A Parallel Island Approach to Multiobjective Feature Selection for Brain-Computer Interfaces

Julio Ortega[1(✉)], Dragi Kimovski[2], John Q. Gan[3], Andrés Ortiz[4], and Miguel Damas[1]

[1] Department of Computer Architecture and Technology, CITIC, University of Granada, Granada, Spain
{jortega,mdamas}@ugr.es

[2] University of Innsbruck, Innsbruck, Austria
dragi@dps.uibk.ac.at

[3] School of Computer Science and Electronic Engineering, University of Essex, Colchester, UK
jqgan@essex.ac.uk

[4] Department of Communications Engineering, University of Málaga, Málaga, Spain
aortiz@ic.uma.es

Abstract. This paper shows that parallel processing is useful for feature selection in brain-computer interfacing (BCI) tasks. The classification problems arising in such application usually involve a relatively small number of high-dimensional patterns and, as curse of dimensionality issues have to be taken into account, feature selection is an important requirement to build suitable classifiers. As the number of features defining the search space is high, the distribution of the searching space among different processors would contribute to find better solutions, requiring similar or even smaller amount of execution time than sequential counterpart procedures. We have implemented a parallel evolutionary multiobjective optimization procedure for feature selection, based on the island model, in which the individuals are distributed among different subpopulations that independently evolve and interchange individuals after a given number of generations. The experimental results show improvements in both computing time and quality of EEG classification with features extracted by multiresolution analysis (MRA), an approach widely used in the BCI field with useful properties for both temporal and spectral signal analysis.

Keywords: Brain-computer interfaces (BCI) · Feature selection · Island model based evolutionary algorithms · Multiresolution analysis (MRA) · Parallel multiobjective optimization

1 Introduction

Many classification tasks in bioinformatics deal with patterns defined by a large number of features. Moreover, these high-dimensional classification problems have frequently to be solved with the number of patterns smaller than the number of features, thus presenting curse of dimensionality problems [1]. Therefore, feature selection is usually required in bioinformatics [2] to eliminate redundant and noisy features in order to improve the accuracy and interpretability of the classifiers.

Among the three different approaches for feature selection (filter, wrapper and embedded methods) [2], our proposal in this paper corresponds to a wrapper procedure. Although wrapper approaches use the classifier performance to evaluate the utility of a given set of features and thus they are classifier-dependent, they are usually recognized as the preferable approaches whenever they would be feasible [3].

One of the issues to be taken into account in the design of a feasible wrapper-based feature selection procedure is the number of possible features because the size of the searching space depends exponentially on that number. In high-dimensional classification problems, several hundreds or even thousands of features usually define a very huge searching space where efficient metaheuristics are required. This paper proposes a parallel multiobjective evolutionary algorithm, in which the individuals of a population represent different feature selections, and the fitness of a given individual is determined through the evaluation of the classifier performance after training it with the corresponding patterns defined by the set of selected features. Parallel processing has been previously considered to take advantage of high performance computer architectures for feature selection [4–8]. In [7, 8] feature selection is approached using parallel multiobjective and cooperative coevolutionary procedures implemented through a master-worker parallel model. In this paper, we propose an island model to implement parallel multiobjective feature selection applied to BCI.

In what follows, Sect. 2 describes our approach to feature selection based on multiobjective optimization and its parallel implementation through the island model. The application considered in this paper corresponds to BCI tasks related with motor imagery, where the features of the patterns are obtained by using Multiresolution Analysis (MRA). The details of the application and the patterns in the database used are provided in Sect. 3. Finally, Sect. 4 describes the experimental results and the conclusions are given in Sect. 5.

2 A Parallel Island Procedure for Multiobjective Feature Selection

As this paper deals with supervised classification problems in which the labels of the training and test patterns are known, it would be possible to evaluate the performance of a classifier from the accuracy obtained after training it (by using the set of training patterns). Nevertheless, other measures that quantify properties such as the generalization capability should be taken into account in order to improve the behavior of the classifier in a real environment, where patterns that have not been used for training have to be processed. To tackle these issues, we propose a multiobjective evolutionary procedure where the selection of features is optimized for both accuracy and generalization capability, both evaluated by using the training patterns.

Figure 1.a shows a scheme of the wrapper method we propose for feature selection based on a multiobjective optimization procedure that searches a vector of decision variables $\mathbf{x} = [x_1, x_2, \ldots, x_n] \in R^n$ to optimize a function vector $\mathbf{f}(\mathbf{x})$, whose scalar

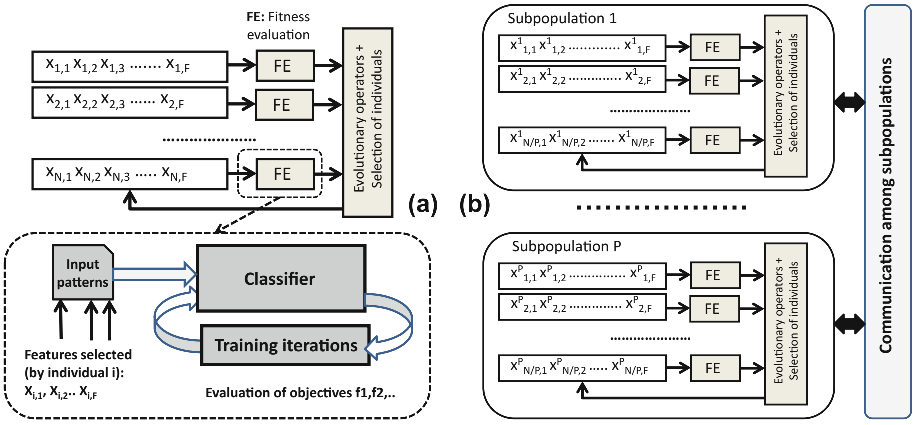
FE: Fitness evaluation
$x_{1,1}\, x_{1,2}\, x_{1,3}$ .... $x_{1,F}$
$x_{2,1}\, x_{2,2}\, x_{2,3}$ ...... $x_{2,F}$
$x_{N,1}\, x_{N,2}\, x_{N,3}$ ..... $x_{N,F}$
FE
Evolutionary operators + Selection of individuals
(a)
(b)
Input patterns
Classifier
Training iterations
Features selected (by individual i): $x_{i,1}, x_{i,2} .. x_{i,F}$
Evaluation of objectives f1,f2,..
Subpopulation 1
$x^1_{1,1}\, x^1_{1,2}$ .... $x^1_{1,F}$
$x^1_{2,1}\, x^1_{2,2}$ .... $x^1_{2,F}$
$x^1_{N/P,1}\, x^1_{N/P,2}$ .... $x^1_{N/P,F}$
Subpopulation P
$x^P_{1,1}\, x^P_{1,2}$ .... $x^P_{1,F}$
$x^P_{2,1}\, x^P_{2,2}$ .... $x^P_{2,F}$
$x^P_{N/P,1}\, x^P_{N/P,2}$ ..... $x^P_{N/P,F}$
Communication among subpopulations

Fig. 1. Wrapper approach to feature selection by evolutionary multiobjective optimization: (a) sequential procedure; (b) island parallel procedure proposed in the paper

values $(f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x}))$ represent the m objectives to optimize. These objectives are usually in conflict, and thus multiobjective optimization should obtain a set of *non-dominated* solutions called Pareto optimal solutions that define the *Pareto front* (no solution in the Pareto front is worse than the others when all the objectives are taken into account), from which it is possible to choose the most convenient solution in specific circumstances. To solve the multiobjective optimization problem we have implemented an evolutionary algorithm based on the NSGA-II algorithm [9], with specific individual codification and genetic operators.

Algorithm 1. Parallel multi-objective feature selection procedure adopted in Fig. 1.b

```
Parallel_NSGAII_featureselection (N individuals, P threads)
    Create P Initialization(i,N/P,SP(i)) threads; // i=1,…,P
    wait(i); // barrier to synchronize P threads;
    Create P Island_evolution(i,N/P,SP(i),commprof,comm,genpar) threads; // i=1,…,P
    wait(i); // barrier to synchronize P threads
    NSGAII_nondomination_sort(SP(1),…,SP(P));// nondomination sort of subpop.
    save results;
    end

Island_evolution(i,N/P,SP(i),commprof,comm,genpar)
    for  j=1 to comm
      for k=1 to genpar
            (SP'(i),f(SP'(i)))=NSGAII_tournament_selection (SP(i),f(SP(i)));
            SP''(i)=Genetic_operators (SP'(i));
            f(SP''(i)) = Evaluation(SP''(i),DS);
            (SP*(i))=NSGAII_nondomination_sort ( SP(i);SP''(i));
            (P,f(P))=NSGAII_replace_chromosome ( SP*(i));
       end;
       communication(SP(1),…,SP(P),commprof);
    end;
```

The main contribution of this paper is a parallel implementation of the procedure summarized in Fig. 1.a based on an island model. This parallel approach, depicted in Fig. 1.b, distributes the N individuals of the population among the available P processors thus defining P subpopulations, each with N/P individuals. The pseudocode description of the parallel procedure is provided in Algorithm 1. The procedure first creates P threads with initialization and evaluate N/P individuals of the population (line 01 of `Parallel_NSGAII_featureselection`). These P threads are synchronized through a barrier in line 02 to perform the evolution of P subpopulations, each including N/P individuals, according to procedure `Island_evolution`. As it can be seen, each thread requires the number of communications (`comm`), the number of generations that the corresponding subpopulation has to complete between communications (`genpar`), and the randomly selected couples of threads that have to communicate after each genpar number of generations (`commprof`). This parallel evolutionary multiobjective procedure, whose behavior is different from the sequential one, allows improvement in the quality of the solutions found by using bigger populations and/or reduction in computing time.

3 Feature Selection in BCI with Multiresolution Analysis

The high-dimensional classification problem considered in this paper deals with brain-computer interfacing (BCI) based on the classification of EEG signals corresponding to motor imagery (MI) tasks. This BCI paradigm uses the series of amplifications and attenuations of short duration occasioned by limb movement imagination, the so called event related desynchronization (ERD) and event related synchronization (ERS). In [10] several approaches for multiobjective feature selection in a MRA (Multiresolution Analysis) system for BCI are proposed and evaluated. A MRA system [11] applies a sequence of successive approximation spaces to describe the target signal, thus being useful whenever the target signal presents different characteristics across the approximation spaces. As a specific example of MRA systems, the discrete wavelet transform (DWT) was applied in [10, 12] to characterize EEGs from motor imagery (MI) tasks.

The patterns used in this work are built, from EEG trials, by a feature extraction procedure based on the MRA described in [12]. Each signal obtained from each electrode contains several segments to which a set of wavelets detail and approximation coefficients are assigned. This way, considering S segments, E electrodes, and L levels of wavelets, each EEG pattern is characterized by $2 \times S \times E \times L$ sets of coefficients. The number of coefficients in each level set depends on the level. In the dataset considered here, which was recorded in the BCI Laboratory at the University of Essex, S = 20 segments, E = 15 electrodes, and L = 6 levels. Therefore, 3600 sets of wavelet coefficients in total in each pattern, with from 4 to 128 coefficients in each set, characterize each pattern: a total of 151200 coefficients. Nevertheless, in [12] only one feature is assigned to each electrode and each level of approximation and detail. It is obtained by computing the variance of the coefficient distribution and normalising the obtained values between 0 and 1. This way, $2 \times S \times E \times L = 3600$ features constitute each pattern. Anyway, as the number of training patterns for each subject is approximately 180, it is clear that an efficient procedure for feature selection is required.

In [12] an approach based on the use of several classifiers is considered to reduce the number of features characterizing the patterns applied to each classifier. Figure 2 describes the structure of the classification procedure based on a set of LDA (linear discriminant analysis) classifiers, in which a module for majority voting based on all the LDA outputs provides the final classification output. This way, a set of $2 \times S \times L$ LDA classifiers with the number of inputs equaling the number of electrodes are adopted, as shown in Fig. 2. This procedure is called OPT0 as the baseline method for performance comparison [10].

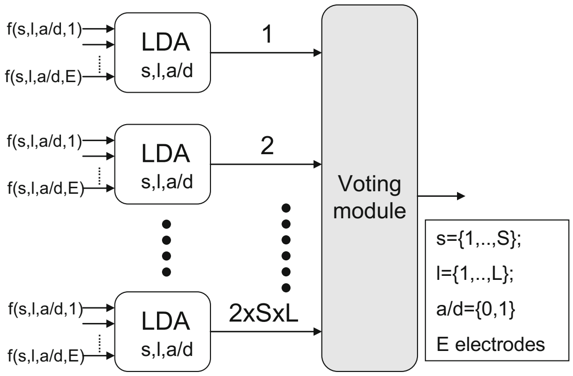

Fig. 2. EEG classification with multiple LDA classifiers based on majority voting, with one LDA classifier per segment, per level and, per type (detail and approximation) of wavelet [12]

In [10] two alternatives for feature selection in BCI with MRA, OPT1 and OPT2, were evaluated and compared with the performance of OPT0. The alternative OPT2 selects a set of LDAs among the $2 \times S \times L$ LDAs in the structure of Fig. 2 through the multiobjective optimization procedure described in Sect. 2. OPT1 is a simpler alternative as it uses only one LDA classifier based on a subset of features selected from all the available features. The two cost functions used in the multiobjective feature selection in OPT1 and OPT2 take into account the labels assigned to the training patterns to identify their corresponding classes. Moreover, to characterize the performance of the classifier while it has been trained or adjusted for a given set of features (i.e., an individual of the population), it is important not only to take into account the accuracy obtained for the training set but also its generalization capabilities, i.e., its accuracy for unseen instances. Thus, the first cost function is related with the Kappa index [13] on training dataset, which takes into account the distribution of the per class error as it is computed as $(p_0 - p_c)/(1 - p_c)$, with p_0 equal to the proportion of coincidences among the classification outputs and the labels of the patterns and p_c being the proportion of patterns on which the coincidence is expected by chance. The second cost function is the average loss function in a 10-fold cross validation analysis to the training patterns. This paper proposes and evaluates parallel implementation of OPT1 and OPT2.

4 Experimental Results

The parallel procedures for OPT1 and OPT2 have been implemented by using the Parallel Computing Toolbox of Matlab® (version 8.3) and executed in a node including two Intel Xeon E5-2620 processors (providing up to 12 threads) at 2.1 GHz and 32 GB DDR3 RAM per node. The experiments were conducted using the dataset recorded in the BCI Laboratory at the University of Essex [12]. For each subject, there is one data file with data recorded for training (training patterns) and another file with data for evaluation (test patterns). Each data file contains about 180 labelled patterns with data from 20 segments (S = 20), six levels (L = 6) of approximation or detail coefficients (a/d = 2), and 15 electrodes (E = 15). The labels correspond to three imagined movements of right hand, left hand, and feet. Each experiment has been repeated ten times in order to complete an analysis to determine the statistical significance of the observed differences among alternatives. The results provided in what follows correspond to the EEG data from subjects 104, 107, and 110 (those achieving the best performance results in [12]).

Table 1 provides the average Kappa index values [13] obtained by the baseline alternative OPT0 [10], which uses the structure of LDA classifiers [12] shown in Fig. 2, and those by the option OPT2 [10], which searches for an optimal selection of

Table 1. Comparison of Kappa indexes and execution times of OPT0 and OPT2 with 120 individuals, 50 generations, 5 and 10 generations/communication, 4 and 8 threads on data from subjects 104, 107, and 110 in the BCI dataset of University of Essex

Sbj.	Procedure	PCnf.	Kappa	Time (s.)	p-val.
104	OPT0		0.564 ± 0.000	–	
	OPT2 (50/50)		0.545 ± 0.035	20892 ± 1668	
	OPT2 (120/50)		0.547 ± 0.040	50135 ± 2222	
	OPT2 (120/50; 4 thr 10 gen/comm.)	PC11	0.523 ± 0.014	13919 ± 325	*0.04
	OPT2 (120/50; 8 thr 10 gen/comm.)	PC12	0.554 ± 0.019	7966 ± 173	0.34
	OPT2 (120/50; 4 thr 5 gen/comm.)	PC13	0.515 ± 0.023	13902 ± 378	*0.04
	OPT2 (120/50; 8 thr 5 gen/comm.)	PC14	0.535 ± 0.034	8095 ± 126	0.75
107	OPT0		0.631 ± 0.000	–	
	OPT2 (50/50)		0.634 ± 0.019	21167 ± 1134	
	OPT2 (120/50)		0.652 ± 0.022	50749 ± 1578	
	OPT2 (120/50; 4 thr 10 gen/comm.)	PC11	0.657 ± 0.020	14214 ± 315	1.00
	OPT2 (120/50; 8 thr 10 it/comm.)	PC12	0.655 ± 0.017	8128 ± 129	0.33
	OPT2 (120/50; 4 thr 5 it/comm.)	PC13	0.645 ± 0.014	14342 ± 134	0.20
	OPT2 (120/50; 8 thr 5 it/comm.)	PC14	0.642 ± 0.018	8225 ± 149	0.06
110	OPT0		0.648 ± 0.000	–	
	OPT2 (50/50)		0.605 ± 0.045	18820 ± 1069	
	OPT2 (120/50)		0.619 ± 0.021	45156 ± 1991	
	OPT2 (120/50; 4 thr 10 it/comm.)	PC11	0.628 ± 0.030	12986 ± 360	0.08
	OPT2 (120/50; 8 thr 10 it/comm.)	PC12	0.631 ± 0.020	7866 ± 237	0.26
	OPT2 (120/50; 4 thr 5 it/comm.)	PC13	0.608 ± 0.034	13064 ± 501	0.15
	OPT2 (120/50; 8 thr 5 it/comm.)	PC14	0.629 ± 0.026	7863 ± 142	0.42

LDAs in the structure of Fig. 2. The data in parentheses for the OPT2 alternatives represent the number of individuals in the population and generations of the evolutionary algorithm and, in the case of parallel executions, the number of threads (4 or 8) and generations (5 or 10) executed by each subpopulation. The different parallel configurations are noted as PC11 to PC14 in the column PCnf. of Table 1. The sequential version of OPT2 with 120 individuals and 50 generations for subjects 104, 107 and 110 requires more than twelve hours, and provides solutions with average Kappa indices equal to 0.547, 0.652 and 0.619, respectively. The parallel versions of OPT2 require less computing times to achieve similar levels of performance than the sequential implementation of OPT2 with 120 individuals. The parallel OPT2 implementations also consume less time than the sequential OPT2 with a population of 50 individuals and even improve the performance of this sequential version of OPT2 for some subjects.

Table 2. Comparison of Kappa indexes and execution times of OPT0 and OPT1 with 2000 individuals, 50 generations and 100 generations, 5 and 10 generations/communication, 4 and 8 threads on data from subjects 104, 107, and 110 in the BCI dataset of University of Essex

Sbj.	Procedure	PCnf.	Kappa	Time (s.)	p-val.
104	OPT0		0.564 ± 0.000	–	
	OPT1 (50/50)		0.510 ± 0.056	4241 ± 375	
	OPT1 (120/50)		0.515 ± 0.047	10316 ± 461	
	OPT1 (960/50; 4 thr 10 gen/comm.)	PC21	0.653 ± 0.053	23279 ± 770	*0.01
	OPT1 (960/100; 4 thr 10 gen/comm.)	PC22	0.606 ± 0.053	44787 ± 414	*0.01
	OPT1 (960/50; 8 thr 10 gen/comm.)	PC23	0.634 ± 0.038	12104 ± 1261	*0.01
	OPT1 (960/50; 8 thr 5 gen/comm.)	PC24	0.698 ± 0.062	12422 ± 1074	*0.01
	OPT1 (960/100; 8thr 10gen/comm.)	PC25	0.673 ± 0.039	25751 ± 2000	*0.01
	OPT1 (1920/100; 8thr 10gen/comm.)	PC26	0.665 ± 0.033	46383 ± 1343	*0.01
107	OPT0		0.631 ± 0.000	–	
	OPT1 (50/50)		0.560 ± 0.041	4317 ± 188	
	OPT1 (120/50)		0.580 ± 0.052	10336 ± 340	
	OPT1 (960/50; 4 thr 10 gen/comm.)	PC21	0.613 ± 0.032	22653 ± 428	*0.05
	OPT1 (960/100; 4 thr 10 gen/comm.)	PC22	0.636 ± 0.037	45039 ± 625	*0.02
	OPT1 (960/50; 8 thr 10 gen/comm.)	PC23	0.603 ± 0.024	10746 ± 18	0.12
	OPT1 (960/50; 8 thr 5 gen/comm.)	PC24	0.609 ± 0.023	10778 ± 33	*0.03
	OPT1 (960/100; 8thr 10gen/comm.)	PC25	0.589 ± 0.028	22537 ± 1059	0.60
	OPT1 (1920/100; 8thr 10gen/comm.)	PC26	0.638 ± 0.029	45838 ± 975	*0.02
110	OPT0		0.648 ± 0.000	–	
	OPT1 (50/50)		0.450 ± 0.045	3867 ± 164	
	OPT1 (120/50)		0.450 ± 0.021	9312 ± 253	
	OPT1 (960/50; 4 thr 10 gen/comm.)	PC21	0.569 ± 0.031	22640 ± 461	*0.01
	OPT1 (960/100; 4 thr 10 gen/comm.)	PC22	0.564 ± 0.065	43796 ± 712	*0.01
	OPT1 (960/50; 8 thr 10 gen/comm.)	PC23	0.566 ± 0.029	10850 ± 191	*0.01
	OPT1 (960/50; 8 thr 5 gen/comm.)	PC24	0.569 ± 0.031	12037 ± 1407	*0.01
	OPT1 (960/100; 8thr 10gen/comm.)	PC25	0.556 ± 0.074	26754 ± 2595	*0.01
	OPT1 (1920/100; 8thr 10gen/comm.)	PC26	0.551 ± 0.028	45355 ± 1413	*0.01

Besides the average Kappa index values obtained by the baseline method OPT0, Table 2 gives the results for the option OPT1 [10], which searches for an optima subset of features that constitute the inputs to only one LDA classifier. Similarly, the data in parentheses in Table 2 represent the number of individuals in the population and generations of the evolutionary algorithm and, in the case of the parallel executions, the number of threads (4 or 8) and generations (5 or 10) executed by each subpopulation. The evaluated parallel configurations are noted as PC21 to PC26 in the column PCnf. of Table 2. The sequential version of OPT1 with 120 individuals and 50 generations for subjects 104, 107 and 110 requires more than four and a half hours, and provides solutions with average Kappa indices equal to 0.515, 0.580 and 0.450, respectively. As can be seen, the parallel versions of OPT1 improve the performance achieved by sequential implementations of this approach with less individuals in the population, and requires much less time than the sequential counterpart with the same population size. Moreover, some parallel configurations of OPT1 reduce the performance differences with respect to OPT2 or even overcome it for some subjects and parallel configurations.

Tables 1 and 2 also show results about the comparison tests to identify which alternatives provide statistically different performance, with p-values obtained from statistical tests with a significance level of 5%. The statistical analysis has been done through a Kruskal-Wallis test, which provides the intervals of the Kruskal-Wallis rank for each considered alternative for OPT2 in Table 1 and OPT1 in Table 2. For subjects 104, 107 and 110, the column noted as p-val. in Tables 1 and 2 gives the p-values obtained after comparing each parallel configuration to the corresponding reference sequential alternative. That is, the parallel configurations PC11 to PC14 in Table 1 have been compared to the sequential execution of OPT2 with 120 individuals and 50 generations, and the parallel configurations PC21 to PC26 in Table 2 to the sequential execution of OPT1, also with 120 individuals and 50 generations. A p-value below 0.05 (marked with an asterisk in columns p-val.) means statistically significant difference. In Table 1, except for PC11 and PC13 for subject 104, the differences in the Kappa indices attained by the evaluated parallel configurations are not statistically significant. Therefore, despite the parallel island evolutionary procedure is different from the sequential evolutionary algorithm, the quality of the results for a given configuration of individuals in the population and a given number of generations is similar to the sequential one, while the parallel procedure provides a significant reduction in the execution time. In the case of PC11 and PC13, the quality of the solutions found decreases only by less than 6% (respectively, 4.3% and 5.8%).

The average Kappa indices in Table 2 show improvements in the quality of solutions by all the parallel configurations (PC21 to PC26) for all subjects (104, 107 and 110), with respect to the reference sequential execution of OPT1 with 120 individuals and 50 generations. These improvements are statistically significant in all the cases but two, PC23 and PC25 for subject 107, where improvements are 4% and 1.6% respectively. The sequential OPT2 provides better results than the sequential OPT1. Moreover, OPT0 provides the best performance for subjects 104 and 110 and a performance close to that of OPT2 for subject 107. It should be taken into account that OPT0 and OPT2 are based on the classifier structure of Fig. 2, which uses more features and more complex classifier structure than OPT1 that requires only one LDA classifier. Table 2 shows that the parallel implementation of OPT1 improves its performance compared to the sequential OPT1 and reduces the performance differences from OPT0 and OPT2. In some cases, it even outperformed OPT0 and OPT2. For example, for subject 104 PC24 (the best parallel configuration for subject 104 with OPT1) provides better results than PC12 (the best configuration for subject 104 with OPT2) and OPT0: 0.698 against 0.554 and 0.564 respectively (a difference statistically significant with p-value equal to 0.004). For the subjects 107 and 110, OPT0 and OPT2 still achieved better performances than some parallel configurations of OPT1. Nevertheless, the differences among the best average Kappa indices for the sequential OPT0 and OPT2 with respect to the sequential OPT1 are reduced in the parallel configurations of OPT1: from 0.072 (0.652 − 0.580) to 0.019 (0.657 − 0.638) for subject 107, and from 0.198 (0.648 − 0.450) to 0.079 (0.648 − 0.569) for subject 110.

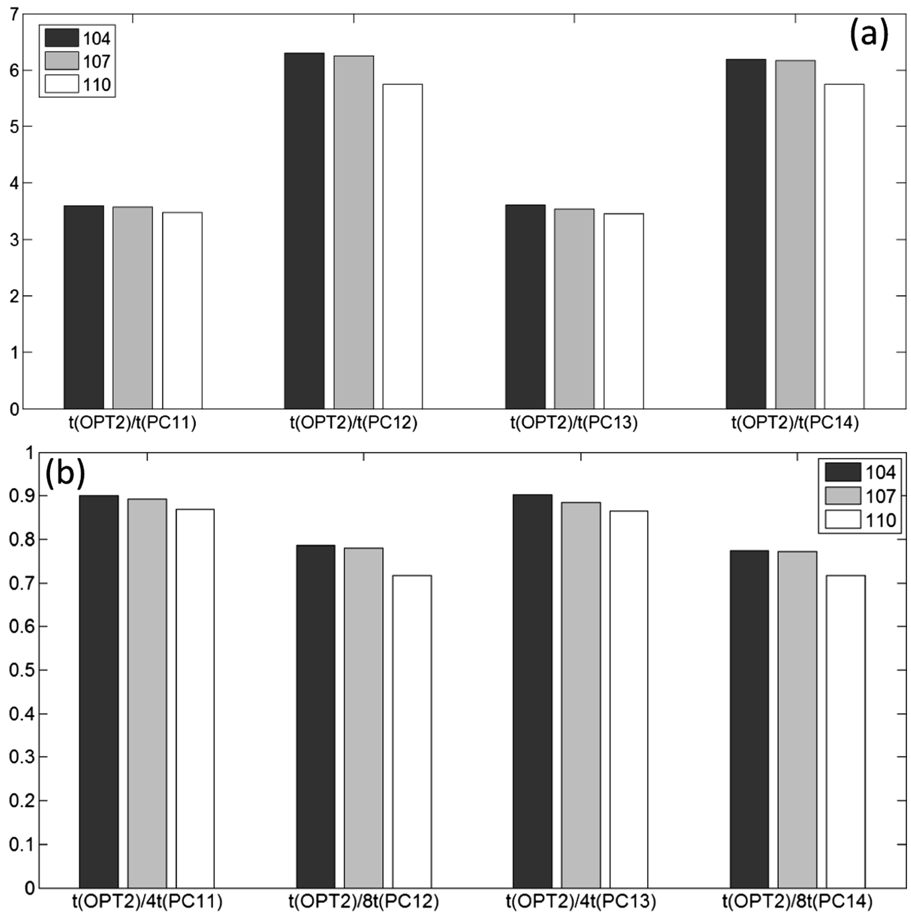

Fig. 3. Speedups (a) and efficiencies (b) for different communication profiles and parallel configurations of OPT2 (PC11 to PC14) for subjects 104, 107 and 110

The parallel approach proposed in this paper reduces the execution time with respect to the corresponding sequential version in all the cases, as shown in Table 1 (column Time(s)). Figure 3 shows the speedups (time of the sequential version divided by the time of a given parallel configuration) and efficiencies (speedups divided by the number of processors) provided by the parallel configurations PC11 to PC14 in Table 1, with respect to the sequential execution of OPT2 with 120 individuals and 50 iterations, t (OPT2)/t(PC11) to t(OPT2)/t(PC14). Two different communication profiles have been considered in our experiments. In the first communication alternative, each subpopulation independently implements 10 iterations (generations) before communicating with another subpopulation randomly selected. In the second one, the subpopulations communicate more frequently as it happens after 5 generations of independent evolution in the subpopulations. As shown in Fig. 3, for a given number of threads, the alternative communicating more frequently (5 iterations/communication) provides lower speedups. Consequently, the efficiencies observed in Fig. 3.b are slightly lower in the alternatives communicating more.

The speedups given in Fig. 4 are based on the comparison of execution time for different parallel configurations of OPT1, as shown in Table 2. In this case, we have not compared the parallel time with the corresponding sequential time due to their large values (more than four days in some cases). Figure 4 provides the speedups by using 4 threads with respect to 8 threads, i.e., t(PC21)/t(PC23) and t(PC22)/t(PC25), to compare parallel configurations with the same configuration of individuals in the population, iterations and communication profiles, although t(PC21)/t(PC23) corresponds to 50 generations and t(PC22)/t(PC25) to 100. The speedup t(PC21)/t(PC24) compares parallel configurations with 8 and 4 threads and the same number of generations (50 generations) but different communication characteristics (10 and 5 generations between communications respectively). Finally, t(PC26)/t(PC25) compares parallel configuration with the same number of threads (8 threads), generations (100 generations), and communication profiles (10 generations between communications) but different number of individuals. All those considered speedups should have a value near two, as shown in Fig. 4.

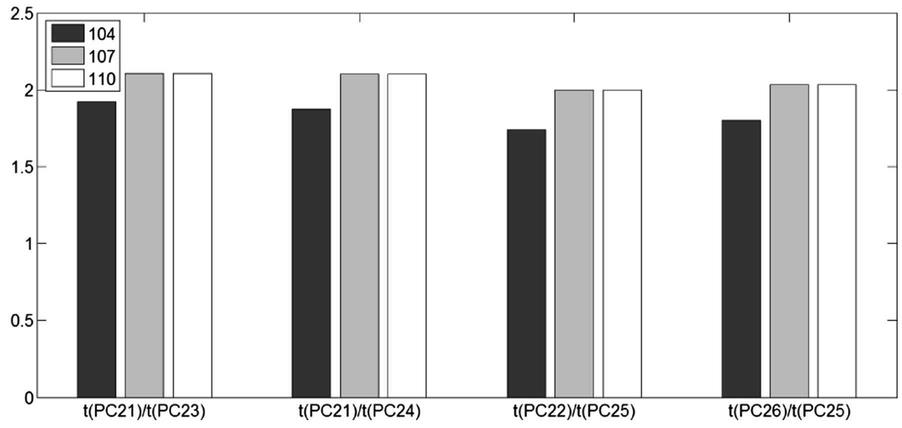

Fig. 4. Speedups of different parallel configurations of OPT1 (PC21 to PC22) for subjects 104, 107 and 110

5 Conclusions

This paper proposes and analyses parallel island implementations of multiobjective feature selection in BCI tasks with MRA. They are able to improve the quality of the solutions by using greater populations and reduce processing time as well.

On the one hand, we have shown that, despite the parallel algorithm is not exactly the same as the sequential one, it decreases the computing time without a statistically significant reduction in the solution qualities given a number of generations and individuals in the population. Although the speedups shown in Fig. 3.a correspond to efficiencies below one (Fig. 3.b), but the efficiency decrease is relatively small. The scalability behavior of the parallel procedure could be considered adequate for the number of threads used in our parallel executions.

On the other hand, the parallel procedure makes it possible to improve the quality of the solutions by using greater populations in the evolutionary algorithm but similar amount of computing time required by the sequential implementation. It has been shown that by using populations with as many individuals as the reference population multiplied by the number of threads used, OPT1using a simple classifier structure is able to match OPT2 using a complicated classifier structure or even outperform it with similar or even lower computing times.

In [8], a parallel cooperative multiobjective approach was proposed for feature selection in high-dimensional EEG data, which not only evolves independent subpopulations but also assigns different areas of the searching space to different subpopulations. This approach allows superlinear speedups in some cases although with some performance loss. The implementation of such approach to distribute the searching space will be also explored in the present parallel island algorithm for BCI with MRA, along with its use in the feature selection problem for other applications and benchmarks to compare with other previous methods.

Acknowledgements. This work was partly funded by grant TIN2015-67020-P (Spanish "Ministerio de Economía y Competitividad" and FEDER funds).

References

1. Raudys, S.J., Jain, A.K.: Small sample size effects in statistical pattern recognition: recommendations for practitioners. IEEE Trans. Pattern Anal. Mach. Intell. **13**(3), 252–264 (1991)
2. Saeys, Y., Inza, I., Larrañaga, P.: A review of feature selection techniques in bioinformatics. Bioinformatics **23**, 2507–2517 (2007)
3. Pudil, P., Somol, P.: Identifying the most informative variables for decision-making problems - a survey of recent approaches and accompanying problems. Acta Oeconomica Pragensia **2008**, 37–55 (2008)
4. de Souza, J.T., Matwin, S., Japkowitz, N.: Parallelizing feature selection. Algorithmica **45** (3), 433–456 (2006)

5. Sun, Z.: Parallel feature selection based on MapReduce. In: Wong, W.E., Zhu, T. (eds.) Computer Engineering and Networking. LNEE, vol. 277, pp. 299–306. Springer, Cham (2014). doi:10.1007/978-3-319-01766-2_35
6. Zao, Z., Zhang, R., Cox, J., Dulin, D., Sarle, W.: Massively parallel feature selection: an approach based on variance preservation. Mach. Learn. **92**(1), 195–220 (2013)
7. Kimovski, D., Ortega, J., Ortiz, A., Baños, R.: Parallel alternatives for evolutionary multi-objective optimization in unsupervised feature selection. Expert Syst. Appl. **42**(9), 4239–4252 (2015)
8. Kimovski, D., Ortega, J., Baños, A.R.: Leveraging cooperation for parallel multiobjective feature selection in high-dimensional EEG data. Concurr. Comput.: Pract. Exp. **27**, 5476–5499 (2015)
9. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J.J., Schwefel, H.-P. (eds.) PPSN 2000. LNCS, vol. 1917, pp. 849–858. Springer, Heidelberg (2000). doi:10.1007/3-540-45356-3_83
10. Ortega, J., Asensio-Cubero, J., Gan, J.Q., Ortiz, A.: Classification of motor imagery tasks for BCI with multiresolution analysis and multiobjective feature selection. BioMed. Eng. OnLine **15**(Suppl. 1), 73 (2016)
11. Daubechies, I.: Ten Lectures on Wavelets. SIAM, Philadelphia (2006)
12. Asensio-Cubero, J., Gan, J.Q., Palaniappan, R.: Multiresolution analysis over simple graphs for brain computer interfaces. J. Neural Eng. **10**(4) (2013). doi:10.1088/1741-2560/10/4/046014
13. Cohen, J.: A coefficient of agreement for nominal scales. Educ. Psychol. Meas. **20**, 37–46 (1960)