1 \vee x_3) \wedge (\neg x_1 \vee x_4 \vee x_2). \quad (2)$$

Several state-of-the-art local search algorithms are enhanced versions of WalkSAT

(WSAT) [2] algorithm. Examples include WalkSAT/Tabu [3], Novelty+ and R-Novelty+ heuristics [4], G2WSAT [5]. Evolutionary algorithms are heuristic algorithms that have been applied to MAX-SAT and many other NP-complete problems. Unlike local search methods that work on a current single solution, evolutionary approaches evolve a set of solutions. GASAT [6] is considered to be the best known genetic algorithm for MAX-SAT. GASAT is a hybrid algorithm that combines a specific crossover and a tabu search procedure. Experiments have shown that GASAT provides very competitive results compared with state-of-art MAX-SAT algorithms.

## 2 Combining GA with VNS (VNS-GA)

Variable Neighbourhood Search (VNS) [7] is a relatively recent meta-heuristic for solving combinatorial and global optimization problems. Unlike many standard metaheuristics where only a single neighbourhood is employed, VNS systematically changes different neighbourhoods within a local search. The idea is that a local optimum reached within one neighbourhood structure is not necessarily the same local optimum of another neighbourhood structure, thus the search can systematically explore different search areas which are defined by different neighbourhood structures.

Genetic Algorithms [8] are stochastic methods for global search and optimization and belong to the group of evolutionary algorithms. They simultaneously examine and manipulate a set of possible solutions. A gene is part of a chromosome, which is the smallest unit of genetic information. Every gene is able to assume different values called allele. All genes of an organism form a genome which affects the appearance of an organism called phenotype. The chromosomes are encoded using a chosen representation and each can be thought of as a point in the search space of candidate solutions. Each individual is assigned a score (fitness) value that allows assessing its quality. The members of the initial population may be randomly generated or by using sophisticated mechanisms by means of which an initial population of high quality chromosomes is produced. The reproduction operator selects (randomly or based on the individual's fitness) chromosomes from the population to be parents and enters them in a mating pool. Parent individuals are drawn from the mating pool and combined so that information is exchanged and passed to off-springs depending on the probability of the cross-over operator. The new population is then subjected to mutation and enters into an intermediate population. The mutation operator acts as an element of diversity into the population and is generally applied with a low probability to avoid disrupting cross-over results. Finally, a selection scheme is used to update the population giving rise to a new generation. The individuals from the set of solutions which is called population will evolve from generation to generation by repeated applications of an evaluation procedure that is based on genetic operators. Over many generations, the population becomes increasingly uniform until it ultimately converges to optimal or near-optimal solutions. The proposed VNS-GA is described in section 2.1 to 2.9.

### 2.1 Neighbourhood selection

Let  $L$  denotes the set of variables of the MAX-SAT problem to be solved. The first phase of the algorithm consists in constructing a set of neighbourhoods satisfying the following property:  $N_1(x) \subset N_2(x) \subset \dots \subset N_{k_{max}}(x)$ . The starting (default) neighbourhood with  $k = 1$  consists of a move based on the flip of a single variable. A flip means assigning the opposite state to a variable (*i.e.*, change True  $\rightarrow$  False or False  $\rightarrow$  True). The first neighbourhood  $N_2$  is

constructed from  $L$  by merging variables. The merging procedure is computed using a randomized algorithm. The variables are visited in a random order. If a variable  $v_i$  has not been matched yet, then a randomly unmatched variable  $v_j$  is selected, and a cluster  $v_k$  consisting of the two variables  $l_i$  and  $l_j$  is created. The set  $N_2$  consists of moves based on flipping predefined clusters each having  $2^1$  variables. The new formed clusters are used to define a new and larger neighbourhood  $N_3$  and recursively iterate the process until the desired number of neighbourhood ( $k_{max}$ ) is reached.

## 2.2 Initial population

The chromosomes which are assignments of values to the variables are encoded as strings of bits, the length of which is the number of variables. The values are represented by 1 and 0 respectively. In this representation, a chromosome  $X$  corresponds to a truth assignment and the search space is the set  $S = \{0, 1\}^n$ . A random initial population is generated from the largest neighbourhood ( $N_{k_{max}}$ ). The random initial population consists in assigning to each chromosome of the population a random value (True or False) to each cluster and all the variables within that cluster will get the same value.

## 2.3 Fitness function

The notion of fitness is fundamental to the application of genetic algorithms. It is a numerical value that expresses the performance of an individual (solution) so that different individuals can be compared. The fitness of a chromosome (individual) in the population is equal to the number of clauses that are unsatisfied by the truth assignment represented by the chromosome.

## 2.4 What neighbourhood to start from

The most crucial part when applying VNS is the selection of the different neighbourhoods according to some strategy for the effectiveness of the search process. The strategy adopted in this work is to let VNS-GA start the search process from the largest neighbourhood  $N_{k_{max}}$  and continues to move towards smaller neighbourhood structures. The motivation behind this strategy is that the order in which the neighbourhood structures have been selected offers a better mechanism for performing diversification and intensification.

The largest neighbourhood  $N_{max}$  allows GA to view any cluster of variables as a single entity leading the search to become guided in faraway regions of the solution space and restricted to only those configurations in the solution space in which the variables grouped within a cluster are assigned the same value. As the switch from one neighbourhood to another implies a decrease in the size of the neighbourhood, the search is intensified around solutions from previous neighbourhoods in order to reach better ones.

## 2.5 Matching process

New solutions are created by combining pairs of individuals in the population and then applying a crossover operator to each chosen pair. Combining pairs of individuals can be viewed as a matching process. In the version of GA used in this work, the individuals are

visited in random order. An unmatched individual  $i_k$  is matched randomly with an unmatched individual  $i_j$ .

## 2.6 Cross-over operator

The task of the crossover operator is to reach regions of the search space with higher average quality. The two-point crossover operator is applied to each matched pair of individuals. The two-point crossover selects two randomly points within a chromosome and then interchanges the two parent chromosomes between these points to generate two new offspring. Recombination can be defined as a process in which a set of configurations (solutions referred as parents) undergoes a transformation to create a set of configurations (referred as offspring). The creation of these descendants involves the location and combinations of features extracted from the parents. Examples of this process can be seen in Tables 1 and 2.

**Table 1.** Two parents before the cross-over operator.

x1:0	x2:0	x3:1	↓ x4 : 0	↓ x5 : 1	↓ x6 : 0	↓ x7 : 1	x8:1	x9:0	x10
x1:0	x2:1	x3:0	↑ x4 : 1	↑ x5 : 0	↑ x6 : 1	↑ x7 : 0	x8:0	x9:1	x10

**Table 2.** Results of the cross-over operator.

x1:0	x2:0	x3:1	↓ x4 : 1	↓ x5 : 0	↓ x6 : 1	↓ x7 : 0	x8:1	x9:0	x10
x1:0	x2:1	x3:0	↑ x4 : 0	↑ x5 : 1	↑ x6 : 0	↑ x7 : 1	x8:0	x9:1	x10

## 2.7 Mutation

The purpose of mutation is to generate modified individuals by introducing new features in the population. By mutation, the alleles of the produced child have a chance to be modified, which enables further exploration of the search space. The mutation operator takes a single parameter  $p_m$ , which specifies the probability of performing a possible mutation. Let  $C = c_1, c_2, \dots, c_m$ , be a chromosome represented by a binary chain where each of whose gene  $c_i$  is either 0 or 1. In our mutation operator, each gene  $c_i$  is mutated through flipping this gene's allele from 0 to 1 or vice versa if the probability test is passed. In case of a large neighbourhood ( $k > 0$ ), the mutation is applied to a cluster of variables.

## 2.8 Selection

The selection acts on individuals in the current population. Based on each individual quality (fitness), it determines the next population. In the roulette method, the selection is stochastic and biased toward the best individuals. The first step is to calculate the cumulative fitness of the whole population through the sum of the fitness of all individuals. After that, the probability of selection is calculated for each individual as being  $P_{selection_i} = f_i / \sum_1^N f_i$ , where  $f_i$  is the fitness of individual  $i$ .

## 2.9 Switch to a smaller Neighbourhood

Once GA has reached the convergence criterion with respect to a neighbourhood  $N_j$ . It switches to another neighbourhood and the assignment reached on the neighbourhood  $N_i$  must be projected on its parent neighbourhood  $N_{i-1}$ . The projection algorithm is simple; if a cluster  $c_j \in N_m$  is assigned the value of true then the merged pair of clusters that it represents,  $c_j, c_k \in N_{m-1}$  are also assigned the true value. Finally, GA is applied at the default neighbourhood.

## 3 Experimental results

### 3.1 Test suite and parameter settings

The performance of VNS-GA is evaluated against GA using a set of random problems (MAX-2SAT, MAX-3SAT). This set is taken from the Ninth MAX- SAT 2014 evaluation (<http://maxsat.ia.udl.cat/benchmarks/>) organized as an affiliated event of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT 2014). Due to the randomization nature of both algorithms, each problem instance was run 50 times with a cut-off parameter (max-time) set to 15 minutes. The tests were carried out on a DELL machine with 800 MHz CPU and 2 GB of memory. The code was written in C++ and compiled with the GNU C compiler version 4.6. The following parameters have been fixed experimentally and are listed below:

$k_{max}$ : The cardinality of the neighbourhood is set such that the number of the formed clusters is 10% of the size of the problem instance (*i.e.*, a problem with 100 problems will lead to  $k_{max}$  equals to 3).

GA is assumed to have reached convergence and switch to a smaller neighbourhood if the fitness of the fittest chromosome remains unchanged during five consecutive generations.

### 3.2 Statistical analysis

The results showing the statistical comparison between the two algorithms are presented in Tables 3 and 4. The mean ( $\bar{x}$ ) and the standard deviation ( $\sigma$ ) of unsolved clauses for VNS-GA and GA algorithms are reported. The range of solutions from each algorithm is also presented in order to show the overlap between solution spaces for any given instance.

Statistical inferential analysis was done with an independent samples t-test which compares the difference in means between the two groups. Comparison using the non-parametric Mann-Whitney U-test gave identical results. The non-parametric effect size measure Probability of Superiority (PS) [9] was used to evaluate the relative dominance of one algorithm over the other. The PS effect size measure is also called  $\hat{A}_{12}$  [10] and is calculated using the rank sum which is a common component in any non-parametric analysis such as the Mann-Whitney U-test [11].

Calculating PS is done according to formula (3) [9, 10, 11], where  $R_1$  is the rank sum of algorithm VNS-GA,  $m$  is the number of observations in the first data sample, and  $n$  is the number of observations in the second data sample.

$$PS = (R_1/m - (m + 1)/2)/n . \quad (3)$$

As can be seen in Table 3 and 4, the VNS-GA algorithm dominates GA for all MAX2SAT instances with less than 1300 clauses. However, for one instance (s2v140c1300-2) GA shows better performance, although the best solution (171 unsolved clauses) is identical for both algorithms. For the five remaining instances, there is no statistically significant difference between the two algorithms.

**Table 3.** Statistical comparison of the solutions for 2SAT-instances given by VNS-GA and GA.

Instance	GA		VNS-GA		$\Delta$ [95%CI of $\Delta$ ]	p	PS	[95% CI of PS]
	x ( $\sigma$ )	Min-Max	x ( $\sigma$ )	Min-Max				
s2v100c1200-1	210.6 (6.2)	192-229	171.5 (3.4)	169-180	39.2 [37.3, 41]	***	1	c
s2v100c1200-2	203.1 (6.3)	186-217	163.9 (1.3)	163-169	39.3 [37.6, 41]	***	1	c
s2v100c1200-3	212.8 (5.5)	199-226	173.7 (2.7)	172-182	39 [37.4, 40.6]	***	1	c
s2v100c1200-4	207.2 (6.2)	191-223	171 (2.2)	169-178	36.2 [34.5, 37.9]	***	1	c
s2v100c1200-5	207.2 (6.2)	191-223	171 (2.2)	169-178	41 [39, 42.9]	***	1	c
s2v100c1200-6	195.5 (7.3)	179-215	154.5 (1.8)	153-160	36.1 [34.6, 37.7]	***	1	c
s2v100c1200-7	204 (5.9)	188-221	167.8 (0.9)	167-172	35.4 [33.8, 37.1]	***	1	c
s2v100c1200-8	207.9 (5.9)	192-222	172.5 (2.3)	171-181	32.8 [31.3, 34.3]	***	1	c
s2v100c1200-10	206.7 (6.7)	184-224	163.5 (3.2)	162-176	43.2 [41.2, 45.1]	***	1	c
s2v120c1200-1	203.2 (5.4)	185-217	163 (2.5)	161-171	40.2 [38.6, 41.7]	***	1	c
s2v120c1200-2	200.1 (5.7)	184-212	161.2 (1.9)	159-167	38.9 [37.4, 40.5]	***	1	c
s2v120c1200-3	204.6 (4.8)	192-217	162.8 (4.7)	160-176	41.8 [40.1, 43.6]	***	1	c
s2v120c1200-4	201.2 (6.3)	185-219	158.2 (1.5)	157-164	43 [41.3, 44.7]	***	1	c
s2v120c1200-5	188.1 (6.7)	172-211	144.2 (1.1)	143-148	43.8 [42.1, 45.6]	***	1	c
s2v120c1200-6	207 (5.6)	194-225	170.2 (2.2)	167-176	36.9 [35.3, 38.4]	***	1	c
s2v120c1200-7	201.2 (5.3)	188-215	164.2 (2.7)	162-170	36.9 [35.4, 38.5]	***	1	c
s2v120c1200-8	203.6 (5.7)	192-224	167.4 (2.1)	165-173	36.2 [34.6, 37.8]	***	1	c
s2v120c1200-9	195 (6.2)	178-209	148.4 (0.7)	148-151	46.5 [44.9, 48.2]	***	1	c
s2v120c1200-10	196.6 (4.8)	186-207	154.3 (1.4)	154-165	42.3 [41, 43.6]	***	1	c
s2v140c1300-1	163.8 (1.8)	162-169	163.6 (1.7)	162-168	0.2 [-0.5, 0.8]	.526	.528	[.456, .607]
s2v140c1300-2	171.9 (1.5)	171-179	172.6 (2.2)	171-181	-0.7 [-1.4, 0]	.014*	.422	[.351, .494]
s2v140c1400-1	183.7 (2.2)	182-192	184.1 (2.3)	182-197	-0.4 [-1.3, 0.4]	.176	.431	[.356, .509]
s2v140c1400-2	179 (2.9)	178-190	179 (3.1)	178-192	0[-1.1, 1.1]	.944	.523	[.465, .579]
s2v140c1500-1	205 (0.7)	205-208	205.6 (1)	205-211	0[-0.3, 0.3]	.844	.534	[.465, .601]
s2v140c1500-2	201 (1.7)	199-207	200.9 (1.7)	199-208	0 [-0.6, 0.7]	.901	.496	[.418, .577]

**Table 3:** Notes:  $\Delta$  = Mean difference, CI = Confidence Interval, p = p-value, PS = Probability of Superiority (i.e. that VNS-GA will have less unsolved clauses than GA for each instance), \*\*\* =  $p < .001$ . \*\* =  $p < .01$ , \* $p < .05$ , c = not possible to calculate CI of PS due to no overlap between VNS-GA and GA. The 95% CI for PS is calculated in Excel using a bootstrapping procedure (random selection with replacement) performed 1000 times. Notes:  $x$  = Mean,  $\sigma$  = Standard Deviation, Min = Minimum observed value, Max = Maximum observed value.

VNS-GA dominates GA on all MAX3SAT instances with 70 variables (s3v70c1000-1 to s3v70c1000-10). GA is significantly better on one instance (s3v80c1000-2) with no statistical difference between the remaining algorithms with 80 variables (s3v80c1000-1 to s3v80-1000-10). When GA is significantly better the difference with respect to best solutions is marginal or non-existent. VNS-GA has a much lower variation of results across the test suite and we take this to mean that the variable neighbourhood scheme stabilizes GA in some way.

**Table 4:** Statistical comparison of the solutions for 3SAT instances given by VNS-GA and GA.

Instance	GA		VNS-GA		$\Delta$ [95% CI of $\Delta$ ]	p	PS	[95% CI of PS]
	x ( $\sigma$ )	Min-Max	x ( $\sigma$ )	Min-Max				
s3v70c1000-1	71.6 (3.5)	65-80	52 (2.6)	47-59	19.6 [18.4, 20.7]	***	1	c
s3v70c1000-2	69.1 (4.3)	57-80	48.2 (4.3)	43-60	20.9 [19.3, 22.5]	***	.999	[.998, 1]
s3v70c1000-3	71.1 (4)	62-80	51 (3.6)	45-60	20.2 [18.8, 21.5]	***	1	c
s3v70c1000-4	72.2 (3.8)	63-81	52.9 (3)	47-61	19.3 [18, 20.6]	***	1	c
s3v70c1000-5	67.8 (4.1)	56-77	45 (2.9)	42-52	22.8 [21.4, 24.1]	***	1	c
s3v70c1000-6	71.9 (3.6)	63-81	54.4 (2.3)	51-61	17.6 [16.4, 18.7]	***	1	c
s3v70c1000-7	70.9 (2.9)	62-78	53.7 (2.1)	49-58	17.2 [16.2, 18.1]	***	1	c
s3v70c1000-8	69.5 (3.8)	60-80	51.8 (2.9)	48-60	17.6 [16.4, 18.9]	***	1	[.998, 1]
s3v70c1000-9	70.9 (3.9)	60-83	52.7 (2.6)	49-61	18.2 [17, 19.4]	***	1	[.998, 1]
s3v70c1000-10	69.7 (4.4)	61-86	48.8 (3)	45-61	20.9 [19.5, 22.3]	***	1	[.998, 1]
s3v80c1000-1	47.8 (2.9)	44-55	47.6 (3.2)	44-57	0.2 [-0.9, 1.3]	.708	.528	[.451, .611]
s3v80c1000-2	46.4 (2.1)	43-52	47.5 (2.7)	43-55	-1.1 [-2, -0.2]	.002**	.383	[.307, .458]
s3v80c1000-3	44.7 (4)	39-57	45.3 (4)	39-54	-0.6 [-2.1, 0.9]	.279	.452	[.372, .530]
s3v80c1000-4	49.3 (2.6)	45-57	50 (2.9)	45-60	-0.8 [-1.8, 0.3]	.055	.417	[.344, .494]
s3v80c1000-5	45.4 (3.1)	41-53	45.4 (3.4)	41-55	0.1 [-1.1, 1.3]	.879	.514	[.438, .590]
s3v80c1000-6	44.8 (3.5)	40-57	44.4 (3.3)	40-52	0.4 [-0.9, 1.6]	.42	.527	[.447, .606]
s3v80c1000-7	43.2 (3.1)	40-52	43.1 (2.9)	40-56	0.1 [-1, 1.2]	.852	.496	[.418, .570]
s3v80c1000-8	45.4 (2.8)	41-58	45.9 (2.7)	41-53	-0.4 [-1.4, 0.6]	.274	.44	[.367, .522]
s3v80c1000-9	41.1 (3)	38-55	41.5 (2.7)	38-49	-0.4 [-1.4, 0.7]	.242	.444	[.373, .522]
s3v80c1000-10	42.8 (2.8)	39-51	43.4 (3.3)	39-55	-0.6 [-1.7, 0.6]	.191	.46	[.381, .538]

**Table 4.** Notes: x = Mean,  $\sigma$  = Standard Deviation, Min = Minimum observed value, Max = Maximal observed value,  $\Delta$  = Mean difference, CI = Confidence Interval, p = p-value, PS = Probability of Superiority (i.e. that VNS-GA will have less unsolved clauses than the GA for each instance), \*\*\* =  $p < .001$ . \*\* =  $p < .01$ , \* $p < .05$ , c = not possible to calculate CI of PS due to no overlap between VNS-GA and GA. The 95% CI for PS is calculated in Excel using a bootstrapping procedure (random selection with replacement) performed 1000 times. Notes:

## 4 Conclusions

In this work, a variable neighbourhood search combined with the genetic algorithm for the maximum satisfiability problem is introduced. VNS follows a simple principle that is based on systematic changes of neighbourhood within the search. The set of neighbourhoods proposed in this paper can easily be incorporated into any metaheuristic when dealing with various combinatorial optimization problems. Starting the search from the largest neighbourhood and moving systematically towards the smallest neighbourhood is a better strategy for performing diversification and intensification. The results indicate that the variable neighbourhood search strategy can enhance the convergence behaviour of GA. It appears clearly from the results that the performance of VNS-GA is better compared to that of GA. The larger the problem, the larger the size of the neighbourhood is needed, and consequently the more efficient is GA at different neighbourhoods. When compared to state of art solvers, VNS-GA is capable of producing similar results than the top ranked solvers while requiring the least amount of time for several test cases. In order to improve VNS-GA, a better strategy would be to construct the different neighbourhoods based on merging variables by exploiting the number of clauses they have in common rather randomly.

## References

- [1] Cook, S. A: The complexity of theorem-proving procedures. Proceedings of the Third ACM Symposium on Theory of Computing, 151–158, (1971).
- [2] Selman, B., Kautz, H. A., Cohen, B.: Noise Strategies for Improving Local Search. Proceedings of AAAI'94, pp. 337–343. MIT Press, (1994).
- [3] McAllester, D., Selman, B. Kautz, H.: Evidence for invariants in local search. proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97), pp 321-326, (1997).
- [4] Hoos, H.: On the run-time behavior of stochastic local search algorithms for SAT. In Proceedings of AAAI-99, pp. 661-666, (1999)
- [5] Li, C.M., Huang, W.Q.: Diversification and determinism in local search for satisfiability. Proceedings of the Eighth International Conference on Theory and Applications of Satisfiability Testing (SAT-05), 569, Lecture Notes in Computer Science, 158–172, (2005).
- [6] Jin-Kao, H., Lardeux, H., Saubion, F.: Evolutionary computing for the satisfiability problem. In Applications of Evolutionary Computing, volume 2611 of LNCS, pp. 258-267, University of Essex, (2003).
- [7] Hansen, P., Jaumard, B., Mladenovic, N., Parreira, A.D.: Variable neighbourhood search for maximum weighted satisfiability problem. Technical Report G-2000-62, Les Cahiers du GERAD, Group for Research in Decision Analysis, 2000.
- [8] Frank, J.: A study of genetic algorithms to find approximate solutions to hard 3CNF problem. In proceedings of Golden West international conference on artificial intelligence, (1994)
- [9] Grissom, R. J., Kim, J. J.: *Effect sizes for research: Univariate and multivariate applications*. Routledge (2012).
- [10] Vargha, A., Delaney, H.D.: A critique and improvement of the CL Common Language Effect Size statistics of McGraw and Wong. Journal of Educational and Behavioral Statistics, 25(2), 101-132, (2000).
- [11] Arcuri, A., Briand, L.: A Hitchhiker's Guide to Statistical Tests for Assessing Randomized Algorithms in Software Engineering. Technical report, simula research laboratory, number 13/2011, (2011).