Expert Systems With Applications 203 (2022) 117379

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A majority–minority cellular automata algorithm for global optimization

Juan Carlos Seck-Tuoh-Mora *, Norberto Hernandez-Romero, Fredy Santander-Baños, Valeria Volpi-Leon, Joselito Medina-Marin, Pedro Lagos-Eulogio

AAI-ICBI-UAEH, Carr Pachuca-Tulancingo Km 4.5, Pachuca 42184 Hidalgo, Mexico

ARTICLE INFO

Keywords:
Global optimization
Majority cellular automata
Metaheuristics
Engineering applications

ABSTRACT

Cellular automata (CA) are discrete dynamical systems that can give rise to complex behaviors under certain conditions. Its operation is based on simple local interactions between its elements. The different dynamical behaviors of CA offer a great diversity of ideas and inspiration to propose new metaheuristics focused on global optimization. One such automata is the one specified by the majority rule, which is capable of implementing logical operations under the right conditions. Taking this rule as inspiration, this work proposes the majority–minority CA algorithm. This algorithm takes different adaptations of the majority rule and its counterpart, the minority rule, to establish different rules that modify vectors of real values in order to achieve a good balance in exploration and exploitation tasks for optimization tasks. The efficiency of the majority–minority CA algorithm is tested with 50 widely used test problems in the literature, using both uni- and multimodals and fixed dimensions. Additionally, 3 engineering applications used in recent literature are also optimized. The numerical results verify the competitiveness of the algorithm compared to other recently published specialized algorithms. The source codes of the proposed algorithm are publicly available at https://github.com/juanseck/MmCAA.git.

1. Introduction

The optimization of engineering problems represents an increasingly complex task due to the significant number of components and interactions between them. Conventional algorithms face difficulties with these types of problems, such as convergence to a local optimum, stagnation and dependence on the correct selection of initial conditions to converge appropriately. Additionally, for their correct operation, the traditional methods based on mathematical programming and the application of the conjugate gradient and quasi-Newton methods require the problem to be optimized to be compliant with specific mathematical properties, such as convexity, continuity and differentiability, which often implies oversimplifying the problem to be addressed (Gogna & Tayal, 2013; Sarker et al., 2003). This issue puts them at a disadvantage in solving problems characterized as being multi-dimensional, multimodal, non-continuous and non-differentiable (Jamil & Yang, 2013). Thus, traditional optimization methods no longer offer quick solutions and satisfactory results.

This situation has led to the search and definition of new optimization methods such as metaheuristics, which can be described as algorithms that adaptively and intelligently combine various procedures to solve complex problems. Although in most cases, it is not possible to show that metaheuristics reach the optimal value of a problem, they are capable of producing good results in a satisfactory amount of time (Kumar & Davim, 2020).

Metaheuristics are also characterized by their easy specification and simple operation that does not depend on the gradient of the functions to be optimized, thus facilitating their computational implementation. It is desired that a metaheuristic can escape the local minima in multi-dimensional problems and be applied in a wide range of problems for the optimization, design and identification of the parameters of engineering systems.

Among the proposed metaheuristics are the swarm intelligence (SI) algorithms, where a set of agents interact locally and without global control to share and generate new information through a simple set of rules that generates collective behavior, emerging with the ability to optimize complex problems (Hassanien & Emary, 2018).

Most SI algorithms are inspired by the behaviors that arise between members of a natural system. Generally, its operation begins with a population that evolves toward values close to the optimum of a problem. Additionally to showing flexibility and adaptability in a great variety of cases, this type of SI algorithm has the advantage of not depending on the selection of initial conditions and not requiring special mathematical conditions of the problem (Cui & Gao, 2012; Slowik, 2021).

* Corresponding author.
E-mail addresses: jseck@uaeh.edu.mx (J.C. Seck-Tuoh-Mora), nhromero@uaeh.edu.mx (N. Hernandez-Romero), sa429458@uaeh.edu.mx (F. Santander-Baños), volpi@uaeh.edu.mx (V. Volpi-Leon), jmedina@uaeh.edu.mx (J. Medina-Marin), plagos@uaeh.edu.mx (P. Lagos-Eulogio).

https://doi.org/10.1016/j.eswa.2022.117379
Received 26 August 2021; Received in revised form 28 January 2022; Accepted 25 April 2022
Available online 6 May 2022

In contrast, the SI algorithm presented in this work is inspired by the behavior of an artificial system known as a cellular automaton, in particular, to use the adaptations of a special type of rule known as the majority rule and its complement, the minority rule (Goles et al., 2018; Martinez et al., 2012). This article aims to present a new optimization method called the majority–minority cellular automata algorithm (MmCAA), which proposes several adaptations of the majority and minority rule for global optimization tasks for problems defined in multiple dimensions. Another objective is to compare the MmCAA with other recent metaheuristics recognized for their efficiency and robustness over a set of test functions with varying degrees of difficulty. Finally, the application of the MmCAA in various engineering problems will be demonstrated to prove its efficiency and versatility in real cases.

An SI algorithm must perform in an efficient and balanced manner two main actions: exploration and exploitation. As the operations for each type of action can be counterproductive and leave the algorithm trapped in a local minimum, an adequate strategy that allows the algorithm to adapt during the application of its different operations, calculate reasonable solutions, avoid getting stuck in the search space and have a premature convergence toward the local minima must be implemented.

The MmCAA algorithm works by first defining a random set of initial solutions. Each of these solutions will generate a neighborhood of new possible solutions, applying adaptations of the majority and minority rule to promote changes to the current solutions that, in some cases, take only the information contained in each solution. In others, information between solutions is communicated to create new solutions. New solutions will be selected based on how much they improve the fitness (or cost) in the problem to be optimized. In the case of this work, the minimization of functions.

The strength of the proposed algorithm is that its specification and implementation are uncomplicated, facilitating its application in optimization tasks and resulting in the achievement of satisfactory results compared with other state-of-the-art metaheuristics.

The contribution of the MmCAA consists of showing that specific cases of cellular automata (CA) that have been studied and applied for other purposes due to their complexity can also be adapted to propose new metaheuristics designed for the optimization of multimodal and non-differentiable problems in multiple dimensions.

As metaheuristics have also been used in other fields of engineering, such as the identification of parameters in complex models, with multiple parameters to be tuned (Kumar & Davim, 2020), the operation of the MmCAA is also tested in various engineering problems. Specifically, the MmCAA is proved for the step-cone pulley design (Rao, 1996; Savsani & Savsani, 2016), the multi-plate disc clutch brake design (Wang et al., 2020) and the speed reducer design (Savsani & Savsani, 2016), comparing the results with other recognized metaheuristics and obtaining competitive results.

The structure of the paper is as follows. Section 2 presents a review of the literature of the latest metaheuristics proposed for the global optimization of functions, emphasizing the SI metaheuristics. Section 3 introduces the basic concepts and main work of the CA with the majority and minority rule. Section 4 explains the MmCAA proposal and the different adaptations of the majority and minority rule, to carry out exploration and exploitation actions concurrently and the general strategy of the MmCAA. Section 5 describes the experimental results obtained by taking 50 test functions in 30 and 500 dimensions. A statistical study with 13 other state-of-the-art algorithms is also presented. Section 6 applies the MmCAA in 3 engineering problems that are taken from specialized literature. These results are also compared with reports from other recent algorithms to show the effectiveness of the MmCAA. The last section provides the conclusions and future proposals for developing new algorithms based on CA.

2. SI metaheuristics review

Metaheuristics based on SI refer to algorithms that mimic the collective intelligence emerging from the global behavior of systems composed of multiple agents that follow simple interaction rules without the existence of any central control. Systems with multiple agents have been an inspiration for the generation of SI algorithms that seek the best set of values to optimize a given objective function with a simple implementation.

SI-based metaheuristic algorithms began to gain popularity in the 90 s. Some of the main contributions were the ant colony optimization (Dorigo, 1992) transforming the optimization problem into one of finding the best path on a weighted graph conducted by a stochastic pheromone model; the particle swarm optimization (PSO) (Kennedy & Eberhart, 1995), where particles search for better solutions inspired by the behavioral models of velocity and position in bird flocking; the shark-search algorithm proposed in Hersovici et al. (1998), where the search analyzes the relevance of neighboring solutions to improve communication and performance.

The initial metaheuristics modeled the behavior patterns of physical or natural systems through a set of equations simplifying the original behavior to develop exploration and exploitation actions. These metaheuristics worked very well with optimization problems in a few dimensions. However, they present difficulties in finding good results in multimodal functions and nonlinear engineering problems. These drawbacks implied the need to evolve toward more sophisticated metaheuristics. The following are some examples: the artificial bee colony algorithm (ABC) (Karaboga, 2005) modeling three essential components, namely, employed foraging bees, unemployed foraging bees and food sources, complemented with two modes of behavior, recruitment of foragers and abandonment of poor sources; the fish school search (Bastos Filho et al., 2008), where agents use a positive gradient to eat and gain weight moving toward better places in the search space; the mosquito host-seeking algorithm (Feng et al., 2009) inspired by the host-seeking behavior of mosquitoes with three kinds of actions, aggregate host-seeking, attraction of the host and social coordination among the swarm; the bat algorithm proposed by Yang (2010), who was motivated by the foraging behavior of micro bats and modeled the natural pulse loudness and emission rate of real bats to propose the formula of velocity and position; or the krill herd algorithm (Gandomi & Alavi, 2012) based on simulating the herding actions of krills, considering the minimum distances of each individual from food and the density of the herd.

The second trend in metaheuristics was working with distinct groups of agents, where each group follows a different behavior to reach a more adaptable optimization process. These new metaheuristics were adequate for multimodal optimization problems and nonlinear engineering applications. Currently, new metaheuristics have to solve problems in dozens and hundreds of dimensions and are required to solve more complex application problems that are defined by a strong nonlinear coupling relationship between their variables. The most recent metaheuristics not only employ different kinds of agents but also implement distinct types of adapting actions and dynamic strategies according to the features of the search space. Some examples are as follows: the cuckoo search algorithm (CSA) (Gandomi et al., 2013), where each agent resembles a cuckoo egg that replaces a not-so-good solution in the nests, and the egg laid by a cuckoo is discovered and the worst nests abandoned with a certain probability; the gray wolf optimizer (GWO) (Mirjalili et al., 2014) that imitates the hierarchy and hunting mechanism of gray wolves, with four types of agents (alpha, beta, delta and omega) and three main steps, namely, searching, encircling and attacking a prey to perform optimization; the elephant herding optimization (Wang et al., 2015), in which agents live in a clan with a leader and other mature agents separate from the group when they reach adulthood; the whale optimization algorithm (Mirjalili & Lewis, 2016) applying a simplification of the hunting method

of humpback whales, called bubble-net feeding method; the drone squadron optimization (de Melo & Banzhaf, 2018), which is an artifact-inspired technique with two core parts, the semi-autonomous drones with a perturbation/movement scheme and the command center that processes the retrieved data to adapt the search to the landscape; the Harris hawks Optimization (HHO) (Heidari et al., 2019), where agents form a team that chase patterns based on the dynamic nature of scenarios to search for better solutions from different directions and motions; or the Slime Mold Algorithm (SMA) (Li et al., 2020), which simulates the positive and negative feedback of the propagation wave of a slime mold.

Only a representative sample of SI metaheuristics has been exposed here to show the development and current trends in their research. In this way, an opportunity area is taking the complexity of artificial systems such as cellular automata, which have been studied for decades and are able to produce complex behaviors and solving complex tasks based on simple interactions, as inspiration for new optimization metaheuristics.

3. Basics of majority rule CA

CAs are discrete dynamical systems proposed by Von Neumann and Ulam to show the feasibility of defining self-reproducing systems that work with uncomplicated local interactions of their components (von Neumann, 1966). The simplicity of the specifications of a CA makes it ideal for implementation in a computer. Its theoretical study and applications have been developed over the years to characterize its ability to produce complex behaviors (McIntosh, 2009; Wolfram, 2002).

A CA comprises cells that initially take a value from a finite set of possible states. The dynamics of the CA lie in discrete steps, so it is a discrete system in time and space. At each step, a cell considers its current state and the states of its close neighbors to update its state in the following time step. In this way, blocks of states are mapped to individual states, which is called the evolution rule. CAs can generate chaotic and complex global behaviors depending on the evolution rule defined by their local mapping. Because of this, CAs have been widely investigated and applied in various engineering and computing problems (Bilan et al., 2020; Hoekstra et al., 2010; Salcido, 2011; Schiff, 2011).

The different evolution rules of CAs can be a source of inspiration for new metaheuristics, given the different complex behaviors that they can produce. Examples of these works can be seen in the scheduling of tasks (Seredynski & Zomaya, 2002), the cellular particle swarm optimization (CPSO) (Shi et al., 2011), the design of adaptive IIR filters (Lagos-Eulogio et al., 2017), the simultaneous optimization of plant layout and task sequencing in a workshop type system (Hernández-Gress et al., 2020) and the global optimization of functions with an adaptation of different evolution rules (Seck-Tuoh-Mora et al., 2021).

One of the evolution rules that has been widely studied in recent works is the majority rule. This rule is elementary; each cell takes the most repeated state of its local neighborhood as a new state. The dynamical behavior of this rule is characterized by the formation of patch patterns that stabilize as the evolution of the system progresses. Its counterpart is the minority rule, which takes the least common element in each neighborhood to update the state of a cell in the next generation. The evolution of the minority rule is characterized by its tendency to avoid a fixed or periodic global state, as a minority state tends to become a majority and vice versa, which causes an oscillating global dynamic.

The majority rule and its variations have been applied in CA to explore and solve various problems, such as a universal circuit specification (Martínez et al., 2010; Parhami et al., 2020), a memory specification to obtain complex behaviors (Goles et al., 2018; Martinez et al., 2012; Xu et al., 2018), or to solve density classification problems (Abilhoa & de Oliveira, 2019; Christensen et al., 2018; Goles & Montealegre, 2020; Laboudi, 2019).

Fig. 1 depicts various dynamic behaviors of the majority rule, the minority rule and the application of the majority rule with probability in one-dimensional CA with two states and different neighborhood sizes.

In these examples, 500 cells are assigned in one state at random. 250 evolutions are used, and time goes from up to down. In the first column, every cell takes one neighbor on each side to obtain an entire neighborhood; in the second column, two neighbors are taken on each side; and in the third column, three neighbors are considered on each side in order to have an entire neighborhood. In all cases, periodic boundary conditions are employed to have entire neighborhoods for all cells.

Evolutions for the majority rule show a periodic pattern almost immediately, whereas for the minority rule, a chaotic pattern characterized by heterogeneous triangular shapes appears. In the third case, every cell picks one of the two rules in a probabilistic way. In the first example, the majority rule is selected with 40% of probability, followed by 70% and 75% in the other cases. With these probabilities, the automata produce complex patterns that combine non-periodic structures with a stable background.

This combination of the majority and minority rules that form complex patterns inspired a new global optimization algorithm where the evolution rules are adapted to work with sequences of real numbers. Thus, the contribution of the proposed algorithm will be to implement, in a balanced way, the exploration and exploitation actions using modifications of the majority and minority rules to work in an alternate and balanced way.

4. MmCAA

The aim of the MmCAA is to emulate the dynamic behavior of the majority and minority rules in CA. Each cell (called a smart-cell) is the set of values defining a solution for the problem to be optimized. The change of each smart-cell is defined by a set of rules, where one is chosen randomly to update its position. Different rules generate different positions, from which the new position with the best cost in the function to be optimized is selected, replacing the position of the smart-cell. With this mechanism, the positions of all smart-cells in the population are improved, and the system iteratively evolves during the optimization process.

Each smart-cell evolves depending on the selected evolution rule, which takes the information from the smart-cell and randomly selected neighbors to improve the position of the smart-cell. In this process of adaptation of the majority and minority rules, some rules consider the information in one or two additional smart-cells to obtain a new position. Other rules only consider the information in the values of the smart-cell to refine the position. The third type of rule only takes into account the best cost obtained up to that moment, to make changes in the position of a smart-cell.

In other words, the MmCAA employs a random spatial model in selecting neighbors, rules and new states, in contrast to the deterministic model, maintaining the same discrete time step as the temporal evolution of a classical CA. Unlike the classic model, each smart-cell in the MmCAA selects its neighbors randomly from the entire set of smart-cells; thus, a grid with fixed positions is not addressed. For each smart-cell, different evolution rules are selected randomly from a possible set of rules instead of having a fixed rule for all smart-cells. This selection of rules forms a new neighborhood of possible new solutions, of which the one with the best fitness will be chosen as the new state of the smart-cell. Similar to the classical model, time in the optimization process advances in discrete steps between one smart-cell generation and another.

The randomness in choosing evolution rules and neighbors allows a smart-cell to access the information contained in the rest of the population, thus generating a neighborhood of new possible positions. This neighborhood structure is suitable to explore the search space

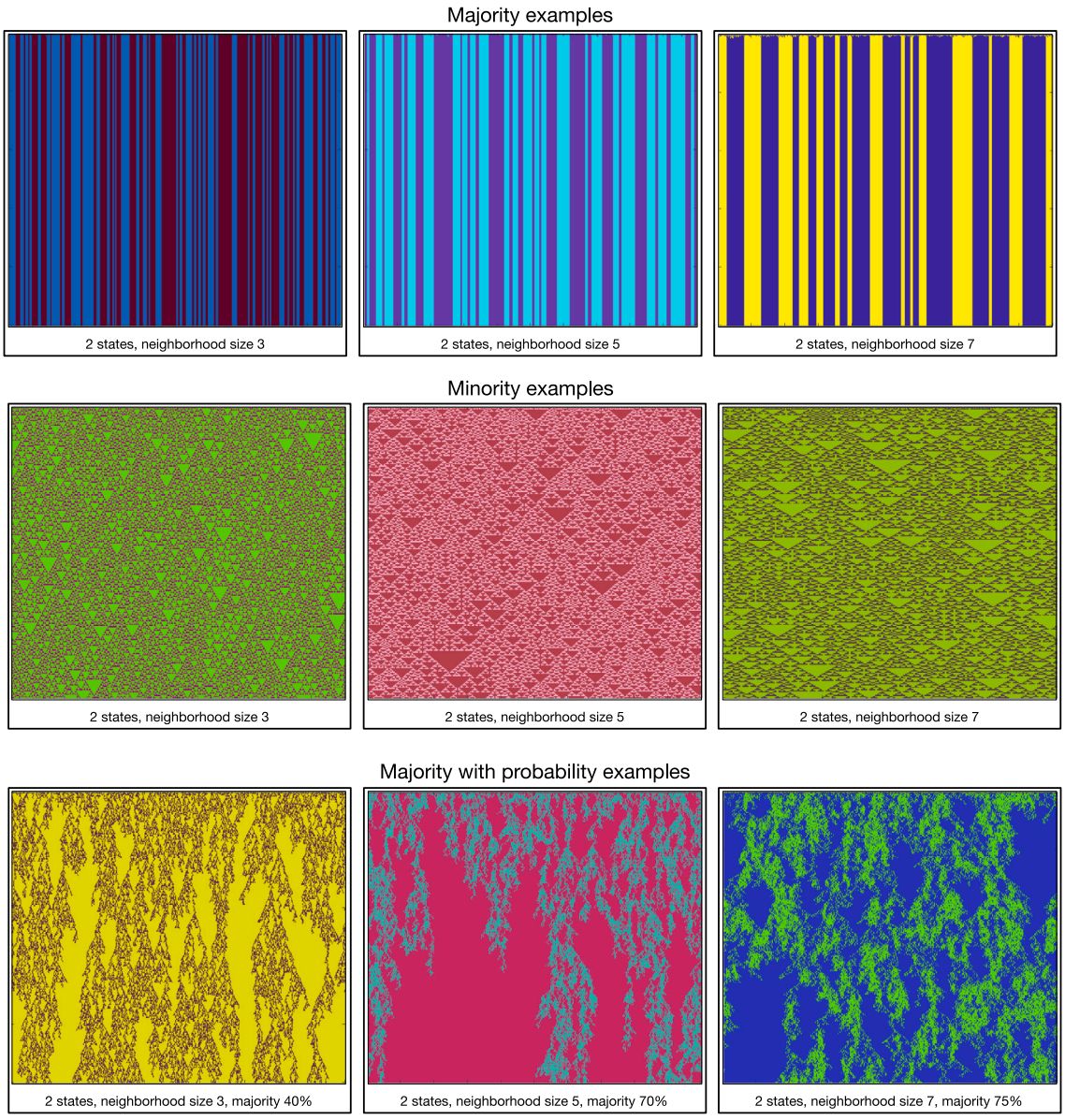


Fig. 1. Examples of CA with 2 states and different neighborhood sizes applying majority, minority and majority with probability evolution rules.

initially and to refine the positions when executing the exploitation of suitable areas later. This same randomness also generates new solutions with a significant change in position during the optimization process, helping the MmCAA escape from local optima and avoid stagnation of solutions.

This work studies the minimization of functions. A function is defined as $f(\boldsymbol{x}) \rightarrow \mathbb{R}$ in n dimensions, with $\boldsymbol{x} \in \mathbb{R}^n$, and each value $x(i)$ in $\boldsymbol{x}$ is bounded between a lower bound $lb(i)$ and an upper bound $ub(i)$. If the bounds are the same for all values in $\boldsymbol{x}$, these limits are simply set as lb and ub.

The MmCAA starts generating a random population S of n_S smart-cells, where each smart-cell is represented as $\boldsymbol{s} \in \mathbb{R}^n$ and its cost is calculated as $f(\boldsymbol{s})$. The best solution of population S is represented by $\boldsymbol{b}_S$ with a cost $f(\boldsymbol{b}_S)$ that is minimal with respect to the other smart-cells in S.

The MmCAA applies a total of six evolution rules on the smart-cells in S. The general strategy of the algorithm is to generate a neighborhood of n_{ne} new positions for each smart-cell, where each smart-cell will take the best solution from these neighbors as a new position if it improves the cost of the original position of the smart-cell or if it passes a probabilistic threshold otherwise. The parameters of the evolution rules are focused on emulating the actions of the majority and minority rules and obtaining an adequate balance of exploration and exploitation actions to conduct a successful optimization process. These evolution rules are explained in the following sections.

4.1. Majority (minority) rule applied to a single smart-cell

Algorithm 1: Majority (minority) rule for a single smart-cell

Result: New smart-cell ***evol***
Input: $\boldsymbol{s}_i$, $prop$;
$elem$ = most repeated element in $\boldsymbol{s}_i$;
forall k *in* $length(\boldsymbol{s}_i)$ **do**
  $\boldsymbol{cam}(k) = \boldsymbol{s}_i(k) - elem$;
end
$\boldsymbol{cam} = \boldsymbol{cam} * prop * rand$;
$\boldsymbol{evol} = \boldsymbol{s}_i - \boldsymbol{cam}$;

The input for the rule described in Algorithm 1 is a smart-cell $\boldsymbol{s}_i$ for $1 \leq i \leq n_S$ and a weight parameter $prop$ to define a limit of change in

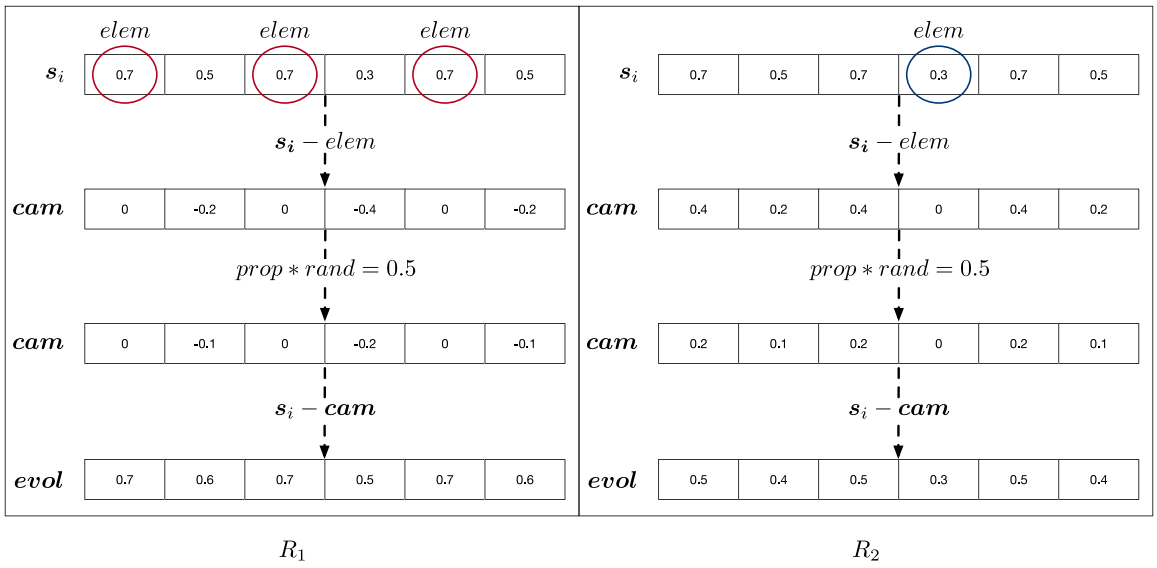


Fig. 2. Rules R_1 and R_2 employed in the MmCAA.

the values of s_i. The rule takes the differences ***cam*** between the values in s_i and the most repeated element *elem* in the smart-cell and *rand* generates a random value between 0 and 1. A new solution ***evol*** is formed by taking the differences between the original smart-cell and ***cam*** weighted randomly between 0 and *prop*. This rule helps bring the values of s_i closer to the most repeated value *elem*. The action of this rule is to produce a self-generated change in s_i to homogenize its values, which is helpful in optimization problems where vectors with good costs have elements with the same value.

The MmCAA uses two versions of this rule (R_1 and R_2). These rules are differentiated in the way the value *elem* is selected in the third line of Algorithm 1. For R_1, *elem* takes the most repeated element in s_i (majority version). For R_2, *elem* selects the least repeated element (minority version) to define the vector ***cam*** later. Fig. 2 depicts the application of these rules, taking smart-cells of 6 values and $prop * rand = 0.5$ to exemplify the application of both rules.

4.2. Rule for rounding values in a smart-cell

Algorithm 2: Rounding rule

Result: New smart-cell ***evol***
Input: s_i, $f(s_i)$, $f(b_S)$, n_r;
$evol = s_i$;
$sum = f(s_i) + f(b_S)$;
$pond = 1 - (f(s_i)/sum)$;
forall k *in* $length(evol)$ **do**
  if $rand <= pond$ **then**
    $evol(k) = round(evol(k), n_r)$;
  end
end

The rule in Algorithm 2 weighs the cost of $f(s_i)$ against that of $f(b_S)$. If the cost of $f(s_i)$ is considerably larger than $f(b_S)$, then the weight *pond* is small and there is not much change in s_i. Otherwise, there will be more changes to calculate a new position.

The change consists of rounding off the n_r to the most significant decimal values of the selected elements in s_i. The least significant decimal digit is the n_rth digit to the right of the decimal point. If the first non-significant digit is less than 5, then the least significant digit remains unchanged, otherwise the least significant digit is incremented by 1. All non-significant digits are removed.

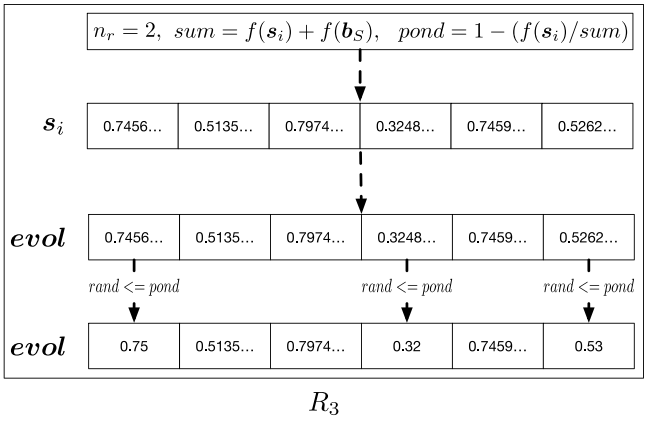


Fig. 3. Rule R_3 used in the MmCAA.

This rule helps auto-generate changes in s_i to round off its values. This rule is applicable for optimization problems to find proper parameters. Only one version (R_3) of this rule is used in the MmCAA. Fig. 3 illustrates the application of this rule, taking $n_r = 2$ as example.

The rules R_1 to R_3 based on the Algorithms 1 and 2 are focused on modifying the information contained in each smart-cell without taking into account its environment; thus, they are rules of exploitation for the position of each smart-cell in the search space.

These rules were already introduced for optimization tasks in Seck-Tuoh-Mora et al. (2021) and are taken up in this work, complemented by other majority and minority rules, to obtain a new optimization algorithm with fewer parameters and operations.

4.3. Majority (minority) rule applied with a neighbor smart-cell

The rule in Algorithm 3 takes the most repeated value of a neighboring smart-cell as a factor change depending on the neighbor's cost weight $f(s_j)$. If the cost $f(s_i)$ is greater than $f(s_j)$, then the weight *pond* is large and there is a major probability that changes are being generated in s_i. The change consists of taking a random proportion that varies between $-prop$ and $prop$ of the most repeated element in s_j to modify each randomly selected position in s_i.

This rule effectively modifies the position of a smart-cell, value by value, especially when the neighboring smart-cell has a much better

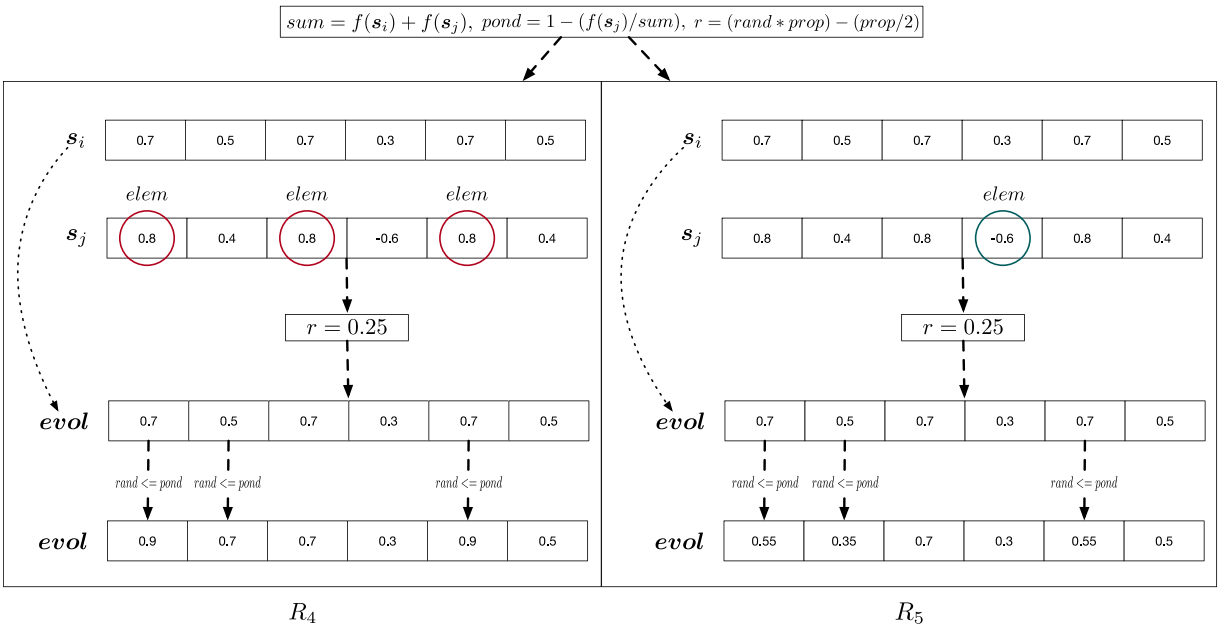


Fig. 4. Rules R_4 and R_5 applied in the MmCAA.

Algorithm 3: Majority (minority) rule with one neighbor

Result: New smart-cell ***evol***
Input: s_i, $f(s_i)$, s_j, $f(s_j)$, $prop$;
$evol = s_i$;
$sum = f(s_i) + f(s_j)$;
$pond = 1 - (f(s_j)/sum)$;
$elem$ = most repeated element in neighbor s_j;
$r = (rand * prop) - (prop/2)$;
forall k *in* $length(evol)$ **do**
  if $rand <= pond$ **then**
    $evol(k) = evol(k) + (r * elem)$;
  end
end

Algorithm 4: Majority rule with two neighbors

Result: New smart-cell ***evol***
Input: s_i, $f(s_i)$, s_{j_1}, $f(s_{j_1})$, s_{j_2}, $f(s_{j_2})$, $prop$;
$evol = s_i$;
$ne_1 = s_i$, $ne_2 = s_{j_1}$, $ne_3 = s_{j_2}$;
$sum = f(s_i) + f(s_{j_1}) + f(s_{j_2})$;
$pond = (f(s_i)/sum)$;
forall k *in* $length(evol)$ **do**
  $elem$ = element $ne_\alpha(k)$ where $\alpha = mod(l, 3), 1 \leq l \leq 3$
    such that $abs(ne_\alpha(k) - ne_{\alpha+1}(k))$ is minimal ;
  $val(k) = elem$;
end
$r = (rand * prop) - (prop/2)$;
forall k *in* $length(evol)$ **do**
  if $rand <= pond$ **then**
    $evol(k) = evol(k) + (r * val(k))$;
  end
end

cost. In the MmCAA, two versions of this rule are used, one taking the most repeated value of the neighbor (R_4) and another taking the least repeated value (R_5). The action of these rules is shown in Fig. 4, taking a value $r = 0.25$ to visualize the effect of each rule in both cases.

Both rules generate new values in s_i that move it away from its original position depending on the information of the neighboring smart-cell. Therefore, they perform a search space exploration action and provide the smart-cell the possibility of escaping from the local optima.

4.4. Majority rule applied with two neighbor smart-cells

The rule in Algorithm 4 randomly selects two neighboring smart-cells to approximate the most common value of each position depending on the cost weight $f(s_i)$. First, the most frequent value in each position in the smart-cell or the two neighbors is detected. If there is no repeated value, the value with the minimum distance concerning any of the neighbors is taken.

If the cost $f(s_i)$ is very large compared to its neighbors, then the weight $pond$ is large and there is a greater probability of generating changes in s_i. The change consists of taking a random proportion between $-prop$ and $prop$ of the most repeated element in each selected position of s_i. The effect of this rule is depicted in Fig. 5 taking value $r = 0.3$ as an example.

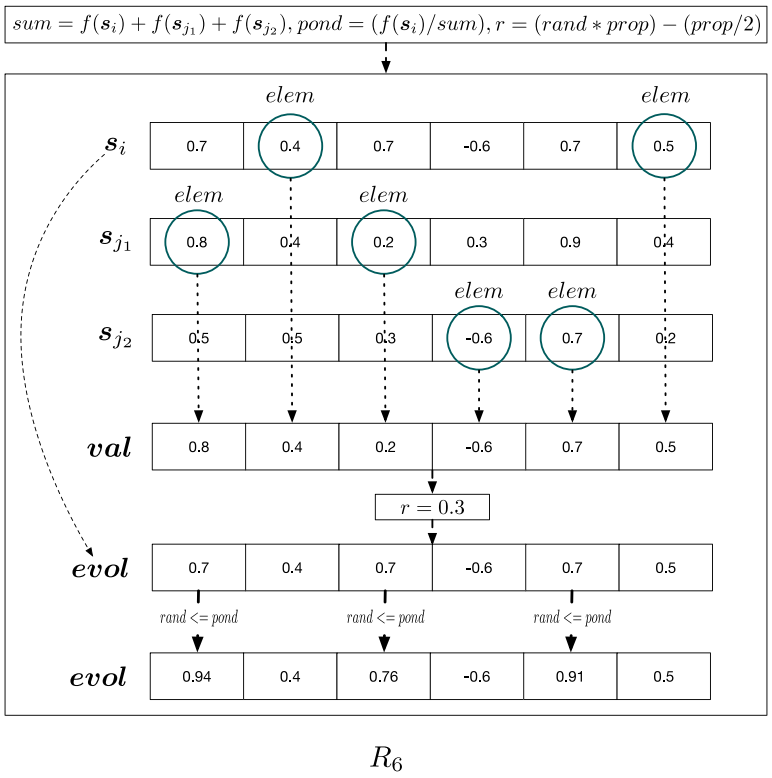
The effect of this rule (R_6) is to modify the values in s_i depending on the most repeated value in each position between the three neighbors. This rule is adequate to refine the position of a smart-cell, value by value, thereby facilitating the conception of new solutions in the search space.

4.5. Complete structure of the MmCAA

The MmCAA has the following input values for its operation: the number n_S of smart-cells, the number n_{ne} of neighbors for each smart-cell, the number of iterations n_{it} and the number of elitist solutions n_{el}. The other input parameters are related to the properties of the f function to be optimized, such as the limits lb and ub and the number n_d of dimensions of f.

The parameters that tune the behavior of the MmCAA include $prop$ used in the algorithms 1, 3 and 4 and the values $lower_r$ and $upper_r$ to define the parameter n_r for the algorithm 2.

The general strategy of the MmCAA is to produce a random population of smart-cells to obtain their cost in f and take some elitist



R_6

Fig. 5. Rule R_6 to complete the MmCAA.

Algorithm 5: The MmCAA

Result: Best smart-cell $\boldsymbol{b}_S$ and fitness value $f(\boldsymbol{b}_S)$
Input: n_S, n_{ne}, n_{it}, n_{el}, lb, ub, n_d, f;
Set parameters $prop$, $lower_r$ y $upper_r$;
Generate random population S of n_S smart-cells;
Evaluate S in f;
forall $i = 2$ *to* n_{it} **do**
  Keep the best n_{el} smart-cells in a new population;
  forall $j = n_{el} + 1$ *to* n_S **do**
    Take $\boldsymbol{s}_j$ and two other random smart-cells $\boldsymbol{s}_{r_1}$, $\boldsymbol{s}_{r_2}$ from S for rules requiring extra solutions;
    forall $k = 1$ *to* n_{ne} **do**
      Choose a random rule R ;
      Obtain $\boldsymbol{evol}_k = R(\boldsymbol{s}_j,$ additional parameters of the rule);
      Check that the n_d values of $\boldsymbol{evol}_k$ are between lb and ub and correct if necessary;
      Calculate $f(\boldsymbol{evol}_k)$;
    end
    Choose the best neighbor $\boldsymbol{evol}$ from the k generated neighbors having a minimum cost $f(\boldsymbol{evol})$;
    if $rand < 0.25$ *or* $f(\boldsymbol{s}_j) > f(\boldsymbol{evol})$ **then**
      $\boldsymbol{s}_j = \boldsymbol{evol}$;
    end
  end
end
Return the best smart-cell $\boldsymbol{b}_S$ and its fitness value $f(\boldsymbol{b}_S)$;

solutions. Elitism helps to conserve the best smart-cells and have their information available to modify the positions of the other solutions. The rest of the smart-cells are updated, taking a neighborhood for each of them. Each neighborhood is formed using new solutions that are a product of the 6 random rules defined by algorithms 1 to 4. These rules produce new solutions based on the information of each smart-cell or the information from other smart-cells as a factor of change.

The best position (with minimum cost) is selected from the generated neighborhood to upgrade the smart-cell, either because it improves its cost or is chosen with a probability of 25%. This probability of replacing one smart-cell with another that has a lower cost helps the algorithm explore new areas of the solution space, escape from the local minima and avoid stagnation. Fig. 6 shows the flow diagram of the MmCAA.

5. Experimentation with benchmark functions

Table 1 presents the 50 test functions used in this work to prove the effectiveness of the MmCAA.

In this set of benchmark functions, the first 16 unimodal functions assess the proposed algorithm's exploitability. The following 17 multimodal test functions show the exploration and escape properties of the local minima. Finally, the remaining 17 functions are fixed-dimension test functions and various local minimums to test the balance of exploration and exploitation movements carried out by the MmCAA. The definitions and mathematical properties of these functions can be reviewed in Jamil and Yang (2013) and Wang et al. (2014). The computational code to implement these functions can also be found in http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/go.htm, http://www.sfu.ca/~ssurjano/optimization.html and http://benchmarkfcns.xyz.

5.1. Defining MmCAA parameters

MATLAB 2015a was used on a 3.1 GHz Intel Xeon CPU machine with 64 GB RAM and a Big Sur operating system to carry out the computational implementation of the MmCAA. The parameters of the number of smart-cells and iterations were taken with values similar to

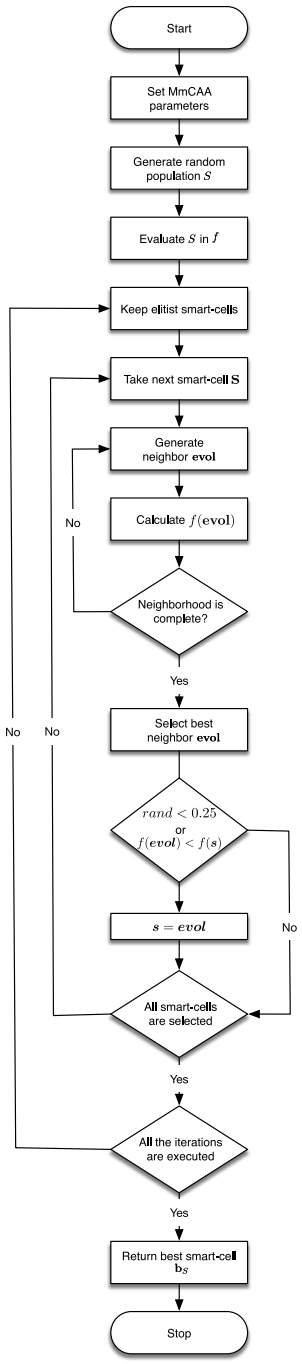


Fig. 6. MmCAA flow chart.

other recently published global optimization algorithms, to make an adequate comparison with other methods.

For the rest of the parameters that define the behavior of the rules in the MmCAA (elitism n_{el}, proportion *prop* and rounding limits $lower_r$ and $upper_r$), an experiment with three levels per parameter was executed, using test functions F_{10} (unimodal), F_{24} (multimodal) in 30 dimensions and F_{43} with fixed dimensions to select the combination of parameters with the best minimization results.

For the elitist part, values 1, 2 and 3 were taken for n_{el}; for the proportion *prop* used in the MmCAA rules, three values 1.4, 1.7 and 2 were tested. To establish n_r in rule 2, values 0, 1 and 2 were analyzed for $lower_r$ as the lower limit, and values 5, 6 and 7 were analyzed for $upper_r$ as the upper limit of rounding. A total of 81 combinations, each

Table 1
Benchmark functions.

No.	Name	No.	Name
F1	Brown	F26	Penalty 2
F2	Exponential	F27	Rastrigin
F3	Powell 2	F28	Salomon
F4	Powell Sum	F29	Schwefel 2.26
F5	Quartic	F30	Shubert 3
F6	Ridge	F31	Streched V Sine Wave
F7	Rosenbrock	F32	Styblinski-Tank3
F8	Schwefel 1.2	F33	Wavy 1
F9	Schwefel 2.20	F34	Beale
F10	Schwefel 2.21	F35	Colville
F11	Schwefel 2.22	F36	Corana
F12	Schwefel 2.23	F37	Cross-in-Tray
F13	Sphere	F38	Drop-Wave
F14	Step 2	F39	Foxholes
F15	Sum Squares	F40	Hartman 1
F16	Zakharov	F41	Hartman 2
F17	Ackley 1	F42	Helical Valley
F18	Alpine 1	F43	Holder-Table
F19	Conditioned Elliptic	F44	Hump
F20	Cosine Mixture	F45	Kowalik
F21	Griewank	F46	Shekel 5
F22	Happy Cat	F47	Shekel 7
F23	Levy	F48	Shekel 10
F24	Pathological	F49	Watson
F25	Penalty 1	F50	Wolfe

Table 2
Parameter settings of the MmCAA.

Number of smart-cells (n_S)	12
Number of neighbors (n_{ne})	6
Number of iterations (n_{it})	500
Number of elitist solutions (n_{el})	2
Rule parameters	
Proportion (*prop*)	1.7
Lower rounding ($lower_r$)	2
Upper rounding ($upper_r$)	6

with 30 independent runs, were tested on the selected functions using the best average value to obtain the most suitable set of parameters. The parameters chosen to perform the comparative tests with other state-of-the-art algorithms are presented in Table 2.

5.2. Participation of the rules in the exploration and exploitation of the search space

The MmCAA applies 6 rules picked randomly from each iteration to generate a neighborhood of new solutions for each smart-cell. The rules are classified with a weight to measure their use, to show what rules contribute to the optimization process for the exploration and exploitation actions.

The rules used for the exploration process are R_4, R_5 and R_6; for the exploitation process, these are R_1, R_2 and R_3. An experiment using the same functions used for the tuning of parameters (F_{10}, F_{24} and F_{43}) was carried out using the values described in Table 2. The best neighbor produced in each iteration was detected. If the fitness of the smart-cell was found to have improved, a value of 10 was accumulated for the exploration rule and a value of 1 for the exploitation rule.

To show that the MmCAA goes from an intense exploration phase toward weighing the exploitation more, the experiment must first show high rule application values at the beginning of the process and then decrease toward low rule application values when the optimization applies more exploitation rules that improve the values of the smart-cell.

For each function, 30 independent runs were applied, and the weights of the rules that improved the smart-cells were averaged for each iteration. The results are presented in Fig. 7.

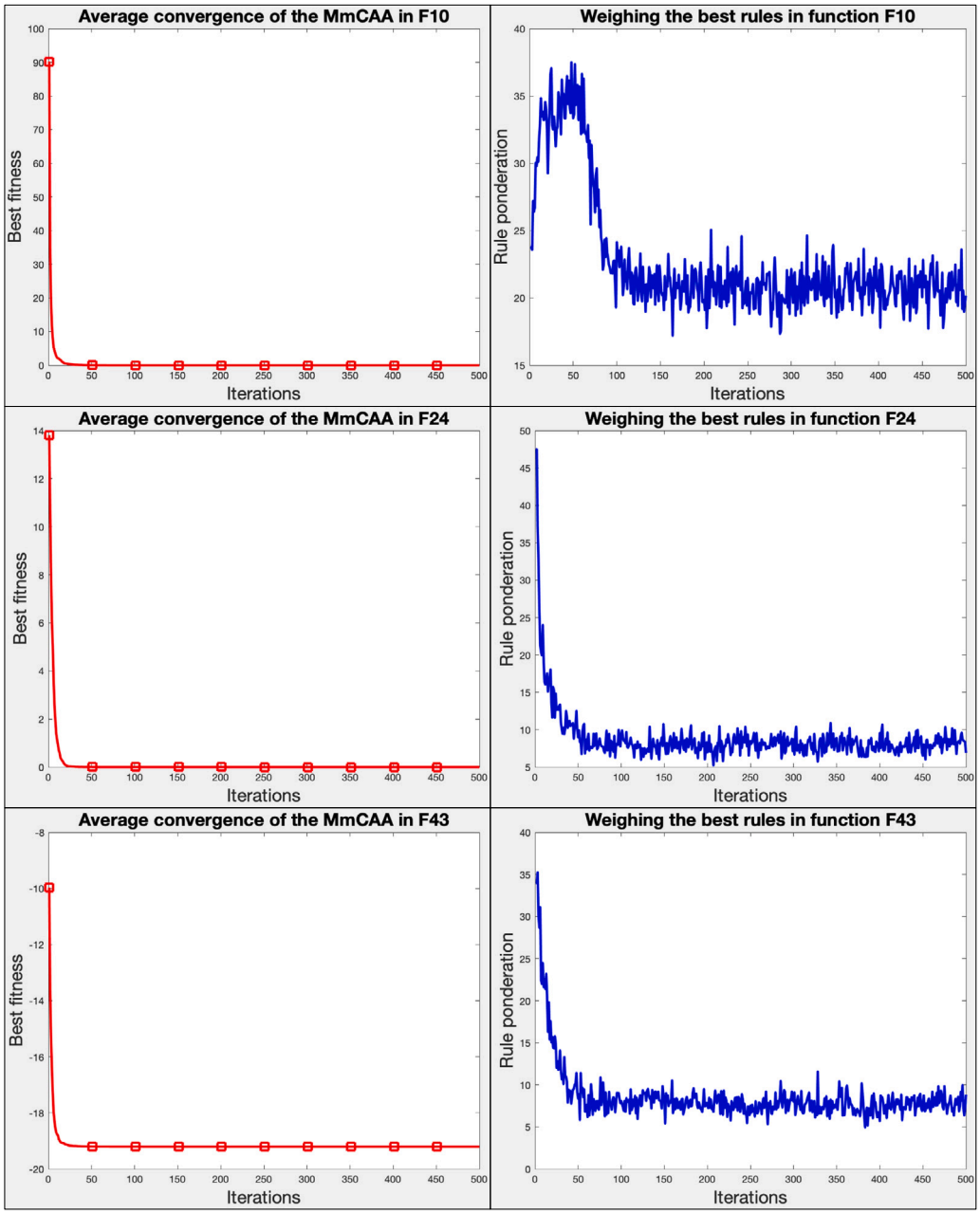


Fig. 7. Weighing the optimization process from exploration to exploitation.

Fig. 7 shows that exploration rules have significant activity in the early stages of the MmCAA and that as the process progresses to later iterations, the exploitation rules become more decisive. In the case of F_{10} (unimodal), the MmCAA tends to increasingly use the exploration rules at the beginning of the process to execute the exploitation phase. This feature demonstrates that the MmCAA has the ability to improve the cost of smart-cells during the optimization process consistently.

5.3. First experiment with test functions in 30 dimensions and fixed dimensions

The MmCAA is compared with 13 algorithms recently proposed in the specialized literature that have shown great performance in global optimization of functions. 10 of these metaheuristics were published in the last two years. These algorithms are as follows: the aquila optimizer (AO) (Abualigah, Yousri et al., 2021), the arithmetic optimization algorithm (AOA) (Abualigah, Diabat et al., 2021), the continuous-state CA algorithm (Seck-Tuoh-Mora et al., 2021), the hunger games search (Yang et al., 2021), the HHO (Heidari et al., 2019), the success-history-based adaptive differential evolution with linear population size reduction (LSH) (Tanabe & Fukunaga, 2014), the LSH with a semi-parameter adaptation hybrid with a covariance matrix adaptation evolution strategy (LSP) (Mohamed et al., 2017), the marine predators algorithm (MPA) (Faramarzi et al., 2020), the modified sine cosine algorithm (MSA) (Gupta et al., 2020), the pure random orthogonal search (PROS) (Plevris et al., 2021), the Pareto-like sequential sampling heuristic (PSS) (Shaqfa & Beyer, 2021), the double adaptive random spare reinforced whale optimization algorithm (RWO) (Chen et al., 2020) and the slime mold algorithm (SMA) (Li et al., 2020).

The computational implementation and parameters of these algorithms were taken directly from the websites referenced in the original works, resulting in a more objective comparison, as the original code of each metaheuristic is being taken.

Table 3
Metaheuristics compared with MmCAA on 30 and fixed dimensional problems.

Fn	Stats	AO	AOA	CCAA	HGS	HHO	LSH	LSP	MmCAA	MPA	MSA	PROS	PSS	RWO	SMA
F1	Avg	5.2E−102	2.4E+00	**0.0E+00**	9.7E−225	8.5E−106	5.4E−15	1.3E−29	**0.0E+00**	1.2E−25	**0.0E+00**	5.8E−04	3.1E−03	3.2E−144	**0.0E+00**
	Std	3.1E−101	1.2E+00	**0.0E+00**	**0.0E+00**	4.3E−105	1.2E−14	2.1E−29	**0.0E+00**	1.6E−25	**0.0E+00**	2.6E−04	1.8E−02	1.3E−143	**0.0E+00**
F2	Avg	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	−1.0E+00	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	−1.0E+00	−1.0E+00	**−1.0E+00**	**−1.0E+00**
	Std	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.9E−16	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	7.5E−06	7.4E−04	**0.0E+00**	**0.0E+00**
F3	Avg	2.5E−104	9.0E−03	**0.0E+00**	8.3E−53	2.9E−104	4.0E−05	3.2E−06	**0.0E+00**	9.1E−14	**0.0E+00**	1.4E−01	8.4E−01	2.4E−58	**0.0E+00**
	Std	1.7E−103	4.4E−02	**0.0E+00**	5.8E−52	1.4E−103	4.1E−05	2.5E−06	**0.0E+00**	4.7E−13	**0.0E+00**	5.7E−02	2.2E+00	1.7E−57	**0.0E+00**
F4	Avg	7.3E−25	**0.0E+00**	**0.0E+00**	1.7E−105	5.3E−131	2.1E−37	1.2E−29	**0.0E+00**	3.3E−62	**0.0E+00**	1.9E−06	8.3E−08	**0.0E+00**	**0.0E+00**
	Std	5.2E−24	**0.0E+00**	**0.0E+00**	1.2E−104	3.7E−130	1.1E−36	7.8E−29	**0.0E+00**	2.3E−61	**0.0E+00**	2.9E−06	1.4E−07	**0.0E+00**	**0.0E+00**
F5	Avg	2.3E−314	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.6E−206	1.3E−25	7.9E−56	**0.0E+00**	2.1E−48	**0.0E+00**	1.4E−08	7.4E−06	2.7E−241	**0.0E+00**
	Std	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.6E−25	2.3E−55	**0.0E+00**	4.7E−48	**0.0E+00**	1.7E−08	3.4E−05	**0.0E+00**	**0.0E+00**
F6	Avg	−3.5E+00	**−5.0E+00**	**−5.0E+00**	−5.0E+00	−5.0E+00	−5.0E+00	−5.0E+00	−4.8E+00	−5.0E+00	−2.9E+00	−5.0E+00	−4.9E+00	**−5.0E+00**	−5.0E+00
	Std	2.9E−01	**0.0E+00**	**0.0E+00**	2.5E−16	3.1E−09	1.3E−07	2.9E−14	9.8E−01	4.9E−07	2.5E+00	1.1E−02	7.1E−02	8.5E−16	6.6E−06
F7	Avg	**9.3E−04**	2.8E+01	2.1E+00	1.1E+01	2.8E−03	2.2E+01	2.2E+01	5.5E−01	2.4E+01	1.3E+01	1.8E+02	4.8E+02	2.3E+01	1.2E+00
	Std	**1.2E−03**	4.0E−01	7.1E+00	1.2E+01	3.9E−03	1.1E+01	7.6E+00	3.9E+00	3.6E−01	1.4E+01	2.9E+02	5.0E+02	1.2E+00	3.9E+00
F8	Avg	6.6E−148	2.3E−03	**0.0E+00**	8.6E−116	2.2E−89	3.5E−01	1.7E−02	**0.0E+00**	3.7E−05	**0.0E+00**	6.9E+03	6.1E+01	3.2E−64	8.4E−309
	Std	4.6E−147	6.9E−03	**0.0E+00**	6.1E−115	1.1E−88	5.6E−01	2.6E−02	**0.0E+00**	7.5E−05	**0.0E+00**	1.9E+03	2.1E+01	1.6E−64	**0.0E+00**
F9	Avg	5.1E−76	8.1E−48	1.2E−300	5.4E−129	4.3E−54	1.3E−05	2.6E−06	**0.0E+00**	3.2E−12	**0.0E+00**	2.5E+00	8.3E+00	3.0E−87	3.7E−163
	Std	3.6E−75	5.7E−47	**0.0E+00**	3.8E−128	2.7E−53	2.5E−05	6.0E−06	**0.0E+00**	3.5E−12	**0.0E+00**	3.7E−01	9.4E+00	7.0E−87	2.2E−162
F10	Avg	1.0E−60	1.5E−02	**0.0E+00**	8.4E−115	2.6E−52	1.3E−01	6.7E−02	1.1E−302	2.5E−09	**0.0E+00**	2.5E+00	1.7E+00	1.3E−39	1.2E−172
	Std	7.3E−60	2.0E−02	**0.0E+00**	4.7E−114	1.7E−51	1.0E−01	5.5E−02	**0.0E+00**	1.2E−09	**0.0E+00**	4.0E−01	1.3E+00	2.2E−39	**0.0E+00**
F11	Avg	2.5E−57	1.1E−54	**0.0E+00**	7.8E−121	7.9E−54	1.9E−03	2.3E−02	**0.0E+00**	2.6E−12	**0.0E+00**	2.5E+00	8.4E+30	9.0E−87	6.7E−156
	Std	1.6E−56	7.4E−54	**0.0E+00**	3.9E−120	5.4E−53	3.1E−03	5.1E−02	**0.0E+00**	1.8E−12	**0.0E+00**	4.8E−01	6.0E+31	4.1E−86	4.7E−155
F12	Avg	**0.0E+00**	4.6E−26	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.0E−26	3.4E−36	**0.0E+00**	2.7E−92	**0.0E+00**	6.9E−04	1.2E+06	**0.0E+00**	**0.0E+00**
	Std	**0.0E+00**	3.2E−25	**0.0E+00**	**0.0E+00**	**0.0E+00**	7.0E−26	1.7E−35	**0.0E+00**	1.2E−91	**0.0E+00**	3.4E−03	8.2E+06	**0.0E+00**	**0.0E+00**
F13	Avg	1.4E−147	1.2E−26	**0.0E+00**	2.6E−256	1.1E−104	8.9E−13	1.6E−18	**0.0E+00**	4.3E−23	**0.0E+00**	4.1E−01	2.3E+00	1.7E−140	**0.0E+00**
	Std	9.9E−147	8.5E−26	**0.0E+00**	**0.0E+00**	4.7E−104	1.8E−12	7.6E−18	**0.0E+00**	7.1E−23	**0.0E+00**	2.0E−01	7.5E+00	9.5E−140	**0.0E+00**
F14	Avg	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	7.4E−01	7.0E−01	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.0E−02	1.0E+01	**0.0E+00**	**0.0E+00**
	Std	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.0E+00	9.7E−01	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.4E−01	1.3E+01	**0.0E+00**	**0.0E+00**
F15	Avg	1.4E−139	4.4E−80	**0.0E+00**	9.6E−249	2.4E−102	1.3E−11	1.2E−15	**0.0E+00**	5.4E−22	**0.0E+00**	6.5E+00	6.0E+01	1.7E−139	7.5E−282
	Std	9.9E−139	2.3E−79	**0.0E+00**	**0.0E+00**	1.2E−101	2.7E−11	6.8E−15	**0.0E+00**	6.4E−22	**0.0E+00**	3.7E+00	1.9E+02	8.8E−139	**0.0E+00**
F16	Avg	7.9E−82	2.1E+02	**0.0E+00**	3.7E−41	1.6E−69	2.8E+00	4.3E−08	**0.0E+00**	2.0E−03	**0.0E+00**	2.6E+02	7.2E−02	1.2E−65	**0.0E+00**
	Std	5.6E−81	5.4E+01	**0.0E+00**	2.6E−40	1.1E−68	2.0E+01	1.4E−07	**0.0E+00**	2.1E−03	**0.0E+00**	5.6E+01	3.1E−02	4.1E−66	**0.0E+00**
F17	AVG	**8.9E−16**	**8.9E−16**	**8.9E−16**	**8.9E−16**	**8.9E−16**	3.3E−01	2.5E−12	**8.9E−16**	2.0E−12	**8.9E−16**	2.6E−01	2.1E+00	4.1E−15	**8.9E−16**
	STD	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	5.5E−01	3.5E−12	**0.0E+00**	1.0E−12	**0.0E+00**	7.7E−02	2.7E+00	1.1E−15	**0.0E+00**
F18	AVG	1.4E−05	**0.0E+00**	1.3E−11	1.6E−122	1.0E−54	1.4E−06	1.5E−04	**0.0E+00**	8.3E−14	**0.0E+00**	1.1E−02	4.5E+00	6.8E−13	7.9E−191
	STD	5.0E−05	**0.0E+00**	4.5E−11	1.2E−121	6.8E−54	1.7E−06	5.9E−04	**0.0E+00**	1.0E−13	**0.0E+00**	2.2E−03	5.7E+00	2.2E−12	**0.0E+00**
F19	AVG	6.4E−115	1.1E−38	**0.0E+00**	2.6E−285	1.8E−99	4.4E−07	2.6E−03	**0.0E+00**	2.3E−19	**0.0E+00**	4.0E+04	5.1E+05	6.2E−137	3.1E−134
	STD	3.6E−114	7.8E−38	**0.0E+00**	**0.0E+00**	8.5E−99	8.8E−07	1.1E−02	**0.0E+00**	2.3E−19	**0.0E+00**	5.0E+04	9.3E+05	3.9E−136	2.2E−133
F20	AVG	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	8.9E−03	8.9E−03	**0.0E+00**	**0.0E+00**	**0.0E+00**	5.3E−04	3.8E−03	**0.0E+00**	**0.0E+00**
	STD	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	3.5E−02	3.5E−02	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.2E−04	1.5E−02	**0.0E+00**	**0.0E+00**
F21	AVG	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.4E−03	2.1E−03	**0.0E+00**	**0.0E+00**	**0.0E+00**	3.7E−02	3.4E−01	2.0E−03	**0.0E+00**
	STD	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	4.8E−03	4.7E−03	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.0E−02	4.2E−01	5.2E−03	**0.0E+00**
F22	AVG	5.2E−02	7.0E−01	5.1E−02	9.5E−02	5.0E−02	4.2E−01	3.0E−01	**−2.2E−18**	3.7E−01	3.7E−03	7.4E−01	1.9E+00	3.3E−01	3.0E−01
	STD	4.5E−02	7.9E−02	2.7E−01	1.3E−01	6.4E−02	1.3E−01	1.2E−01	**1.6E−17**	5.0E−02	3.9E−03	1.8E−01	6.7E+00	4.7E−02	1.6E−01
F23	AVG	8.8E−05	2.5E+00	**1.5E−32**	7.2E−07	7.8E−06	1.4E−01	3.6E−03	**1.5E−32**	4.7E−02	1.4E−07	1.9E−03	4.7E−03	5.9E−01	6.2E−04
	STD	1.5E−04	7.8E−02	**0.0E+00**	9.5E−07	9.6E−06	2.0E−01	1.8E−02	**0.0E+00**	6.3E−02	2.5E−07	9.2E−04	1.7E−02	1.9E−01	7.2E−04
F24	AVG	2.4E−07	**0.0E+00**	3.1E−08	2.0E−05	3.2E−04	7.6E+00	7.0E+00	1.9E−09	1.7E+00	6.0E−07	3.0E+00	1.2E+01	1.3E+00	6.9E−05
	STD	4.5E−07	**0.0E+00**	9.1E−08	3.4E−05	1.7E−03	3.6E−01	4.0E−01	6.4E−09	9.8E−01	9.0E−07	5.3E−01	2.5E−01	1.0E+00	1.6E−04
F25	AVG	7.5E−07	4.0E−01	**1.6E−32**	2.9E−09	1.7E−06	2.1E−02	6.2E−03	**1.6E−32**	6.3E−10	5.9E−09	2.8E−03	4.3E+00	5.5E−03	6.1E−04
	STD	1.8E−06	4.1E−02	**0.0E+00**	3.1E−09	2.2E−06	7.8E−02	2.5E−02	**0.0E+00**	2.6E−10	8.9E−09	4.0E−03	6.5E+00	5.3E−03	8.5E−04
F26	AVG	6.6E−06	2.8E+00	**1.3E−32**	4.2E−08	1.5E−05	2.2E−03	6.6E−04	**1.3E−32**	6.6E−04	9.0E−08	2.0E−02	2.3E−01	2.3E−01	2.7E−03
	STD	1.1E−05	1.0E−01	**2.8E−48**	3.3E−08	2.0E−05	4.4E−03	2.6E−03	**2.8E−48**	2.6E−03	2.0E−07	1.0E−02	4.8E−01	1.3E−01	3.8E−03
F27	AVG	**0.0E+00**	**0.0E+00**	1.2E+00	**0.0E+00**	**0.0E+00**	6.7E+00	1.2E+01	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.0E−01	1.4E+01	**0.0E+00**	**0.0E+00**
	STD	**0.0E+00**	**0.0E+00**	5.9E+00	**0.0E+00**	**0.0E+00**	2.1E+00	3.1E+00	**0.0E+00**	**0.0E+00**	**0.0E+00**	9.2E−02	6.6E+00	**0.0E+00**	**0.0E+00**
F28	AVG	3.8E−04	1.0E−01	6.2E−02	1.1E−121	7.9E−53	4.0E−01	5.2E−01	**0.0E+00**	1.2E−01	**0.0E+00**	1.4E+00	6.6E−01	9.4E−02	7.2E−154
	STD	2.7E−03	7.7E−09	4.9E−02	7.9E−121	4.9E−52	9.6E−02	1.2E−01	**0.0E+00**	4.0E−02	**0.0E+00**	3.2E−01	1.2E−01	2.4E−02	5.1E−153

(*continued on next page*)

Table 3 (*continued*).

F29	AVG	1.5E−05	2.7E+00	**1.3E−32**	6.3E−08	6.2E−05	3.1E−03	2.9E−02	**1.3E−32**	1.1E−03	2.0E−07	1.6E+00	1.7E+06	2.9E−01	2.3E−02
	STD	2.4E−05	4.7E−01	**2.8E−48**	5.8E−08	7.9E−05	9.4E−03	6.3E−02	**2.8E−48**	3.3E−03	9.2E−07	5.9E−01	7.9E+06	1.8E−01	2.5E−02
F30	AVG	−4.5E+02	−2.3E+02	−4.5E+02	−4.5E+02	−4.4E+02	−4.1E+02	−3.8E+02	**−4.5E+02**	−3.8E+02	−4.5E+02	−4.5E+02	−1.4E+02	−4.1E+02	−4.5E+02
	STD	3.6E−03	1.3E+01	1.3E−03	1.8E−02	2.8E+01	7.4E+00	8.4E+00	**4.0E−04**	1.7E+01	5.2E−04	2.2E−02	5.4E+01	1.5E+01	9.2E−02
F31	AVG	2.9E−39	**0.0E+00**	**0.0E+00**	1.4E−58	3.6E−28	1.9E−01	8.0E−02	**0.0E+00**	1.2E−06	**0.0E+00**	4.7E+00	1.3E+01	5.2E−46	4.8E−54
	STD	4.6E−39	**0.0E+00**	**0.0E+00**	9.5E−58	1.1E−27	4.3E−01	2.3E−01	**0.0E+00**	8.4E−07	**0.0E+00**	4.7E−01	7.7E+00	8.3E−46	3.4E−53
F32	AVG	−1.2E+03	−4.2E+02	**−1.2E+03**	−1.2E+03	−1.2E+03	−1.0E+03	−1.0E+03	−1.2E+03	−1.1E+03	−1.2E+03	−1.2E+03	−8.0E+02	−1.0E+03	−1.2E+03
	STD	2.8E−03	2.5E+01	**0.0E+00**	4.4E−04	5.0E−03	3.7E+01	2.9E+01	8.9E−06	3.1E+01	5.2E−05	3.1E+00	2.7E+02	6.4E+01	5.5E−02
F33	AVG	**0.0E+00**	**0.0E+00**	4.6E−02	**0.0E+00**	**0.0E+00**	3.0E−02	4.4E−02	**0.0E+00**	2.7E−03	**0.0E+00**	6.1E−04	6.2E−02	**0.0E+00**	**0.0E+00**
	STD	**0.0E+00**	**0.0E+00**	9.2E−02	**0.0E+00**	**0.0E+00**	1.1E−02	2.7E−02	**0.0E+00**	1.0E−02	**0.0E+00**	2.3E−04	8.2E−02	**0.0E+00**	**0.0E+00**
F34	AVG	7.7E−05	1.4E−01	3.0E−02	2.2E−28	1.5E−13	**0.0E+00**	**0.0E+00**	1.3E−05	9.0E−22	6.3E−06	8.3E−01	6.2E−02	1.6E−14	7.4E−10
	STD	7.3E−05	3.2E−01	1.5E−01	1.5E−27	1.0E−12	**0.0E+00**	**0.0E+00**	3.9E−05	1.8E−21	9.6E−06	1.9E+00	2.1E−01	5.7E−14	2.1E−09
F35	AVG	3.0E−04	4.4E−01	**0.0E+00**	3.4E−05	1.3E−04	3.6E−28	**0.0E+00**	**0.0E+00**	2.1E−11	2.4E−06	2.8E+00	1.2E−02	3.7E−02	1.1E−03
	STD	3.8E−04	3.4E−01	**0.0E+00**	9.8E−05	4.6E−04	2.4E−27	**0.0E+00**	**0.0E+00**	2.5E−11	4.2E−06	1.4E+00	1.7E−02	1.8E−02	2.2E−03
F36	AVG	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.6E+00	1.8E+01	**0.0E+00**	**0.0E+00**
	STD	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.8E+00	2.3E+01	**0.0E+00**	**0.0E+00**
F37	AVG	−2.1E+00	−2.1E+00	−2.1E+00	**−2.1E+00**	−2.1E+00	**−2.1E+00**	**−2.1E+00**	**−2.1E+00**	**−2.1E+00**	−2.1E+00	−2.1E+00	−2.1E+00	**−2.1E+00**	−2.1E+00
	STD	4.3E−06	4.8E−07	2.0E−15	9.0E−16	4.5E−09	9.0E−16	9.0E−16	4.8E−10	**6.2E−16**	2.3E−09	5.8E−06	1.3E−05	7.5E−16	8.2E−11
F38	AVG	**−1.0E+00**	**−1.0E+00**	−9.8E−01	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	−1.0E+00	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	−9.1E−01	−9.9E−01	−9.8E−01	**−1.0E+00**
	STD	**0.0E+00**	**0.0E+00**	2.8E−02	**0.0E+00**	**0.0E+00**	**0.0E+00**	9.0E−03	**0.0E+00**	**0.0E+00**	**0.0E+00**	7.3E−02	1.4E−02	2.8E−02	**0.0E+00**
F39	AVG	1.3E+00	8.0E+00	1.0E+00	1.2E+00	1.2E+00	**1.0E+00**	1.0E+00	1.0E+00	1.0E+00	1.0E+00	1.0E+00	1.0E+00	2.8E+00	1.0E+00
	STD	5.3E−01	4.4E+00	1.4E−01	1.4E+00	7.4E−01	**0.0E+00**	1.4E−01	2.4E−11	1.3E−16	7.1E−11	1.2E−10	2.8E−10	3.4E+00	4.8E−14
F40	AVG	−3.9E+00	−3.9E+00	**−3.9E+00**	**−3.9E+00**	−3.9E+00	**−3.9E+00**	**−3.9E+00**	−3.9E+00	**−3.9E+00**	−3.9E+00	−3.6E+00	−3.9E+00	**−3.9E+00**	−3.9E+00
	STD	2.5E−03	2.9E−03	8.0E−16	1.3E−15	5.9E−04	1.3E−15	1.3E−15	5.4E−05	9.5E−16	3.8E−03	7.1E−01	3.7E−06	**7.4E−16**	4.8E−08
F41	AVG	−3.2E+00	−3.1E+00	−3.3E+00	−3.3E+00	−3.2E+00	−3.3E+00	−3.3E+00	−3.3E+00	**−3.3E+00**	−3.1E+00	−3.3E+00	−3.3E+00	−3.3E+00	−3.2E+00
	STD	6.4E−02	6.9E−02	1.7E−02	6.0E−02	8.1E−02	4.6E−02	5.4E−02	1.6E−03	**1.9E−12**	1.0E−01	5.1E−02	6.0E−02	5.8E−02	5.3E−02
F42	AVG	5.2E−03	3.8E−08	**0.0E+00**	7.6E−296	2.0E−04	1.4E−54	1.8E−102	1.6E−07	1.2E−19	2.1E−06	2.3E−04	1.8E−03	1.1E−17	4.3E−11
	STD	4.9E−03	5.5E−08	**0.0E+00**	**0.0E+00**	5.5E−04	4.7E−54	6.1E−102	1.1E−06	2.4E−19	2.9E−06	3.6E−04	2.4E−03	6.7E−17	9.6E−11
F43	AVG	−1.9E+01	−1.9E+01	−1.9E+01	**−1.9E+01**	−1.9E+01	**−1.9E+01**	**−1.9E+01**	−1.9E+01	−1.9E+01	−1.9E+01	−1.9E+01	−1.9E+01	−1.9E+01	−1.9E+01
	STD	1.4E−03	1.2E+00	5.2E−09	2.4E−15	8.4E−15	**1.8E−15**	1.9E−15	1.4E−04	2.1E−13	2.5E−03	4.4E−06	4.4E−05	5.0E−15	1.9E−07
F44	AVG	−1.0E+00	−1.0E+00	−1.0E+00	**−1.0E+00**	−1.0E+00	**−1.0E+00**	**−1.0E+00**	−1.0E+00	**−1.0E+00**	−1.0E+00	−1.0E+00	−1.0E+00	**−1.0E+00**	−1.0E+00
	STD	1.6E−04	8.2E−08	5.0E−13	**2.2E−16**	3.2E−12	2.3E−16	**2.2E−16**	1.7E−06	4.5E−16	8.0E−07	2.1E−06	8.1E−06	5.5E−16	5.4E−11
F45	AVG	4.6E−04	1.4E−02	8.8E−04	6.0E−04	3.3E−04	7.1E−04	**3.1E−04**	3.1E−04	3.1E−04	3.3E−04	1.4E−02	5.5E−03	3.3E−04	4.3E−04
	STD	1.4E−04	2.8E−02	2.8E−03	2.6E−04	2.5E−05	2.8E−03	**1.0E−19**	2.1E−06	2.6E−15	6.0E−05	1.5E−02	8.4E−03	1.3E−04	2.1E−04
F46	AVG	−1.0E+01	−4.0E+00	**−1.0E+01**	−1.0E+01	−5.7E+00	−1.0E+01	−9.9E+00	−1.0E+01	−1.0E+01	−1.0E+01	−5.8E+00	−5.2E+00	−8.4E+00	−1.0E+01
	STD	4.9E−03	8.3E−01	**1.3E−11**	7.2E−01	1.7E+00	1.0E+00	1.0E+00	1.9E−07	1.4E−11	1.1E−04	3.1E+00	3.1E+00	2.4E+00	7.2E−05
F47	AVG	−1.0E+01	−4.3E+00	−1.0E+01	−1.0E+01	−5.8E+00	**−1.0E+01**	−1.0E+01	−1.0E+01	−1.0E+01	−1.0E+01	−5.0E+00	−5.8E+00	−8.4E+00	−1.0E+01
	STD	5.0E−03	1.3E+00	1.5E−12	7.5E−01	1.8E+00	**1.6E−15**	7.5E−01	2.2E−05	2.0E−11	1.3E−04	3.2E+00	3.4E+00	2.6E+00	1.2E−04
F48	AVG	−1.1E+01	−4.2E+00	−1.1E+01	−1.0E+01	−5.9E+00	**−1.1E+01**	**−1.1E+01**	−1.1E+01	−1.1E+01	−1.1E+01	−4.5E+00	−5.0E+00	−8.7E+00	−1.1E+01
	STD	2.4E−03	1.4E+00	5.6E−12	1.1E+00	1.9E+00	**1.7E−15**	1.8E−15	1.8E−05	2.1E−11	1.2E−04	3.0E+00	3.1E+00	2.6E+00	1.1E−04
F49	AVG	5.7E−02	3.1E−01	5.6E−03	1.5E−01	5.5E−02	2.3E−03	**2.3E−03**	1.4E−02	2.3E−03	1.8E−02	6.5E+00	8.4E−02	1.5E−02	2.1E−02
	STD	4.1E−02	1.1E+00	4.1E−03	7.2E−01	4.7E−02	**1.3E−17**	1.3E−17	3.4E−03	1.9E−10	1.3E−02	9.6E+00	8.9E−02	1.0E−02	2.9E−02
F50	AVG	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.4E−119	9.5E−119	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.7E−04	2.6E−04	**0.0E+00**	**0.0E+00**
	STD	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	6.3E−119	2.3E−118	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.6E−04	2.5E−04	**0.0E+00**	**0.0E+00**

Table 4
Wilcoxon rank-sum test and ranking of the compared algorithms on 30D and fixed problems.

Algorith	+ /−/≈	Avg	Rank
AO	38/1/11	4.96	8
AOA	31/4/15	7.30	12
CCAA	11/9/30	2.48	2
HGS	30/6/14	3.44	4
HHO	33/4/13	4.76	7
LSH	37/11/2	6.06	11
LSP	35/10/5	5.70	10
MmCAA	−/ − /−	2.32	1
MPA	28/14/8	4.74	6
MSA	20/1/29	3.26	3
PROS	46/2/2	8.76	13
PSS	46/3/1	9.56	14
RWO	33/7/10	5.16	9
SMA	23/7/20	3.70	5

Table 5
Wilcoxon rank-sum test and ranking of the compared algorithms on 500D and fixed problems.

Algorith	+/−/≈	Avg	Rank
AO	39/0/11	4.76	7
AOA	39/2/9	7.84	12
CCAA	14/9/27	2.60	2
HGS	28/7/15	3.76	4
HHO	35/3/12	4.54	5
LSH	37/10/3	7.14	11
LSP	39/8/3	6.74	10
MmCAA	−/−/−	2.24	1
MPA	28/12/10	4.78	8
MSA	20/2/28	3.61	3
PROS	45/2/3	8.46	13
PSS	45/2/3	10.20	14
RWO	31/5/14	4.80	9
SMA	29/5/16	4.56	6

In the first experiment, the first 33 functions in 30 dimensions and the last 17 functions of fixed dimensions listed in Table 1 were used, resulting in 30 independent runs for each algorithm and for obtaining the average value and the standard deviation for each test function.

For the MmCAA, $n_S = 12$ and $n_{ne} = 6$ were taken. The remaining algorithms used $(n_S)(n_{ne}) = 72$ individuals. All algorithms applied $n_{it} = 500$ iterations. Table 3 presents the results of all algorithms for unimodal, multimodal and fixed-dimension functions. The best results are shown in bold with orange background.

In the case of unimodal functions, the MmCAA obtained 13 of the 16 best average values, performing similar to the CCAA and MSA and superior to the other algorithms. The MmCAA also obtained the best values with respect to the standard deviation for 14 of the 16 functions, demonstrating a good exploitability feature. In case of multimodal functions, the MmCAA obtained best average values for 15 functions out of the 17. This performance is superior to the rest of the algorithms, where the CCAA has 10 and MSA has only 9 better average values. The MmCAA also reached the best values in the standard deviation, showing yet again a good exploration ability. For the remaining functions with fixed dimensions, the MmCAA generated 5 of the best average values, falling behind the LSH, LSP, HGS, MPA and CCAA algorithms. The MmCAA obtained 4 of the best values with respect to standard deviation, showing a competitive balance for exploration and exploitation actions in this type of problems.

Table 4 applies the Wilcoxon rank-sum test to compare the MmCAA with the other algorithms, function by function. The + symbol indicates that the MmCAA obtained a better statistically significant result, the ≈ symbol describes that there is no significant difference, and the - symbol represents a comparatively worse result. The next column shows the average rank of each algorithm obtained by considering each function individually. The last row shows the final position of each algorithm according to this rank. The MmCAA has the best overall range (2.32), followed by the CCAA (2.48) and the MSA (3.26). In this experiment, the MmCAA compares favorably with all the other algorithms and has a better average rank, considering all the test functions.

5.4. Second experiment with test functions in 500 dimensions and fixed dimensions

The second experiment used the same 33 scalable test functions, but with 500 dimensions, and took the last 17 test functions with fixed dimensions again. The same statistical study was carried out, taking the average value and the standard deviation of each algorithm in 30 independent runs and then applying the Wilcoxon rank-sum test and analyzing the statistically significant differences between the performance of the MmCAA with the rest of the algorithms.

Same as in the first experiment, the MmCAA used $n_S = 12$ and $n_{ne} = 6$ to define the number of smart-cells and the neighborhood size, and the other algorithms used $(n_S)(n_{ne}) = 72$ individuals. All algorithms applied $n_{it} = 500$ iterations per run. Table 6 shows the corresponding results obtained.

The MmCAA obtained 14 of the 16 best average values for unimodal functions and the lowest standard deviation for 15 of the 16 problems, outperforming all the other metaheuristics. In the case of multimodal test functions, the MmCAA scored 16 of the 17 best average values, beating all other methods. The MmCAA also achieved the best standard deviation values in 16 of the 17 cases. In the second run, to optimize functions with fixed dimensions, the MmCAA generated 6 of the best average values and 5 of the best standard deviation values. These results yet again demonstrate a good balance between exploring and exploiting the search space again, behind the HGS, LSH and LSP methods now, with performance comparable to the CCAA and MPA and ahead of the other remaining 8 algorithms.

Table 5 presents the Wilcoxon rank-sum test for the second experiment, the average rank of each metaheuristic and its overall place. In all cases, the MmCAA obtained a positive balance in the Wilcoxon rank-sum test and has the best average rank (2.24), followed by the CCAA (2.60) and the MSA (3.61). The competitiveness of the MmCAA is comparable to recent algorithms that are already recognized for their excellent results.

Compared to the CCAA, an algorithm with the same inspiration, the MmCAA performed better, especially in the test functions with 500 dimensions. Regarding its operation and computational implementation, the MmCAA uses only 3 parameters to operate its rules ($prop$, $lower_r$ and $upper_r$) in contrast to the 7 parameters required by the CCAA. Additionally, the MmCAA only uses 6 rules for its operation, unlike the 10 rules used in the CCAA. In this way, the MmCAA is a simpler algorithm to implement and tune than the CCAA, in addition to being based on a single type of behavior such as majority CA; this shows the possibility of proposing new metaheuristics based on other types of complex behaviors observed in CA for further research.

Fig. 8 presents several examples of convergence curves for some problems with 30, 500 and fixed dimensions. Only the curves of the best algorithms are presented in each instance.

5.5. Convergence behavior of the MmCAA

In metaheuristics, a solution must make abrupt position changes at the beginning of the algorithm when exploring the search space, and these changes must be reduced as the optimization process advances, thereby demonstrating the exploitation action performed by the algorithm. This dynamic causes the metaheuristic to converge during the optimization process (Van Den Bergh & Engelbrecht, 2006).

An experiment was conducted to analyze the convergence of the MmCAA and show that it develops a correct balance between the

Table 6
Metaheuristics compared with MmCAA on 500 and fixed dimensional problems.

Fn	Stats	AO	AOA	CCAA	HGS	HHO	LSH	LSP	MmCAA	MPA	MSA	PROS	PSS	RWO	SMA
F1	Avg	7.5E−101	2.5E+17	**0.0E+00**	6.6E−248	1.8E−104	1.2E+03	1.2E+02	**0.0E+00**	1.3E−18	**0.0E+00**	2.4E+00	2.5E+04	3.5E−84	4.8E−279
	Std	2.0E−100	2.6E+17	**0.0E+00**	**0.0E+00**	1.3E−103	1.8E+02	3.3E+01	**0.0E+00**	8.5E−19	**0.0E+00**	2.1E−01	1.1E+04	9.2E−84	**0.0E+00**
F2	Avg	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	−3.0E−01	−3.6E−01	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	−9.1E−01	−5.7E−13	**−1.0E+00**	**−1.0E+00**
	Std	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	5.4E−02	7.3E−02	**0.0E+00**	**0.0E+00**	**0.0E+00**	8.4E−03	4.6E−13	**0.0E+00**	**0.0E+00**
F3	Avg	1.7E−105	2.0E+03	**0.0E+00**	1.2E−219	7.6E−103	2.1E+03	3.1E+02	**0.0E+00**	1.8E−17	**0.0E+00**	4.3E+02	1.2E+05	7.3E−68	6.0E−201
	Std	6.0E−105	2.1E+02	**0.0E+00**	**0.0E+00**	5.1E−102	3.4E+02	4.4E+01	**0.0E+00**	1.4E−17	**0.0E+00**	5.4E+01	6.5E+03	6.8E−68	**0.0E+00**
F4	Avg	2.9E−15	**0.0E+00**	5.2E−207	1.3E−103	2.0E−127	1.5E−18	6.3E−16	**0.0E+00**	3.5E−61	**0.0E+00**	3.9E−04	1.7E−05	**0.0E+00**	**0.0E+00**
	Std	1.4E−14	**0.0E+00**	**0.0E+00**	8.2E−103	1.4E−126	6.2E−18	4.0E−15	**0.0E+00**	2.4E−60	**0.0E+00**	6.1E−04	1.0E−04	**0.0E+00**	**0.0E+00**
F5	Avg	5.1E−310	1.0E−40	**0.0E+00**	**0.0E+00**	1.4E−200	6.5E+01	4.4E+01	**0.0E+00**	3.4E−34	**0.0E+00**	2.7E−01	9.9E+03	6.9E−142	**0.0E+00**
	Std	**0.0E+00**	7.3E−40	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.5E+01	9.0E+00	**0.0E+00**	4.1E−34	**0.0E+00**	9.1E−02	6.3E+02	1.2E−141	**0.0E+00**
F6	Avg	−5.5E−02	−5.0E+00	−4.0E−01	−5.0E+00	−2.7E+00	5.0E+00	−7.0E−01	−9.9E−02	−5.0E+00	−1E−162	−2.7E+00	3.3E+01	**−5.0E+00**	−2.4E+00
	Std	2.7E−01	7.6E−05	1.4E+00	1.5E−08	2.5E+00	6.0E−01	2.4E−01	7.0E−01	1.8E−03	**0.0E+00**	1.7E−01	5.3E−01	6.2E−16	2.4E+00
F7	Avg	2.8E−02	5.0E+02	3.9E+01	2.3E+02	4.9E−02	9.2E+06	2.6E+05	**0.0E+00**	5.0E+02	1.5E+02	8.0E+04	1.5E+09	5.0E+02	4.9E+01
	Std	4.9E−02	5.8E−02	1.4E+02	2.5E+02	6.1E−02	1.9E+06	7.4E+04	**0.0E+00**	3.6E−01	2.3E+02	1.5E+04	7.3E+07	8.7E−01	7.8E+01
F8	Avg	1.9E−116	3.4E+01	1.1E−07	3.4E+04	2.2E−46	2.3E+05	1.5E+05	**0.0E+00**	7.2E+03	**0.0E+00**	2.6E+06	4.1E+06	7.2E−61	1.4E−30
	Std	9.5E−116	2.5E+01	7.4E−07	2.4E+05	1.5E−45	4.1E+04	3.3E+04	**0.0E+00**	6.4E+03	**0.0E+00**	2.9E+05	5.3E+05	9.5E−62	9.7E−30
F9	Avg	2.5E−61	7.8E+00	1.3E−299	5.4E−124	1.4E−51	3.0E+03	2.0E+03	**0.0E+00**	7.6E−09	**0.0E+00**	6.8E+02	1.3E+04	1.1E−53	5.4E−06
	Std	1.8E−60	5.4E−01	**0.0E+00**	3.8E−123	8.7E−51	1.6E+02	1.3E+02	**0.0E+00**	6.1E−09	**0.0E+00**	3.1E+01	2.3E+02	2.6E−53	3.8E−05
F10	Avg	6.8E−77	1.7E−01	**0.0E+00**	6.5E−118	9.0E−52	3.6E+01	3.0E+01	2.9E−241	4.0E−05	**0.0E+00**	8.7E+01	9.1E+01	2.9E−32	1.0E−50
	Std	4.8E−76	1.3E−02	**0.0E+00**	3.2E−117	4.4E−51	2.6E+00	3.0E+00	**0.0E+00**	2.6E−05	**0.0E+00**	1.7E+00	2.6E+00	7.1E−34	6.1E−50
F11	Avg	6.3E−78	7.9E+00	**0.0E+00**	9.7E+269	3.4E−53	3.2E−5	8.1E+152	**0.0E+00**	4.8E+216	3.2E−43	4.1E−15	6.4E−63	1.3E−56	2.8E+00
	Std	1.8E−77	4.3E−01	**0.0E+00**	5.4E−01	1.7E−52	4.6E+01	4.8E−01	**0.0E+00**	7.1E+01	3.8E−02	6.2E+01	2.7E−02	7.2E−56	1.3E+01
F12	Avg	**0.0E+00**	5.3E−08	**0.0E+00**	**0.0E+00**	**0.0E+00**	3.6E+15	3.3E+14	**0.0E+00**	1.3E−58	**0.0E+00**	9.2E+10	1.8E+20	6.0E−322	**0.0E+00**
	Std	**0.0E+00**	2.5E−08	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.9E+15	2.3E+14	**0.0E+00**	5.3E−58	**0.0E+00**	3.4E+11	1.9E+19	**0.0E+00**	**0.0E+00**
F13	Avg	3.6E−154	5.9E−01	**0.0E+00**	2.8E−219	9.4E−102	3.4E+04	1.4E+04	**0.0E+00**	5.5E−16	**0.0E+00**	1.9E+03	5.7E+05	3.8E−81	1.4E−109
	Std	2.5E−153	3.7E−02	**0.0E+00**	**0.0E+00**	3.6E−101	4.3E+03	2.8E+03	**0.0E+00**	3.5E−16	**0.0E+00**	1.7E+02	1.6E+04	6.8E−81	9.7E−109
F14	Avg	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	3.8E+04	2.0E+04	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.9E+03	5.7E+05	**0.0E+00**	**0.0E+00**
	Std	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	4.4E+03	4.0E+03	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.6E+02	1.5E+04	**0.0E+00**	**0.0E+00**
F15	Avg	5.6E−152	1.4E+02	**0.0E+00**	1.6E−226	6.4E−101	6.9E+06	3.0E+06	**0.0E+00**	1.1E−13	**0.0E+00**	4.7E+05	1.2E+08	8.8E−79	2.7E−03
	Std	2.9E−151	6.9E+00	**0.0E+00**	**0.0E+00**	2.7E−100	6.6E+05	5.8E+05	**0.0E+00**	6.5E−14	**0.0E+00**	4.9E+04	4.8E+06	1.8E−78	1.5E−02
F16	Avg	4.6E+02	7.3E+07	1.9E−03	3.1E+03	2.3E+03	5.0E+03	2.2E+03	**0.0E+00**	4.7E+01	**0.0E+00**	7.4E+03	8.4E+03	1.0E−48	1.7E+03
	Std	1.3E+03	3.3E+08	4.9E−03	2.4E+03	3.0E+03	7.4E+02	3.9E+02	**0.0E+00**	8.6E+00	**0.0E+00**	3.5E+02	5.2E+02	6.1E−48	3.8E+03
F17	AVG	**8.9E−16**	9.5E−03	**8.9E−16**	**8.9E−16**	**8.9E−16**	1.3E+01	8.5E+00	**8.9E−16**	1.3E−09	**8.9E−16**	4.2E+00	2.1E+01	4.3E−15	**8.9E−16**
	STD	**0.0E+00**	4.8E−04	**0.0E+00**	**0.0E+00**	**0.0E+00**	4.7E−01	3.3E−01	**0.0E+00**	4.1E−10	**0.0E+00**	1.0E−01	2.3E−01	7.0E−16	**0.0E+00**
F18	AVG	2.9E−04	1.7E−05	3.1E−11	1.6E−117	1.2E−52	1.4E+02	7.0E+01	**0.0E+00**	1.4E−10	4.5E−06	1.0E+01	7.1E+02	9.7E−14	3.3E−03
	STD	9.1E−04	4.4E−05	1.1E−10	1.1E−116	7.5E−52	1.0E+01	8.6E+00	**0.0E+00**	1.1E−10	3.2E−05	8.0E−01	2.7E+01	5.1E−13	2.3E−02
F19	AVG	2.1E−148	1.4E+04	**0.0E+00**	8.0E−226	3.1E−94	2.2E+08	1.4E+08	**0.0E+00**	4.7E−12	**0.0E+00**	1.4E+08	4.6E+09	1.2E−77	2.6E−01
	STD	1.5E−147	2.1E+03	**0.0E+00**	**0.0E+00**	2.2E−93	4.2E+07	2.5E+07	**0.0E+00**	3.3E−12	**0.0E+00**	3.8E+07	3.7E+08	1.9E−77	1.6E+00
F20	AVG	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.6E+01	2.8E+01	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.4E+00	1.0E+02	**0.0E+00**	**0.0E+00**
	STD	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.8E+00	1.4E+00	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.0E−01	1.7E+00	**0.0E+00**	**0.0E+00**
F21	AVG	**0.0E+00**	2.6E−03	**0.0E+00**	**0.0E+00**	**0.0E+00**	9.2E+00	4.5E+00	**0.0E+00**	2.2E−18	**0.0E+00**	1.5E+00	1.4E+02	6.4E−04	**0.0E+00**
	STD	**0.0E+00**	2.3E−04	**0.0E+00**	**0.0E+00**	**0.0E+00**	8.7E−01	6.6E−01	**0.0E+00**	1.6E−17	**0.0E+00**	6.1E−02	4.1E+00	2.8E−03	**0.0E+00**
F22	AVG	3.9E−01	9.7E−01	4.6E−02	7.7E−01	2.5E−01	3.3E+04	1.3E+04	**−4.4E−18**	9.1E−01	7.0E−02	1.4E+03	5.7E+05	6.0E−01	8.7E−01
	STD	3.5E−01	4.0E−02	2.3E−01	7.1E−01	1.8E−01	3.1E+03	2.3E+03	**2.2E−17**	1.8E−02	1.4E−01	1.9E+02	1.6E+04	7.1E−02	7.4E−01
F23	AVG	1.8E−02	4.5E+01	**1.5E−32**	8.7E−04	2.3E−04	1.5E+02	3.0E+01	**1.5E−32**	3.4E+01	7.0E−01	7.1E+00	1.9E+03	3.1E+01	3.3E−02
	STD	2.5E−02	1.2E−01	**0.0E+00**	1.9E−03	4.0E−04	1.9E+01	3.3E+00	**0.0E+00**	9.4E−01	5.0E+00	5.4E−01	5.3E+01	1.6E+00	6.2E−02
F24	AVG	5.0E−06	3.5E+01	3.7E−06	2.9E−03	3.8E−03	2.3E+02	2.3E+02	**3.1E−08**	6.1E+01	9.8E−06	1.1E+02	2.4E+02	3.6E+01	2.4E−03
	STD	1.0E−05	3.9E+00	8.3E−06	4.5E−03	1.7E−02	8.8E−01	1.4E+00	**7.5E−08**	3.1E+01	1.7E−05	2.5E+00	1.2E+00	1.8E+01	4.6E−03
F25	AVG	9.9E−08	1.1E+00	**9.4E−34**	6.0E−07	3.1E−07	8.4E+04	1.5E+01	**9.4E−34**	2.8E−01	1.9E−09	8.4E−01	2.5E+09	2.2E−01	5.2E−04
	STD	1.4E−07	1.1E−02	**1.7E−49**	9.7E−07	3.1E−07	7.5E+04	5.8E+00	**1.7E−49**	1.9E−02	3.0E−09	8.9E−02	2.0E+08	3.0E−02	8.4E−04
F26	AVG	5.3E−05	5.0E+01	**1.3E−32**	1.4E−04	6.4E−05	5.3E+06	8.5E+03	**1.3E−32**	4.8E+01	2.0E+00	7.6E+01	5.6E+09	3.2E+01	4.1E−01
	STD	6.2E−05	2.9E−02	**2.8E−48**	2.9E−04	7.2E−05	2.0E+06	7.2E+03	**2.8E−48**	4.4E−01	9.8E+00	1.0E+01	3.9E+08	1.2E+00	9.0E−01
F27	AVG	**0.0E+00**	2.5E−06	**0.0E+00**	**0.0E+00**	**0.0E+00**	3.4E+03	7.4E+02	**0.0E+00**	**0.0E+00**	**0.0E+00**	3.2E+02	5.4E+03	**0.0E+00**	**0.0E+00**
	STD	**0.0E+00**	4.7E−06	**0.0E+00**	**0.0E+00**	**0.0E+00**	6.1E+02	9.0E+01	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.3E+01	7.7E+01	**0.0E+00**	**0.0E+00**

(*continued on next page*)

Table 6 (*continued*).

F28	AVG	1.7E−03	2.4E−01	1.6E−02	9.9E−123	3.4E−51	2.9E+01	1.7E+01	**0.0E+00**	2.0E−01	**0.0E+00**	3.2E+01	8.1E+01	8.6E−02	6.8E−23
	STD	1.2E−02	6.7E−02	3.7E−02	4.5E−122	2.3E−50	1.3E+00	1.7E+00	**0.0E+00**	5.1E−17	**0.0E+00**	1.1E+00	1.2E+00	3.5E−02	4.8E−22
F29	AVG	3.7E−04	2.9E+14	**1.3E−32**	1.5E−04	2.6E−04	5.7E+11	6.6E+11	**1.3E−32**	4.8E+01	7.4E−07	1.3E+09	1.1E+14	3.2E+01	1.6E+01
	STD	6.8E−04	6.5E+13	**2.8E−48**	4.5E−04	4.6E−04	1.3E+11	1.6E+11	**2.8E−48**	4.3E−01	3.2E−06	8.2E+08	5.4E+12	9.7E−01	1.8E+01
F30	AVG	−7.4E+03	−2.3E+03	−7.4E+03	−7.2E+03	−7.4E+03	−1.5E+03	−1.5E+03	**−7.4E+03**	−2.3E+03	−7.4E+03	−7.1E+03	−5.4E+02	−2.5E+03	−7.4E+03
	STD	3.0E−02	3.6E+01	8.5E−02	9.3E+02	7.3E−01	5.2E+01	3.7E+01	**2.1E−03**	8.1E+01	1.1E−02	2.0E+01	1.6E+02	7.6E+01	3.4E+00
F31	AVG	3.3E−38	1.8E−08	**0.0E+00**	4.0E−60	8.9E−27	1.0E+03	7.5E+02	**0.0E+00**	6.5E−05	**0.0E+00**	4.3E+02	1.9E+03	8.5E−33	5.8E−03
	STD	7.2E−38	6.5E−08	**0.0E+00**	2.1E−59	4.3E−26	3.2E+01	3.2E+01	**0.0E+00**	4.5E−05	**0.0E+00**	1.2E+01	1.5E+01	1.2E−32	3.2E−02
F32	AVG	−2.0E+04	−6.9E+02	**−2.0E+04**	−2.0E+04	−2.0E+04	5.4E+06	1.2E+06	−2.0E+04	−1.1E+04	−2.0E+04	−1.2E+03	9.3E+08	−1.1E+04	−1.9E+04
	STD	3.7E−02	4.8E+01	**1.9E−12**	2.6E−01	9.5E−02	1.0E+06	5.1E+05	8.0E−05	2.0E+02	1.0E−03	9.9E+03	4.2E+07	4.1E+02	1.3E+02
F33	AVG	**0.0E+00**	**0.0E+00**	7.1E−03	**0.0E+00**	**0.0E+00**	7.7E−01	5.1E−01	**0.0E+00**	**0.0E+00**	**0.0E+00**	8.3E−02	7.0E−01	**0.0E+00**	**0.0E+00**
	STD	**0.0E+00**	**0.0E+00**	3.5E−02	**0.0E+00**	**0.0E+00**	6.8E−03	1.8E−01	**0.0E+00**	**0.0E+00**	**0.0E+00**	3.4E−03	6.6E−03	**0.0E+00**	**0.0E+00**
F34	AVG	8.4E−05	1.3E−01	1.5E−02	1.5E−02	9.4E−16	**0.0E+00**	**0.0E+00**	2.4E−05	2.1E−21	3.8E−06	7.4E−01	4.6E−02	1.2E−14	6.4E−10
	STD	8.5E−05	3.0E−01	1.1E−01	1.1E−01	3.2E−15	**0.0E+00**	**0.0E+00**	4.8E−05	6.1E−21	4.6E−06	1.7E+00	1.8E−01	2.2E−14	1.5E−09
F35	AVG	2.2E−04	4.0E−01	**0.0E+00**	4.4E−05	6.0E−05	1.0E−29	**0.0E+00**	**0.0E+00**	2.2E−11	2.9E−06	2.4E+00	1.3E−02	3.4E−02	1.7E−03
	STD	4.1E−04	6.0E−01	**0.0E+00**	1.1E−04	1.1E−04	7.0E−29	**0.0E+00**	**0.0E+00**	2.1E−11	4.4E−06	1.7E+00	1.7E−02	1.7E−02	4.4E−03
F36	AVG	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	3.4E+00	1.4E+01	**0.0E+00**	**0.0E+00**
	STD	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	7.1E+00	2.2E+01	**0.0E+00**	**0.0E+00**
F37	AVG	−2.1E+00	−2.1E+00	−2.1E+00	**−2.1E+00**	−2.1E+00	**−2.1E+00**	**−2.1E+00**	**−2.1E+00**	**−2.1E+00**	−2.1E+00	−2.1E+00	−2.1E+00	**−2.1E+00**	−2.1E+00
	STD	3.5E−07	5.2E−07	8.6E−16	9.0E−16	5.0E−09	9.0E−16	9.0E−16	3.3E−10	**5.5E−16**	1.3E−09	4.9E−06	1.8E−05	7.4E−16	6.2E−11
F38	AVG	**−1.0E+00**	**−1.0E+00**	−9.9E−01	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	**−1.0E+00**	−9.3E−01	−9.8E−01	−9.9E−01	**−1.0E+00**
	STD	**0.0E+00**	**0.0E+00**	2.7E−02	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	6.2E−02	2.0E−02	2.4E−02	**0.0E+00**
F39	AVG	1.5E+00	9.1E+00	**1.0E+00**	1.2E+00	1.2E+00	**1.0E+00**	1.0E+00	1.0E+00	1.0E+00	1.0E+00	1.0E+00	1.0E+00	2.9E+00	1.0E+00
	STD	1.5E+00	4.1E+00	1.5E−16	1.4E+00	7.5E−01	**0.0E+00**	1.4E−01	4.1E−11	1.3E−16	5.7E−11	2.3E−10	3.4E−10	3.5E+00	6.1E−14
F40	AVG	−3.9E+00	−3.9E+00	**−3.9E+00**	**−3.9E+00**	−3.9E+00	**−3.9E+00**	**−3.9E+00**	−3.9E+00	**−3.9E+00**	−3.9E+00	−3.3E+00	−3.9E+00	**−3.9E+00**	−3.9E+00
	STD	2.4E−03	2.9E−03	8.4E−16	1.3E−15	1.5E−03	1.3E−15	1.3E−15	3.0E−05	9.5E−16	3.8E−03	9.1E−01	4.3E−06	**6.5E−16**	3.4E−08
F41	AVG	−3.2E+00	−3.1E+00	−3.3E+00	−3.3E+00	−3.2E+00	−3.3E+00	−3.3E+00	−3.3E+00	**−3.3E+00**	−3.1E+00	−3.3E+00	−3.3E+00	−3.3E+00	−3.2E+00
	STD	8.3E−02	7.8E−02	8.6E−09	6.6E−02	8.0E−02	4.2E−02	4.2E−02	1.3E−03	**2.2E−12**	1.0E−01	5.6E−02	6.0E−02	6.0E−02	5.4E−02
F42	AVG	4.5E−03	4.4E−08	**0.0E+00**	3.8E−288	1.1E−04	6.1E−55	1.8E−102	**0.0E+00**	1.0E−19	1.6E−06	2.5E−04	1.7E−03	5.3E−19	3.9E−11
	STD	4.5E−03	6.1E−08	**0.0E+00**	**0.0E+00**	2.7E−04	1.8E−54	9.3E−102	**0.0E+00**	3.2E−19	1.3E−06	4.5E−04	2.1E−03	2.2E−18	1.1E−10
F43	AVG	−1.9E+01	−1.9E+01	−1.9E+01	−1.9E+01	−1.9E+01	−1.9E+01	**−1.9E+01**	−1.9E+01	−1.9E+01	−1.9E+01	−1.9E+01	−1.9E+01	−1.9E+01	−1.9E+01
	STD	1.9E−03	1.2E+00	6.3E−09	3.7E−15	9.0E−15	4.1E−15	**1.8E−15**	2.9E−04	1.5E−13	1.3E−03	5.5E−06	8.6E−05	5.7E−15	2.5E−07
F44	AVG	−1.0E+00	−1.0E+00	−1.0E+00	**−1.0E+00**	−1.0E+00	**−1.0E+00**	**−1.0E+00**	−1.0E+00	−1.0E+00	−1.0E+00	−1.0E+00	−1.0E+00	−1.0E+00	−1.0E+00
	STD	2.2E−04	6.2E−08	5.1E−13	**2.2E−16**	9.0E−12	2.5E−16	2.4E−16	2.9E−06	2.4E−16	1.0E−06	2.5E−06	3.8E−06	3.2E−16	2.7E−11
F45	AVG	4.4E−04	1.3E−02	5.3E−04	6.1E−04	3.6E−04	3.1E−04	**3.1E−04**	3.1E−04	3.1E−04	4.0E−04	1.7E−02	5.0E−03	3.5E−04	4.6E−04
	STD	8.6E−05	1.8E−02	1.6E−03	3.0E−04	1.8E−04	**1.3E−19**	1.4E−19	1.7E−06	2.9E−15	2.6E−04	1.7E−02	7.9E−03	1.8E−04	2.1E−04
F46	AVG	−1.0E+01	−4.0E+00	−1.0E+01	**−1.0E+01**	−5.4E+00	−1.0E+01	−9.8E+00	−1.0E+01	−1.0E+01	−1.0E+01	−5.7E+00	−4.8E+00	−8.8E+00	−1.0E+01
	STD	4.8E−03	1.1E+00	5.7E−13	**3.0E−15**	1.2E+00	7.1E−01	1.2E+00	1.1E−07	1.4E−11	9.4E−05	3.3E+00	2.9E+00	2.3E+00	1.0E−04
F47	AVG	−1.0E+01	−4.4E+00	−1.0E+01	−1.0E+01	−5.2E+00	**−1.0E+01**	−1.0E+01	−1.0E+01	−1.0E+01	−1.0E+01	−4.6E+00	−4.9E+00	−7.7E+00	−1.0E+01
	STD	2.7E−03	1.4E+00	3.0E−13	7.5E−01	7.5E−01	**1.6E−15**	7.5E−01	2.0E−05	1.5E−11	1.0E−04	2.9E+00	2.9E+00	2.8E+00	1.4E−04
F48	AVG	−1.1E+01	−4.3E+00	−1.1E+01	−1.1E+01	−5.4E+00	**−1.1E+01**	−1.0E+01	−1.1E+01	−1.1E+01	−1.1E+01	−4.2E+00	−5.5E+00	−8.8E+00	−1.1E+01
	STD	3.7E−03	1.8E+00	2.4E−12	3.0E−15	1.3E+00	**1.6E−15**	7.6E−01	2.1E−05	1.9E−11	1.1E−04	3.1E+00	3.6E+00	2.5E+00	1.0E−04
F49	AVG	4.3E−02	1.1E−01	4.8E−03	5.8E−02	5.5E−02	2.3E−03	**2.3E−03**	1.3E−02	2.3E−03	2.2E−02	5.8E+00	8.9E−02	1.6E−02	2.1E−02
	STD	3.1E−02	6.1E−01	2.9E−03	1.7E−01	4.8E−02	**1.0E−17**	1.2E−17	3.5E−03	1.8E−10	2.6E−02	8.9E+00	8.9E−02	1.1E−02	3.6E−02
F50	AVG	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	4.8E−120	8.1E−119	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.2E−04	4.0E−04	**0.0E+00**	**0.0E+00**
	STD	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.1E−119	1.9E−118	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.7E−04	3.4E−04	**0.0E+00**	**0.0E+00**

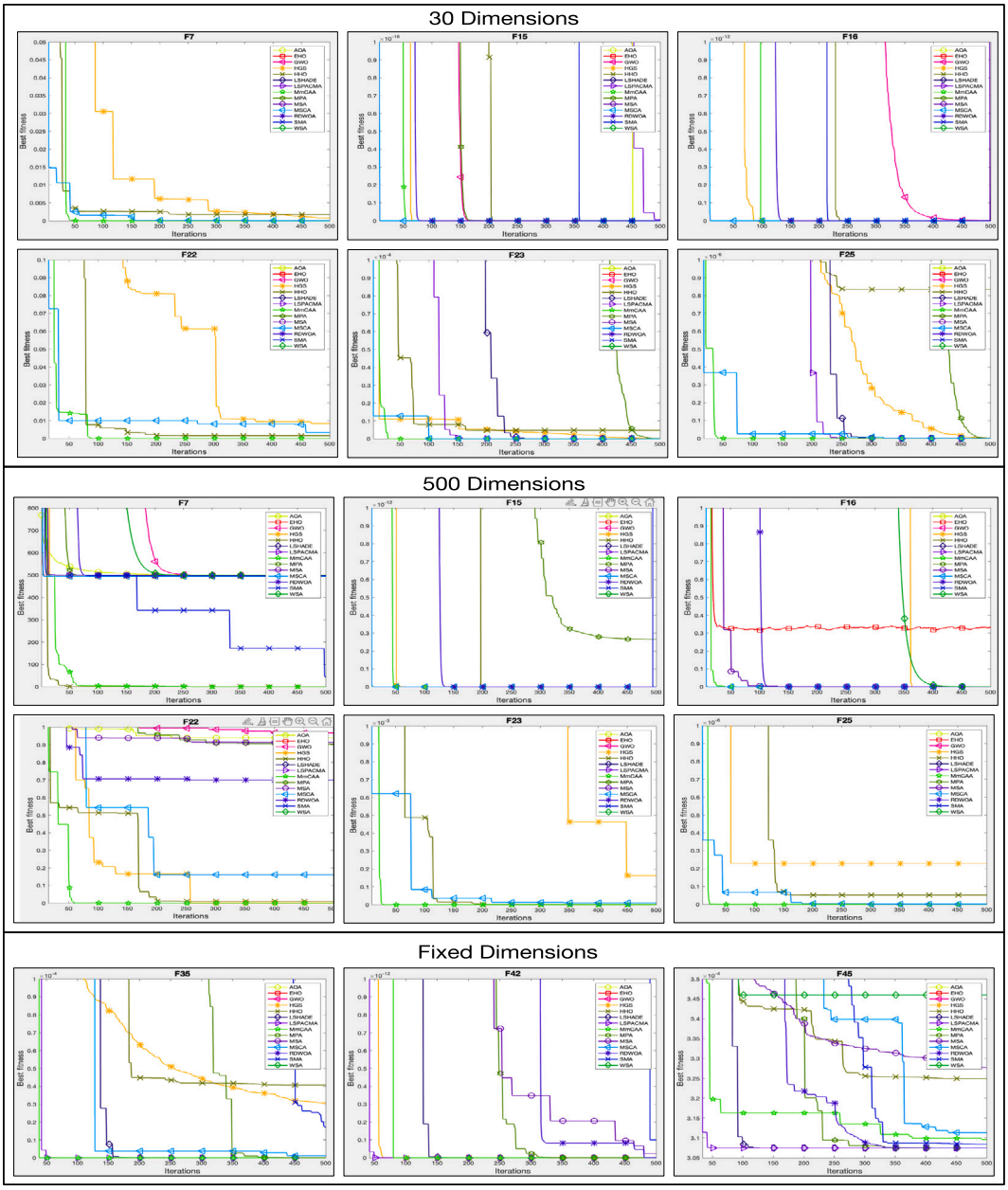


Fig. 8. Convergence curves of the different algorithms for test functions.

Table 7
Comparison of metaheuristics for the SCP problem.

Algorithm	f_{min}
MDDE	14.488
MmCAA	15.9919
TLBO	16.63451
ABC	16.63466
PVS	16.63496
CSA	16.72545
FPA	16.7416

Table 8
Comparison of metaheuristics for the MPDCB problem.

Algorithm	$y(1)$	$y(2)$	$y(3)$	$y(4)$	$y(5)$	f_{min}
MBFPA	70	90	1	600	2	0.235242457900804
MmCAA	70	90	1	619	2	0.23524
HHO	69.9999999992493	90	1	1000	3	0.259768993
TLBO	70	90	1	810	3	0.313656
WCA	70	90	1	910	3	0.313656
PVS	70	90	1	980	3	0.31366

exploration and exploitation of the search space. The experiment takes 50 independent runs with the parameters from Table 2 and calculates the average of the best values. The evolution of these average values during the iterations is presented in Fig. 9, where the first 3 functions are unimodal with 500 dimensions, the next 3 are multimodal with 500 dimensions, and the last 3 are functions with 2 dimensions. First, an example of each function in 2 dimensions is depicted. The second column shows the average movement of the first value in the position of the best smart-cell. The third column describes the average Euclidean distance between the previous and the current position of the best smart-cell. The average weighting from the exploration process to the exploitation process presented is also calculated in Fig. 7. The last column shows the average convergence of the 50 runs.

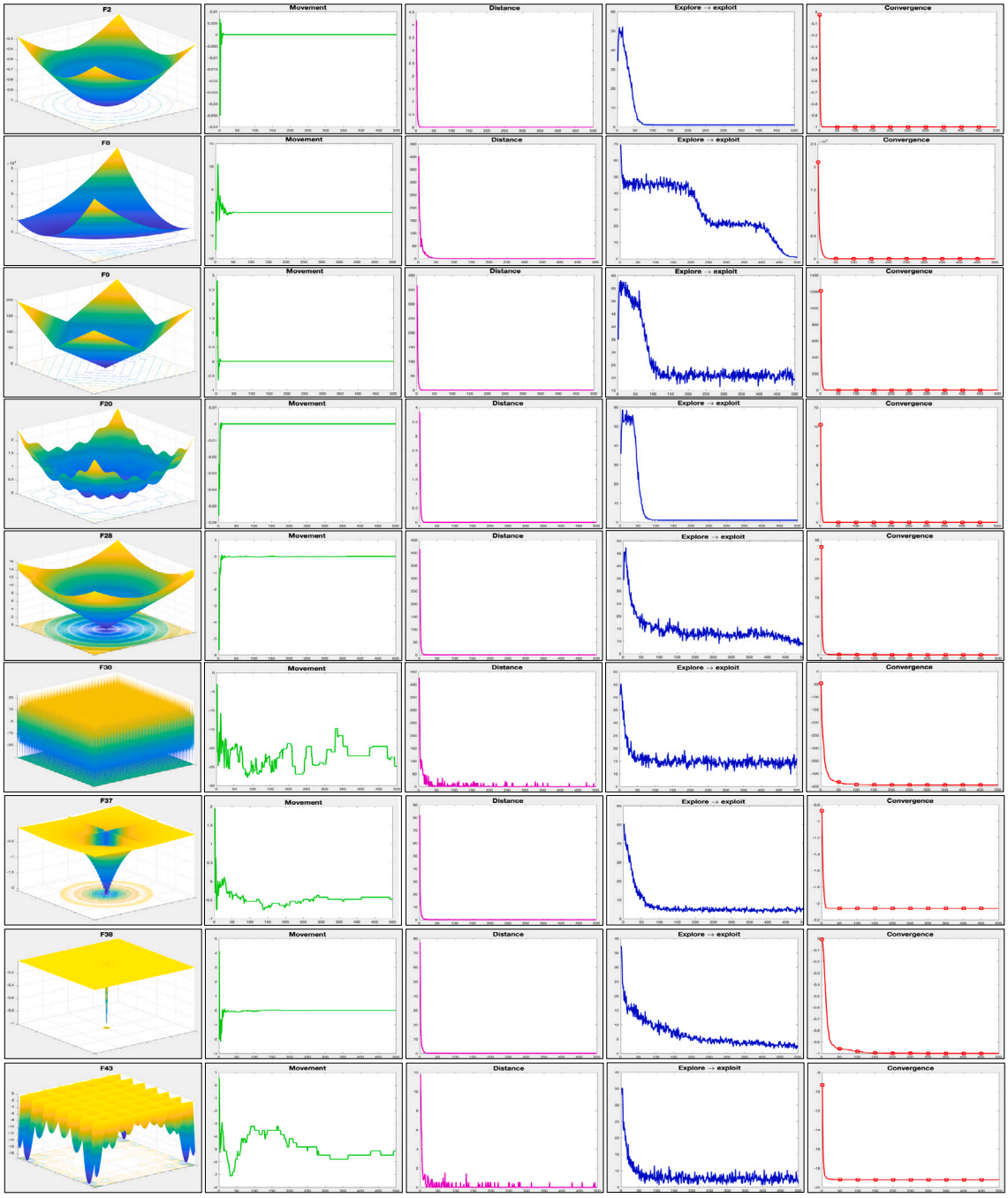


Fig. 9. Average of movement, distance, exploration–exploitation and convergence of the best solutions in the MmCAA.

It can be seen in this analysis that the MmCAA performs an extensive exploration of the search space with abrupt changes in the position and distance with respect to the previous positions of the best smart-cell due to the rules R_4 to R_6. As the optimization process progresses, these changes are no longer as abrupt when the exploitation rules R_1 to R_3 become preponderant to replace the smart-cell with better, closer positions, which is clearly observed in the fourth column in Fig. 9. This demonstrates the MmCAA's efficient balance of exploration and exploitation to achieve a decreasing movement of each smart-cell by optimizing the different types of functions.

6. Application of the MmCAA in engineering problems

6.1. Step-cone pulley design (SCP)

This design problem consists of optimizing the design of a 4 step-cone pulley that has a minimum weight (Rao, 1996; Savsani & Savsani, 2016). In this problem, we have 5 design variables, 4 of which correspond to the diameters of the pulleys ($y(1)$, $y(2)$, $y(3)$ and $y(4)$) at each step and the last design variable is the pulley width ($y(5)$). This design is subject to 11 restrictions; three of them are equalities, and the others are inequalities, which ensures that the same length of the belt is used

in all steps, tension rates and power transmitted by the belt. The system to be designed can be seen in Fig. 10, and the mathematical formulation is presented in Eq. (1).

$$min\ f(\boldsymbol{y}) = \rho y(5)\left(y(i)^2\left(1+\left(\frac{n(i)}{N}\right)^2\right)\right),\ 1 \le i \le 4$$

Subject to:

$$h_i(\boldsymbol{y}) = C(1) - C(i+1) = 0 \qquad 1 \le i \le 3$$
$$g_i(\boldsymbol{y}) = -R(i) \le 2 \qquad 1 \le i \le 4$$
$$g_{i+4}(\boldsymbol{y}) = (0.75 \times 745.6998) - P(i) \le 0 \qquad 1 \le i \le 4$$

where:

$$\boldsymbol{n} = (750, 450, 250, 150) \qquad N = 350 \qquad \rho = 7200\ \text{kg/m}^3 \tag{1}$$

$$C(i) = \frac{\pi y(i)}{2}\left(1+\frac{n(i)}{n}\right) + \frac{\left(\frac{n(i)}{n}-1\right)^2}{4a} + 2a \qquad a = 3\ \text{mm}$$

$$R(i) = \exp\left(\mu\left(\pi - 2sin^{-1}\left(\left(\frac{n(i)}{N}-1\right)\frac{y(i)}{2a}\right)\right)\right) \qquad \mu = 0.35$$

$$P(i) = sty(5)\,(1-R(i))\,\frac{\pi y(i) n(i)}{60} \qquad s = 1.75\ \text{MPa} \qquad t = 8\ \text{mm}$$

with the limits $0 \le y(1), y(2) \le 60$ and $0 \le y(3), y(4), y(5) \le 90$. For this problem, 50 independent runs were made considering $n_S = 5$, $n_{ne} = 6$ and $n_{it} = 500$ to have a setup comparable with the 6 best metaheuristic algorithms described in Savsani and Savsani (2016) for this design. In Savsani and Savsani (2016) only the best values obtained for the cost function are presented, not the obtained design values. The values obtained by the MmCAA are $\boldsymbol{y} = (38.3596, 52.619, 70.02, 84.706, 89.86)$. The comparison of the minimization against the other algorithms is displayed in Table 7.

It can be seen that the MmCAA obtains the second best minimum weight among the 7 algorithms; thus, the proposed algorithm represents a suitable option to optimize the design of a step-cone pulley.

6.2. Multi-plate disc clutch brake design problem (MPDCB)

The MPDCB is an engineering problem recently treated by the latest optimization algorithms (Heidari et al., 2019; Wang et al., 2020). In this problem, the intention is to minimize the total weight of a multiple disc clutch brake. For this case, there are 5 $\boldsymbol{y}$ variables to define: the internal and external radius, $y(1)$ and $y(2)$, the thickness of the discs $y(3)$, the acting force $y(4)$ and the number of friction surfaces $y(5)$. The problem is also conditioned by 8 constraints that depend on the design's geometry and the operating requirements. The system to be designed can be seen in Fig. 11, and the mathematical formulation is presented in Eq. (2).

$$min\ f(\boldsymbol{y}) = \pi(y(2)^2 - y(1)^2)y(3)(y(5)+1)\rho$$

Subject to:

$$g_1(\boldsymbol{y}) = y(2) - y(1) - \Delta r \ge 0 \qquad g_2(\boldsymbol{y}) = L_{max} - (y(5)+1)(y(3)+\delta) \ge 0$$
$$g_3(\boldsymbol{y}) = P_{max} - P_{rz} \ge 0 \qquad g_4(\boldsymbol{y}) = P_{max} v_{sr\,max} - P_{rz} v_{sr} \ge 0$$
$$g_5(\boldsymbol{y}) = v_{sr\,max} - v_{sr} \ge 0 \qquad g_6(\boldsymbol{y}) = T_{max} - T \ge 0$$
$$g_7(\boldsymbol{y}) = M_h - sM_s \ge 0 \qquad g_8(\boldsymbol{y}) = T \ge 0$$

where:

$$\rho = 0.0000078\ \text{kg/mm}^3 \qquad \Delta r = 20\ \text{mm} \qquad L_{max} = 30\ \text{mm}$$
$$\delta = 0.5 \qquad P_{max} = 1\ \text{MPa} \qquad P_{rz} = \frac{y(4)}{A}\ \text{N/mm}^2$$
$$A = \pi(y(2)^2 - y(1)^2)\ \text{mm}^2 \qquad v_{sr\,max} = 10\ \text{m/s} \qquad v_{sr} = \frac{\pi R_{sr} n}{30}\ \text{mm/s}$$
$$R_{sr} = \frac{2}{3}\frac{y(2)^3 - y(1)^3}{y(2)^2 y(1)^2}\ \text{mm} \qquad n = 250\ \text{rpm} \qquad T_{max} = 15\ \text{s}$$
$$T = \frac{I_z \pi n}{30(M_h + M_f)}\ \text{mm} \qquad I_z = 55\ \text{kg/m}^2 \qquad M_h = \frac{2}{3}\mu y(4) y(5)\frac{y(2)^3 - y(1)^3}{y(2)^2 - y(1)^2}$$
$$M_f = 3\ \text{N m} \qquad \mu = 0.5 \qquad s = 1.5 \qquad M_s = 40\ \text{N m} \tag{2}$$

with limits $60 \le y(1) \le 80$, $90 \le y(2) \le 110$, $1 \le y(3) \le 3$, $600 \le y(4) \le 1000$ and $2 \le y(5) \le 9$. For this problem, 50 independent runs were made, taking $n_S = 5$, $n_{ne} = 6$ and $n_{it} = 500$ to have a setup comparable with the 5 different algorithms reported in Heidari et al. (2019) and Wang et al. (2020) for this model. The reference values and the results obtained by the MmCAA are presented in Table 8.

It can be seen that the MmCAA obtains a design comparable to the MBFPA and that it improves on the other algorithms in minimizing the objective function. Thus, the proposed algorithm is suitable for optimizing the design of a multi-plate disc clutch brake.

6.3. Speed reducer design problem (SR)

The SR is another design problem to be considered by the latest optimization algorithms for comparison in engineering applications (Braik, 2021; Wang et al., 2020). In this case, the intention is to minimize the total weight of a speed reducer, where the design depends on 7 variables $\boldsymbol{y}$ to define: the face width $y(1)$, module of teeth $y(2)$, the number of teeth on pinion $y(3)$, length of the first shaft between bearings $y(4)$, the length of the second shaft between bearings $y(5)$, the diameter of the first shaft $y(6)$ and the diameter of the second shaft $y(7)$.

The problem has 11 restrictions that depend mainly on the stress conditions to which the parts of the system are subjected. The design to be specified can be seen in Fig. 12 and the mathematical formulation is presented in Eq. (3).

$$min\ f(\boldsymbol{y}) = 0.785y(1)y(2)^2\left(3.333y(3)^2 + 14.9334y(3) - 42.0934\right) - 1.508y(1)$$
$$\left(y(6)^2 + y(7)^2\right) + 7.4777y(1)\left(y(6)^3 + y(7)^3\right) + 1.508y(1)\left(y(4)y(6)^2 + y(5)y(7)^2\right)$$

Subject to:

$$g_1(\boldsymbol{y}) = \frac{27}{y(1)y(2)^2 y(3)} - 1 \le 0 \qquad g_2(\boldsymbol{y}) = \frac{397.5}{y(1)y(2)y(3)^2} - 1 \le 0$$
$$g_3(\boldsymbol{y}) = \frac{1.93y(4)^3}{y(1)y(3)y(6)^4} - 1 \le 0 \qquad g_4(\boldsymbol{y}) = \frac{1.93y(4)^3}{y(1)y(3)y(7)^4} - 1 \le 0$$
$$g_5(\boldsymbol{y}) = \frac{1}{100y(6)^3}\sqrt{\left(\frac{745y(4)}{y(2)y(3)}\right)^2 + 16.9(10^6)} - 1 \le 0$$
$$g_6(\boldsymbol{y}) = \frac{1}{85y(7)^3}\sqrt{\left(\frac{745y(4)}{y(2)y(3)}\right)^2 + 157.5(10^6)} - 1 \le 0$$
$$g_7(\boldsymbol{y}) = \frac{y(2)y(3)}{40} - 1 \le 0 \qquad g_8(\boldsymbol{y}) = \frac{5y(2)}{y(1)} - 1 \le 0$$
$$g_9(\boldsymbol{y}) = \frac{y(1)}{12y(2)} - 1 \le 0 \qquad g_{10}(\boldsymbol{y}) = \frac{1.5y(6)+1.9}{y(4)} - 1 \le 0$$
$$g_{11}(\boldsymbol{y}) = \frac{1.1y(7)+1.9}{y(5)} - 1 \le 0 \tag{3}$$

with limits $2.6 \le y(1) \le 3.6$, $0.7 \le y(2) \le 0.8$, $17 \le y(3) \le 28$, $7.3 \le y(4), y(5) \le 8.3$, $2.9 \le y(6) \le 3.9$ and $5.0 \le y(7) \le 5.5$. The number of independent runs made for this problem were also 50, taking $n_S = 10$, $n_{ne} = 20$ and $n_{it} = 1000$ to have a setup comparable with the 16 different algorithms reported in Braik (2021) and Wang et al. (2020) for this problem. The reference values and the results obtained by the MmCAA are presented in Table 9.

It can be seen that among all the algorithms, the MmCAA obtains the second best design to minimize the objective function. Thus, the proposed algorithm is useful for optimizing the design of a speed reducer.

7. Conclusions and future work

This work presents a new algorithm for global optimization based on the majority and minority rules of CA called MmCAA. This algorithm also applies concepts inspired by CA such as local interaction and neighborhoods between smart-cells. The randomness and concurrency of applying the different rules generate an adequate balance

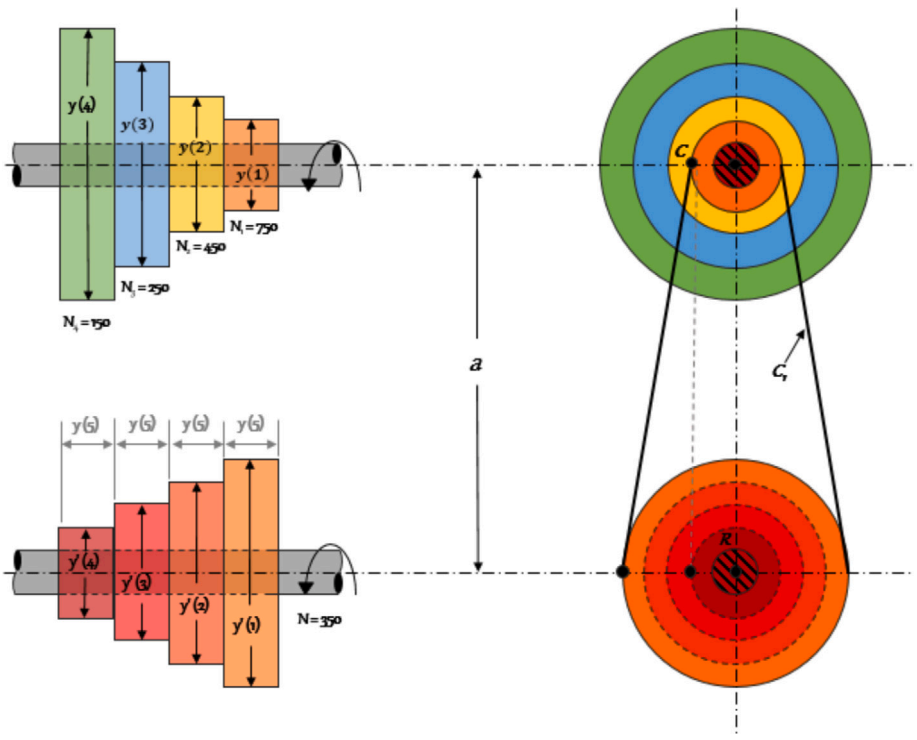


Fig. 10. Step-cone pulley system.

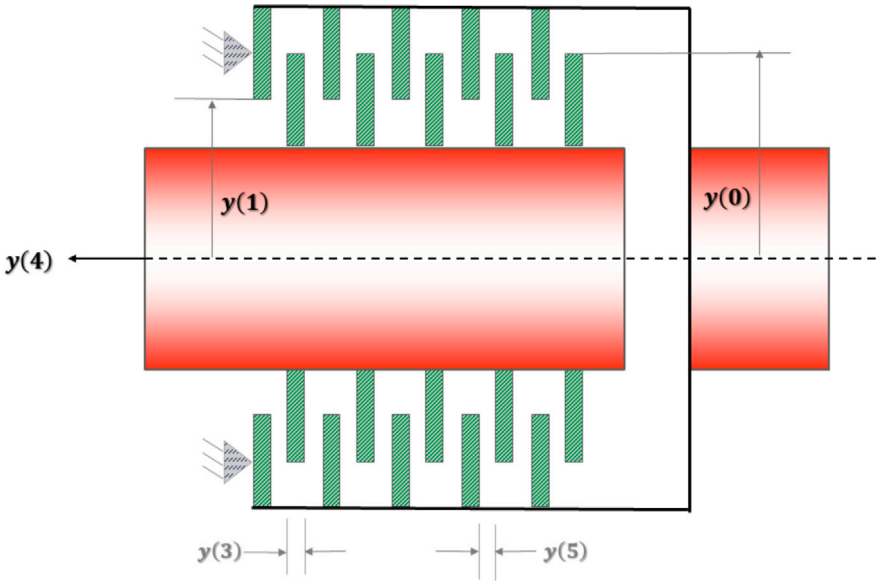


Fig. 11. Multi-plate disc clutch brake system.

between exploration and exploitation actions as well as the exchange of information between solutions during the optimization process.

Algorithm tuning and benchmarking were conducted with 50 test functions: 16 unimodal and 17 multimodal. Both groups tested on 30 and 500 dimensions and, additionally, 17 functions with fixed dimensions. The three groups tested the exploration and exploitation capabilities of the MmCAA and contrasted them with other 13 recently published and recognized algorithms. The experiments showed that the MmCAA behaved satisfactorily, making it sufficiently competitive against the other algorithms.

To test the application of the proposed algorithm in engineering problems, 3 design cases used in recent literature were used to compare the MmCAA with the results obtained by other recent algorithms. The MmCAA also demonstrated its ability to find quality solutions for these types of problems, thus showing compatibility with recent metaheuristics for engineering problems.

For future research, the plan is to continue exploring the different dynamic behaviors of CA, such as the complexity of specific rules (Rule 54, Rule 110 or LIFE) or the non-trivial behavior of groups of rules (reversibility, reaction–diffusion, traffic or lattice gases) to serve as an inspiration for continuing to define new metaheuristic algorithms based on simple local behaviors. Another line of research is to adapt recent techniques for handling multi-objective and discrete natural problems,

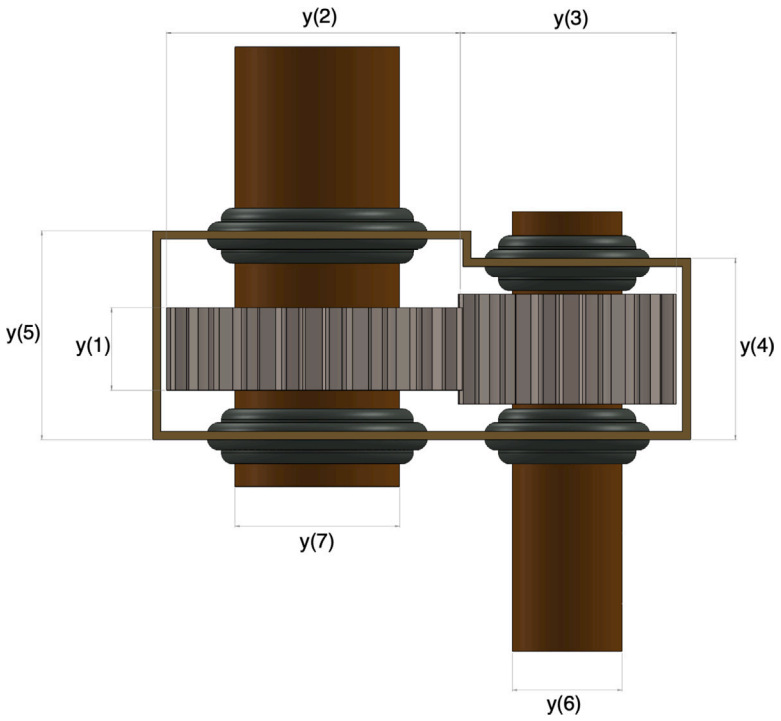


Fig. 12. Speed reducer system.

Table 9
Comparison of metaheuristics for the SR problem.

Algorithm	$y(1)$	$y(2)$	$y(3)$	$y(4)$	$y(5)$	$y(6)$	$y(7)$	f_{min}
MBFPA	3.5	0.7	17	7.3	7.7153199122	3.35021466	5.28665446	2994.341315
MmCAA	3.5	0.7	17	7.3	7.716	3.3503	5.28653	2994.4287
CSA	3.5	0.7	17	7.3	7.715320	3.350215	5.286654	2994.4710
WCA	3.5	0.7	17	7.3	7.715319	3.350214	5.286654	2994.471066
HEAA	3.500022	0.7	17.000012	7.300427	7.715377	3.350230	5.286663	2994.499107
PVS	3.49999	0.0.6999	17	7.3	7.8	3.3502	5.2866	2996.3481
PSODE	3.5	0.7	17	7.3	7.8	3.350214	5.2866832	2996.348167
MDE	3.50001	0.7	17	7.300156	7.800027	3.350221	5.286685	2996.356689
SHO	3.50159	0.7	17	7.3	7.815454	3.351272	5.288745	2998.5507
GWO	3.506690	0.7	17	7.3	7.815726	3.357847	5.286768	3001.2883
MVO	3.508502	0.7	17	7.3	7.816034	3.358073	5.286777	3002.9281
PSO	3.500019	0.7	17	8.3	7.803454	3.352412	5.286715	3005.7631
MFO	3.507524	0.7	17	7.3	7.802364	3.323541	5.287524	3009.5712
HS	3.520124	0.7	17	8.3	7.802354	3.366970	5.288719	3029.002
SCA	3.508755	0.7	17	7.3	7.814353	3.461020	5.289213	3030.5636
GSA	3.6	0.7	17	8.3	7.802442	3.369658	5.289224	3051.1209
GA	3.510253	0.7	17	8.3	7.818452	3.362201	5.287723	3067.5611

such as scheduling, routing, dispatch or classification problems, among others.

CRediT authorship contribution statement

Juan Carlos Seck-Tuoh-Mora: Conceptualization, Methodology, Software, Validation, Formal analysis, Writing – original draft. **Norberto Hernandez-Romero:** Conceptualization, Methodology, Validation, Writing – original draft. **Fredy Santander-Baños:** Investigation, Methodology, Writing – original draft. **Valeria Volpi-Leon:** Investigation, Methodology, Writing – original draft. **Joselito Medina-Marin:** Conceptualization, Formal analysis, Methodology. **Pedro Lagos-Eulogio:** Conceptualization, Validation, Methodology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This study was supported by the National Council for Science and Technology (CONACYT), Mexico with project number CB- 2017-2018-A1-S-43008 and FOP16-2021-01-320109. Fredy Santander-Baños was supported by CONACYT, Mexico grant number 639377.

References

Abilhoa, W. D., & de Oliveira, P. P. B. (2019). Density classification based on agents under majority rule: Connectivity influence on performance. In *International symposium on distributed computing and artificial intelligence* (pp. 163–170). Springer.

Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., & Gandomi, A. H. (2021). The arithmetic optimization algorithm. *Computer Methods in Applied Mechanics and Engineering, 376*, Article 113609.

Abualigah, L., Yousri, D., Abd Elaziz, M., Ewees, A. A., Al-qaness, M. A., & Gandomi, A. H. (2021). Aquila optimizer: A novel meta-heuristic optimization algorithm. *Computers & Industrial Engineering, 157*, Article 107250.

Bastos Filho, C. J. A., de Lima Neto, F. B., Lins, A. J. C. C., Nascimento, A. I. S., & Lima, M. P. (2008). A novel search algorithm based on fish school behavior. In

2008 IEEE international conference on systems, man and cybernetics (pp. 2646–2651). http://dx.doi.org/10.1109/ICSMC.2008.4811695.

Bilan, S. M., Bilan, M. M., & Motornyuk, R. L. (2020). *New methods and paradigms for modeling dynamic processes based on cellular automata*. IGI Global.

Braik, M. S. (2021). Chameleon Swarm Algorithm: A bio-inspired optimizer for solving engineering design problems. *Expert Systems with Applications*, *174*, Article 114685.

Chen, H., Yang, C., Heidari, A. A., & Zhao, X. (2020). An efficient double adaptive random spare reinforced whale optimization algorithm. *Expert Systems with Applications*, *154*, Article 113018.

Christensen, J. D., Griffin, D. B., & Peak, D. (2018). Undecided cliques promote consensus in the directed majority automaton. *International Journal of Unconventional Computing*, *13*.

Cui, Z., & Gao, X. (2012). Theory and applications of swarm intelligence. *Neural Computing and Applications*, *2*(21), 205–206.

Dorigo, M. (1992). *Learning and natural algorithms* (Ph.D. thesis), Politecnico di Milano.

Faramarzi, A., Heidarinejad, M., Mirjalili, S., & Gandomi, A. H. (2020). Marine predators algorithm: A nature-inspired metaheuristic. *Expert Systems with Applications*, *152*, Article 113377.

Feng, X., Lau, F., & Gao, D. (2009). A new bio-inspired approach to the traveling salesman problem. In *International conference on complex sciences, Vol. 5* (pp. 1310–1321). Springer.

Gandomi, A. H., & Alavi, A. H. (2012). Krill herd: A new bio-inspired optimization algorithm. *Communications in Nonlinear Science and Numerical Simulation*, *17*(12), 4831–4845. http://dx.doi.org/10.1016/j.cnsns.2012.05.010, URL: https://www.sciencedirect.com/science/article/pii/S1007570412002171.

Gandomi, A., Yang, X., & Alavi, A. (2013). Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems.. *Engineering with Computers*, *29*, 17–35. http://dx.doi.org/10.1007/s00366-011-0241-y.

Gogna, A., & Tayal, A. (2013). Metaheuristics: review and application. *Journal of Experimental & Theoretical Artificial Intelligence*, *25*(4), 503–526.

Goles, E., & Montealegre, P. (2020). The complexity of the asynchronous prediction of the majority automata. *Information and Computation*, *274*, Article 104537.

Goles, E., Montealegre, P., Perrot, K., & Theyssier, G. (2018). On the complexity of two-dimensional signed majority cellular automata. *Journal of Computer and System Sciences*, *91*, 1–32.

Gupta, S., Deep, K., Mirjalili, S., & Kim, J. H. (2020). A modified sine cosine algorithm with novel transition parameter and mutation operator for global optimization. *Expert Systems with Applications*, *154*, Article 113395.

Hassanien, A. E., & Emary, E. (2018). *Swarm intelligence: Principles, advances, and applications*. CRC Press.

Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems*, *97*, 849–872.

Hernández-Gress, E. S., Seck-Tuoh-Mora, J. C., Hernández-Romero, N., Medina-Marín, J., Lagos-Eulogio, P., & Ortíz-Perea, J. (2020). The solution of the concurrent layout scheduling problem in the job-shop environment through a local neighborhood search algorithm. *Expert Systems with Applications*, *144*, Article 113096.

Hersovici, M., Jacovi, M., Maarek, Y., Pelleg, D., Shtalhaim, M., & Ur, S. (1998). The shark-search algorithm. An application: Tailored web site mapping. *Computer Networks and ISDN Systems*, *30*, 317–326.

Hoekstra, A. G., Kroc, J., & Sloot, P. M. (2010). *Simulating complex systems by cellular automata*. Springer.

Jamil, M., & Yang, X.-S. (2013). A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, *4*(2), 150–194.

Karaboga, D. (2005). *An idea based on honey bee swarm for numerical optimization*: *Technical Report*, Erciyes University, Engineering Faculty, Computer Engineering Department.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization.. In *The IEEE international conference on neural networks (ICNN '95)* (pp. 1942–1948). IEEE.

Kumar, K., & Davim, J. P. (2020). *Optimization using evolutionary algorithms and metaheuristics: Applications in engineering* (1st ed.). CRC Press.

Laboudi, Z. (2019). An effective approach for solving the density classification task by cellular automata. In *2019 4th world conference on complex systems (WCCS)* (pp. 1–8). IEEE.

Lagos-Eulogio, P., Seck-Tuoh-Mora, J., Hernandez-Romero, N., & Medina-Marin, J. (2017). A new design method for adaptive IIR system identification using hybrid CPSO and DE. *Nonlinear Dynamics*, *88*, 2371–2389.

Li, S., Chen, H., Wang, M., Heidari, A. A., & Mirjalili, S. (2020). Slime mould algorithm: A new method for stochastic optimization. *Future Generation Computer Systems*.

Martinez, G. J., Adamatzky, A., & Alonso-Sanz, R. (2012). Complex dynamics of elementary cellular automata emerging from chaotic rules. *International Journal of Bifurcation and Chaos*, *22*(02), Article 1250023.

Martínez, G. J., Morita, K., Adamatzky, A., & Margenstern, M. (2010). Majority adder implementation by competing patterns in life-like rule B2/S2345. In *International conference on unconventional computation* (pp. 93–104). Springer.

McIntosh, H. V. (2009). *One dimensional cellular automata*. Luniver Press.

de Melo, V. V., & Banzhaf, W. (2018). Drone squadron optimization: a novel self-adaptive algorithm for global numerical optimization. *Neural Computing and Applications*, *30*(10), 3117–3144.

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, *95*, 51–67.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software*, *69*, 46–61.

Mohamed, A. W., Hadi, A. A., Fattouh, A. M., & Jambi, K. M. (2017). LSHADE with semi-parameter adaptation hybrid with CMA-ES for solving CEC 2017 benchmark problems. In *2017 IEEE congress on evolutionary computation (CEC)* (pp. 145–152). IEEE.

von Neumann, J. (1966). *Theory of self-reproducing automata*. Champain, IL: University of Illionois Press.

Parhami, B., Abedi, D., & Jaberipur, G. (2020). Majority-logic, its applications, and atomic-scale embodiments. *Computers and Electrical Engineering*, *83*, Article 106562.

Plevris, V., Bakas, N. P., & Solorzano, G. (2021). Pure random orthogonal search (PROS): A plain and elegant parameterless algorithm for global optimization. *Applied Sciences*, *11*(11), 5053.

Rao, S. S. (1996). Further topics in optimization. In *Engineering optimization: Theory and practice* (pp. 779–783). New Age International Publishers.

Salcido, A. (2011). *Cellular automata: Innovative modelling for science and engineering*. BoD–Books on Demand.

Sarker, R., Kamruzzaman, J., & Newton, C. (2003). Evolutionary optimization (EvOpt): a brief review and analysis. *International Journal of Computational Intelligence and Applications*, *3*(04), 311–330.

Savsani, P., & Savsani, V. (2016). Passing vehicle search (PVS): a novel metaheuristic algorithm. *Applied Mathematical Modelling*, *40*(5–6), 3951–3978.

Schiff, J. L. (2011). *Cellular automata: A discrete view of the world, Vol. 45*. John Wiley & Sons.

Seck-Tuoh-Mora, J. C., Hernandez-Romero, N., Lagos-Eulogio, P., Medina-Marin, J., & Zuñiga-Peña, N. S. (2021). A continuous-state cellular automata algorithm for global optimization. *Expert Systems with Applications*, *177*, Article 114930.

Seredynski, F., & Zomaya, A. Y. (2002). Sequential and parallel cellular automata-based scheduling algorithms. *IEEE Transactions on Parallel and Distributed Systems*, *13*(10), 1009–1023.

Shaqfa, M., & Beyer, K. (2021). Pareto-like sequential sampling heuristic for global optimisation. *Soft Computing*, 1–20.

Shi, Y., Liu, H., Gao, L., & Zhang, G. (2011). Cellular particle swarm optimization. *Information Sciences*, *181*(20), 4460–4493.

Slowik, A. (2021). *Swarm intelligence algorithms (Two volume set)*. CRC Press.

Tanabe, R., & Fukunaga, A. S. (2014). Improving the search performance of SHADE using linear population size reduction. In *2014 IEEE congress on evolutionary computation (CEC)* (pp. 1658–1665). IEEE.

Van Den Bergh, F., & Engelbrecht, A. P. (2006). A study of particle swarm optimization particle trajectories. *Information Sciences*, *176*(8), 937–971.

Wang, G.-G., Deb, S., & Coelho, L. d. S. (2015). Elephant herding optimization. In *2015 3rd international symposium on computational and business intelligence (ISCBI)* (pp. 1–5). IEEE.

Wang, G.-G., Gandomi, A. H., Yang, X.-S., & Alavi, A. H. (2014). A novel improved accelerated particle swarm optimization algorithm for global numerical optimization. *Engineering Computations*.

Wang, Z., Luo, Q., & Zhou, Y. (2020). Hybrid metaheuristic algorithm using butterfly and flower pollination base on mutualism mechanism for global optimization problems. *Engineering with Computers*, 1–34.

Wolfram, S. (2002). *A new kind of science, Vol. 5*. Champaign, IL: Wolfram media.

Xu, J., Li, E., Chen, F., & Jin, W. (2018). Chaotic properties of elementary cellular automata with majority memory. *Chaos, Solitons & Fractals*, *115*, 84–95.

Yang, X. (2010). A new metaheuristic bat-inspired algorithm. In *Nature inspired cooperative strategies for optimization (NICSO 2010), Vol. 284* (pp. 65–74). Springer.

Yang, Y., Chen, H., Heidari, A. A., & Gandomi, A. H. (2021). Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. *Expert Systems with Applications*, *177*, Article 114864.