Improved archiving and search strategies for Multi Agent Collaborative Search

Lorenzo A. Ricciardi, Massimiliano Vasile

Abstract This paper presents a new archiving strategy and some modified search heuristics for the Multi Agent Collaborative Search algorithm (MACS). MACS is a memetic scheme for multi-objective optimisation that combines the local exploration of the neighbourhood of some virtual agents with social actions to advance towards the Pareto front. The new archiving strategy is based on the physical concept of minimising the potential energy of a cloud of points each of which repels the others. Social actions have been modified to better exploit the information in the archive and local actions dynamically adapt the maximum number of coordinates explored in the pattern search heuristic. The impact of these modifications is tested on a standard benchmark and the results are compared against MOEA/D and a previous version of MACS. Finally, a real space related problem is tackled.

1 Introduction

Multi Agent Collaborative Search is a memetic algorithm to solve multi-objective optimisation problems that was proposed some time ago to solve robust optimisation problems in space mission design [1, 2]. In MACS, a population of virtual agents is deployed at random locations in the search space. Each agent locally explores its neighbourhood performing a set of local search actions, also named individual actions. Then the population as a whole performs a set of social actions, to concurrently advance towards the front. An external archive is used to store the current best representation of the Pareto set. In a recent version of MACS, called MACS2, a

Lorenzo A. Ricciardi
Advanced Space Concepts Laboratory, University of Strathclyde, Glasgow, UK
e-mail: lorenzo.ricciardi@strath.ac.uk

Massimiliano Vasile
Advanced Space Concepts Laboratory, University of Strathclyde, Glasgow, UK
e-mail: massimiliano.vasile@strath.ac.uk

combination of Pareto dominance and Tchebycheff scalarisation was introduced to select potential improvements towards the Pareto front. Previous studies by Vasile and Zuiani [3, 4, 5, 6] showed the effectiveness of this approach on different benchmark and challenging real problems, testing numerous strategies both for the individual and the social actions. Since then MACS2 was successfully used for the design of space missions for the removal of space debris by means of low-thrust, many revolutions orbits, and for the design of the initial, low-thrust rising phase for the technology demonstrator mission DESTINY. Both are real engineering multi-objective optimisation problems for which no previous solution was known, and involved the concurrent minimisation of fuel consumption, mission time, and, for the DESTINY mission, radiation exposition time.

A thorough analysis of the behaviour of the algorithm, however, has revealed that the archiving procedure was suboptimal as it was not retaining some good isolated non-dominated solutions while keeping solutions in densely populated regions of the Pareto front. The final archive was therefore giving a rather uneven representation of the Pareto front, with poorly distributed solutions. This poor distribution had also an indirect impact on the effectiveness of the search itself, since social actions were exploiting the information in the archive.

In order to address this issue, a new archiving strategy is proposed in this paper. The new strategy, which from now on will be called Energy Based Archiving (EBA), is physically based on the simple idea of minimising the energy of a cloud of points which exert repulsion on each other. Given an initial set with $r+q$ elements, the new archiving procedure selects the subset with r elements with the lowest possible energy. The energy is simply defined as the inverse of the sum of the normalised squared distances of the points in criteria space. Thus, the lowest energy state is associated to the most evenly spread distribution of the points. Note that, the archiving strategy is not specific to MACS, but can be applied to any multi-objective optimisation algorithm, and, as will be shown, can also improve the results obtained by other algorithms.

The paper presents also a set of modified search heuristics that takes advantage of the higher quality of the front stored by the new archiver. One modification improves the exploitation of the information in the archive for the generation of social actions. The other is a dynamic adjustment of the maximum number of coordinates that are explored when local actions are implemented.

The paper is structured as follows: after a preliminary description of multi-objective optimisation in general, the discussion will focus on implementation details and the algorithms will be presented in pseudo-code. Tests will be carried out on the standard benchmark of UF functions from the CEC 2009 competition [7], which are known to have complex Pareto sets. MACS with the new archiving and search heuristics (which for clarity will be called MACS2.1) will be compared with MOEA/D [8], the winner of the CEC 2009 competition, and with MACS2 [5]. It will also be proved that the new archiving algorithm improves the IGD and averaged Hausdorff distance of the solutions given by MOEA/D. The impact of the modified search heuristics will also be investigated, and MACS2.1 and MACS2 will finally be compared on a real challenging space related problem.

2 Problem formulation

This paper is focused on finding the feasible set of solutions that solves the following problem:

$$\min_{\mathbf{x}\in D} \mathbf{f}(\mathbf{x}) \tag{1}$$

where D is a hyperrectangle defined as $D = \{x_j | x_j \in [b_j^l \; b_j^u] \subseteq \Re, \; j = 1,...,n\}$ and $\mathbf{f}$ is the vector function:

$$\mathbf{f} : D \to \Re^m, \quad \mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_m(\mathbf{x})]^T \tag{2}$$

The optimality of a particular solution is defined through the concept of dominance: with reference to problem (1), a vector $\mathbf{y} \in D$ is dominated by a vector $\mathbf{x} \in D$ if $f_l(\mathbf{x}) \leq f_l(\mathbf{y})$ for all $l = 1,...,m$ and there exists k so that $f_k(\mathbf{x}) \neq f_k(\mathbf{y})$. The relation $\mathbf{x} \prec \mathbf{y}$ states that $\mathbf{x}$ dominates $\mathbf{y}$. A decision vector in D that is not dominated by any other vector in D is said to be Pareto optimal. All non-dominated decision vectors in D form the Pareto set D_P and the corresponding image in criteria space is the Pareto front

Starting from the concept of dominance, it is possible to associate, to each solution in a finite set of solutions, the scalar dominance index:

$$I_d(\mathbf{x}_i) = |\{i^* \mid i, i^* \in N_p \wedge \mathbf{x}_{i^*} \prec \mathbf{x}_i\}| \tag{3}$$

where the symbol $|.|$ is used to denote the cardinality of a set and N_p is the set of the indices of all the solutions. All non-dominated and feasible solutions $\mathbf{x}_i \in D$ with $i \in N_p$ form the set:

$$X = \{\mathbf{x}_i \in D \mid I_d(\mathbf{x}_i) = 0\} \tag{4}$$

The set X is a subset of D_P, therefore, the solution of problem (1) translates into finding the elements of X. If D_P is made of a collection of compact sets of finite measure in $\Re^n$, then once an element of X is identified it makes sense to explore its neighbourhood to look for other elements of X. On the other hand, the set of non dominated solutions can be disconnected and its elements can form islands in D. Hence, multiple parallel exploration can increase the collection of elements of X.

2.1 Tchebycheff Scalarisation

In Tchebycheff approach to the solution of problem (1), a number of scalar optimization problems is solved in the form:

$$\min_{\mathbf{x}\in D} g(\mathbf{f}(\mathbf{x}), \lambda, \mathbf{z}) = \max_{l=1,...,m} \{\lambda_l | f_l(\mathbf{x}) - z_l |\} \tag{5}$$

where $\mathbf{z} = [z_1, ..., z_m]^T$ is the reference objective vector whose components are $z_l = \min_{\mathbf{x} \in D} f_l(\mathbf{x})$, for $l = 1, ..., m$, and λ_l is the l-th component of a weight vector λ. By solving a number of problems (5), with different weight vectors, one can obtain different Pareto optimal solutions. Although the final goal is always to find the set X_g, using the solution of problem (5) or index (3) has substantially different consequences in the way samples are generated and selected. In the following, the solution to problem (5) will be used as selection criterion in combination with index (3).

3 Implementation

This section briefly summarises the main features of MACS2, the new archiving procedure and the modified heuristics.

With reference to Algorithm 1, MACS2 starts by initialising a population of n_{pop} agents at random locations within the search domain D with a Latin Hypercube Sampling. Non-dominated agents are copied in the archive A to form the first approximation of the Pareto set. The archive A has specified maximum size max_{arch}. A set of n_λ m-dimensional unit vectors λ_k (defining m directions in criteria space) is then generated, sampled uniformly from a quarter of circle or eighth of a sphere for bi and tri-objective problems or through Latin Hypercube Sampling for higher dimensional problems. The first m λ_k vectors form a base in $\Re^m$, so that the solution vectors that best optimise each individual objective function are always in the final approximation of the Pareto set. Line 8 of Algorithm 1 initialises a utility function (see Zhang et al. [8]) that is used to monitor the progress of each agent. A user-defined fraction p_{social} of agents is specified, and each social agent is then randomly associated to a particular scalarised sub-problem and to the corresponding weight vector λ_k (see Line 9). After initialising the velocity $\mathbf{V}_i$ of each agent (Line 10) the main loop starts (Line 12). Until a maximum number of function evaluations is reached, the agents perform first local search actions, described in the next section, to move towards the Pareto set. A local action is considered successful when it generates a dominating solution or a solution that satisfies the Tchebycheff scalarisation criterion corresponding to a particular sub-problem. After all local actions have been completed, the archive A is updated with all non-dominated solutions. A total of n_{social} agents then perform the social actions described in Section 3.2 and Algorithm 3.

In the new implementation of MACS2 proposed in this paper, when a finite size archive is assumed, the new archiving strategy, described in Section 3.3, is employed to choose which candidates are added to A. Furthermore, as the archive fills up, the number of coordinates scanned by the pattern search local action (see Algorithm 2) is gradually reduced until only one direction at a time is considered. At this point the archive is completely full. This heuristic is motivated by the fact that a well populated and distributed archive contains a lot of information that can be used to generate new samples at a lower cost than a full pattern search. Every n_λ iterations,

the utility function will be updated and sub-problems with the lowest utility function will be changed (see Line 18-20 of Algorithm 1).

3.1 Individualistic actions

In the following we describe the individualistic search actions. Each agent has a repertoire of three different actions, namely: *inertia, pattern search and differential evolution*. Each agent performs each action sequentially until an improvement is registered (i.e. the algorithm generates either a dominant solution or a solution that satisfies Tchebycheff criterion, if the agent is associated to a λ_k). The pseudo-code is given in Algorithm 2.

Inertia

If the previous moves defined a search direction $\mathbf{V}_i$ in parameter space, inertia generates a new sample in the same direction (lines 3-8). The trial position for the $i-th$ agent, $\mathbf{x}_{trial}$ is, defined as:

$$\mathbf{x}_{trial} = \mathbf{x}_i + \alpha \mathbf{V}_i \tag{6}$$

where α is a random number between 0 and 1. If $\mathbf{x}_{trial}$ is outside the admissible domain D, α is contracted with a simple backtracking procedure so that $\mathbf{x}_{trial}$ falls on the boundary of D. In the case a number of components of $\mathbf{x}_i$ lower than n is already equal to either their lower or upper limit and $\mathbf{x}_i + \alpha \mathbf{V}_i$ is outside D, then the corresponding components of $\mathbf{V}_i$ are set to zero before the backtracking procedure is applied. This heuristic is introduced to improve the exploration of the boundary of the search space.

Pattern Search

If inertia gives no improvement or is not performed ($\mathbf{V}_i = 0$), a simple pattern search strategy is implemented. This heuristic changes only one randomly chosen component j of $\mathbf{x}_i$ at a time (lines 14-20). The trial position $\mathbf{x}_{trial}$ is thus equal to $\mathbf{x}_i$, except for the $j-th$ component, which is:

$$x_{trial,j} = x_{ij} + \alpha \Delta_j \rho_i \tag{7}$$

where this time α is a random number between -1 and 1, Δ_j is the difference between the upper and lower boundaries for variable j and ρ_i defines the size of a hyperrectangle centred in $\mathbf{x}_i$. If direction $\alpha \Delta_j \rho_i$ is not successful, the opposite direction $-sign(\alpha)\beta \Delta_j \rho_i$ is attempted with β a random number between 0 and 1. If also this move fails, a new random direction (different from the previous ones) is chosen.

This strategy is repeated until either an improvement is found (i.e. a dominant solution is generated or Tchebycheff criterion is satisfied), or a specified maximum number of directions has been explored. In this version of MACS, the maximum number of directions is dynamically adjusted as

$$max\_dirs = round\left(n-(n-1)\frac{curr\_arch\_size}{max\_arch\_size}\right) \tag{8}$$

where *max_dirs* is the maximum number of dimensions to scan, n is the number of coordinates, *curr_arch_size* is the current size of the archive and *max_arch_size* is the specified maximum size of the archive (lines 11-21). If a good sample is found, it is used to compute vector $\mathbf{V}_i$.

Differential Evolution

If pattern search did not lead to an improvement, a differential evolution step is taken, by combining vector $\mathbf{x}_i$ with 3 randomly chosen agents $\mathbf{x}_{\mathbf{i_1}}, \mathbf{x}_{\mathbf{i_2}}$ and $\mathbf{x}_{\mathbf{i_3}}$ (lines 24-28). The displacement vector is then given by:

$$\mathbf{dx}_i = \alpha\mathbf{e}\left((\mathbf{x}_i - \mathbf{x}_{i_1}) + F\left(\mathbf{x}_{i_2} - \mathbf{x}_{i_3}\right)\right) \tag{9}$$

where α is a random number between 0 and 1, F is a user specified constant and $\mathbf{e}$ is a mask vector whose elements are either 0 or 1 as follows:

$$e_j = \begin{cases} 1, & \text{if } \alpha_2 < CR \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

where α_2 is a random number between 0 and 1, and CR is another user specified constant. The trial position for the differential evolution move finally reads:

$$\mathbf{x}_{trial} = \mathbf{x}_i + \mathbf{dx}_i \tag{11}$$

The feasibility check for this new position is performed exactly as for the inertia case: reducing α or suppressing some components of $\mathbf{dx}_i$.

Local neighbourhood size management

If all local actions have failed, the local neighbourhood size ρ_i is reduced by a user defined factor ρ_{contr}. After a user defined maximum number of contractions $\rho_{max,contr}$, ρ_i is reset to ρ_{ini}. Conversely, if one action is successful, ρ_i is increased by a factor ρ_{contr}, up to the maximum value ρ_{ini} (lines 30-37).

3.2 Social actions

Social actions are implemented following the same principle as in Zuiani and Vasile [5]: a fraction p_{social} of the total population of the agents implements a DE type of heuristic by picking agents either from the population or from the archive (lines 4 or 6). The probability of picking agents from the archive or from the population is determined by:

$$p_{arch\_vs\_pop} = 1 - e^{-\frac{current\_size\_archive}{num\_agents}} \tag{12}$$

In MACS2 each social agent was immediately moved to the trial position if the trial position was satisfying Tchebycheff criterion. The modified heuristic proposed in this paper, instead, updates the archive with all trial vectors that are non-dominated by any other element of the archive. After the archive is updated, each agent performing social actions is then moved to the location of the element of the archive that best improves the corresponding Tchebycheff sub-problem, unless that location is already occupied by another agent (lines 15-22). This new heuristic better exploits the information in the archive and at the same time does not exclude non-dominated trial vectors that do not satisfy the Tchebycheff condition before checking the content of the archive. The pseudo-code for social actions is given in Algorithm 3.

3.3 The new archiving strategy

All non-dominated solutions found by the agents are stored in an external archive A. The archiving process is a fundamental part of the optimisation process. Not only does the archive store an approximation of the Pareto set, but it also represents a source of information for the implementation of social actions. Thus, not only is a well distributed archive desirable but it is also a necessity to improve exploration. The algorithm described in this paper attempts to generate the most evenly distributed Pareto front possible with the available set of solution vectors. The heuristic is based on the physical concept of minimisation of an energy. It draws inspiration from the fact that a set of equally charged particles in a sphere will move towards its surface and spread uniformly. In this case, however, the particles are not free to move, but can only occupy specified positions.

Suppose that at iteration k the archive is full and is composed of r elements. Let $\mathbf{y}_i$ and $\mathbf{y}_i$ be the position of element i and j in objective space, then one can define the generalised energy of the archive as:

$$E = \sum_{i=1}^{r} \sum_{j=i+1}^{r} \frac{1}{(\mathbf{y}_i - \mathbf{y}_j)^T (\mathbf{y}_i - \mathbf{y}_i)} \tag{13}$$

This energy is simply the inverse of the sum of the squared distances of the points of the archive in the criteria space. Suppose now that there are q non-dominated candidate solutions which also do not dominate any of the elements in the archive.

The problem of choosing which candidate substitutes which element of the archive is reformulated as finding the subset of r elements from the set of $r+q$ elements that minimises the energy E.

Note that a direct update of the archive using E is not feasible if r and q are big, because the total number of possible combinations is $\binom{r+q}{r}$. As an alternative, the following procedure is proposed.

Let the archive A be not full. If there is enough space in A to add all the candidates, the archive is simply updated adding those elements (lines 1-2). A symmetric matrix M, containing the reciprocals of the squared distances of all the elements in the archive, is updated (line 4). From M, the total energy of the archive E and a helper vector $\mathbf{E_2}$, are computed (lines 6-7). $\mathbf{E_2}$ is needed to simplify and speed up some computations, as will be explained later. If the archive is full, instead, the following procedure is applied.

Given the elements in the archive A and a set of candidate elements C, for each element in A the energy E is recalculated assuming that that element was replaced by an element in C (line 23). If the lowest variation of E is negative, the element of C that gives that variation and the element in the archive are swapped. If there has been at least one replacement, the whole process is repeated until no more improvements can be detected or a maximum specified number of iterations is exceeded (lines 19-29). In this study, we specified a maximum of 100 iterations. In case the archive is not full but there is not enough space to add all the candidates, the above mentioned algorithm adds, sequentially, the candidates which give the least increase of the total energy of the archive (lines 8-12). No swapping between candidates and agents in the archive is performed in this case, only addition of candidates until the archive is full, and the corresponding update of M and $\mathbf{E_2}$ (lines 13-14). The pseudo-code for the archiving procedure is given in Algorithm 4 .

The actual Matlab implementation stores in a symmetric matrix the inverse of all pairwise squared distances between the elements currently in the archive. Deletion, addition and substitution of elements are performed as block matrix operations to save time. In the i-th entry of the $\mathbf{E_2}$ vector is stored the energy the archive would have if element i were removed. This way, the computation of the energy with a substitution of one element is linear in the number of candidates, because the baseline value (i.e the energy of the archive without replacement) is already stored, and only the contribution of the new candidate needs to be computed. Finally, in order to avoid scaling problems when objectives have very different length scales, a normalisation of the elements in both the candidate set and the archive in criteria space is performed and repeated whenever one of the elements in the archive that optimise each individual objective function is replaced. The overall algorithm is called Energy Based Archiving (EBA).

As an example of the results provided by this archiving strategy, we considered a hypothetical Pareto front with 100 elements and tried to extract the q elements with the EBA algorithm, with the archiving algorithm employed in MACS2 and the one implemented in NSGA-II [9]. Figure 1 shows the results provided by the three archiving strategies for $q = 10$ and $q = 25$. The EBA strategy gives a good spreading of the extracted elements, slightly better than the one obtained with the strategy in

MACS2 and much better than that obtained by the strategy implemented in NSGA-II. The hypothetical Pareto front is composed of a set of random samples taken from the ZDT4 Pareto front. All the algorithms extract the same number of elements from this set. The IGD produced by each algorithm is shown in table 1.

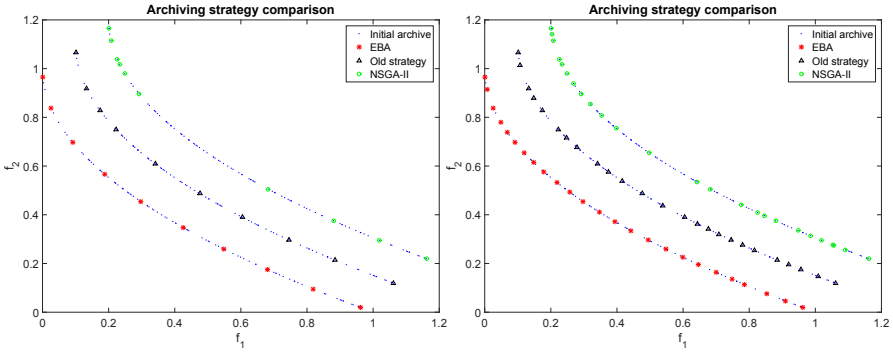
Table 1: IGD of the different archiving strategies

	MACS 2.1 archiver	MACS2 archiver	NSGA-II archiver
10 points	3.68e-2	3.90e-2	9.44e-2
25 points	1.52e-2	1.55e-2	2.44e-2

The EBA strategy requires two sets of operations corresponding to two steps: the fill-in of the archive and computation of the energy E, and its minimisation. The two steps never occur at the same time when the archive is updated. The computation of each reciprocal of squared distance, in (13), for an m dimensional space, requires $3m$ operations: m differences of homologous coordinates, m squares of differences, $m-1$ sums of squares and 1 reciprocal of sums. If the reciprocal of pairwise squared distances of the r elements of the archive is stored in an r by r symmetric matrix M with zero diagonal elements, the number of reciprocal of squared distances to be computed is $\frac{r(r-1)}{2}$, for a total of $\frac{3mr(r-1)}{2}$ operations. With this matrix, the computation of the total energy E of the archive requires the sum of the elements of the upper (or lower) triangular part of the matrix, for a total of $\frac{r(r-1)}{2}$ sums. Starting from the energy E and matrix M the already mentioned r components vector $\mathbf{E_2}$ is computed. $\mathbf{E_2}$ contains in its i-th entry the energy the archive would have if element i were excluded from the archive. The computation of the elements of this vector is conveniently performed by subtracting the sum of all elements of the i-th row of M from the energy E. Thus, a total of r^2 operations is required: 1 subtraction and $r-1$ sums for each component of $\mathbf{E_2}$. This completes the fill-in step of the archive, for a total of $\frac{3mr(r-1)}{2} + \frac{r(r-1)}{2} + r^2 \sim \mathscr{O}(mr^2)$ operations. Note that during a run of MACS2.1 the archive grows gradually, so the construction of the matrix M, energy E and vector $\mathbf{E_2}$ is performed incrementally, rather than all at once.

The energy minimisation step requires the computation of the reciprocal of square distances from each element of A to each candidate in C, for a total of rq combinations or $3mrq$ operations. At this point, the energy the archive would have if element i were substituted with the candidate j is computed: this is conveniently performed by summing $\mathbf{E_2}(i)$ to the reciprocal of the square distances from the j-th candidate to each other element in the archive, for a total of r^2 operations: r sums for each of element of the archive. Now, the minimum energy over all possible combinations of candidates E_{new} is compared against the energy of the archive E. Suppose all the tentative energies are stored in an r by q matrix and its lower value entry E_{new} is located in the position (i^*, j^*). If E_{new} is lower than E, then a new minimum is found, and the candidate in position j^* substitutes the element in position i^*. At this point, it is required to update all elements of the i-th row (or column) of the matrix M with the new reciprocal squared distances. This requires again $3m(r-1)$ opera-

tions, but can be avoided with proper bookkeeping (i.e if the matrix containing the reciprocal of square distances from each candidate to each element of the archive is stored). Finally, the update of the vector $\mathbf{E_2}$ is performed exactly as before. If the energy minimisation step is performed n_{it} times, the total number of operations is $3mrq + n_{it}\left(r^2 + r^2\right)$. Thus, the total cost of the EBA archiving algorithm is $\mathscr{O}\left(3mr^2/2\right)$ for the fill-in step, and $\mathscr{O}(n_{it}r^2)$ for the minimisation step (assuming $q \leq r$). In order to verify the computational complexity of the EBA strategy, we performed a statistical analysis on the dependence of n_{it} on q by sampling the Pareto front of the ZDT4 benchmark function with 100 points plus an increasing number of additional candidates. 200 independent runs were made for each number of additional candidates, and an $\mathscr{O}(q^{\frac{1}{2}})$ relation was discovered. Thus, the average cost of the minimisation step was found to be $\mathscr{O}(r^2 q^{\frac{1}{2}})$. It is also important to note that the computational complexity is linear in the number of dimensions of the Pareto front, making EBA a promising method for many-objective optimisation problems.



(a) 10 points selected

(b) 25 points selected

Fig. 1: Outcomes of different archiving strategies from the same initial archive. The fronts have been shifted to enhance the comparison

Algorithm 1 MACS2

Set $n_{feval,max}$, max_{arch}, n_{pop}, p_{social}, F, CR, ρ_{ini}, ρ_{contr}, $\rho_{max,contr}$
Set $n_{social} = n_{pop} p_{social}$
Set iteration counter $h = 0$
Initialise population $P_h, n_{feval} = 0$
Initialise neighbourhood size $\rho_i = \rho_{ini}\ \forall i \in \{1,..,n_{pop}\}$
Insert the non-dominated elements of P_0 in the archive A
Initialise n_λ vectors λ_k for $k \in \{1,..,n_\lambda\}$ such that $||\lambda_k|| = 1$
Initialise utility function $U_k = 1\ \forall k \in \{1,..,n_\lambda\}$
Select n_{social} active sub-problems to follow
Initialise the vector of agents' velocities $\mathbf{V}_i = 0\ \forall i \in \{1,..,n_{pop}\}$
while $n_{feval} < n_{feval,max}$ **do**
    h=h+1
    update number of directions to scan in pattern search
    Perform local actions through Algorithm 2
    Update archive A with non dominated elements through Algorithm 4
    Perform social actions through Algorithm 3
    Update archive A with non dominated elements through Algorithm 4
    if $mod(h, n_{social}) == 0$ **then**
        Update utility function U_k and the n_{social} active sub-problems
    end if
end while

Algorithm 2 Individualistic actions

for $i = 1 : n_{pop}$ **do**
    Set improved=FALSE
    if $||\mathbf{V}_i|| \neq 0$ **then**
        Perform Inertia move
        Evaluate move
        if successful **then**
            set improved=TRUE
        end if
    end if
    if not improved **then**
        counter=0
        while counter$\leq max_pat_search_dirs$ & not improved **do**
            counter=counter+1
            Pick random direction
            Perform Pattern Search
            Evaluate move
            if successful **then**
                set improved=TRUE
                set $\mathbf{V}_i = x_{i,old} - x_i$
            end if
        end while
    end if
    if not improved **then**
        Perform Differential Evolution
        Evaluate move
        if successful **then**
            set improved=TRUE
        end if
    end if
    if not improved **then**
        Contract ρ_i
        if ρ_i has contracted more than $\rho_{max,contr}$ times **then**
            $\rho_i = \rho_{ini}$
        end if
    else
        De-contract ρ_i unless this would cause ρ_i to be greater than ρ_{ini}
    end if
end for

Algorithm 3 Social actions

choose random number r between 0 and 1
compute $p = 1 - e^{-\frac{curr\_arch\_size}{num\_agents}}$
if $r \leq p$ **then**
    perform DE between social agents and random points from archive
else
    perform DE between social agents and random points from current population
end if
add candidate solutions in archive through Algorithm 4
if there are at least as many agents in the archive as objective functions **then**
    if there's exactly as many agents in the archive as objective functions **then**
        n_{move} = num objective functions
    else
        n_{move} = min(num agents in archive, num agents performing social actions)
    end if
    create pool of n_{move} agents to be moved. Agents following exclusively one of the objectives are always chosen
    for all agents in pool **do**
        find the agent in archive better solving current agent's sub-problem
        if archive position is better than current position **then**
            move current agent to that position
            hide that position in archive for current run of social actions, to prevent multiple agents moving in the same position
        end if
    end for
end if

Algorithm 4 Energy Based Archiving

if there's room for all candidates in archive **then**
    Add them to the archive
    for all candidates **do**
        Update the symmetric matrix M containing the reciprocal of the squared distance of each pair of elements
    end for
    Update the total energy E of the archive
    Update the vector $\mathbf{E_2}$
else
    if only some candidates can be added **then**
        while archive is not full **do**
            Choose the candidate which gives the least possible addition of energy to the archive and add it
            Update M, E and $\mathbf{E_2}$
        end while
    else if the archive is full **then**
        Set improved=TRUE
        iterations=0
        while improved and $iterations < max_{it}$ **do**
            improved=FALSE
            iterations=iterations+1
            Create a matrix containing the energy that the archive would have if each element of the archive were substituted with each candidate
            Locate the minimum entry E_{new} of this matrix
            if $E_{new} < E$ **then**
                E_{new} is at position (i^*, j^*)
                Swap candidate j^* with element i^*
                Set improved=TRUE
                Update E, M and $\mathbf{E_2}$
            end if
        end while
    end if
end if

4 Test cases

The test set used in this paper is a mix of the first seven UF functions proposed in the CEC2009 competition on multi-objective optimisation, the function ZDT4 proposed by Zitzler et al. [10] and a real case of space mission design.

4.1 CEC 2009 UF functions

The UF functions have a complex Pareto set and are a good benchmark to test the archiving procedure. The version of MACS2 with the new archiving procedure and the modified heuristics, called MACS2.1 from now on, was tested and compared against the version of MOEA/D that won the CEC2009 competition [8] and against MACS2.

On the UF test set, each algorithm was run 200 times for each of the functions UF1-7 on a Linux workstation with 8 GB of RAM and an Intel i7-4790 cpu. The settings for MACS2.1 are reported in Table 2 while for MOEA/D the parameters suggested by its authors in [8] were used. The algorithms are compared against the Inverse Generational Distance (IGD) metric, which was used to rank the solutions in the CEC2009 competition, and against the Averaged Hausdorff distance. Both metrics are described and extensively analysed in [11]. As pointed out by Schutze et al [11], the IGD metric is sensistive to the number of elements in the reference Pareto front and in the computed one. Hence the inclusion of the Averaged Hausdorff distance in this comparison. Mean and variance of the IGD and Averaged Hausdorff distance for each problem and algorithm are reported in Table 3, together with result of the Wilcoxon hypothesis test. In 6 of the 7 cases analysed in this paper, the results obtained by MACS2.1 have lower mean IGD and mean Averaged Hausdorff distance, and the variances of those metrics are 1 to 3 orders of magnitude lower for MACS2.1, meaning that the results or MACS2.1 are much more repeatable. The low values of the Wilcoxon test confirm that the underlying distributions of the metrics are indeed different.

To better appreciate the effect of the archiver, MACS2.1 was then run with the same settings but with the archiving strategy employed by MACS2 (as described in [5]), while the final results of MOEA/D where filtered with the EBA algorithm instead of using the filter employed by MOEA/D. Tables 4 and 5 summarise the results of this test. The EBA strategy improved the quality of the Pareto front found by MOEA/D in 4 cases without worsening the others, and improved the results of MACS2.1 in 3 cases with no significant variation in the other cases. The amount of the improvement depends on the quality and size of the the archive: a closer examination of the UF5 and UF6 cases showed that none of the 200 archives had 100 non-dominated elements, hence EBA simply gave the same result as the strategy implemented in MACS2.

For UF4 the high IGDs are caused by a relatively high distance between the computed front and the true one, more than by a poor distribution of the points, while

Table 2: Settings for MACS, CEC problems

$n_{feval,max}$	n_{pop}	ρ_{ini}	F	CR	p_{social}	ρ_{contr}	$\rho_{max,contr}$
300000	150	1	0.9	0.9	0.2	0.5	5

Problem	MACS2.1 IGD	MOEA/D IGD	Wilcoxon test IGD	MACS2.1 Hausdorff	MOEA/D Hausdorff	Wilcoxon test Hausdorff
UF1	**4.09e-3**	4.41e-3	3.70e-59	**1.65e-2**	1.12e-1	3.14e-11
	(9.58e-9)	(1.69e-8)		(5.53e-5)	(6.16e-2)	
UF2	**4.43e-3**	6.24e-3	3.70e-59	**2.09e-2**	7.48e-2	4.52e-53
	(1.23e-7)	(1.57e-6)		(4.41e-5)	(9.46e-3)	
UF3	1.84e-2	**7.16e-3**	3.70e-59	1.46e-1	**6.38e-2**	5.08e-34
	(1.09e-5)	(2.47e-5)		(2.61e-2)	(1.83e-2)	
UF4	**2.93e-2**	6.14e-2	3.70e-59	**4.99e-2**	1.11e-1	4.83e-67
	(7.50e-7)	(2.50e-5)		(2.74e-5)	(3.17e-4)	
UF5	**5.80e-2**	2.98e-1	3.70e-59	**1.32e-1**	7.96e-1	4.83e-67
	(5.58e-5)	(7.45e-3)		(9.56e-4)	(2.20e-1)	
UF6	**2.74e-2**	2.68e-1	3.70e-59	**8.86e-2**	6.27e-1	7.03e-67
	(6.10e-5)	(4.34e-2)		(2.06e-3)	(1.14e-1)	
UF7	**4.15e-3**	4.77e-3	1.46e-34	**2.96e-2**	1.67e-1	1.26e-08
	(5.61e-8)	(3.17e-6)		(9.40e-4)	(6.83e-2)	

Table 3: Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 and MOEA/D, CEC2009 problems. Also reported the unsigned Wilcoxon test results

for UF3 there is a relative lack of points in the upper left region of the Pareto front. For this comparison, the Averaged Hausdorff distance does not show any relevant improvement. This is due to the fact the Averaged Hausdorff distance penalizes the outliers (as clearly stated in [11]), and as such is a worst case measure. The EBA algorithm was conceived to maximise the spreading of the overall solution, but cannot guarantee the worst case distance from each point of the reference front to each point on the computed one, so the observed metrics are not surprising.

We then compared MACS2.1 versus MACS2 [5]. As it can be seen from Table 6, MACS2.1 improves over MACS2 in 5 out of 7 cases for the IGD, although it produces worse results on UF4 and UF5. The Averaged Hausdorff distance for this case favour MACS 2.1 only in 3 over 7 cases.

To better understand which heuristic is contributing to give the different results of MACS2.1 with respect to MACS2, we compared the results obtained by MACS2.1 with dynamic adjustment of the maximum number of directions in the pattern search, against MACS2.1 with a fixed maximum number of direction equal to $2n$. This because in MACS2 the number of directions scanned by pattern search is fixed and equal to $2n$. Results in Table 7 show that in all cases except for UF4 and UF5, the dynamic adjustment of the maximum number of directions scanned by pattern search has a positive effect on the IGD. The comparison of the Averaged Hausdorff distance shows a substantial parity between the two approaches, meaning that the dynamic strategy does not improve the position of the outliers. It can be also be appreciated that in the static case, both the IGD and the Averaged Hausdorff distance associated to the UF1 to UF5 cases are close to the values obtained

Problem	MOEA/D+EBA IGD	MOEA/D IGD	Wilcoxon test IGD	MOEA/D+EBA Hausdorff	MOEA/D Hausdorff	Wilcoxon test Hausdorff
UF1	**4.11e-3**	4.41e-3	8.16e-54	1.11e-1	1.12e-1	3.67e-1
	(1.71e-8)	(1.69e-8)		(6.16e-2)	(6.16e-2)	
UF2	**6.00e-3**	6.24e-3	2.53e-04	7.48e-2	7.48e-2	9.94e-1
	(1.58e-6)	(1.57e-6)		(9.46e-3)	(9.46e-3)	
UF3	**6.88e-3**	7.16e-3	2.54e-07	6.30e-2	6.38e-2	1.19e-1
	(2.54e-5)	(2.47e-5)		(1.83e-2)	(1.83e-2)	
UF4	6.13e-2	6.14e-2	7.92e-01	1.11e-1	1.11-e1	8.45e-1
	(2.49e-5)	(2.50e-5)		(3.12e-4)	(3.17e-4)	
UF5	2.98e-1	2.98e-1	9.97e-01	7.96e-1	7.96e-1	9.98e-1
	(7.45e-3)	(7.45e-3)		(2.20e-1)	(2.20e-1)	
UF6	2.68e-1	2.68e-1	9.95e-01	6.27e-1	6.27e-1	9.96e-1
	(4.34e-2)	(4.34e-2)		(1.14e-1)	(1.14e-1)	
UF7	**4.48e-3**	4.77e-3	1.24e-32	1.67e-1	1.67e-1	9.08e-1
	(3.19e-6)	(3.17e-6)		(6.83e-2)	(6.83e-2)	

Table 4: Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MOEA/D with EBA archiving and standard MOEA/D, CEC2009 problems. Also reported the Wilcoxon test result

by MACS2. This is not surprising since in MACS2.1 DE is performed after pattern search, so if pattern search scans all possible coordinates it will most probably find an improvement and thus DE will not be performed. This also means that in the UF6 and UF7 cases some other heuristic of MACS2.1 is instead contributing.

With the same rationale as the previous analysis, we compared MACS2.1 with the new implementation of social moves against MACS2.1 with the old implementation of the social moves. Table 8 shows that the IGD of the new version is better than the old version in 4 over 7 cases, statistically the same in 1 case and worse in 2 cases, while the Averaged Hausdorff distance of the new social moves is better in 3 cases, statistically the same in 1 case and worse in 3 cases. Thus, this modification does not seem to give a clear contribution. However, further studies are required to assess the interaction of all the combinations of the proposed modifications, especially between the update of the sub-problems through the utility function and the social actions.

As a final rigorous performance test for the CEC cases, we compared the success rate of each algorithm on each of the UF functions. The success rate is defined as the number of runs in which the IGD falls below a given threshold over the total number of runs. Thresholds were chosen to differentiate the results as much as possible but using rather simple values. Table 9 summarises the results. As it is evident, MACS2.1 has overall good performance, outperforming MOEA/D in all cases except for UF3. The introduction of EBA in MOEA/D can improve its results by 10-30% on some problems. MACS2.1 is also generally better than MACS2: although for UF2, UF4 and UF5 the latter has a success rate 20% higher than the former, MACS2.1 is more than 20% better in the other problems, up to 90% better for UF7. In MACS2.1, an overall 5 to 30% improvement is given by the EBA archiving strategy, while the dynamic setting of the maximum number of coordinates can improve

results up to 90% or worsen them up to 15%. Similarly, the new implementation of the social moves can improve results up to 50% or worsen them up to 20%. Overall, the proposed version of MACS2.1 seems to have more consistent results on the entire set of problems, never falling behind by more than 25% over any other algorithm on any problem.

Problem	MACS2.1 IGD	MACS2.1 NO EBA IGD	Wilcoxon test IGD	MACS2.1 Hausdorff	MACS2.1 NO EBA Hausdorff	Wilcoxon test Hausdorff
UF1	**4.09e-3**	4.32e-3	1.88e-54	1.65e-2	1.61e-2	9.88e-1
	(9.58e-9)	(1.04e-8)		(5.53e-5)	(1.49e-5)	
UF2	**4.43e-3**	4.70e-3	3.84e-25	2.09e-2	2.09e-2	8.49e-1
	(1.23e-7)	(6.56e-8)		(4.41e-5)	(4.13e-5)	
UF3	1.84e-2	1.85e-2	6.56e-01	1.46e-1	1.41e-1	6.45e-1
	(1.09e-5)	(9.72e-6)		(2.61e-2)	(2.25e-2)	
UF4	2.93e-2	2.92e-2	5.29e-01	4.99e-2	5.04e-2	2.46e-1
	(7.50e-7)	(1.03e-6)		(2.74e-5)	(3.13e-5)	
UF5	5.80e-2	5.84e-2	7.84e-01	1.32e-1	1.35e-1	2.19e-1
	(5.58e-5)	(5.63e-5)		(9.56e-4)	(9.28e-4)	
UF6	2.74e-2	2.66e-2	2.58e-01	8.86e-2	9.61e-2	8.12e-2
	(6.10e-5)	(3.71e-5)		(2.06e-3)	(2.64e-3)	
UF7	**4.15e-3**	4.49e-3	1.40e-35	2.96e-2	2.79e-2	1.52e-1
	(5.61e-8)	(6.01e-8)		(9.40e-4)	(5.80e-5)	

Table 5: Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 with EBA archiving vs MACS2.1 without EBA archiving on the CEC2009 problems. Also reported the Wilcoxon test result

Problem	MACS2.1 IGD	MACS2 IGD	Wilcoxon test IGD	MACS2.1 Hausdorff	MACS2 Hausdorff	Wilcoxon test Hausdorff
UF1	**4.09e-3**	4.39e-3	1.50e-59	**1.65e-2**	2.80e-2	9.26e-35
	(9.58e-9)	(2.48e-8)		(5.53e-5)	(3.14e-4)	
UF2	**4.43e-3**	4.49e-3	6.33e-09	2.09e-2	**1.57e-2**	5.08e-24
	(1.23e-7)	(1.32e-8)		(4.41e-5)	(7.61e-6)	
UF3	**1.84e-2**	2.41e-2	8.25e-50	1.46e-1	**6.75e-2**	1.07e-29
	(1.09e-5)	(4.98e-6)		(2.61e-2)	(3.60e-4)	
UF4	2.93e-2	**2.63e-2**	2.15e-66	4.99e-2	**4.43e-2**	1.21e-26
	(7.50e-7)	(2.96e-7)		(2.74e-5)	(2.01e-5)	
UF5	5.80e-2	**5.29e-2**	2.81e-11	1.32e-1	**1.22e-1**	2.09e-05
	(5.58e-5)	(4.81e-5)		(9.56e-4)	(1.09e-3)	
UF6	**2.74e-2**	3.41e-2	7.30e-17	**8.86e-2**	1.02e-1	2.03e-04
	(6.10e-5)	(1.06e-4)		(2.06e-3)	(2.60e-3)	
UF7	**4.15e-3**	6.54e-3	3.84e-66	**2.96e-2**	4.93e-2	5.68e-39
	(5.61e-8)	(4.96e-6)		(9.40e-4)	(4.37e-4)	

Table 6: Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 vs. MACS2 on the CEC2009 problems. Also reported the Wilcoxon test result

Problem	MACS2.1 IGD	MACS2.1 static IGD	Wilcoxon test IGD	MACS2.1 Hausdorff	MACS2.1 static Hausdorff	Wilcoxon test Hausdorff
UF1	**4.09e-3**	4.40e-3	5.46e-61	**1.65e-2**	2.54e-2	4.74e-39
	(9.58e-9)	(1.98e-8)		(5.53e-5)	(1.29e-4)	
UF2	**4.43e-3**	4.47e-3	2.48e-06	2.09e-2	**1.60e-2**	2.64e-21
	(1.23e-7)	(2.14e-8)		(4.41e-5)	(1.24e-5)	
UF3	**1.84e-2**	2.51e-2	4.05e-09	1.46e-1	1.36e-1	4.67e-01
	(1.09e-5)	(1.09e-5)		(2.61e-2)	(1.43e-2)	
UF4	2.93e-2	**2.66e-2**	2.20e-65	4.99e-2	**4.54e-2**	2.70e-20
	(7.50e-7)	(3.42e-7)		(2.74e-5)	(2.26e-5)	
UF5	5.80e-2	**5.47e-2**	9.51e-06	1.32e-1	**1.24e-1**	1.21e-03
	(5.58e-5)	(4.98e-5)		(9.56e-4)	(7.51e-4)	
UF6	**2.74e-2**	3.04e-2	1.27e-05	**8.86e-2**	9.78e-2	3.48e-03
	(6.10e-5)	(7.43e-5)		(2.06e-3)	(2.13e-3)	
UF7	**4.15e-3**	5.08e-3	9.58e-66	**2.96e-2**	4.76e-2	9.84e-48
	(5.61e-8)	(1.59e-6)		(9.40e-4)	(2.33e-4)	

Table 7: Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 with EBA archiving and dynamic setting of maximum number of coordinates for pattern search vs MACS2.1 with EBA archiving and static setting of maximum number of coordinates for pattern search on the CEC 2009 problems. Also reported the Wilcoxon test result

Problem	MACS2.1 IGD	MACS2.1 old social IGD	Wilcoxon test IGD	MACS2.1 Hausdorff	MACS2.1 old social Hausdorff	Wilcoxon test Hausdorff
UF1	**4.09e-3**	4.14e-3	1.96e-04	**1.65e-2**	2.53e-2	2.71e-06
	(9.58e-9)	(2.63-e8)		(5.53e-5)	(1.54e-3)	
UF2	4.43e-3	**4.11e-3**	8.44e-36	2.09e-2	**1.54e-2**	2.36e-27
	(1.23e-7)	(1.83e-8)		(4.41e-5)	(9.18e-6)	
UF3	**1.84e-2**	1.95e-2	2.95e-04	1.46e-1	**8.41e-2**	5.16e-31
	(1.09e-5)	(4.03e-6)		(2.61e-2)	(1.24e-2)	
UF4	2.93e-2	**2.76e-2**	5.87e-48	4.99e-2	**4.64e-2**	2.22e-15
	(7.50e-7)	(7.04e-7)		(2.74e-5)	(2.39e-5)	
UF5	5.80e-2	5.75e-2	5.33e-01	1.32e-1	1.30e-1	4.49e-01
	(5.58e-5)	(5.01e-5)		(9.56e-4)	(8.26e-4)	
UF6	**2.74e-2**	3.05e-2	2.47e-05	**8.86e-2**	1.01e-1	5.35e-05
	(6.10e-5)	(6.63e-5)		(2.06e-3)	(2.67e-3)	
UF7	**4.15e-3**	4.60e-3	4.13e-39	**2.96e-2**	3.31e-2	3.39e-13
	(5.61e-7)	(1.14e-7)		(9.40e-4)	(7.54e-5)	

Table 8: Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 vs MACS2.1 with old social moves on the CEC2009 problems. Also reported the Wilcoxon test result

%IGD < τ	MACS2.1 NO EBA	MACS2.1 NO EBA	MOEA/D	MOEA/D + EBA	MACS2.1 static pat	MACS2.1 old social	MACS2
UF1 (τ=4.5e-3)	100	94.5	85.0	98.0	76.0	99.0	79.5
UF2 (τ=5.0e-3)	93.5	88.5	0.5	13.0	99.5	100	100
UF3 (τ=2.0e-2)	68.5	67.5	95.0	95.0	9.0	61.0	3.5
UF4 (τ=3.0e-2)	78.5	78.5	0	0	100	100	100
UF5 (τ=5.0e-2)	16.0	12.0	0	0	25.0	15.0	36.5
UF6 (τ=3.0e-2)	70.0	74.5	0	0	56.0	53.5	36.5
UF7 (τ=4.5e-3)	92.0	60.0	64.5	92.0	1.5	40.5	1.5

Table 9: Success rates for the IGD of the CEC2009 functions for all the tested algorithms and their variants

4.2 ZDT4 and 3 impulse problem

To further test the capabilities of MACS2.1 we run it 200 times on the ZDT4 test function and on a real space trajectory optimisation problem. In the space trajectory design problem the goal is to optimise three impulsive manoeuvres to transfer a spacecraft from a circular Low Earth Orbit, with a radius of 7000km, to a circular Geostationary orbit, with a radius of 42000km (for further details on the problem the interested reader can refer to [3]). The motivation behind the choice of the ZDT4 and the 3 impusle test case is that both of them are characterised by many local Pareto fronts. The settings for MACS2.1 are reported in table 10, 200 solutions per run were maintained in the archive. The performance of MACS2.1 was compared against the perfomance of MACS2, whose settings were specified as in [5].

Note that, for the 3 impulse case the true Pareto front is unknown, thus for self consistence a global Pareto front was extracted from all the 400 runs and used as a reference Pareto front for the calculation of all the metrics.

Tables 11 and 12 report the metrics computed for both the ZDT4 and 3 impulse problem and the corresponding success rates. On the ZDT4 case MACS2.1 outperforms MACS2.

The 3 impulse case gives less clear results instead. Although many interesting areas of the global Pareto front are due to MACS2.1, the mean IGD associated to its solutions is higher than the mean IGD of the solutions computed by MACS2. This is due to the fact (see Figure 2) that MACS2 is contributing to the global Pareto front with twice as many points than MACS2.1, and all those points are concentrated in a central area, while the contribution from MACS2.1 is as expected more widespread and surprisingly a bit scarce in the central area. Thus, the typical run of MACS2 will generate many points close to the over represented region, while the typical run of

MACS2.1 will generate less points in that area but more widely spread points, and this results in the averaged IGD of MACS2.1 to be higher than that of MACS2. The Averaged Hausdorff distance instead does not suffer from this kind of bias, thus the lower value of this metric is associated to the fronts computed by MACS2.1.

Table 10: Settings of MACS2.1 on the 3 impulse and ZDT4 problems (in brackets)

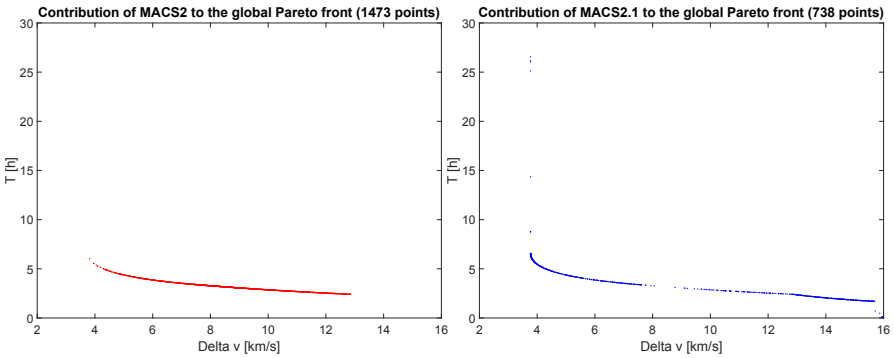
$n_{feval,max}$	n_{pop}	ρ_{ini}	F	CR	p_{social}	ρ_{contr}	$\rho_{max,contr}$
30000	10	1	0.9	0.9	1	0.5	5
(15000)	(10)	(1)	(0.9)	(0.9)	(1)	(0.5)	5

Problem	MACS2.1 IGD	MACS2 IGD	Wilcoxon test	MACS2.1 Hausdorff	MACS2 Hausdorff	Wilcoxon test
zdt4	**7.6e4-3**	8.01e-1	3.00e-64	**2.52e-2**	2.62e+0	1.20e-62
	(1.05e-4)	(2.48e-1)		(7.59e-4)	(3.60e+0)	
3 imp	2.78e-1	**1.17e-2**	1.32e-43	**2.89e+2**	3.45e+2	7.41e-31
	(2.81e-2)	(1.07e-3)		(7.11e+3)	(6.05e+3)	

Table 11: Mean (variance in brackets) for the IGD and averaged Hausdorff distances for MACS2.1 vs MACS2 on zdt4 and triple impulse problems. Also reported the Wilcoxon test result

% IGD $<$ τ	MACS2	MACS2.1	% Hausdorff $<$ τ	MACS2	MACS2.1
zdt4(τ=1e-2)	1.5	83.5	zdt4(τ=5e-2)	3.0	97.5
3imp(τ=1e-1)	29.5	3.0	3imp(τ=3e+2)	5.0	82.5

Table 12: ZDT4 and triple impulse success rates for MACS2 and MACS2.1 and the various metrics



(a) Contribution of MACS2 (b) Contribution of MACS2.1

Fig. 2: Contribution of the different algorithms to the global Pareto front

5 Conclusions

In this paper we presented a new archiving strategy and some modified search heuristics for MACS2. The results computed by the new algorithm, called MACS2.1, are overall better than those computed by MOEA/D on the UF test set. MACS2.1 outperformed also MACS2 in 5 over 7 of the UF functions and on the ZDT4 test case. In the 3 impulse case, MACS2.1 contributes with a wide spread of points to the Pareto front not concentrated in the central area densely covered by MACS2. The better IGD scored by MACS2 in this case can be explained by an uneven distribution of points on the global Pareto front. This is confirmed by the fact that MACS2.1 shows a better Averaged Hausdorff distance, a metric which does not resent from this kind of bias.

The effectiveness of the archiving strategy at extracting well spread Pareto fronts was shown both by comparing the results obtained by MACS2.1 with and without EBA, and by comparing the results obtained by MOEA/D with and without EBA.

The effect of the modified strategies employed in the current version of MACS was also investigated. From the results of our tests, we can conclude that the strategy to dynamically change the number of directions scanned by the pattern search algorithm can have a deep positive impact on some problems and a slightly negative impact on others. The new implementation of the social actions gives less clear results instead: further studies are required to assess how it is influenced by the choice of the sub problems, and how do these new strategies interact in general. Interesting future research options involve the use of more sophisticated local search heuristics in combination with a Monotonic Basin Hopping strategy already successfully tested in [6].

References

[1] Federico Zuiani and Massimiliano Vasile. Preliminary design of debris removal missions by means of simplified models for low-thrust, many-revolution transfers. *International Journal of Aerospace Engineering*, 2012, 2012.

[2] Federico Zuiani, Yasuhiro Kawakatsu, and Massimiliano Vasile. Multi-objective optimisation of many-revolution, low-thrust orbit raising for destiny mission. In *23rd AAS/AIAA Space Flight Mechanics Conference*, 2013.

[3] M. Vasile and F. Zuiani. Multi-agent collaborative search: an agent-based memetic multi-objective optimization algorithm applied to space trajectory design. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 225(11):1211–1227, 2011.

[4] Federico Zuiani and Massimiliano Vasile. Improved individualistic actions for multi-agent collaborative search. *In EVOLVE*, 2013.

[5] Federico Zuiani and Massimiliano Vasile. Multi agent collaborative search based on tchebycheff decomposition. *Computational Optimization and Applications*, 56(1):189–208, 2013.

[6] Federico Zuiani and Massimiliano Vasile. Multi agent collaborative search with thecycheff decomposition and monotonic basin hopping. *In BIOMA*, May 2012.

[7] Qingfu Zhang, Aimin Zhou, Shizheng Zhao, Ponnuthurai Nagaratnam Suganthan, Wudong Liu, and Santosh Tiwari. Multiobjective optimization test instances for the cec 2009 special session and competition. *University of Essex, Colchester, UK and Nanyang technological University, Singapore, special session on performance assessment of multi-objective optimization algorithms, technical report*, pages 1–30, 2008.

[8] Qingfu Zhang, Wudong Liu, and Hui Li. The performance of a new version of moea/d on cec09 unconstrained mop test instances. In *IEEE Congress on Evolutionary Computation*, volume 1, pages 203–208, 2009.

[9] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197, 2002.

[10] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary computation*, 8(2):173–195, 2000.

[11] Oliver Schütze, Xavier Esquivel, Adriana Lara, and Carlos A Coello Coello. Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization. *Evolutionary Computation, IEEE Transactions on*, 16(4):504–522, 2012.