A Parallel Version of SMS-EMOA for Many-Objective Optimization Problems

Raquel Hernández Gómez[1(✉)], Carlos A. Coello Coello[1], and Enrique Alba[2]

[1] Computer Science Department, CINVESTAV-IPN (Evolutionary Computation Group), 07360 Mexico City, Mexico
rhernandez@computacion.cs.cinvestav.mx, ccoello@cs.cinvestav.mx

[2] Universidad de Málaga, 29071 Malaga, Spain
eat@lcc.uma.es

Abstract. In the last decade, there has been a growing interest in multi-objective evolutionary algorithms that use performance indicators to guide the search. A simple and effective one is the $\mathcal{S}$-Metric Selection Evolutionary Multi-Objective Algorithm (SMS-EMOA), which is based on the hypervolume indicator. Even though the maximization of the hypervolume is equivalent to achieving Pareto optimality, its computational cost increases exponentially with the number of objectives, which severely limits its applicability to many-objective optimization problems. In this paper, we present a parallel version of SMS-EMOA, where the execution time is reduced through an asynchronous island model with micro-populations, and diversity is preserved by external archives that are pruned to a fixed size employing a recently created technique based on the Parallel-Coordinates graph. The proposed approach, called $\mathcal{S}$-PAMICRO (PArallel MICRo Optimizer based on the $\mathcal{S}$ metric), is compared to the original SMS-EMOA and another state-of-the-art algorithm (HypE) on the WFG test problems using up to 10 objectives. Our experimental results show that $\mathcal{S}$-PAMICRO is a promising alternative that can solve many-objective optimization problems at an affordable computational cost.

1 Introduction

Numerous real-world problems can be formulated as Multi-Objective Optimization Problems (MOPs), which involve several (often conflicting) objectives to be optimized at the same time. In general, a MOP is formally described as follows:

$$\text{Minimize} \quad \boldsymbol{F}(\boldsymbol{x}) := (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})) \tag{1}$$

$$\text{subject to} \quad \boldsymbol{x} \in \mathcal{S}, \tag{2}$$

C.A. Coello Coello—Author gratefully acknowledges support from CONACyT project no. 221551.

E. Alba—Author is partially funded by the Spanish MINECO and FEDER project TIN2014-57341-R (http://moveon.lcc.uma.es).

J. Handl et al. (Eds.): PPSN XIV 2016, LNCS 9921, pp. 568–577, 2016.
DOI: 10.1007/978-3-319-45823-6_53

where $\boldsymbol{x}$ is the *vector of decision variables*, $\mathcal{S} \subset \mathbb{R}^n$ is the *feasible region set* and $\boldsymbol{F}(\boldsymbol{x})$ is the vector of m ($\geq$2) *objective functions* ($f_i : \mathbb{R}^n \rightarrow \mathbb{R}$). The aim is to seek from among the set of all values, which satisfy the constraint functions defined in Eq. (2), the particular set $\boldsymbol{x}^*$ that yields the optimum values for all the objective functions.

Multi-Objective Evolutionary Algorithms (MOEAs) are stochastic, population-based, search techniques; which are well-suited for solving a wide variety of complex MOPs. In the last decades several MOEAs have been proposed (see, for example, [4, Chap. 2] and [18]), with the vast majority relying on two concepts: *Pareto dominance*[1] as their primary selection mechanism, followed by a *density estimator*.[2] The former favors non-dominated solutions over dominated ones, whereas the latter induces a total order of *incomparable solutions*,[3] preserving *diversity*[4] at the same time.

One of the main concerns is that Pareto-based MOEAs face difficulties to reach the *Pareto optimal front*[5] when dealing with many-objective optimization problems ($m \geq 4$) [9,11,13]. This is due to the fact that most or all solutions in the population quickly become non-dominated with respect to the rest, and the best individuals are identified only by the density estimator. Thus, in some cases good locally non-dominated solutions in terms of convergence might be discarded at the expense of keeping good solutions in terms of diversity, in spite of the fact that they may be distant from the Pareto optimal front [1]. To address this issue, a new trend is the incorporation of *performance indicators*[6] into the selection mechanism of a MOEA [2,6,19]. The *hypervolume indicator* [4, p. 257] is, with no doubt, a natural choice, (see for example [6,19]) since it is the only unary indicator that is known to be Pareto compliant. Also, it has been proven that maximizing the hypervolume is equivalent to reaching the Pareto optimal set [7]. However, the main drawback of this sort of approach is its computational cost, which increases exponentially with the number of objectives [3], making it prohibitive for many-objective optimization problems.

In this work, we focus on the $\mathcal{S}$-Metric Selection Evolutionary Multi-Objective Algorithm (SMS-EMOA) [6], due to its simplicity and superiority over Pareto- and Aggregation-based algorithms [6,10,16]. This optimizer is a steady state evolutionary algorithm that ranks individuals according to Pareto dominance and uses the hypervolume as its density estimator. The worst-case complexity of SMS-EMOA is $\mathcal{O}(|P|^m)$ [17]. Parallelizing SMS-EMOA arises as a possible alternative to reduce its computational cost, where at least two strate-

[1] A solution $\boldsymbol{x} \in \mathcal{S}$ dominates a solution $\boldsymbol{y} \in \mathcal{S}$ ($\boldsymbol{x} \prec \boldsymbol{y}$) if and only if $\forall i \in \{1, \ldots, m\}$, $f_i(\boldsymbol{x}) \leq f_i(\boldsymbol{y})$ and $\exists j \in \{1, \ldots, m\}$, $f_j(\boldsymbol{x}) < f_j(\boldsymbol{y})$.

[2] A density estimator models the distribution of a population, by measuring the similarity degree among individuals.

[3] Two solutions $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{S}$ are incomparable if neither $\boldsymbol{x} \prec \boldsymbol{y}$ nor $\boldsymbol{y} \prec \boldsymbol{x}$ holds.

[4] *Diversity* refers to achieving a uniform distribution of solutions covering all regions of the objective function space.

[5] $POF := \{\boldsymbol{F}(\boldsymbol{x}) \in \mathbb{R}^m : \boldsymbol{x} \in \mathcal{S}, \nexists \boldsymbol{y} \in \mathcal{S}, \boldsymbol{y} \prec \boldsymbol{x}\}$.

[6] A performance indicator, defined as $I : \mathbb{R}^m \rightarrow \mathbb{R}$, measures the quality of an approximation set (the final population of a MOEA).

gies are possible [14]: (1) parallelization of the computations, in which the operations applied to an individual are performed in parallel, and (2) parallelization of the population, in which the population is partitioned and each subpopulation evolves in semi-isolation (individuals can be exchanged between subpopulations). Klinkenberg et al. [10] and Lopez et al. [12] have studied the first approach. In [10], a variation of SMS-EMOA parallelized the evaluations of individuals using a surrogate model, whose purpose was to approximate the function values. In [12], the exact hypervolume contributions of SMS-EMOA were parallelized through the use of Graphics Processing Units (GPUs). To the best of our knowledge, our work is the first attempt to incorporate the second sort of approach (parallelization of the population) into SMS-EMOA.

In order to get a better grasp of the variability of the execution time of SMS-EMOA, we sampled several points on DTLZ1 [4, p. 200], varying the number of objective functions and the population size on a PC Intel(R) Core(TM) i7 CPU 950 @ 3.07 GHz $\times$ 8 with 3.8 GB memory, using the same parameters in all experiments [6]. The average resulting surface is shown in Fig. 1. An interesting observation is that, regardless of the number of objectives, time was almost negligible when using small populations (less than 20 individuals). This fact is considered in our proposal, where we use micro-populations in an asynchronous island model [15]. Furthermore, diversity is improved by external archives that are kept to a constant size by a recently proposed density estimator [8], which is scalable in objective space.

The remainder of this paper is organized as follows. Section 2 is devoted to the description of our proposed parallel MOEA. In Sect. 3 we present our experimental results. Finally, Sect. 4 provides our conclusions and some potential lines of future research.

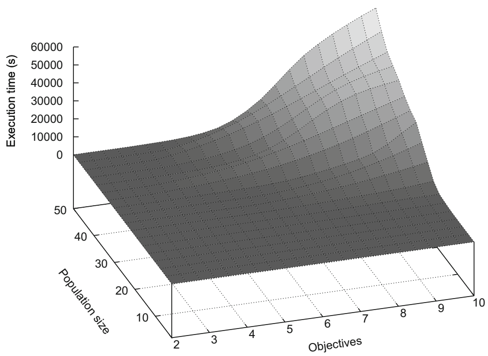

Fig. 1. Average execution time of SMS-EMOA.

2 Our Proposed Approach

The PArallel MICRo Optimizer based on the $\mathcal{S}$ metric ($\mathcal{S}$-PAMICRO) draws ideas from the island model, where the overall population is split into l micro-populations, called *islands*. Every island evolves independently a serial SMS-EMOA with an external archive of size $l \times |P|$, where $|P|$ corresponds to the micro-population size. In this approach, the islands are connected in a logical unidirectional ring, exchanging $nmig$ solutions occasionally[7] in an asynchronous fashion. The goal of $\mathcal{S}$-PAMICRO is to reduce the execution time of SMS-EMOA, hopefully also improving the quality of solutions in high dimensional spaces, because of the separated search of the islands, which changes the behavior of the serial version and yields a new kind of algorithm [14,15].

Algorithm 1. Outline of an island in $\mathcal{S}$-PAMICRO

Input: MOP, stopping criterion, island identification i, number of islands l, number of migrants $nmig$, and frequency of migration $fmig$.
Output: Final sub-population A
1: $A \leftarrow \emptyset$ {initialize external archive}
2: $n \leftarrow l|P|$ {archive size limit}
3: Initialize micro-population P at random
4: **while** the stopping criterion is not satisfied **do**
5: $P \leftarrow$ SMS-EMOA(MOP, $fmig$, P) {execute during $fmig$ evaluations of the objective vector}
6: $R \leftarrow$ Check the arrival of migrants from $(l + i - 1) \pmod{l}$ island
7: $A \leftarrow A \cup P \cup R$
8: **if** $|A| > n$ **then**
9: $A \leftarrow$ Pruning(A,n) {see Algorithm 2}
10: $S \leftarrow$Uniform_Random_Selection(A,$nmig$) {$nmig$ random solutions are selected from A}
11: Send copies of S to the $(i + 1) \pmod{l}$ island
12: $P \leftarrow$ Elitist_Ranking_Replacement($P \cup R$) {dominated individuals are likely to be discarded}
13: **return** A

In Algorithm 1, we present the pseudocode of an island in $\mathcal{S}$-PAMICRO. First, the external archive A and its maximum size are specified. Next, the micro-population P is initialized at random. In line 5, SMS-EMOA is executed during $fmig$ function evaluations. Then, an island receives, without blocking, the immigrants R from the source island, according to the adopted topology. In line 7, the external archive is updated, adding the current micro-population as well as the immigrants. In lines 8 and 9, the external archive is truncated if it exceeds its limits, using the technique described in the next paragraph. In the following two lines, the candidates to be migrated are selected by using the policy of uniform-random migration [15], in which $nmig$ individuals are randomly

[7] This is known as *migration*.

selected from the archive and a copy of them is sent to the destination island. In line 12, the micro-population is updated, replacing $|R|(<|P|)$ individuals with the immigrants. Here, we employed elitist-ranking replacement [15], where immigrants are combined with the current population, and then they are ranked using Pareto dominance, and the worst solutions are removed. This elitist mechanism preserves the currently best solutions for the next iteration, assuring proximity to the Pareto optimal front. At the end, the final sub-populations of all islands $i \in \{0, 1, \ldots, l-1\}$ are collected and adjusted to the size $l \times |P|$, using the same pruning technique. This operation is performed by a designated island.

Algorithm 2. Pruning

Input: Population P, desired size n
Output: Reduced population P
1: $\{F_1, \ldots, F_k\} \leftarrow$ Rank population P in k fronts according to Pareto dominance.
2: Calculate $\boldsymbol{z}^{min}$ and $\boldsymbol{z}^{max}$
3: Normalize population $p.\boldsymbol{y} \leftarrow \frac{p.\boldsymbol{y} - \boldsymbol{z}^{min}}{\boldsymbol{z}^{max} - \boldsymbol{z}^{min}}$, $\forall p \in P, p.\boldsymbol{y} \in \mathbb{R}^m$
4: **while** $|P| > n$ **do**
5:   **if** $|F_k| \leq |P| - n$ **then** {Remove members of the k-th front}
6:     $r \leftarrow F_k$
7:     $k \leftarrow k - 1$
8:   **else**
9:     $D \leftarrow$ Calculate density of P based on the Parallel-Coordinates graph
10:     $r \leftarrow \arg\max_{\boldsymbol{p} \in F_k} D[p]$
11:     $F_k \leftarrow F_k \setminus \{r\}$
12:   $P \leftarrow P \setminus \{r\}$
13: **return** P

Our pruning technique is explained in Algorithm 2. First the population is ranked using the well-known non-dominated sorting procedure [4, p. 93]. In lines 2 and 3, the population is normalized in the objective space by means of two reference points: z^{min}, composed of the best objective values found so far, and z^{max}, formed with those vectors parallel to the axes with the lowest Euclidean norm. Next, all members of the worst current k-th front are removed if the size of this front is less or equal than the number of individuals to be removed (lines 5–7). Otherwise, the individual with the highest density value is eliminated from the current front (lines 9–11) until the desired size is achieved.

The density estimator, originally proposed in [8], is based on a visualization technique, called Parallel Coordinates. In this technique, a graph is built in the 2-dimensional plane where m copies of the real line $\mathbb{R}$ are placed perpendicular to the x-axis and a solution in $\mathbb{R}^m$ is represented by a series of connected line segments with vertices on the parallel axes. The core idea in the density estimator is to represent the Parallel Coordinates of each distinct pair of objective functions as a 2D matrix, where the $m(m-1)/2$ graphs are attached next to each other and only normalized individuals are considered. The dimension of this matrix depends on a resolution parameter (γ). An element of the matrix identifies the level of overlapping line segments and those individuals covering a wide area of the matrix have a better density estimator. Interested readers are referred to [8] for more details.

$\mathcal{S}$-PAMICRO was developed in the *EMO Project*,[8] our framework for Evolutionary Multi-Objective Optimization. This software is implemented in C language and MPICH.[9]

3 Experimental Results

In this section, we investigate the effectiveness of $\mathcal{S}$-PAMICRO on the Walking-Fish-Group (WFG) test suite [4, p. 209]. In this benchmark, properties, such as non-separability, multi-modality, deceptiveness and bias, are preserved as we increase the number of objectives, making these problems harder to solve for a MOEA. The decision variables (n) and the position-related parameter (k) are specified in Table 1.

We compared the results of our proposed algorithm with respect to SMS-EMOA, its parallel version using the asynchronous island model without external archives (pSMS-EMOA), and the Hypervolume Estimation Algorithm (HypE) [2] for 2, 3, 5 and 10 objectives. HypE ranks the population by means of Pareto dominance and its secondary selection criterion is based on the estimation of the hypervolume contributions using Monte Carlo sampling (for 2 and 3 objectives, the exact value is computed). All the MOEAs were implemented in the EMO Project, using real-numbers encoding.

The variation operators were polynomial-based mutation and simulated binary crossover (SBX) [5]. The crossover rate and its distribution index were set to 0.9 and 20, for 2 and 3 objectives, and 1.0 and 30 for many-objective problems. The mutation rate and its distributed index was set to $1/n$ and 20, respectively. For HypE, the number of sampling points was fixed to 20,000 and the resolution parameter of $\mathcal{S}$-PAMICRO (γ), as suggested in [8], is shown in Table 1.

Table 1. Parameters adopted in our experiments

m	WFG		MOEAs	pMOEAs		$feval$	$\mathcal{S}$-PAMICRO
	n	k	$\|P\|$	$\|P\|$	l		γ
2	24	4	100	10	10	40,000	3
3	24	4	120	10	12	50,000	2
5	47	8	196	11	18	50,000	2
10	105	18	276	11	25	80,000	2

The stopping criterion consisted of reaching a maximum number of objective function evaluations ($feval$), limiting the execution time to no more than two hours for each run. For fair comparisons, the parameters were similar in the

[8] Available at http://computacion.cs.cinvestav.mx/~rhernandez.
[9] https://www.mpich.org.

sequential and parallel cases. The population size $|P|$ of the sequential algorithms (SMS-EMOA/HypE) and the parallel MOEAs (pSMS-EMOA/$\mathcal{S}$-PAMICRO) are defined in Table 1, as well as the number of islands or processors (l) in the latter case. Here, l is equivalent to the division of the overall population size among the micro-population size. Experiments were carried on a Cluster of 10 PCs Intel(R) Core(TM) i7 CPU 950 @ 3.07 GHz $\times$ 8 with 3.8 GB memory. The frequency of migration, $fmig$, was set to 80 function evaluations and the number of migrants $nmig$ was set to 2 (these values were empirically determined). We performed 30 independent runs for all scenarios. For comparing results, we adopted the hypervolume indicator, bounded by the reference points $(3, 5, 7, \ldots)$ for the instances WFG1 and WFG3; and $(2.2, 4.2, 6.2, \ldots)$ for the rest of the problems. We applied the Wilcoxon rank sum test (one-tailed) to the mean hypervolume indicator values, in order to determine whether $\mathcal{S}$-PAMICRO performed better than the other MOEAs at the significance level of 5 %.

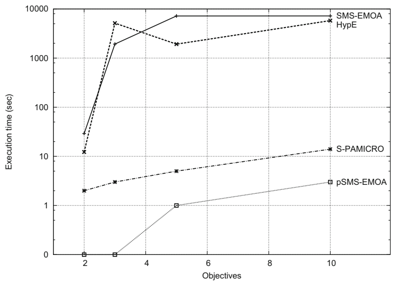

Fig. 2. Average execution time of optimizers.

The average execution time, using a logarithmic scale for the y-axis, is shown in Fig. 2. As it can be observed, $\mathcal{S}$-PAMICRO spent considerably less time than SMS-EMOA and HypE. For example, with 10 objectives, a run of our proposed approach took only 16 s out of the two hours that were allowed to the other MOEAs. Using 5 objective functions, $\mathcal{S}$-PAMICRO ended in 5 s, in contrast to the 26 min spent by HypE. Even in low dimensionality, our algorithm could reduce the run time a little bit. Furthermore, the overhead of handling the external archive in $\mathcal{S}$-PAMICRO is relatively low, compared to pSMS-EMOA that was the fastest optimizer.

On the other hand, interesting results with respect to the quality of solutions were obtained. In Table 2, we present the hypervolume indicator values of all the experiments. An arrow pointing upwards ($\uparrow$) means that our algorithm outperformed in a significantly better way, the other MOEAs compared. Conversely, an arrow pointing downwards ($\downarrow$) means that our algorithm was significantly beaten. An asterisk ($*$) means that the algorithm was interrupted because the allowed execution time was exceeded. In the majority of the cases for 5 and 10 objectives, $\mathcal{S}$-PAMICRO obtained the best results, outperforming SMS-EMOA,

Table 2. Median and standard deviation of the hypervolume indicator on the WFG benchmark. The two best values are shown in gray scale, where a darker tone corresponds to the best value.

m	HypE	SMS-EMOA	pSMS-EMOA	$\mathcal{S}$-PAMICRO
		WFG1		
2	5.17e+00 4.11e-1 ↑	4.45e+00 3.63e-1 ↑	3.66e+00 2.59e-1 ↑	6.61e+00 9.65e-1
3	5.66e+01 1.62e+0 ↓	5.28e+01 2.50e+0 ↑	4.23e+01 3.08e+0 ↑	5.56e+01 3.71e+0
5	2.82e+03 1.17e+2 ↑	3.18e+03 7.20e+1 * ↑	3.91e+03 4.83e+1 ↑	5.16e+03 3.88e+2
10	4.19e+09 1.81e+8 ↑	1.88e+09 2.62e+8 * ↑	5.28e+09 5.76e+7 ↑	5.87e+09 2.33e+8
		WFG2		
2	5.46e+00 2.79e-2 ↑	5.47e+00 1.25e-1 ↑	5.39e+00 1.71e-1 ↑	5.49e+00 4.00e-2
3	5.34e+01 4.21e+0 ↓	4.47e+01 4.47e+0	5.18e+01 2.00e+0 ↑	5.32e+01 2.50e-1
5	4.24e+03 3.00e+2 ↑	4.41e+03 3.32e+2 * ↑	4.66e+03 1.52e+1 ↑	4.75e+03 2.00e+1
10	4.66e+09 3.22e+8 ↑	3.80e+09 2.86e+8 * ↑	4.91e+09 1.75e+8 ↑	4.93e+09 1.96e+8
		WFG3		
2	1.09e+01 3.06e-2 ↑	1.09e+01 2.09e-2 ↑	1.08e+01 3.23e-2 ↑	1.09e+01 4.50e-2
3	7.59e+01 2.19e-1 ↑	7.60e+01 1.52e-1	7.48e+01 1.06e-1 ↑	7.61e+01 3.61e-1
5	5.55e+03 1.55e+2 ↑	6.84e+03 5.88e+1 * ↑	6.93e+03 3.11e+1 ↑	7.22e+03 5.86e+1
10	8.37e+09 1.38e+8 ↓	7.64e+09 1.95e+8 * ↑	5.91e+09 3.30e+8 ↑	8.19e+09 1.98e+9
		WFG4		
2	2.91e+00 3.46e-3 ↓	2.90e+00 1.08e-2	2.77e+00 2.05e-2 ↑	2.90e+00 2.10e-2
3	2.96e+01 5.19e-2 * ↓	2.97e+01 5.43e-2 ↓	2.66e+01 2.41e-1 ↑	2.88e+01 4.45e+0
5	1.69e+03 9.10e+1 ↑	2.50e+03 6.71e+1 * ↑	3.13e+03 7.15e+1 ↑	3.47e+03 1.16e+2
10	1.86e+09 1.03e+8 * ↓	1.37e+09 6.15e+7 * ↓	2.00e+09 4.38e+8 ↓	1.22e+09 5.81e+8
		WFG5		
2	2.59e+00 2.40e-3 ↑	2.58e+00 2.82e-3 ↑	2.53e+00 1.21e-2 ↑	2.59e+00 8.62e-3
3	2.74e+01 7.07e-1 * ↓	2.73e+01 1.38e-1 ↓	2.52e+01 1.92e-1 ↑	2.70e+01 1.46e-1
5	1.96e+03 1.33e+2 ↑	2.47e+03 5.10e+1 * ↑	2.75e+03 1.50e+2 ↑	3.31e+03 9.51e+1
10	1.95e+09 1.06e+8 * ↑	1.04e+09 3.14e+7 * ↑	1.04e+09 3.47e+8 ↑	3.99e+09 6.24e+8
		WFG6		
2	2.65e+00 5.79e-2 ↑	2.64e+00 5.43e-2 ↑	2.56e+00 3.93e-2 ↑	2.68e+00 2.11e-2
3	2.77e+01 2.68e-1	2.79e+01 2.12e-1 ↓	2.52e+01 3.86e-1 ↑	2.77e+01 4.05e-1
5	1.80e+03 1.37e+2 ↑	2.08e+03 7.00e+1 * ↑	2.93e+03 6.19e+1 ↑	3.39e+03 6.23e+1
10	1.83e+09 1.28e+8 ↑	9.82e+08 3.55e+7 * ↑	2.02e+09 2.55e+8 ↑	3.83e+09 5.36e+8
		WFG7		
2	2.92e+00 1.60e-3 ↓	2.91e+00 1.05e-2 ↓	2.84e+00 1.25e-2 ↑	2.91e+00 3.05e-1
3	2.97e+01 2.72e-2 * ↓	2.99e+01 1.35e-2 ↓	2.73e+01 2.64e-1 ↑	2.93e+01 1.95e-1
5	1.82e+03 1.10e+2 ↑	2.66e+03 7.07e+1 * ↑	3.20e+03 7.84e+1 ↑	3.55e+03 4.62e+1
10	2.22e+09 1.08e+8 ↓	1.26e+09 5.23e+7 *	1.12e+09 2.77e+8	8.52e+08 7.72e+8
		WFG8		
2	2.25e+00 1.46e-2 ↓	2.24e+00 1.13e-2 ↓	2.10e+00 2.99e-2 ↑	2.24e+00 3.37e-2
3	2.34e+01 2.82e-1 ↑	2.52e+01 8.04e-2 ↓	2.19e+01 4.28e-1 ↑	2.43e+01 5.25e-1
5	1.52e+03 1.20e+2 ↑	2.26e+03 5.62e+1 * ↑	2.55e+03 1.16e+2 ↑	2.86e+03 3.62e+2
10	1.84e+09 1.29e+8 ↓	1.06e+09 4.60e+7 * ↓	1.53e+09 3.69e+8 ↓	4.64e+08 7.71e+8
		WFG9		
2	2.30e+00 2.61e-1 ↑	2.78e+00 2.34e-1 ↑	2.63e+00 2.09e-1 ↑	2.81e+00 4.88e-1
3	2.16e+01 1.56e+0 * ↑	2.82e+01 1.77e+0 ↓	2.25e+01 1.10e+0 ↑	2.74e+01 6.78e+0
5	1.75e+03 1.65e+2 ↑	2.36e+03 1.12e+2 * ↑	2.57e+03 6.33e+1	2.61e+03 8.93e+2
10	1.66e+09 1.10e+8 ↑	1.12e+09 6.31e+7 * ↑	1.87e+09 3.46e+8 ↑	2.31e+09 9.27e+8

HypE and pSMS-EMOA. While with 2 and 3 objectives, our proposal only surpassed pSMS-EMOA, being competitive with SMS-EMOA and HypE.

In summary, we observed that $\mathcal{S}$-PAMICRO could achieve much better results than SMS-EMOA and HypE in high dimensionality, spending much less computational time. For this reason, we claim that our proposed approach is a promising alternative for solving many-objective optimization problems.

4 Conclusions and Future Work

This paper presented a parallel version of the $\mathcal{S}$-Metric Selection Evolutionary Multi-Objective Algorithm (SMS-EMOA). The new approach, called PArallel MICRo Optimizer based on the $\mathcal{S}$ metric ($\mathcal{S}$-PAMICRO), draws ideas from the asynchronous island model with relatively small populations. Diversity is preserved through external archives that are pruned to a limit size, using a recently proposed technique that is based on automatic image analysis. We compared our proposal with respect to HypE (Hypervolume Estimation Algorithm), and with respect to the serial version of SMS-EMOA and another parallel version of it. We observed that $\mathcal{S}$-PAMICRO is a viable alternative for solving many-objective optimization problems at an affordable computational time. In fact, the execution time seems to be dominated by polynomial terms and not the exponential terms when using micro-populations. The model of the execution time of $\mathcal{S}$-PAMICRO is $1.526m$–1.632, using least-squares approximation. Further studies are nevertheless required, adopting more benchmarks and comparing to other state-of-the-art MOEAs. We are also interested in studying the effects of the additional parameters related to the migration operator.

References

1. Adra, S.F., Fleming, P.J.: Diversity management in evolutionary many-objective optimization. IEEE Trans. Evol. Comput. **15**(2), 183–195 (2011)
2. Bader, J., Zitzler, E.: HypE: an algorithm for fast hypervolume-based many-objective optimization. Evol. Comput. **19**(1), 45–76 (2011)
3. Bringmann, K., Friedrich, T.: Don't be greedy when calculating hypervolume contributions. In: FOGA 2009: Proceedings of the Tenth ACM SIGEVO Workshop on Foundations of Genetic Algorithms, Orlando, Florida, USA, pp. 103–112. ACM, January 2009
4. Coello Coello, C.A., Lamont, G.B., Van Veldhuizen, D.A.: Evolutionary Algorithms for Solving Multi-objective Problems, 2nd edn. Springer, New York (2007). ISBN 978-0-387-33254-3
5. Deb, K., Agrawal, R.B.: Simulated binary crossover for continuous search space. Complex Syst. **9**, 115–148 (1995)
6. Emmerich, M.T.M., Beume, N., Naujoks, B.: An EMO algorithm using the hypervolume measure as selection criterion. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) EMO 2005. LNCS, vol. 3410, pp. 62–76. Springer, Heidelberg (2005)

7. Fleischer, M.: The measure of Pareto optima applications to multi-objective metaheuristics. In: Fonseca, C.M., Fleming, P.J., Zitzler, E., Deb, K., Thiele, L. (eds.) EMO 2003. LNCS, vol. 2632, pp. 519–533. Springer, Heidelberg (2003)
8. Hernández Gómez, R., Coello Coello, C.A.: A multi-objective evolutionary algorithm based on parallel coordinates. In: Proceedings of the 2016 Genetic and Evolutionary Computation Conference (GECCO 2016), New York, NY, USA. ACM Press (2016, in press)
9. Ishibuchi, H., Tsukamoto, N., Nojima, Y.: Evolutionary many-objective optimization: a short review. In: IEEE Congress on Evolutionary Computation, CEC 2008 (IEEE World Congress on Computational Intelligence), pp. 2419–2426, June 2008
10. Klinkenberg, J.-W., Emmerich, M.T.M., Deutz, A.H., Shir, O.M., Bäck, T.: A reduced-cost SMS-EMOA using Kriging, self-adaptation, and parallelization. In: Ehrgott, M., Naujoks, B., Stewart, T.J., Wallenius, J. (eds.) Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems, vol. 634, pp. 301–311. Springer, Heidelberg (2010)
11. Li, B., Li, J., Tang, K., Yao, X.: Many-objective evolutionary algorithms: a survey. ACM Comput. Surv. **48**(1), 13:1–13:35 (2015)
12. Lopez, E.M., Antonio, L.M., Coello Coello, C.A.: A GPU-based algorithm for a faster hypervolume contribution computation. In: Gaspar-Cunha, A., Henggeler Antunes, C., Coello Coello, C.A. (eds.) EMO 2015. LNCS, vol. 9019, pp. 80–94. Springer, Heidelberg (2015)
13. Lücken, C., Barán, B., Brizuela, C.: A survey on multi-objective evolutionary algorithms for many-objective problems. Comput. Optim. Appl. **58**(3), 707–756 (2014)
14. Luna, F., Alba, E.: Parallel multiobjective evolutionary algorithms. In: Kacprzyk, J., Pedrycz, W. (eds.) Springer Handbook of Computational Intelligence, pp. 1017–1031. Springer, Heidelberg (2015)
15. Van Veldhuizen, D.A., Zydallis, J.B., Lamont, G.B.: Considerations in engineering parallel multiobjective evolutionary algorithms. IEEE Trans. Evol. Comput. **7**(2), 144–173 (2003)
16. Wagner, T., Beume, N., Naujoks, B.: Pareto-, aggregation-, and indicator-based methods in many-objective optimization. In: Obayashi, S., Deb, K., Poloni, C., Hiroyasu, T., Murata, T. (eds.) EMO 2007. LNCS, vol. 4403, pp. 742–756. Springer, Heidelberg (2007)
17. While, L., Bradstreet, L., Barone, L.: A fast way of calculating exact hypervolumes. IEEE Trans. Evol. Comput. **16**(1), 86–95 (2012)
18. Zhou, A., Bo-Yang, Q., Li, H., Zhao, S.-Z., Suganthan, P., Zhang, Q.: Multiobjective evolutionary algorithms: a survey of the state of the art. Swarm Evol. Comput. **1**(1), 32–49 (2011)
19. Zitzler, E., Künzli, S.: Indicator-based selection in multiobjective search. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 832–842. Springer, Heidelberg (2004)