MDPI

Article

Optimization: parallel stochastic methods and mixed termination rules

Vasileios Charilogis[1], Ioannis G. Tsoulos[2,*], Anna Maria Gianni[3]

[1] Department of Informatics and Telecommunications, University of Ioannina, Greece; v.charilog@uoi.gr
[2] Department of Informatics and Telecommunications, University of Ioannina, Greece; itsoulos@uoi.gr
[3] Department of Informatics and Telecommunications, University of Ioannina, Greece; am.gianni@uoi.gr
* Correspondence: itsoulos@uoi.gr;
† Current address: Department of Informatics and Telecommunications, University of Ioannina, Greece.
‡ These authors contributed equally to this work.

Abstract: Parallel optimization constitutes a powerful tool for solving optimization problems in various domains. By harnessing multiple computational resources simultaneously, optimization methods can achieve faster convergence and improved performance. This specific parallel implementation involves the concurrent execution of different methods, such as evolutionary algorithms, swarm-based optimization, and the utilization of multiple restarts. The primary objective is the efficient exploration of the search space and the attainment of optimal solutions in shorter time frames without squandering computational power. However, defined termination criteria are essential to prevent uncontrolled execution of the algorithm, aiming to conserve computational resources and time. Within the scope of this study, an innovative combination of termination rules and a mechanism for transferring optimal solutions among different methodological approaches is proposed. The proposed enhancements have been tested on a series of well-known optimization problems from relevant literature, and the results are reported here.

Keywords: Global optimization; Parallel techniques; Termination rules; Evolutionary techniques

1. Introduction

The global minimization problem of a continuous and differentiable function $f : S \to R, S \subset R^n$ can be formulated as

$$x^* = \arg\min_{x \in S} f(x). \quad (1)$$

where the set S is has as follows:

$$S = [a_1, b_1] \otimes [a_2, b_2] \otimes \ldots [a_n, b_n]$$

Such problems occur frequently in areas such as physics [1–3], chemistry [4–6], economics [7,8], medicine [9,10] etc. A series of methods have been proposed in the recent literature to handle problems of Equation 1. Methods used to handle problems of Equation 1 are usually divided into two categories: deterministic and stochastic methods. In the first category, the most common method is the interval method [11,12], where the set S is divided through a series of steps into subregions and some subregions that do not contain the global solution can be removed using some pre-defined criteria. On the other hand, there is a variety of stochastic methods that are easier to implement than the deterministic ones. Among them, there are methods based on considerations derived from Physics, such as the Simulated Annealing method[14], the Henry's Gas Solubility Optimization (HGSO) [15], the Gravitational Search Algorithm (GSA) [16], the Small World Optimization Algorithm (SWOA) [17], etc. Also, a series of evolutionary based techniques have also been suggested for Global Optimization problems, such as Genetic Algorithms [18], the

Citation: Charilogis V.; Tsoulos I.G.; Gianni A.M; Optimization: parallel stochastic methods and mixed termination rules. *Journal Not Specified* **2023**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

Differential Evolution method [19,20], Particle Swarm Optimization (PSO) [21,22], Ant Colony Optimization (ACO) [23], Bat Algorithm (BA) [24], Whale Optimization Algorithm (WOA) [25], Grasshopper Optimization Algorithm (GOA) [26], etc.

However, the above optimization methods require significant amounts of computational power and time; therefore, parallelization of the methods is essential. Recently, a variety of methods proposed that can take advantage of parallel processing, such as parallel techniques [28–30], or methods that utilize the GPU architectures [31–33] etc. Parallel optimization represents a significant approach in the field of optimization problem-solving and is applied across a wide range of applications, such as optimization of machine learning model parameters [34–36], control system design and optimization [37–39], energy and resource management [40–43], as well as problems related to sustainable development and enhancing sustainability [44–46].

By harnessing multiple computational resources, parallel optimization allows for the simultaneous execution of multiple algorithms, leading to faster convergence and improved performance. These computational resources can communicate with each other to exchange information and synchronize processes, thereby contributing to faster convergence towards common solutions. Additionally, leveraging multiple resources enables more effective handling of exceptions and errors, while increased computational power for conducting more trials or utilizing more complex models leads to enhanced performance [47]. Of course, this process requires the development of suitable algorithms and techniques to effectively manage and exploit available resources. Each parallel optimization algorithm requires a coherent strategy for workload distribution among the available resources, as well as an efficient method for collecting and evaluating results.

Parallel techniques have been developed by various researchers, such as combining Simulated Annealing and parallel techniques [48], parallel Particle Swarm Optimization methods [49], application of radial basis functions in parallel stochastic optimization [50] etc. Also, since the Genetic Algorithms is a method that can be parallelized easily, there are several researchers that have examined them thoroughly in the relevant literature [51,52].

In this paper, a new optimization method is proposed which is a mixture of existing global optimization techniques running in parallel on a number of available computing units. Each technique is executed independently of the others and periodically the optimal values retrieved from them are distributed to the rest of the computing units using the propagation techniques presented here. In addition, for the most efficient termination of the overall algorithm, intelligent termination techniques based on stochastic observations are used, which are suitably modified to adapt to the parallel computing environment. The proposed algorithm is graphically outlined in Figure 1. The methods that are used in each processing unit may include Particle Swarm Optimization (PSO) , the Multistart method and Differential Evolution (DE). These techniques have been adopted since they have been widely used in the relevant literature and provide the possibility to be parallelized relatively easily.

The following sections are organized as follows: In section 2, the three main algorithms participating in the overall algorithm are described. In section 3 the algorithm for parallelizing the three methods is described, along with the proposed mechanism for propagating the optimal solution to the remaining methods. In section 4, experimental models and experimental results are described. Finally, in section 5, the conclusions from the application of the current work are discussed.

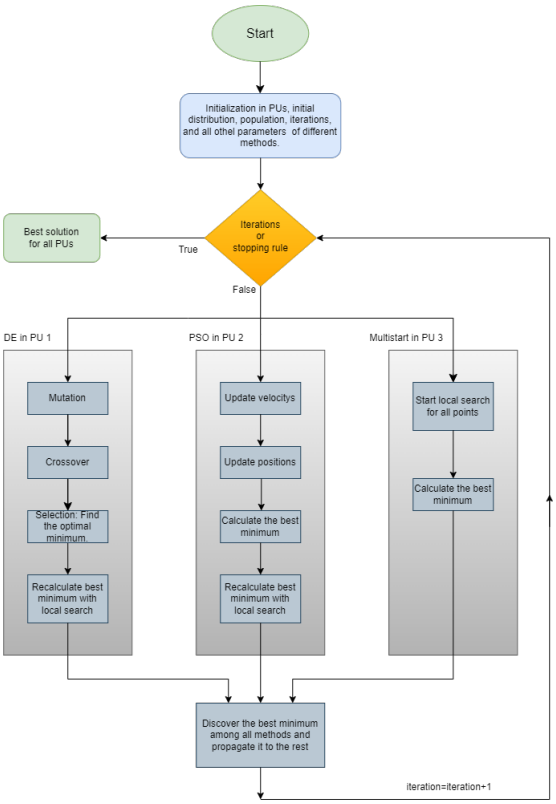


Figure 1. Flowchart of the overall process. The DE acronym stands for the Differential Evolution method, the PSO acroym stands for the Particle Swarm Optimization method and the PU acronym represents the Parallel Unit that executes the algorithm.

2. Adopted Algorithms

The methods that may executed in each processing unit are fully described in this section.

2.1. The method Particle swarm optimization

PSO is an optimization method inspired by the behavior of swarms in nature. In PSO, a set of particles moves in the search space seeking the optimal solution. Each particle has a current position and velocity, and it moves based on its past performance and that of its neighboring particles. PSO continuously adjusts the movement of particles aiming for convergence to the optimal solution [53–55]. This method can be programmed easily and the set of the parameters to be set is limited. Hence, it has been used in a series of practical problems, such as problems that arise in physics [56,57], chemistry [58,59], medicine [60,61], economics [62] etc. The main steps of the PSO method are presented in the Algorithm 1.

Algorithm 1 The main steps of the PSO method.

1. **Initialization**.
   (a) **Set** iter $= 0$ (iteration counter).
   (b) **Set** the number of particles N_p.
   (c) **Set** the maximum number of iterations allowed $\text{iter}_{\max}$
   (d) **Set** the local search rate $p_l \in [0,1]$.
   (e) **Initialize** randomly the positions of the m particles $x_1, x_2, ..., x_m$, with $x_i \in S \subset R^n$
   (f) **Initialize** randomly the velocities of the m particles $u_1, u_2, ..., u_m$, with $u_i \in S \subset R^n$
   (g) **For** $i = 1..N_p$ do $p_i = x_i$. The p_i vector are the best located values for every particle i.
   (h) **Set** $p_{\text{best}} = \arg\min_{i \in 1..m} f(x_i)$
2. **Termination Check**. Check for termination. If termination criteria are met then stop.
3. **For** $i = 1..N_p$ **Do**
   (a) **Update** the velocity:
   $$u_i = \omega u_i + r_1 c_1 (p_i - x_i) + r_2 c_2 \left(p_{\text{best}} - x_i\right)$$
   The parameters $r_1,\ r_2$ are random numbers with $r_1 \in [0,1]$ and $r_2 \in [0,1]$.
   The constant number $c_1,\ c_2$ are in the range $[1,2]$.
   The variable ω is called inertia, with $\omega \in [0,1]$.
   (b) **Update** the position
   $$x_i = x_i + u_i$$
   (c) **Set** $r \in [0,1]$ a random number. If $r \leq p_m$ then $x_i = \text{LS}(x_i)$, where $\text{LS}(x)$ is a local search procedure.
   (d) **Evaluate** the fitness of the particle i, $f(x_i)$
   (e) **If** $f(x_i) \leq f(p_i)$ then $p_i = x_i$
4. **End For**
5. **Set** $p_{\text{best}} = \arg\min_{i \in 1..m} f(x_i)$
6. **Set** iter $=$ iter $+ 1$.
7. **Goto** Step 2

2.2. The method Differential evolution

The DE method relies on differential operators and is particularly effective in optimization problems that involve searching through a continuous search space. By employing differential operators, DE generates new solutions that are then evaluated and adjusted until the optimal solution is achieved [63,64]. The method was used in a series of problems, such as electromagnetics [65], energy consumption problems [66], job shop scheduling [67], image segmentation [68] etc. The Differential Evolution operates through the following steps: Initially, initialization takes place with a random population of solutions in the search space. Then, each solution is evaluated based on the objective function. Subsequently, a process is iterated involving the generation of new solutions through modifications, evaluation of these new solutions, and selection of the best ones for the next generation. The algorithm terminates when a termination criterion is met, such as achieving sufficient improvement in performance or exhausting the number of iterations. The main steps of the DE method are presented in Algorithm 2.

Algorithm 2 The main steps of the DE method.

1. **INPUT:**
   (a) The population size $N_d \geq 4$. The members of this population are also called agents.
   (b) The crossover probability CR $\in [0,1]$.
   (c) The differential weight $F \in [0,2]$.
2. **OUTPUT**:
   (a) The agent x_{best} with the lower function value $f(x_{\text{best}})$.
3. **Initialize** all agents in S.
4. **While** termination criteria are not hold **do**
   (a) **For** $i = 1 \ldots N_d$ **do**
      i. **Select** as x the agent i.
      ii. **Select** randomly three agents a, b, c with the property $a \neq b$, $b \neq c$, $c \neq a$.
      iii. **Select** a random position $R \in \{1, \ldots, n\}$
      iv. **Create** the vector $y = [y_1, y_2, \ldots, y_n]$ with the following procedure
      v. **For** $j = 1, \ldots, n$ **do**
         A. **Set** $r_i \in [0,1]$ a random number.
         B. **If** $r_j <$ CR **or** $j = R$ **then** $y_j = a_j + F \times (b_j - c_j)$ **else** $y_j = x_j$.
      vi. If $y \in S$ AND $f(y) \leq f(x)$ then $x = y$.
      vii. **EndFor**
   (b) **EndFor**
5. **End While**

2.3. The method Multistart

In contrast, the Multistart method follows a different approach than previous methods. Instead of focusing on a single initial point, it repeats the search from various initial points in the search space. This allows for a broader exploration of the search space and increases the chances of finding the optimal solution**.** The Multistart method is particularly useful when optimization algorithms may get trapped in local minima [69–71]. The approach of multiple starts belongs to the simplest techniques for global optimization. At the outset of the process, an initial distribution of points is made in Euclidean space. Subsequently, local optimization begins simultaneously from these points. The discovered minima are compared, and the best one is retained as the global minimum. Local optimizations rely on the Broyden Fletcher Goldfarb Shanno (BFGS) method [72]. The main steps of the multistart method are presented in Algorithm 3.

Algorithm 3 The main steps of the Multistart method.

1. **Initialization** step.
   (a) **Set** N_m as the total number of samples.
   (b) **Set** (x^*, y^*) as the global minimum. Initialize y^* to a very large value.
2. **Sampling** step.
   (a) **For** $i = 1 \ldots N_m$ **Do**
      i. **Sample** a point $x_i \in S$
      ii. $y_i = \text{LS}(x_i)$. Where LS(x) is a local search procedure.
      iii. **If** $y_i \leq y^*$ then $x^* = x_i, y^* = y_i$
   (b) **EndFor**

3. The proposed method

In the present work, a mixture of termination rules is proposed as a novel technique, which can be used without partitioning in any stochastic global optimization technique. Also, a new parallel global optimization technique is proposed that utilizes the new termination rule. In the following subsections, the new termination rule is fully described, followed by the new optimization method.

3.1. The new termination rule

The proposed termination rule is a mixture of several stochastic termination rules proposed in the recent literature. The first algorithm will be called *best - fitness* and in this termination technique, at each iteration k, the difference between the current best value $f_{min}^{(k)}$ and the previous best value $f_{min}^{(k+1)}$ is calculated, i.e., the absolute difference:

$$\left| f_{\min}^{(k)} - f_{\min}^{(k-1)} \right| \tag{2}$$

If this difference is zero for a series of predefined consecutive iterations N_k, then the method terminates. In the second termination rule, which will be called *mean - fitness*, the average function value for each iteration is computed. If this value remains relatively unchanged over a certain number of consecutive iterations, it indicates that the method may not be making significant progress towards discovering a new global minimum. Therefore, termination is warranted. Hence, in every iteration k, we compute:

$$\delta^{(k)} = \left| \sum_{i=1}^{\mathrm{NP}} \left| f_i^{(k)} \right| - \sum_{i=1}^{\mathrm{NP}} \left| f_i^{(k-1)} \right| \right| \tag{3}$$

The value NP denotes the number of particles or chromosomes that participate in the algorithm. The termination rule is defined as follows: terminate if $\delta^{(k)} \leq \epsilon$ for a predefined number N_k of iterations. The third stopping rule was the so - called DoubleBox stopping rule, that was initially proposed in the work of Tsoulos [73]. According to this criterion, the algorithm terminates when one of the following conditions is met:

- The iteration k count exceeds a predefined limit of iterations k_{max}
- The relative variance $\sigma^{(k)}$ falls below half of the variance $\sigma^{(k_{last})}$ of the last iteration $k_{(last)}$ where a new optimal functional value was found.

$$k \geq N_k \, or \, \sigma^{(k)} \leq \frac{\sigma^{(klast)}}{2} \tag{4}$$

This proposed termination criterion is the combination of the aforementioned similarity types. Optimization termination is achieved when any homogeneity condition is satisfied, resulting in improved speed. Therefore, the following relationships holds:

$$\left| f_{\min}^{(k)} - f_{\min}^{(k-1)} \right| or$$

$$\delta^{(k)} = \left| \sum_{i=1}^{\mathrm{NP}} \left| f_i^{(k)} \right| - \sum_{i=1}^{\mathrm{NP}} \left| f_i^{(k-1)} \right| \right| or$$

$$\sigma^{(iteration)} \leq \frac{\sigma^{(klast)}}{2} \, or$$

$$k \geq N_k$$

3.2. The proposed algorithm

In the scientific literature, parallelization typically involves distributing the population among parallel computing units to save computational power and time [77,78]. In the present study, we concurrently compute the optimal solutions of three different stochastic optimization methods originating from different classification categories. The proposed algorithm is shown in Algorithm 4.

Algorithm 4 The proposed overall algorithm

1. **Set** as N_I the total number of parallel processing units.
2. **Set** as N_k the total number of allowed iterations.
3. **Set** $k = 0$ the iteration number.
4. **For** $j = 1, .., N_I$ do in parallel
   (a) **Execute** an iteration of any stochastic algorithm mentioned in section 2.
   (b) **Find** the best element from all optimization methods and **propagate** it to the rest of processing units.
5. **End For**
6. **Update** $k = k + 1$
7. **Check** the proposed termination rule. If the termination rule is valid, then goto step 7a else goto step 4.
   (a) **Terminate** and report the best value from all processing units.

4. Experiments

All experiments conducted were repeated 30 times to ensure the reliability of the algorithm producing the results. The parallelization was achieved using the OpenMP library [82], while the implementation of the method was done in ANSI C++ within the optimization package OPTIMUS, available at https://github.com/itsoulos/OPTIMUS. The values used for all the parameters are shown in Table 1. The values of the parameters used in the experiments were such that there is a balance between the ability of the algorithms to discover the global minimum with certainty on the one hand, but also on their speed.

Table 1. The following parameters were considered for conducting the experiments

Parameter	Value	Explanation
N_e	120	Total elements
N_k	200	Maximum number of iterations
N_s	15 (DE, PSO and Multistart)	Similarity max count
F	0.8 (DE)	Differential weight for DE
CR	0.9 (DE)	Crossover Probability for DE

4.1. Test functions

The test functions [79,80] presented below exhibit varying levels of difficulty in solving them; hence, a periodic local optimization mechanism has been incorporated. Periodic local optimization plays a crucial role in increasing the success rate in locating the minimum of functions. This addition appears to lead to a success rate approaching 100% for all functions, regardless of their characteristics such as dimensionality, minima, scalability, and symmetry. A study by Z.-M. Gao and colleagues [81] specifically examines the issue of symmetry and asymmetry in the test functions. The test functions used in the conducted experiments are shown in Table 2. The application of a local search method at the end of every method used in the experiments ensures that the global optimization method will discover a true minima of the used test function.

NAME	FORMULA	DIMENSION
Bent Cigar	$f(x) = x_1^2 + 10^6 \sum_{i=2}^{n} x_i^2$	10
BF1	$f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1) - \frac{4}{10}\cos(4\pi x_2) + \frac{7}{10}$	2
BF2	$f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1)\cos(4\pi x_2) + \frac{3}{10}$	2
Branin	$f(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$	2
CM	$f(x) = \sum_{i=1}^{n} x_i^2 - \frac{1}{10}\sum_{i=1}^{n}\cos(5\pi x_i)$	4
Discus	$f(x) = 10^6 x_1^2 + \sum_{i=2}^{n} x_i^2$	10
Easom	$f(x) = -\cos(x_1)\cos(x_2)\exp\left((x_2-\pi)^2 - (x_1-\pi)^2\right)$	2
Exp	$f(x) = -\exp\left(-0.5\sum_{i=1}^{n} x_i^2\right), \quad -1 \leq x_i \leq 1$	$n = 4, 16$
Griewank2	$f(x) = 1 + \frac{1}{200}\sum_{i=1}^{2} x_i^2 - \prod_{i=1}^{2}\frac{\cos(x_i)}{\sqrt{(i)}}$	2
Griewank10	$\mathrm{f}(x) = 1 + \frac{1}{200}\sum_{i=1}^{10} x_i^2 - \prod_{i=1}^{10}\frac{\cos(x_i)}{\sqrt{(i)}}$	10
Gkls[74]	$f(x) = \mathrm{Gkls}(x, n, w)$	$n = 2, 3\ w = 50, 100$
Hansen	$f(x) = \sum_{i=1}^{5} i\cos[(i-1)x_1 + i]\sum_{j=1}^{5} j\cos[(j+1)x_2 + j]$	2
Hartman3	$f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	3
Hartman6	$f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	6
High Elliptic	$f(x) = \sum_{i=1}^{n}\left(10^6\right)^{\frac{i-1}{n-1}} x_i^2$	10
Potential[75]	$V_{LJ}(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right]$	$n = 9, 15, 21, 30$
Rastrigin	$f(x) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2)$	2
Shekel5	$f(x) = -\sum_{i=1}^{5}\frac{1}{(x-a_i)(x-a_i)^T + c_i}$	4
Shekel7	$f(x) = -\sum_{i=1}^{7}\frac{1}{(x-a_i)(x-a_i)^T + c_i}$	4
Shekel10	$f(x) = -\sum_{i=1}^{10}\frac{1}{(x-a_i)(x-a_i)^T + c_i}$	4
Sinusoidal[76]	$f(x) = -(2.5\prod_{i=1}^{n}\sin(x_i - z) + \prod_{i=1}^{n}\sin(5(x_i - z))), \quad 0 \leq x_i \leq \pi$	$n = 4, 8$
Test2N	$f(x) = \frac{1}{2}\sum_{i=1}^{n} x_i^4 - 16x_i^2 + 5x_i$	$n = 4, 9$
Test30N	$\frac{1}{10}\sin^2(3\pi x_1)\sum_{i=2}^{n-1}\left((x_i-1)^2(1+\sin^2(3\pi x_{i+1}))\right) + (x_n-1)^2(1+\sin^2(2\pi x_n))$	$n = 3, 4$

Table 2. The test functions used in the conducted experiments.

4.2. Experimental results

All tables presented here demonstrate the average function calls obtained for each test problem. The average function can be considered as a measure of the speed of each method, since each function has a different complexity and therefore, the average execution time would not be a good indication of the computational time required. In the first set of experiments we examined if the presence of the suggested stopping rule, which is the combination of a series of stopping rules, affects the optimization techniques without parallelization. In Table 3, we observe the performance of all termination rules for the DE method. From these, both statistical and quantitative comparisons arise, depicted in Figure 2.

Table 3. Average function calls for the Differential evolution method, using different different termination rules

Problem	ITERATION	BEST	MEAN	DOUBLEBOX	ALL
BF1	32561	6761	10511	10369	6114
BF2	30339	7693	9222	11524	7560
BRANIN	21650	4639	10982	4907	4289
CAMEL	27444	6116	13290	9905	5940
CIGAR10	31854	7647	4348	21116	3111
CM4	31504	7913	4175	11851	4079
DISCUS10	25477	5191	2042	13614	2034
EASOM	19591	1917	15103	1721	1721
ELP10	25870	6046	24675	12519	6031
EXP4	27134	5216	18151	6385	5040
EXP16	28161	5588	27387	7552	5339
GKLS250	25362	5227	2641	8379	2641
GKLS350	24645	5624	3298	17437	3206
GRIEWANK2	29342	8027	10458	14756	6915
GRIEWANK10	38706	9664	37839	17312	9539
POTENTIAL3	26013	5278	24823	13871	5256
PONTENTIAL5	32876	8225	32439	20828	7742
PONTENTIAL6	35041	8467	34946	19169	8197
PONTENTIAL10	41410	10330	41300	26308	10643
HANSEN	23069	5219	23050	14245	4263
HARTMAN3	24165	4613	17966	7333	4566
HARTMAN6	25963	5734	17109	9354	5550
RASTRIGIN	29501	5912	17240	8987	5999
ROSENBROCK8	34964	8503	34429	20980	8051
ROSENBROCK16	41395	10052	41335	26156	8307
SHEKEL5	25810	6064	25800	19877	5933
SHEKEL7	26177	5597	26168	20274	5677
SHEKEL10	26489	6037	26439	22960	5100
SINU4	24646	5751	24620	12511	4776
SINU8	25355	6340	25477	18741	4980
TEST2N4	25987	5848	24947	11640	5727
TEST2N9	26698	7303	26571	16165	6288
TEST30N3	25560	3169	8395	8134	2869
TEST30N4	25021	2841	10381	7853	2714
Total	**965780**	**214552**	**677557**	**474733**	**186197**

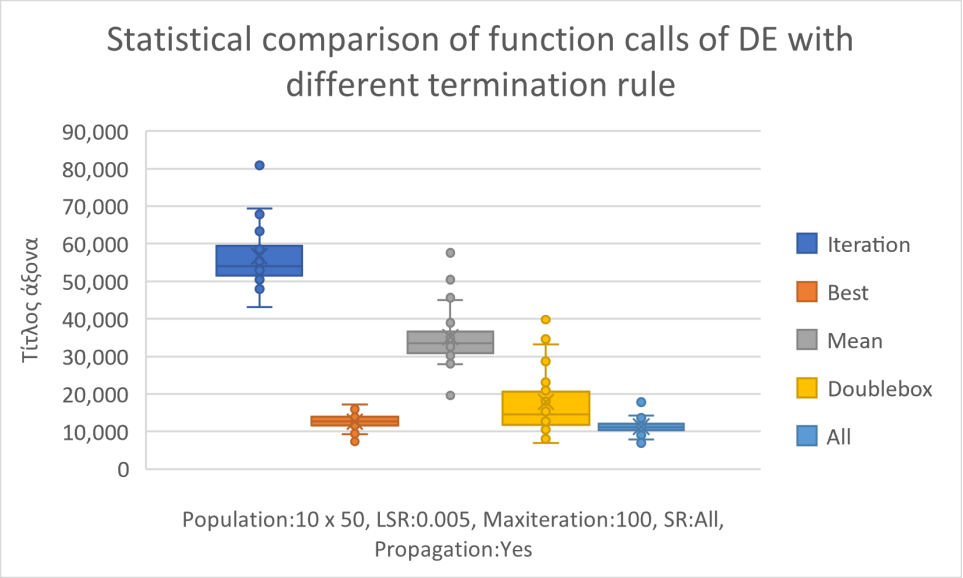


Figure 2. Statistical comparison of function calls with different termination rule of differential evolution

As it is evident, the combination of the stopping rules reduces the required number of function calls that are required to obtain the global minimum by the DE method. The proposed termination scheme outperforms by a percentage that varies from 13% to 73% the other termination rules used here. Subsequently, the same experiment was conducted for the PSO method and the results are shown in Table 4 and the statistical comparison is outlined graphically in Figure 3.

Table 4. Particle swarm optimization: Function calls with different termination rules

Problem	ITERATION	BEST	MEAN	DOUBLEBOX	ALL
BF1	33113	3418	3127	3122	3005
BF2	32804	3265	2995	2919	2889
BRANIN	27762	2601	2549	2431	2417
CAMEL	28511	2801	2675	2547	2547
CIGAR10	39038	3987	3674	3638	3638
CM4	33051	3863	3299	3144	3144
DISCUS10	29548	2611	2409	2405	2405
EASOM	26251	2420	2232	2232	2232
ELP10	1907	1705	1741	1586	1586
EXP4	28364	2751	2558	2558	2558
EXP16	28536	2898	2676	2676	2676
GKLS250	27495	2796	2553	2422	2422
GKLS350	28350	2947	2645	3833	2558
GRIEWANK2	32596	4405	2839	4282	2820
GRIEWANK10	37070	5327	4561	4797	4248
POTENTIAL3	32890	3289	3231	3222	3170
PONTENTIAL5	50595	4926	4881	7282	4730
PONTENTIAL6	62076	6699	5537	7534	5077
PONTENTIAL10	78807	10582	6737	29865	6539
HANSEN	29527	3788	2687	3062	2587
HARTMAN3	29673	2807	2743	2550	2550
HARTMAN6	31739	3081	2915	2809	2809
RASTRIGIN	32717	3558	2990	2829	2829
ROSENBROCK8	33115	4279	4263	4036	3969
ROSENBROCK16	35519	5761	5432	5170	5170
SHEKEL5	29461	3132	2823	12476	2816
SHEKEL7	30003	3064	2855	11520	2856
SHEKEL10	30043	3128	2942	14786	2866
SINU4	27786	3047	2740	2673	2657
SINU8	28129	3177	2824	4113	2828
TEST2N4	28950	3037	2839	2681	2681
TEST2N9	30368	4188	3106	3105	3067
TEST30N3	27643	2943	3130	2762	2762
TEST30N4	27836	3202	3185	2915	2915
Total	**1111273**	**125483**	**110393**	**169982**	**106023**

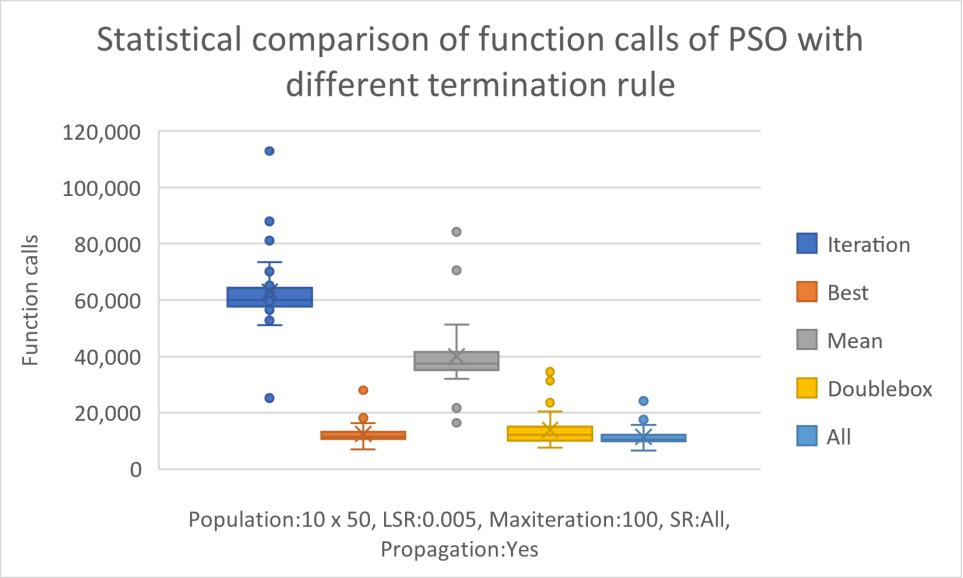


Figure 3. Statistical comparison of function calls with different termination rule of particle swarm optimization

Once more, the proposed termination scheme outperforms the other stopping rules in almost every test function. Likewise, the same experiment was carried out for the Multistart method and the results are depicted in Table 5 and the statistical comparison is shown in Figure 4.

Table 5. Multistart: Function calls with different termination rules

Problem	ITERATION	BEST	MEAN	DOUBLEBOX	ALL
BF1	479374	53873	479374	50762	50762
BF2	276214	37670	239220	35711	35711
BRANIN	107283	11063	11046	10521	10521
CAMEL	163508	16083	153559	15255	15255
CIGAR10	59097	14937	14937	14697	14697
CM4	726033	62236	62192	58581	58581
DISCUS10	50456	6296	6296	6056	6056
EASOM	55130	5412	55130	51132	5412
ELP10	58917	14757	14981	14517	14517
EXP4	54174	10022	54174	9774	9774
EXP16	54680	10528	54680	10280	10280
GKLS250	58547	6202	47948	5908	5908
GKLS350	81988	7560	42145	7087	7087
GRIEWANK2	188429	19179	188429	17877	17877
GRIEWANK10	620223	65206	567153	61818	61818
POTENTIAL3	126632	17773	126632	17161	17161
PONTENTIAL5	226873	31347	226873	30249	30249
PONTENTIAL6	247144	35457	247144	33883	33883
PONTENTIAL10	283230	44957	283230	43618	43618
HANSEN	201543	18568	201543	17541	17541
HARTMAN3	162934	17395	162934	16562	16562
HARTMAN6	179073	22010	179073	21015	21015
RASTRIGIN	275610	24696	275610	23015	23015
ROSENBROCK8	62010	17850	17850	17610	17610
ROSENBROCK16	69460	25300	25300	25060	25060
SHEKEL5	151648	17725	151648	16972	16972
SHEKEL7	154373	17894	154373	17127	17127
SHEKEL10	159825	17931	159825	17135	17135
SINU4	155634	16429	155634	15647	15647
SINU8	170167	19519	170167	18674	18674
TEST2N4	154092	16486	154092	15806	15806
TEST2N9	160651	22035	160651	20716	20716
TEST30N3	161628	16934	161628	16145	16145
TEST30N4	159753	16686	159753	15907	15907
Total	**6296333**	**758016**	**5165224**	**769819**	**724099**

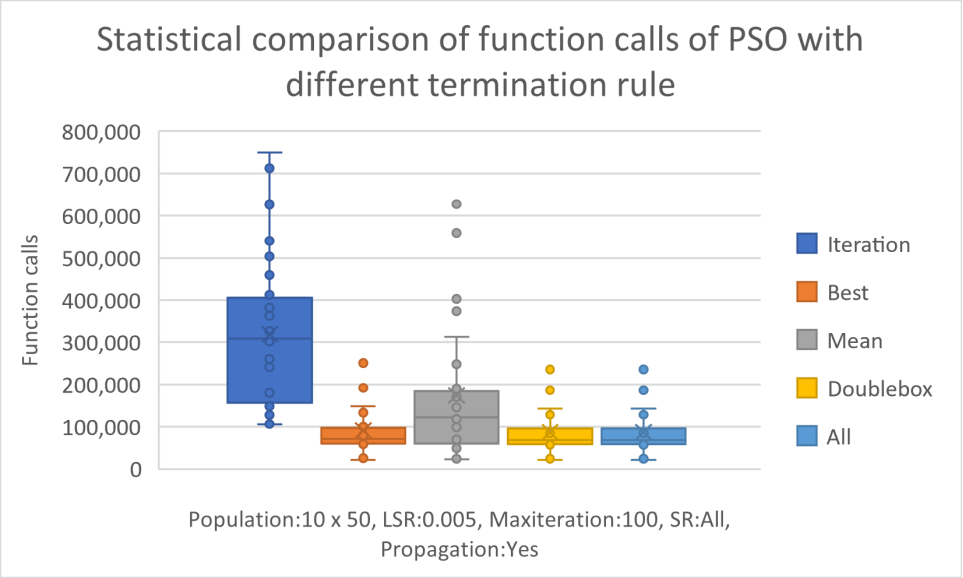


Figure 4. Statistical comparison of function calls with different termination rule of multistart

Observing the tables with the corresponding statistical and quantitative comparisons, it is evident that the calls to the objective function for the proposed termination rule are fewer than any other rule in any method.

The proposed termination rule was also applied to the current parallel optimization technique for the test function described previously. The experimental results for the so - called *DoubleBox* stopping rule and the given parallel method are show in Table 6. The columns in the table stand for the following:

1. Column Problem denotes the test function used.
2. Column 1x500 denotes the application of the proposed technique with one processing unit and 500 particles.
3. Column 2x250 stands for the usage of two processing units. In each unit, the number of particles was set to 250.
4. Column 5x100 denotes the incorporation of 5 processing units into the proposed method. In each processing unit the number of particles was set to 100.
5. Column 10x50 stands for the usage of 10 processing units. In each processing unit, the number of particles was set to 50.

Table 6. Experiments using the proposed optimization technique and the Doublebox stopping rule. The experiment was performed on the test functions described previously. Numbers in cells denote average function calls.

Problem	1x500	2x250	5x100	10x50
BF1	36018	28282	18419	10859
BF2	42191	37555	16568	10710
BRANIN	45165	45726	45757	34381
CAMEL	54786	54782	54253	53839
CIGAR10	18387	18300	18307	11950
CM4	60796	60501	60287	60200
DISCUS10	25115	24586	14565	8957
EASOM	40376	40211	40100	40099
ELP10	20193	14745	11211	7691
EXP4	54779	54676	54416	54412
EXP16	52768	52205	52563	52560
GKLS250	51980	51240	51233	51231
GKLS350	48577	48761	48702	4100
GRIEWANK2	56767	56690	51170	26076
GRIEWANK10	20941	20721	14870	11765
POTENTIAL3	49418	49800	49109	49002
PONTENTIAL5	59827	59656	59367	59322
PONTENTIAL6	62720	62398	62294	62280
PONTENTIAL10	72899	72745	72943	72578
HANSEN	45064	45689	45510	45424
HARTMAN3	47307	47112	47111	47100
HARTMAN6	49370	49302	49312	49222
RASTRIGIN	57617	56577	56513	56400
ROSENBROCK8	22997	22903	14139	14001
POSENBROCK16	42117	34856	21392	13722
SHEKEL5	50631	50198	50233	44662
SHEKEL7	51212	51200	51200	45610
SHEKEL10	51741	51607	51587	47761
SINU4	48140	48100	48111	48099
SINU8	48761	48434	48423	48204
TEST2N4	48904	48307	48354	48354
TEST2N9	48838	48555	48600	48335
TEST30N3	51640	51307	51194	51143
TEST30N4	48992	48549	48540	48487
Total	**1587034**	**1556276**	**1476353**	**1338536**

In this experiment, the total number of particles was set to 500, in order to have credibility in the values recorded. The results indicate that the increase in processing units may reduce the required number of function calls. The same series of experiments was also conducted using the so - called *mean fitness* termination rule, that has been described in equation 3. The obtained results are presented in Table 7.

Table 7. Experiments using the proposed optimization technique and the mean - fitness stopping rule. Numbers in cells denote average function calls.

Problems	1x500	2x250	5x100	10x50
BF1	49714	41284	41089	19596
BF2	58272	46270	41324	15878
BRANIN	45165	45665	44894	7268
CAMEL	54786	54989	54651	10295
CIGAR10	63026	62206	60277	27217
CM4	60796	60333	40072	12977
DISCUS10	50988	50690	36151	14040
EASOM	40376	34178	21355	7377
ELP10	50397	16787	9334	7809
EXP4	54749	54445	29812	11024
EXP16	52768	51233	24782	10722
GKLS250	51980	50333	37696	9641
GKLS350	48577	48442	28420	9337
GRIEWANK2	56767	40471	30632	12356
GRIEWANK10	68439	58410	53818	30317
POTENTIAL3	49418	49328	33657	10140
PONTENTIAL5	59827	58259	34436	11515
PONTENTIAL6	62720	62715	34158	14417
PONTENTIAL10	72899	72800	64964	16780
HANSEN	45064	45582	32013	9318
HARTMAN3	47307	47983	31033	8899
HARTMAN6	49370	47983	27884	9602
RASTRIGIN	54637	53551	33095	10169
ROSENBROCK8	67885	57314	46961	25383
POSENBROCK16	77207	66172	59098	41461
SHEKEL5	50631	49800	32025	10389
SHEKEL7	51212	50630	29136	11193
SHEKEL10	51741	50321	33462	11200
SINU4	48140	52788	32412	9721
SINU8	48761	55864	32685	10410
TEST2N4	48904	48800	28959	8643
TEST2N9	48838	47001	36281	8085
TEST30N3	51514	50100	39828	11699
TEST30N4	48992	46875	42564	11954
Total	**1841867**	**1729602**	**1258958**	**456832**

And in this series of experiments, the columns of the table retain the same meaning as in Table 6. Furthermore, once again, it is observed that the increase in the number of computing units significantly reduces the required number of function calls to find the global minimum. Additionally, there is a significant reduction in the number of function calls required to be compared to the previous termination rule. Finally, the proposed termination rule is utilized in the parallel optimization technique and the experimental results are outlined in Table 8.

Table 8. Experiments using the proposed optimization technique and proposed stopping rule. Numbers in cells denote average function calls.

Problems	1x500	2x250	5x100	10x50
BF1	10839	6543	6375	5933
BF2	9838	6317	5978	5716
BRANIN	5787	5003	5076	4936
CAMEL	12204	5889	5703	5434
CIGAR10	7631	6575	6582	6500
CM4	16222	8935	6837	6392
DISCUS10	5997	5039	5001	5252
EASOM	4701	4791	4772	4751
ELP10	6230	5363	5471	5205
EXP4	10277	5542	5620	5533
EXP16	9577	5518	5544	5532
GKLS250	13328	5993	5205	5011
GKLS350	10131	7373	5094	4869
GRIEWANK2	8504	8284	6371	5761
GRIEWANK10	8535	7585	7229	6922
POTENTIAL3	11338	5666	5830	5695
PONTENTIAL5	12414	7310	7126	6996
PONTENTIAL6	14760	8450	7477	7561
PONTENTIAL10	17729	11928	10431	9338
HANSEN	8379	6862	4883	4970
HARTMAN3	10674	5395	5219	5078
HARTMAN6	10841	5486	5649	5348
RASTRIGIN	13657	8283	5850	5292
ROSENBROCK8	9456	6751	7232	6979
POSENBROCK16	9853	7993	8377	8086
SHEKEL5	10959	5749	5810	5884
SHEKEL7	10651	5828	5843	5698
SHEKEL10	1533	5863	6112	5588
SINU4	10291	6427	5371	5251
SINU8	9993	7139	6274	5583
TEST2N4	10242	7516	5573	5129
TEST2N9	12998	10137	6112	5566
TEST30N3	6228	5512	5402	5485
TEST30N4	5540	5749	5750	5350
Total	**337337**	**228794**	**207179**	**198624**

And in this case, it is observed that the increase of parallel processing units significantly reduces the required number of function calls, as in the two previous termination rules. However, the proposed termination rule dramatically reduces the number of required function calls and consequently the computation time compared to previous termination rules. This reduction in fact intensifies as the number of computing nodes increases. Also, for the same set of experiments, a statistical test (Wilcoxon test) was conducted and the it is graphically presented in Figure 5.

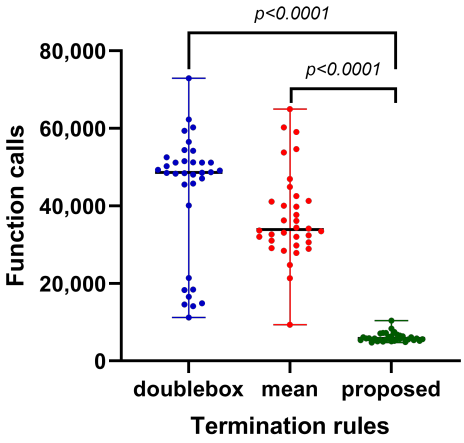


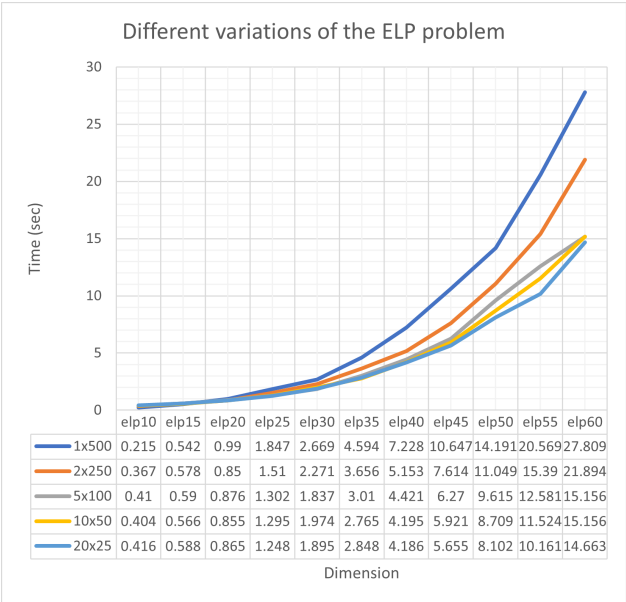
Figure 5. Wilcoxon rank-sum test results for the comparison of different termination rules as used in the proposed parallel optimization technique and for 5 processing units.

A p-value of less than 0.05 (two-tailed) was used to determine statistical significance and is indicated in bold.

An additional experiment was performed in order to demonstrate the efficiency of the proposed technique on high-dimensional functions. For this reason, the high elliptic function was selected. The function is defined as

$$f(x) = \sum_{i=1}^{n} \left(10^6\right)^{\frac{i-1}{n-1}} x_i^2$$

where the n stands for the dimension of the function. The results from the application of the current parallel method to this function when the variable n varies from $n = 10$ to $n = 60$ are graphically illustrated in Figure 6.



	elp10	elp15	elp20	elp25	elp30	elp35	elp40	elp45	elp50	elp55	elp60
1x500	0.215	0.542	0.99	1.847	2.669	4.594	7.228	10.647	14.191	20.569	27.809
2x250	0.367	0.578	0.85	1.51	2.271	3.656	5.153	7.614	11.049	15.39	21.894
5x100	0.41	0.59	0.876	1.302	1.837	3.01	4.421	6.27	9.615	12.581	15.156
10x50	0.404	0.566	0.855	1.295	1.974	2.765	4.195	5.921	8.709	11.524	15.156
20x25	0.416	0.588	0.865	1.248	1.895	2.848	4.186	5.655	8.102	10.161	14.663

Figure 6. Execution time for the proposed method when applied to the ELP problem for different dimensions.

As in the previous experiments, increasing the number of computing units leads to a reduction in the required number of function calls and as can be seen in this graph, this in turn leads to a reduction in computing time.

5. Conclusions

In this paper, the use of mixed termination rules for global optimization methods was thoroughly presented, and a new global optimization method that takes full advantage of parallel computing structures was afterward proposed. The use of mixed termination rules leads a series of computational global optimization techniques to find the global minimum of the objective function faster, as was also shown by the experimental results. The termination rules exploited are based on asymptotic criteria and are general enough to be applicable to any global optimization problem and stochastic global optimization technique.

Furthermore, the new stopping rule was utilized in a novel stochastic global optimization technique that involves some basic global optimization techniques. This method is designed to be executed in parallel computation environments. At each step of the method, a different global optimization method is also executed on each parallel computing unit, and these different techniques exchange optimal solutions with each other. In current work, the global optimization techniques of Differential Evolution, Multistart and Particle Swarm optimization were used in the processing units, but the method can be generalized to use other stochastic optimization techniques, such as genetic algorithms or simulated annealing. Also, the overall method terminates with the combined termination rule proposed here, and the experimental results performed on a series of well - known test problems from the relevant literature seem to be very promising.

Future extensions of the current work may include the incorporation of more advanced termination rules as well as the execution of alternative global optimization methods in the processing units, such as genetic algorithms or variants of the simulated annealing.

Author Contributions: V.C., I.G.T and A.M.G conceived the idea and methodology and supervised the technical part regarding the software. V.C. conducted the experiments. A.M.G. performed the statistical analysis. I.G.T. and all other authors prepared the manuscript. All authors have read and agreed to the published version of the manuscript.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: This research has been financed by the European Union : Next Generation EU through the Program Greece 2.0 National Recovery and Resilience Plan , under the call RESEARCH – CREATE – INNOVATE, project name “iCREW: Intelligent small craft simulator for advanced crew training using Virtual Reality techniques" (project code:TAEDK-06195).

Conflicts of Interest: The authors have no conflicts of interest to declare.

References

1. M. Honda, Application of genetic algorithms to modelings of fusion plasma physics, Computer Physics Communications **231**, pp. 94-106, 2018.
2. X.L. Luo, J. Feng, H.H. Zhang, A genetic algorithm for astroparticle physics studies, Computer Physics Communications **250**, 106818, 2020.
3. T.M. Aljohani, A.F. Ebrahim, O. Mohammed, Single and Multiobjective Optimal Reactive Power Dispatch Based on Hybrid Artificial Physics–Particle Swarm Optimization, Energies **12**, 2333, 2019.
4. P.M. Pardalos, D. Shalloway, G. Xue, Optimization methods for computing global minima of nonconvex potential energy functions, Journal of Global Optimization **4**, pp. 117-133, 1994.
5. A. Liwo, J. Lee, D.R. Ripoll, J. Pillardy, H. A. Scheraga, Protein structure prediction by global optimization of a potential energy function, Biophysics **96**, pp. 5482-5485, 1999.
6. J. An, G.He, F. Qin, R. Li, Z. Huang, A new framework of global sensitivity analysis for the chemical kinetic model using PSO-BPNN, Computers & Chemical Engineering **112**, pp. 154-164, 2018.
7. Zwe-Lee Gaing, Particle swarm optimization to solving the economic dispatch considering the generator constraints, IEEE Transactions on **18** Power Systems, pp. 1187-1195, 2003.

8. M. Basu, A simulated annealing-based goal-attainment method for economic emission load dispatch of fixed head hydrothermal power systems, International Journal of Electrical Power & Energy Systems **27**, pp. 147-153, 2005.
9. Y. Cherruault, Global optimization in biology and medicine, Mathematical and Computer Modelling **20**, pp. 119-132, 1994.
10. Eva K. Lee, Large-Scale Optimization-Based Classification Models in Medicine and Biology, Annals of Biomedical Engineering **35**, pp 1095-1109, 2007.
11. M.A. Wolfe, Interval methods for global optimization, Applied Mathematics and Computation **75**, pp. 179-206, 1996.
12. T. Csendes and D. Ratz, Subdivision Direction Selection in Interval Methods for Global Optimization, SIAM J. Numer. Anal. **34**, pp. 922–938, 1997.
13. Kawachi, M., Ando, N. (1989). Genetic Algorithms in Search, Optimization & Machine Learning. Artificial Intelligence, Vol.: 7, Issue: 1, pp.: 168. Doi: https://doi.org/10.11517/jjsai.7.1_168
14. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P. (1983). Optimization by simulated annealing. Science, Vol. 220, Issue: 4598, pp.: 671–680. DOI: 10.1126/science.220.4598.671 [CrossRef] [PubMed]
15. Hashim, F.A., Houssein, E.H., Mabrouk, M.S., Al-Atabany,W., Mirjalili, S. (2019). Henry gas solubility optimization: A novel physics based algorithm. Future Generation Computer Systems: Vol.: 101, pp.: 646–667. Doi: https: 10.1016/j.future.2019.07.015
16. Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S. (2009). GSA: A gravitational search algorithm. Information Sciences: Vol.: 179, pp.:2232–2248. https://doi.org/10.1016/j.ins.2009.03.004
17. Du, H., Wu, X., Zhuang, J. (2006). Small-World Optimization Algorithm for Function Optimization. In Proceedings of the International Conference on Natural Computation, Xi'an, China, 24–28 September 2006, pp. 264–273. Doi: 10.1007/11881223_33
18. Goldberg, D.E., Holland, J.H. (1988). Genetic Algorithms and Machine Learning. Machine Learning (3): pp.: 95–99. Doi: 10.1023/A:1022602019183
19. Das, S. , Suganthan, P.N. (2011). Differential evolution: A survey of the state-of-the-art. IEEE Transactions on Evolutionary Computation, Vol.: 15, Issue:1, pp.: 4–31. Doi: 10.1109/TEVC.2010.2059031
20. V. Charilogis , I.G. Tsoulos, A. Tzallas, E. Karvounis, Modifications for the Differential Evolution Algorithm, Symmetry **14**, 447, 2022.
21. Kennedy, J., Eberhart, R. (1995). Particle Swarm Optimization. In Proceedings of the ICNN'95—International Conference on Neural Networks, Perth,WA, Australia, 27 November–1 December 1995, Vol.: 4, pp. 1942–1948. Doi: 10.1109/ICNN.1995.488968
22. V. Charilogis, I.G. Tsoulos, Toward an Ideal Particle Swarm Optimizer for Multidimensional Functions, Information **13**, 217, 2022.
23. Dorigo, M., Maniezzo, V., Colorni, A. (1996). Ant system: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), Vol.: 26, pp.: 29–41. Doi: 10.1109/3477.484436
24. Yang, X.S., Gandomi, A.H. (2012). Bat algorithm: A novel approach for global engineering optimization. Engineering Computations, Vol.: 29, pp.: 464–483. Doi: https://doi.org/10.1108/02644401211235834
25. Mirjalili, S., Lewis, A. (2016). The whale optimization algorithm. Advances in Engineering Softwar, Vol.: 95, pp.: 51–67. Doi: https://doi.org/10.1016/j.advengsoft.2016.01.008
26. Saremi, S., Mirjalili, S., Lewis, A. (2017). Grasshopper optimisation algorithm: Theory and application. Advances in Engineering Software, Vol.: 105, pp.: 30–47.Doi: https://doi.org/10.1016/j.advengsoft.2017.01.004.
27. Rao, R.V., Savsani, V.J., Vakharia, D. (2011). Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. Computer-Aided Design, Vol.: 43, pp.: 303–315. Doi: https://doi.org/10.1016/j.cad.2010.12.015
28. J. F. Schutte, J. A. Reinbolt, B. J. Fregly, R. T. Haftka, A. D. George, Parallel global optimization with the particle swarm algorithm, International journal for Numerical methods in Engineering **61**, pp. 2296-2315, 2004.
29. J. Larson and S.M. Wild, Asynchronously parallel optimization solver for finding multiple minima, Mathematical Programming Computation **10**, pp. 303-332, 2018.
30. I.G. Tsoulos, A. Tzallas, D. Tsalikakis, PDoublePop: An implementation of parallel genetic algorithm for function optimization, Computer Physics Communications **209**, pp. 183-189, 2016.
31. R. Kamil, S. Reiji, An Efficient GPU Implementation of a Multi-Start TSP Solver for Large Problem Instances, Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation, pp. 1441-1442, 2012.
32. Van Luong T., Melab N., Talbi EG. (2011) GPU-Based Multi-start Local Search Algorithms. In: Coello C.A.C. (eds) Learning and Intelligent Optimization. LION 2011. Lecture Notes in Computer Science, vol 6683. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-25566-3_24
33. K. Barkalov, V. Gergel, Parallel global optimization on GPU, J Glob Optim **66**, pp. 3–20, 2016.
34. Low, Y., Gonzalez, J., Kyrola, A., Bickson,D., Guestrin, C., Hellerstein, J. (2010). GraphLab: A New Framework For Parallel Machine Learning. Source: arXiv, Computer Science.
35. Yangyang, L., Liu, G., Lu, G., Jiao, L., Marturi, N. & Shang, R. (2019). Hyper-Parameter Optimization Using MARS Surrogate for Machine-Learning Algorithms. IEEE Transactions on Emerging Topics in Computational Intelligence, pp(99):1-11. DOI: 10.1109/TETCI.2019.2918509.
36. Yamashiro, H. & Nonaka, H. (2021). Estimation of processing time using machine learning and real factory data for optimization of parallel machine scheduling problem. ScienceDirect: Operations Research Perspectives, Vol.:8, 2021, Doi: https://doi.org/10.1016/j.orp.2021.100196 .
37. Kim, H.S. & Tsai, L. (2003). Design Optimization of a Cartesian Parallel Manipulator. Journal of Mechanical Design,125(1):43-51. Doi: https://doi.org/10.1115/1.1543977 .

38. Oh, S., Jong Jang,H. & Pedrycz, W. (2009). The design of a fuzzy cascade controller for ball and beam system: A study in optimization with the use of parallel genetic algorithms. ScienceDirect: Engineering Applications of Artificial Intelligence, 22(2):261-271. Doi: https://doi.org/10.1016/j.engappai.2008.07.003 .
39. Fatehi, M., Toloei, A., Zio, E., Niaki, S.T.A. & Kesh,B. (2023). Robust optimization of the design of monopropellant propulsion control systems using an advanced teaching-learning-based optimization method. ScienceDirect: Engineering Applications of Artificial Intelligence, Vol.:126, Part: A.
40. Cai, J., Yang, H., Lai, T. & Xu, K.(2023). Parallel pump and chiller system optimization method for minimizing energy consumption based on a novel multi-objective gorilla troops optimizer. ScienceDirect: Journal Of Building Engineering, Vol.: 76, Doi: https://doi.org/10.1016/j.jobe.2023.107366
41. Yu, Y. & Shahabi, L. (2023). Engineering Application of Artificial Intelligence: Optimal infrastructure in microgrids with diverse uncertainties based on demand response, renewable energy sources and two-stage parallel optimization algorithm. ScienceDirect. Vol.:123, Part b. Doi: https://doi.org/10.1016/j.engappai.2023.106233
42. Ramirez-Gil, F.J., Pere-Madrid, C.M., Nelli Silva, E.C. & Montealerge-Rubio, W. (2021). Sustainable Computing: Informatics and Systems: Parallel computing for the topology optimization method: Performance metrics and energy consumption analysis in multiphysics problems. Vol.:30, Doi: https://doi.org/10.1016/j.suscom.2020.100481
43. Tavakolan, M., Mostafazadeh, F., Eirdmousa, S.J., Safari, A. & Mirzai, K. (2022). A parallel computing simulation-based multi-objective optimization framework for economic analysis of building energy retrofit: A case study in Iran. ScienceDirect: Journal of Building Engineering, Vol.: 45, Doi: https://doi.org/10.1016/j.jobe.2021.103485
44. Lin, G. (2020). Parallel optimization n based operational planning to enhance the resilience of large-scale power systems. Missis-sippi State University, Scholars Junction. Online: https://scholarsjunction.msstate.edu/cgi/viewcontent.cgi?article=4435&context=td
45. Pang, M. & Shoemaker, C.A. (2023). Comparison of parallel optimization algorithms on computationally expensive groundwater remediation designs. Sience of the Total Environment: 857(3), Doi: https://doi.org/10.1016/j.scitotenv.2022.159544
46. Ezugwu, A. (2023). A general Framework for Utilizing Metaheuristic Optimization for Sustainable Unrelated Parallel Machine Scheduling: A concise overview. Arxiv, Computer Science, Neural and Evolutionary Computing. Doi: https://doi.org/10.48550/arXiv.2311.12802
47. Censor, Y. & Zenios, S. (1997). Parallel Optimization: Theory, Algorithms and Applications. Publisher: Oxford University Press, USAISBN: ISBN-13: 978-0195100624. DOI: 10.1093/oso/9780195100624.001.0001
48. E. Onbaşoğlu, L. Özdamar, Parallel simulated annealing algorithms in global optimization. Journal of global optimization **19**, pp. 27-50, 2001.
49. J.F. Schutte, J.A. Reinbolt, B.J. Fregly, R.T. Haftka, A.D. George, Parallel global optimization with the particle swarm algorithm. International journal for numerical methods in engineering **61**, pp. 2296-2315, 2004.
50. Regis, R. G., & Shoemaker, C. A. (2009). Parallel stochastic global optimization using radial basis functions. INFORMS Journal on Computing, 21(3), 411-426.
51. R. Shonkwiler, Parallel genetic algorithms. In ICGA (pp. 199-205), 1993.
52. E. Cantú-Paz, A survey of parallel genetic algorithms, Calculateurs paralleles, reseaux et systems repartis **10**, pp. 141-171, 1998.
53. Marini, F. & Walczak, B. (2015). Particle swarm optimization (PSO). A tutorial. Chemometrics and Intelligent Laboratory Systems. Volume 149, Part B, p.:153-165. Doi: https://doi.org/10.1016/j.chemolab.2015.08.020
54. García-Gonzalo, E. & Fernández-Martínez, J. L. (2012). A Brief Historical Review of Particle Swarm Optimization (PSO). Journal of Bioinformatics and Intelligent Control, Volume 1, Number 1, June 2012, pp. 3-16(14). American Scientific Publishers. DOI:https://doi.org/10.1166/jbic.2012.1002
55. Jain, M., Saihjpal, V., Singh, N. & Singh, S.B. (2022). An Overview of Variants and Advancements of PSO Algorithm. MDPI, Applied Sciences: 12, 8392. Doi:https://doi.org/10.3390/ app12178392 .
56. Anderson Alvarenga de Moura Meneses, Marcelo Dornellas, Machado Roberto Schirru, Particle Swarm Optimization applied to the nuclear reload problem of a Pressurized Water Reactor, Progress in Nuclear Energy **51**, pp. 319-326, 2009.
57. Ranjit Shaw, Shalivahan Srivastava, Particle swarm optimization: A new tool to invert geophysical data, Geophysics **72**, 2007.
58. C. O. Ourique, E.C. Biscaia, J.C. Pinto, The use of particle swarm optimization for dynamical analysis in chemical processes, Computers & Chemical Engineering **26**, pp. 1783-1793, 2002.
59. H. Fang, J. Zhou, Z. Wang et al, Hybrid method integrating machine learning and particle swarm optimization for smart chemical process operations, Front. Chem. Sci. Eng. **16**, pp. 274–287, 2022.
60. M.P. Wachowiak, R. Smolikova, Yufeng Zheng, J.M. Zurada, A.S. Elmaghraby, An approach to multimodal biomedical image registration utilizing particle swarm optimization, IEEE Transactions on Evolutionary Computation **8**, pp. 289-301, 2004.
61. Yannis Marinakis. Magdalene Marinaki, Georgios Dounias, Particle swarm optimization for pap-smear diagnosis, Expert Systems with Applications **35**, pp. 1645-1656, 2008.
62. Jong-Bae Park, Yun-Won Jeong, Joong-Rin Shin, Kwang Y. Lee, An Improved Particle Swarm Optimization for Nonconvex Economic Dispatch Problems, IEEE Transactions on Power Systems **25**, pp. 156-16**216**6, 2010.
63. Feoktistov, V. (2006). Differential Evolution. In Search of Solutions. Optimization and Its Appl;ications, Springer. Doi: https://doi.org/10.1007/978-0-387-36896-2
64. Bilal, M.P., Zaheer, H., Garcia-Hernandez, L. & Abraham, A. (2020). Differential Evolution: A review of more than two decades of research. Engineering Applications of Artificial Intelligence 90 (2020) 103479. Doi: https://doi.org/10.1016/j.engappai.2020.103479

65. P. Rocca, G. Oliveri, A. Massa, Differential Evolution as Applied to Electromagnetics, IEEE Antennas and Propagation Magazine. **53**, pp. 38-49, 2011.
66. W.S. Lee, Y.T. Chen, Y. Kao, Optimal chiller loading by differential evolution algorithm for reducing energy consumption, Energy and Buildings **43**, pp. 599-604, 2011.
67. Y. Yuan, H. Xu, Flexible job shop scheduling using hybrid differential evolution algorithms, Computers & Industrial Engineering **65**, pp. 246-260, 2013.
68. L. Xu, H. Jia, C. Lang, X. Peng, K. Sun, A Novel Method for Multilevel Color Image Segmentation Based on Dragonfly Algorithm and Differential Evolution, IEEE Access **7**, pp. 19502-19538, 2019.
69. Marti, R., Resende, M.G.C. & Ribeiro, C. (2013). Multi-start methods for combinatorial optimization. European Journal of Operational Research Volume 226, Issue 1, 1 April 2013, Pages 1-8. Doi: https://doi.org/10.1016/j.ejor.2012.10.012
70. Marti, R., Moreno-Vega, J. & Duarte, A. (2010). Advanced Multi-start Methods. Handbook of Metaheuristics, pp: 265–281.
71. Tu, W. & Mayne, R.W. (2002). Studies of multi-start clustering for global optimization. International Journal for Numerical Methods in Engineering. Doi: https://doi.org/10.1002/nme.400.
72. Dai, Y. H. (2002). Convergence properties of the BFGS algoritm. SIAM Journal on Optimization, 13(3), 693-701.
73. Tsoulos, I.G. Modifications of real code genetic algorithm for global optimization. Appl. Math. Comput. 2008, 203, 598–607.
74. M. Gaviano, D.E. Ksasov, D. Lera, Y.D. Sergeyev, Software for generation of classes of test functions with known local and global minima for global optimization, ACM Trans. Math. Softw. **29**, pp. 469-480, 2003.
75. J.E. Lennard-Jones, On the Determination of Molecular Fields, Proc. R. Soc. Lond. A **106**, pp. 463–477, 1924.
76. Z.B. Zabinsky, D.L. Graesser, M.E. Tuttle, G.I. Kim, Global optimization of composite laminates using improving hit and run, In: Recent advances in global optimization, pp. 343-368, 1992.
77. Charilogis, V.; Tsoulos, I.G. A Parallel Implementation of the Differential Evolution Method. Analytics 2023, 2, 17–30.
78. Charilogis, V.; Tsoulos, I.G.; Tzallas, A. (2023). An Improved Parallel Particle Swarm Optimization. SN Computer Science (2023) 4:766
79. C.A. Floudas, P.M. Pardalos, C. Adjiman, W. Esposoto, Z. Gümüs, S. Harding, J. Klepeis, C. Meyer, C. Schweiger, Handbook of Test Problems in Local and Global Optimization, Kluwer Academic Publishers, Dordrecht, 1999.
80. M. Montaz Ali, Charoenchai Khompatraporn, Zelda B. Zabinsky, A Numerical Evaluation of Several Stochastic Algorithms on Selected Continuous Global Optimization Test Problems, Journal of Global Optimization **31**, pp 635-672, 2005.
81. Gao, Z.M., Zhao, J., Hu,Y.R., Chen, H.F. (2021).The Challenge for the Nature - Inspired Global Optimization Algorithms: Non-Symmetric Benchmark Functions. IEEE Access, July 26, 2021.
82. R. Chandra, L. Dagum, D. Kohr, D. Maydan,J. McDonald and R. Menon, Parallel Programming in OpenMP, Morgan Kaufmann Publishers Inc., 2001.