A Multi-Objective Evolutionary Algorithm Fitness Function for Case-Base Maintenance

Eduardo Lupiani[1], Susan Craw[2], Stewart Massie[2],
Jose M. Juarez[1], and Jose T. Palma[1]

[1] University of Murcia, Spain
{elupiani,jmjuarez,jtpalma}@um.es
[2] The Robert Gordon University, Scotland, UK
{s.craw,s.massie}@rgu.ac.uk

Abstract. Case-Base Maintenance (CBM) has two important goals. On the one hand, it aims to reduce the size of the case-base. On the other hand, it has to improve the accuracy of the CBR system. CBM can be represented as a multi-objective optimization problem to achieve both goals. Multi-Objective Evolutionary Algorithms (MOEAs) have been recognised as appropriate techniques for multi-objective optimisation because they perform a search for multiple solutions in parallel. In the present paper we introduce a fitness function based on the Complexity Profiling model to perform CBM with MOEA, and we compare its results against other known CBM approaches. From the experimental results, CBM with MOEA shows regularly good results in many case-bases, despite the amount of redundant and noisy cases, and with a significant potential for improvement.

1 Introduction

Case-Base Maintenance (CBM) has as its main goals control of the number of cases within the case-base and maintaining the accuracy of the CBR system to resolve problems [14]. Redundant cases have a negative impact on the performance of the system, and noisy cases adversely affect the accuracy of the proposed solutions. Therefore, CBM algorithms usually try to remove both redundant cases and noisy cases.

There is a wealth of approaches to perform CBM in the literature [1,7,16,18–22]. The CNN algorithm only deletes redundant cases, focusing on retrieval efficiency [8]. The RNN algorithm extends CNN to consider noise cases as well [7]. The ENN algorithm only removes noisy cases, and RENN consists of multiple iterations of ENN, taking the output of one repetition as an input for the following iteration [21]. The family of algorithms DROP1, DROP2 and DROP3 were introduced to reduce redundancy and noisy cases [22]. The DROP family introduces the concept of *associate*, in an attempt to classify cases as redundant or noisy. COV-FP algorithms [20] and ICF [1] exploit the concepts *coverage* and *reachability* in order to reduce both the redundancy and noise levels [19].

S.J. Delany and S. Ontañón (Eds.): ICCBR 2013, LNAI 7969, pp. 218–232, 2013.

The main disadvantage of the aforementioned algorithms, with the exception of the COV-FP series, is their sensitivity to the order in which cases are examined. That is, given the same case-base, these CBM algorithms could provide different outcomes depending on the order of the cases in the case-base. Additionally, another commonplace feature of all these algorithms is their greedy approach to CBM goals, and the use of a lazy learning approach, such as k-nearest neighbour [3]. Furthermore, since each algorithm has a fixed deletion policy, the suitability of the algorithm to perform CBM is directly related to the redundancy and noise levels. For instance, those algorithms focused on case reduction underperform in cases-bases with few representative cases.

Consequently, CBM may be understood as a multi-objective optimization problem, minimising the case-base size and error rate at the same time. However, other objectives must be taken into consideration. In particular, it would be useful to estimate the optimum number of cases to resolve the entire problem domain [13], and to select a set of cases from the original case-base that maintains or improves accuracy.

In the last decades, Multi-Objective Evolutionary Algorithms (MOEAs) have been applied successfully in multi-objective optimization problems [2]. Therefore, CBM could be approached as a MOEA. To this end, considering that CBM algorithms should generate a case-base without redundant cases or noisy cases, and that is as small as possible to resolve the entire problem domain, three objectives based on Complexity Profiling [15] can be considered. Hence, MOEAs should get a good well-maintained case-base irrespective of the redundancy and noise levels of the original case-base.

In this work we propose to represent CBM as a 3-objective optimization problem, and we present a CBM algorithm based on MOEA using a novel fitness function.

The remainder of this work is as follows: in the next section we review the background of Complexity Profiling and the basic principles of MOEA. In section 3 we propose a fitness function for MOEA to perform CBM. In section 4, we evaluate the MOEA with different case-bases, and other CBM algorithms. Finally, in section 5 we present our conclusions and future work.

2 Background

2.1 Complexity Profiling

Massie *et al.* [16] introduced Complexity Profiling to estimate the proportion of redundant and noisy cases, as well as the existing error rate in the case-base. The foundation of this approach is a local complexity, which is an approximation to find the proportion of cases with the same solution in the nearest neighbour set of the case. Expression 1 describes the complexity function for a case:

$$complexity(c, k) = 1 - \frac{1}{k} \sum_{i=1}^{k} p(c, i), \tag{1}$$

where k is the number of nearest neighbours to consider and $p(c, i)$ is the proportion of cases within the case's i-nearest neighbours that belong to the same solution as c. The codomain for *complexity* function is $[0, 1]$. The more the *complexity* of a case is, the more likely the case would be noisy.

Complexity Profiling is a global measure of the case-base, and it is composed by three different indicators:

1. the *error rate* is the average of all the local complexities measures;
2. the *noise* is the proportion of all the complexity measures with values greater than ϵ; and
3. the *redundancy* is the proportion of all the complexity measures with values equal to ρ.

The error, noise and redundancy are defined formally as follow:

$$error(M, k) = \frac{1}{|M|} \sum_{c \in M} complexity(c, k). \tag{2}$$

$$noise(M, k) = \frac{|\{c \in M | complexity(c, k) \geq \epsilon\}|}{|M|}. \tag{3}$$

$$redundancy(M, k) = \frac{|\{c \in M | complexity(c, k) = \rho\}|}{|M|}, \tag{4}$$

where M is a case-base, $c \in M$ is a case within M, and k is the number of neighbours of c. Experiments with $\epsilon = 0.5$ and $\rho = 0$ confirm that Complexity Profiling is a good predictor of accuracy and noise [16].

2.2 Multi-Objective Evolutionary Algorithms

Evolutionary Algorithms (EAs) are inspired in biological evolution [9], since they simulate biological processes to search for a solution to an optimization problem. EAs represent the problem with a string of binary values. The string is known as an individual, and each of its binary values as genes. For each individual the EA applies a function known as the fitness function, which indicates the suitability of the individual to resolve the optimization problem. The search for the best individual is an iterative process. Starting with a set of individuals known as the population, an EA uses three operations on it to create the next generation of individuals: reproduction, crossover and mutation. The reproduction operation aims to select the better individuals according to their fitness values. Crossover is applied only to selected individuals to create new individuals, usually exchanging their genes. Mutation flips randomly the genes of the individual to increase the diversity of individuals. At the end of the iteration process, the individuals within the final population are potential solutions to the optimization problem. Hence, a strategy is needed to choose the final solution as well.

Multi-Objective Evolutionary Algorithm (MOEA) is an EA that searches for a solution to a problem according to two or more optimization objectives. Unlike

EA, MOEA fitness function returns a value per each objective [23]. Expression 5 defines formally the optimization problem to minimize n objectives:

$$minimize(\Phi(x)) = minimize(\phi_1(x), \phi_2(x), \ldots, \phi_n(x)), \tag{5}$$

where x is an individual, Φ is the fitness function, and each ϕ_n is the fitness function associated to an objective. Given the fitness values of two individuals, it is possible to define a relation of dominance between them [5]. This dominance determines which individual is closer to the optimization objectives. Expression 6 defines formally the relation:

$$\begin{aligned} x \prec y \iff & \\ \forall \phi_i(x), \phi_i(y) \in \Phi(x) : \phi_i(x) \leq \phi_i(y) \wedge & \\ \exists \phi_j(x), \phi_j(y) \in \Phi(x) : \phi_j(x) < \phi_j(y), & \end{aligned} \tag{6}$$

where x and y are the individuals, $x \prec y$ expresses that x dominates y, and n is the number of objectives. MOEA generates generations of individuals, where non dominated individuals have higher odds of survival.

3 Multi-objective Optimization Fitness Function for Case-Base Maintenance

So to perform CBM with a MOEA, we need to set up the representation of the problem. On the one hand, we need to represent a case-base as an individual of the population. On the other hand, we need to define a fitness function to evaluate the suitability of the individual.

3.1 Case-Base Representation

The case-base is a string of binary values that creates an individual. The length of the string is the cardinality of the case-base. That is, each gene (binary value) of the individual (string) represents the presence of the case in the case-base.

Let M be the original case-base, denoted by $M = \{c_1, c_2, \ldots, c_n\}$, where c_i the i-th case of M ($|M| = n$). The space of all possible individuals of M is denoted by X. An individual $x \in X$ is formally defined as $x = x_1 x_2 \ldots x_{n-1} x_n$, where x_i is the i-th gene of the individual with values of $x_i \in \{true, false\}$.

In order to map the cases from the original case-base (M) to the elements of the individual, we introduce the following function:

$$\begin{aligned} \mathcal{M} : & X \rightarrow \wp(M) \\ \mathcal{M}(x) = \overline{x} = & \{c_i \in M | x_i = true\}. \end{aligned} \tag{7}$$

For example, given the individual x with all elements set to true, $\mathcal{M}(x) = M$, otherwise if all elements are set to false then $\mathcal{M}(x) = \emptyset$. For the sake of clarity, we use the notation $\overline{x}$ as the case-base equivalent to the individual x.

3.2 Fitness Function to Perform CBM

We propose a fitness function based on Complexity Profiling to solve an optimization problem with three objectives:

1. to minimize the difference between the current number of cases in the solution and the estimated number of non redundant cases;
2. to minimize the number of redundant cases; and
3. to minimize the error rate level.

The first objective aims to estimate the minimum number of cases, the second is focused on avoiding case-bases with redundant cases, and the third leads the search to find a case-base with the minimum error rate. According to these objectives, the resulting case-base is expected to have smoother frontiers between the clusters of cases of different solutions, and with few cases within the clusters.

The formal description of the fitness function is shown as follows:

$$\begin{aligned} \Phi : X, \mathbb{N} &\rightarrow \mathbb{R}^3 \\ \Phi(x,k) &= (f_{size}(\overline{x},k), redundancy(\overline{x},k), error(\overline{x},k)). \end{aligned} \tag{8}$$

Note that the domain of the fitness function is an individual x, and a natural number k that sets the number of neighbours to consider in all the functions.

Function f_{size} is defined as follows:

$$\begin{aligned} f_{size} : X, \mathbb{N} &\rightarrow \mathbb{R} \\ f_{size}(x,k) &= ((|M| * (1 - redundancy(\overline{x},k)) - length(x))^2, \end{aligned} \tag{9}$$

where $length(x)$ is the number of elements of x set to true and $(|M| * (1 - redundancy(M,k)) - length(x)$ is the distance between the current number of cases in the solution and the estimated number of non redundant cases that the case-base should contain. This objective is squared to penalize those individuals with a greater number of cases.

The values returned by functions $redundancy(\overline{x},k)$ and $error(\overline{x},k)$ in the fitness function (expression 8) oppose each other since a lower error rate means a higher redundancy and vice versa.

3.3 NSGA-II

In this work we consider the well-known NSGA-II [5], a non-dominated sorting based MOEA. Given two individuals x and y representing two case-bases, and the fitness function $\Phi(x,k)$, the dominance relation for NSGA-II is defined as:

$$\begin{aligned} &(f_{size}(x) \leq f_{size}(y) \wedge redundancy(x) \leq redundancy(y) \wedge error(x) \leq error(y)) \wedge \\ &(f_{size}(x) < f_{size}(y) \vee redundancy(x) < redundancy(y) \vee error(x) < error(y)). \end{aligned} \tag{10}$$

For the sake of clarity we have omitted the parameter k of each function.

The main contributions of NSGA-II are a fast non-dominated sorting function and two operators to sort the individuals: a density estimation of the individuals in the population covering the same solution and a crowded comparison operator.

The *fast-nondominated-sort* algorithm details are shown in Alg.1. This function given a population P returns a list of the non-dominated fronts $\mathcal{F}$, where the individuals in front $\mathcal{F}_i$ dominates those individuals in front $\mathcal{F}_{i+1}$. That is, the first front contains the non-dominated individuals, the second front has those individuals dominated only once, the third contains individuals dominated up to twice, and so on. The individuals in the same front could have similar case-base representations; to avoid this situation NSGA-II uses the crowded comparison operator $\geq_n$, because individuals with lower density are preferred. To define formally the operator $\geq_n$, let x, y be two individuals, then $x \geq_n y$ if $(x_{rank} < y_{rank})$ or $((i_{rank} = j_{rank}) \wedge (i_{density} > j_{density})$, where x_{rank} represents the front where the individual belongs. The *crowding-distance-assignment* procedure calculates the density per each individual (Alg.2).

Parameters are set up at the beginning, such as the number of generations and number of individuals N for population. Each generation t implies an iteration of the algorithm, where two populations P_t and Q_t of N individuals are used. When NSGA-II starts, the initial population P_0 is generated randomly. Furthermore, binary tournament selection, recombination, and mutation operators are used with individuals from P_0 to create a child population Q_0. Once P_0 and Q_0 are initialized, NSGA-II runs its main loop, which we can see in Alg.3. In each iteration, population P_t and Q_t are joined to create the population R_t, whose number of individuals is $2N$. After that, the individuals in R_t are sorted according to their dominance and crowding distances. The sorted individuals are added to population P_{t+1}. At the end of each iteration P_{t+1} is truncated to N individuals, and Q_{t+1} is generated using binary tournament selection, recombination, and mutation operators.

Once NSGA-II finishes, the final population P_t will contain as much individuals as potential solutions, and the non-dominated individuals are mapped to their corresponding case-bases. The case-base with the minimum error rate is chosen as the solution of the CBM algorithm. If two or more case-bases have the same error rate, then the algorithm chooses the first case-base found.

For further details of NSGA-II algorithms see [5].

3.4 Interpreting the MOEA Approach

A MOEA using our proposed fitness function tends to search for the minimum error rate and to delete the maximum number of cases, without exceeding a threshold of number of non-redundant cases that corresponds to $|M| * (1 - redundancy(M, k))$ (expression 8). Figure 2 depicts the target cases-bases of the fitness function for Iris dataset. That is, case-bases with a lower number of cases and with a similar error rate to the original case-base. To build the figure, we have created 1000 case-bases selecting from 5 to 70 random cases from Iris. Therefore, we have 1000 case-bases of 5 cases, 1000 cases-bases of 6 cases, and

Alg.1 fast-nondominated-sort(P)

Require: A population P
Ensure: list of the non-dominated fronts $\mathcal{F}$

for $p \in P$ **do**
  for $q \in P$ **do**
    if $p \prec q$ **then**
      $S_p \leftarrow S_p \bigcup \{q\}$
    else
      if $q \prec p$ **then**
        $n_p \leftarrow n_p + 1$
      end if
    end if
  end for
  if $n_p = 0$ **then**
    $\mathcal{F}_1 \leftarrow \mathcal{F}_1 \bigcup \{p\}$
  end if
end for
$i = 1$
while $\mathcal{F}_i \neq \emptyset$ **do**
  $\mathcal{H} \leftarrow \emptyset$
  for $p \in \mathcal{F}_i$ **do**
    for $q \in S_p$ **do**
      $n_q \leftarrow n_q - 1$
      if $n_q = 0$ **then**
        $\mathcal{H} \leftarrow \mathcal{H} \bigcup \{q\}$
      end if
    end for
  end for
  $i = i + 1$
  $\mathcal{F}_i \leftarrow \mathcal{H}$
end while
return $\mathcal{F}$

Alg.2 crowding-distance-assignment($\mathcal{I}$)

Require: A set of individuals $\mathcal{I}$
Ensure: Each individual within $\mathcal{I}$ with a density measure.

$l \leftarrow |\mathcal{I}|$
for $i \in [1, N]$ **do**
  $\mathcal{I}[i] \leftarrow 0$
end for
for each objective m **do**
  $\mathcal{I} \leftarrow \text{sort}(\mathcal{I}, m)$
  $\mathcal{I}[1]_{\text{density}} \leftarrow \infty$
  $\mathcal{I}[l]_{\text{density}} \leftarrow \infty$
  for $i \in [2, (l-1)]$ **do**
    $\mathcal{I}[i]_{\text{density}} \leftarrow \mathcal{I}[i]_{\text{density}} + (\mathcal{I}[i+1].m - \mathcal{I}[i-1].m)$
  end for
end for

Alg.3 NSGA-II main loop

Require: A fitness function Φ.
Ensure: a population P_t of potential solution

$R_t \leftarrow P_t \bigcup Q_t$
$t \leftarrow 0$, $i \leftarrow 1$
$\mathcal{F} \leftarrow$ fast-nondominated-sort(P)
while $|P_{t+1}| < N$ **do**
  crowding-distance-assignment($\mathcal{F}_i$)
  $P_{t+1} \leftarrow P_{t+1} \bigcup \mathcal{F}_i$
  $i \leftarrow i + 1$
end while
sort($P_{t+1}, \geq_n$)
$P_{t+1} \leftarrow P_{t+1}[0 : N]$
$Q_{t+1} \leftarrow$ make-new-pop(P_{t+1})
$t \leftarrow t + 1$

Fig. 1. NSGA-II algorithm and main functions [5]

so on. Finally, a Hold-Out evaluation is used to measure the error rate of each case-base, using 60% of the cases as the training set and 40% as the test set. For each set of 1000 cases-bases the error rate given by the Hold-Out evaluation is averaged. The plot shows for each case-base size the average error rate, and the maximum and minimum observed values for each case-base size.

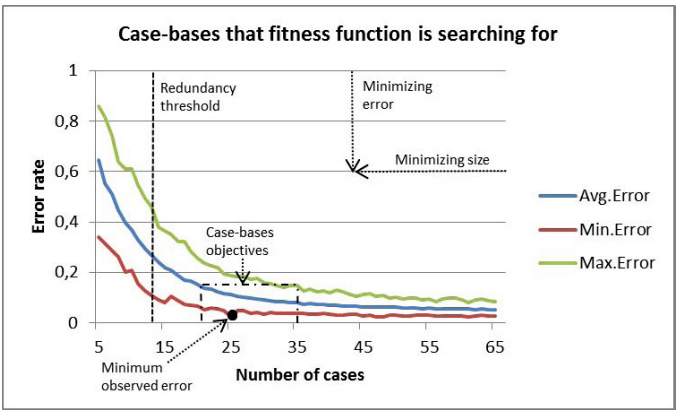

Fig. 2. Evolution of the error rate for Iris dataset. A lower number of cases is correlated with higher error rates. The fitness function goal is to find a case-base located in between the redundancy threshold and the minimum observed value for this experimentation.

4 Experimental Evaluation

4.1 Experiments and Results

We have considered two measurements to study the suitability of our fitness functions to perform CBM: the Reduction Rate, and the Competence Improvement [18]. The reason to use these two measures is because the suitability of CBM algorithms is strongly related to the number of cases deleted by the CBM process, and to the accuracy of the CBR system that will use the maintained case-base. The measurements are defined as follows:

1. The Reduction Rate is the average number of cases removed, that is:

$$reduction(M', M) = \frac{|M'|}{|M|}. \tag{11}$$

2. The Competence Improvement, which quantifies the proportional improvement in accuracy of the CBR system. Note that this error is not related to the Complexity Profiling error measure. This measure is formally defined as follows:

$$CI(M', M) = \frac{eval\_error(M)}{eval\_error(M')}, \tag{12}$$

where M is the initial case-base and M' is the case-base after the maintenance, with $M' \subseteq M$, and $eval\_error(M)$ is the proportion of times that the CBR system returns a wrong solution to the input problems using the Hold-Out approach described below. Values of $CI(M', M) > 1$ mean an improvement in accuracy, values $CI(M', M) < 1$ mean an underperformance, and otherwise it means no improvement at all. The CBR system is evaluated using a Hold-Out approach executed 10 times, as other authors suggest [4,17,18,20]. In particular, the Hold-Out is performed considering 30% of the cases as the training set. The retrieval of similar cases is performed using a k-NN approach. The value $k = 3$ is set for both the k-NN and the calculation of Complexity Profiling.

Table 1. Error rate (exp. 2), redundancy (exp. 4) and noise level (exp. 3) given by Complexity Profiling with $k = 3$. The values in bold represent the redundant and noise datasets, respectively.

	error rate	*redundancy*	*noise*
australian	0.277	**0.636**	0.284
contraceptive	0.716	0.133	**0.764**
diabetes	0.435	0.44	**0.451**
flags	0.6	0.289	**0.655**
glass	0.436	0.444	**0.453**
ionosphere	0.18	**0.772**	0.18
iris	0.064	**0.913**	0.06
liver-bupa	0.586	0.215	**0.597**
lymph	0.367	0.487	**0.392**
segment	0.056	**0.917**	0.058
sonar	0.218	**0.716**	0.245
vehicle	0.447	0.43	**0.468**
vowel	0.044	**0.9**	0.028
wine	0.069	**0.899**	0.056
zoo	0.083	**0.881**	0.089

In order to test the suitability of our proposal, we evaluate NSGA-II with our fitness function for CBM, using different standard datasets, and we do a comparative analysis considering some representative CBM algorithms from the literature. The results of each evaluation are the Reduction Rate and Competence Improvement measurements. In particular, our experiments consider:

– 15 datasets from the UCI repository [6]: australian, contraceptive, diabetes, flags, glass, ionosphere, iris, liver-bupa, lymph, segment, sonar, vehicle, vowel, wine and zoo. Each dataset has no missing values, and the nominal or string values in the datasets have been replaced by equivalent integer values. Finally, each record of the dataset is considered as a case, and the last attribute makes up the solution. Table 1 shows the levels of error rate, redundancy and noise given by Complexity Profiling using $k = 3$, $\epsilon = 0.5$ and $\rho = 0$ for expressions 2, 3 and 4. We consider a dataset as noisy where its noise level is higher than 0.4 and redundant when its redundancy level is higher than 0.5. Thus, there are eight redundant datasets and seven noisy datasets.

– 7 CBM algorithms: CNN, RNN, RENN, DROP1, DROP2, DROP3 and ICF.
– NSGA-II as MOEA using our fitness function. The number of individuals is 100, the number of generations 250, the mutation probability is 0.05 and the crossover probability is 0.9.

Figures 3, 4, 5 depict the result of Competence Improvement for each CBM algorithm and case-base. Each column represents the result CNN, RNN, RENN, DROP1, DROP2, DROP3 and ICF. NSGA-II returns a final population with individuals representing case-bases. In this experimentation we only consider three of them: the *minimum* as the solution case-base with the maximum error rate, the *maximum* as the case-base with the minimum error rate, and the case base with the minimum Complexity Profiling error, which is returned by NSGA-II. The lines in figures 3, 4, 5 represent the results of the maximum, minimum and the output case-bases obtained by NSGA-II using our fitness function.

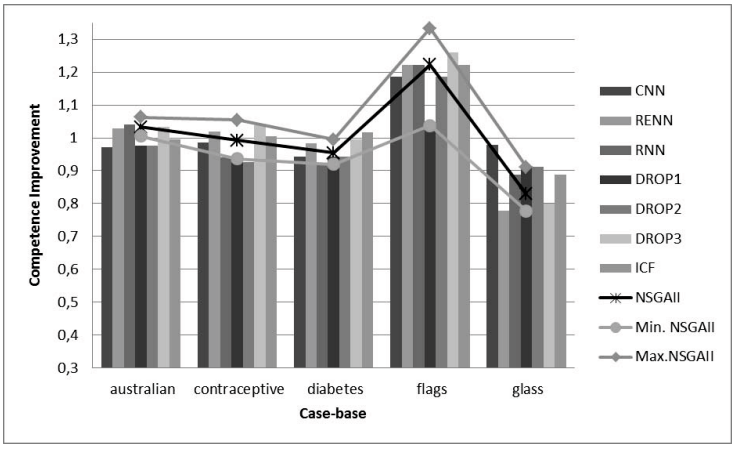

Fig. 3. Results for Competence Improvement for australian, contraceptive, diabetes, flags and glass case-bases for each CBM algorithm. Higher means better.

Table 2 shows the reduction rate results for each case-base and CBM algorithm. The highest reduction is highlighted in bold. For the sake of clarity, table 2 only shows the reduction given by the output case-base in NSGA-II, because maximum and minimum are very similar.

Concerning the duration of a CBM execution, we must take into consideration the size of the case-base, the number of features to describe the problem and solution, and the complexity to compute the similarity between cases. NSGA-II has the longest runtime among all the algorithms because other factors are implicated, such the crossover and mutation operator, the assigned probability to apply, and especially the amount of individuals in the population and the number of generations.

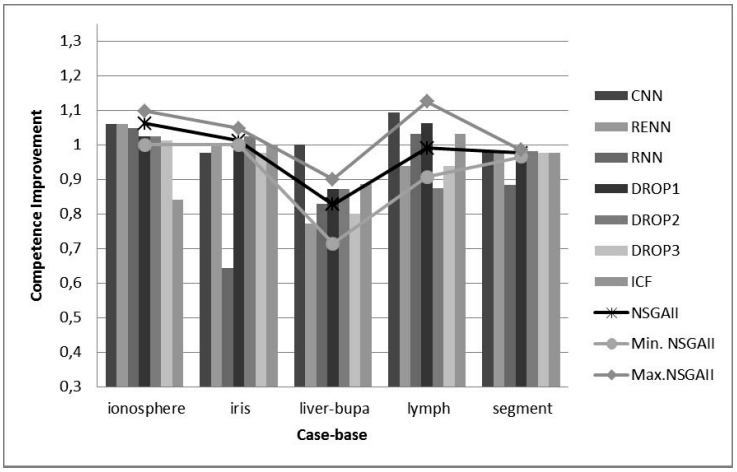

Fig. 4. Results for Competence Improvement for ionosphere, iris, liver-bupa, lymph and segment case-bases for each CBM algorithm. Higher means better.

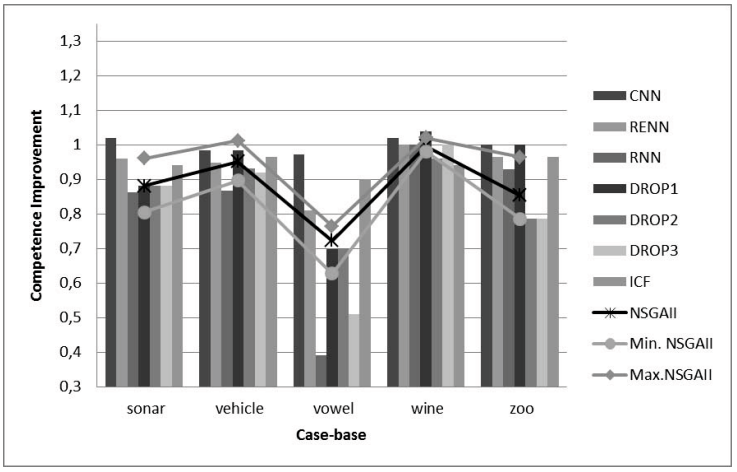

Fig. 5. Results for Competence Improvement for sonar, vehicle, vowel, wine and zoo case-bases for each CBM algorithm. Higher means better.

4.2 Discussion

According to the experiments, the case-base returned by the NSGA-II algorithm achieves the best improvements in competence (australian, ionosphere, and segment), or reaches a solution very close to the best observed among the other CBM algorithms. When the best case-bases of NSGA-II are considered, this algorithm achieves the best results in many of the datasets (australian, contraceptive, flags, ionosphere, iris, lymph, segment and vehicle). Additionally, the case-base with the worst Competence Improvement with NSGA-II only returns

Table 2. Reduction rate for each dataset and CBM algorithm

Datasets	CNN	RENN	RNN	DROP1	DROP2	DROP3	ICF	**NSGAII**
australian	0.595	0.162	0.598	0.5	0.5	0.581	0.415	**0.616**
contraceptive	0.246	0.552	0.266	0.5	0.5	**0.778**	0.686	0.47
diabetes	0.445	0.304	0.477	0.497	0.501	**0.652**	0.48	0.59
flags	0.259	0.533	0.37	0.496	0.504	**0.77**	0.637	0.705
glass	0.409	0.383	0.436	0.497	0.503	0.691	0.557	**0.713**
ionosphere	0.69	0.163	**0.763**	0.5	0.5	0.5834	0.425	0.698
iris	0.827	0.019	**0.856**	0.49	0.5	0.51	0.327	0.215
liver-bupa	0.271	0.408	0.383	0.5	0.5	**0.704**	0.588	0.642
lymph	0.534	0.223	0.612	0.495	0.505	0.612	0.466	**0.766**
segment	0.844	0.054	**0.881**	0.499	0.5	0.527	0.413	0.479
sonar	0.579	0.214	0.593	0.503	0.503	0.607	0.4	**0.651**
vehicle	0.413	0.325	0.455	0.499	0.501	**0.663**	0.498	0.588
vowel	0.471	0.108	**0.704**	0.499	0.5	0.555	0.327	0.372
wine	0.782	0.065	**0.815**	0.492	0.5	0.532	0.315	0.299
zoo	0.714	0.071	**0.771**	0.443	0.5	0.543	0.529	0.184
average	0.539	0.239	0.599	0.494	0.5012	0.621	0.471	0.533

the worst results in liver and sonar datasets among the CBM algorithm considered. In some datasets with high levels of redundancy, the best competence improvement results of NSGA-II are beaten by algorithms specialized in removing redundant cases, such as CNN. Similarity, NSGA-II is beaten in some noisy datasets by those algorithms specialized in deleting noisy cases, such as ICF in diabetes. However, overall the average Competence Improvement achieved by NSGA-II is consistent in all the experiments.

It is also worth mentioning that the worst case-base in the final population of NSGA-II are often very close to the competence improvement of the rest of the algorithms. However, our eager approach to choosing the final case-base seems insufficient for picking the best maintained case-base, suggesting that it is not enough to consider only the minimum error rate.

Figure 6 plots one point for each CBM algorithm, and each point corresponds to the Reduction Rates and Competence Improvement resulting from the average of all the Reduction Rates and Competence Improvement from the experiments. The figure depicts how difficult it is to achieve both great reductions and accuracy improvement at the same time, because a larger reduction results in worsening accuracy. The only exceptions to this tendency are CNN and the best results given by NSGA-II.

Finally, to identify whether a CBM algorithm deletes noisy or redundant cases, the Pearson product-moment correlation coefficient is computed between the error, redundancy and noise measure, which are returned by Complexity Profiling, and the accuracy and reduction rate given by the evaluation process. This correlation ranges from -1 to $+1$. Values in the interval $(-1, 0)$ indicates a negative correlation, values in the interval $(0, 1)$ note a positive correlation. That is, values close to -1 means the CBM algorithm does not delete that kind of cases, and values close to 1 point out that CBM deletes aggressively that type of case. Table 3 shows the coefficient values for each pair of results.

The NSGA-II correlation coefficients (table 3) indicate that the number of deleted cases is correlated with both noisy and redundancy levels. Thus, it seems

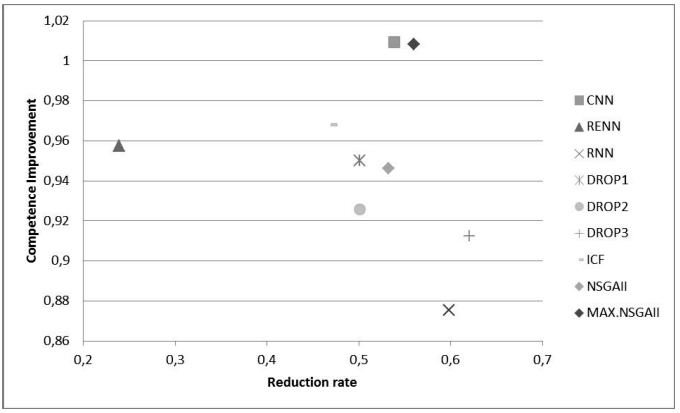

Fig. 6. Distribution resulting of averaging the datasets results for each CBM algorithm according to the reduction rate and competence improvement

that the fitness function aims the search of the maintained case-base deleting redundant cases and smoothing the frontiers between clusters of cases. Noisy cases are deleted more aggressively than the redundant cases though. In particular, NSGA-II achieves lower reduction rates in datasets with many redundant cases and few noise cases such as iris, vowel, wine and zoo. On the contrary, the reduction rate is greater in noisy datasets, such as contraceptive, diabetes, flags, glass, liver-bupa, lymph and vehicle. The coefficients also show that RENN, DROP3 and ICF are focused on deleting noisy cases, albeit DROP3 and ICF remove redundant cases as well. Moreover, CNN and RNN delete mainly redundant cases. DROP1 and DROP2 are focused on deleting cases near the borders. The rest of the CBM algorithms delete both types of cases equally.

Table 3. Correlation between Complexity Profiling error and accuracy, Complexity Profiling redundant level and reduction rate, and Complexity Profiling noise level and reduction rate

Pearson correlation	CNN	RENN	RNN	DROP1	DROP2	DROP3	ICF	NSGAII
Error & accuracy	-0,97	-0,91	-0,48	-0,86	-0,83	-0,64	-0,90	-0,85
Redundant & reduct. rate	0,91	-0,96	0,96	-0,33	-0,28	-0,95	-0,86	-0,60
Noise & reduction rate	-0,89	0,97	-0,96	0,31	0,30	0,97	0,88	0,59

5 Conclusions and Future Work

In this work we propose a multi-objective evolutionary approach to solve some tasks of Case-Base Maintenance. In particular, we present a novel fitness function based on Complexity Profiling [15]. We test the suitability of the approach on different datasets and compare the performance achieved to that of existing CBM algorithms from the literature.

Previous works are mainly focused on reducing either the number of redundant cases or noisy cases [1, 7, 8, 20–22], or aimed at selecting attributes [10, 12] or to both enhance the accuracy and reduce the size of the case-base [11]. However, the fitness function proposed in this work measures the redundancy of the case-base, the number of noisy cases and the error rate of the system. Therefore, this function aims to maintain the case-base following three objectives. The experiments show that the fitness function aims the search of the maintained case-base, to those case-bases with less redundant cases and smoother frontiers between clusters of cases.

The results obtained in the experiments show that the evolutionary approach outperforms general CBM approaches in many datasets, obtaining promising results even in worst cases. However, in our opinion, the most remarkable result of our proposal is the regularity of the behaviour with most datasets.

The runtime could be a limitation, in particular where CBM can not be performed off-line and the CBR system is stopped until the CBM process finishes. For this reason MOEA are not suitable in all scenarios. Nevertheless, using MOEA could be suitable when the case-base is built for the first time from a raw set of data, and where time is not the most important restriction. In this scenario, selection of an individual case-base from the final population could be done through an evaluation process using Cross-Validation or Hold-Out.

The use of genetic operators is limited in this work. Therefore, the next step will focus on the definition of specific crossover and mutation operators based on coverage and reachability.

Acknowledgements. This work was partially funded by the Seneca Research Foundation of the Region of Murcia under project 15277/PI/10, and by the Spanish Ministry of Science and Innovation+European FEDER+PlanE funds under the project TIN2009-14372-C03-01.

References

1. Brighton, H., Mellish, C.: On the consistency of information filters for lazy learning algorithms. In: Żytkow, J.M., Rauch, J. (eds.) PKDD 1999. LNCS (LNAI), vol. 1704, pp. 283–288. Springer, Heidelberg (1999)
2. Coello, C.C., Lamont, G., van Veldhuizen, D.: Evolutionary Algorithms for Solving Multi-Objective Problems. Genetic and Evolutionary Computation (2007)
3. Cover, T., Hart, P.: Nearest neighbor pattern classification. IEEE Transactions on Information Theory 13(1), 21–27 (1967)
4. Cummins, L., Bridge, D.: Maintenance by a committee of experts: The MACE approach to case-base maintenance. In: McGinty, L., Wilson, D.C. (eds.) ICCBR 2009. LNCS, vol. 5650, pp. 120–134. Springer, Heidelberg (2009)
5. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2) (2002)
6. Frank, A., Asuncion, A.: UCI machine learning repository (2010), `http://archive.ics.uci.edu/ml`

7. Gates, G.: Reduced nearest neighbor rule. IEEE Transactions on Information Theory 18(3), 431 (1972)
8. Hart, P.: Condensed nearest neighbor rule. IEEE Transactions on Information Theory 14(3), 515+ (1968)
9. Holland, J.H.: Adaptation in Natural And Artificial Systems. MIT Press (1975)
10. Ishibuchi, H., Nakashima, T., Nii, M.: Genetic-algorithm-based instance and feature selection. In: Frasson, C., McCalla, G.I., Gauthier, G. (eds.) ITS 1992. LNCS, vol. 608, pp. 95–112. Springer, Heidelberg (1992)
11. Cano, J.R., Herrera, F., Lozano, M.: Evolutionary stratified training set selection for extracting classification rules with trade off precision-interpretability. Data & Knowledge Engineering 60(1), 90–108 (2007)
12. Kim, K., Han, I.: Maintaining case-based reasoning systems using a genetic algorithms approach. Expert Systems with Applications 21(3), 139–145 (2001)
13. Leake, D., Wilson, M.: How many cases do you need? Assessing and predicting case-base coverage. In: Ram, A., Wiratunga, N. (eds.) ICCBR 2011. LNCS, vol. 6880, pp. 92–106. Springer, Heidelberg (2011)
14. Leake, D.B., Wilson, D.C.: Categorizing case-base maintenance: Dimensions and directions. In: Smyth, B., Cunningham, P. (eds.) EWCBR 1998. LNCS (LNAI), vol. 1488, pp. 196–207. Springer, Heidelberg (1998)
15. Massie, S., Craw, S., Wiratunga, N.: Complexity-guided case discovery for case based reasoning. In: 20th National Conference on Artificial Intelligence, AAAI 2005, vol. 1, pp. 216–221 (2005)
16. Massie, S., Craw, S., Wiratunga, N.: Complexity profiling for informed case-base editing. In: Roth-Berghofer, T.R., Göker, M.H., Güvenir, H.A. (eds.) ECCBR 2006. LNCS (LNAI), vol. 4106, pp. 325–339. Springer, Heidelberg (2006)
17. McKenna, E., Smyth, B.: Competence-guided case-base editing techniques. In: Blanzieri, E., Portinale, L. (eds.) EWCBR 2000. LNCS (LNAI), vol. 1898, pp. 186–197. Springer, Heidelberg (2000)
18. Pan, R., Yang, Q., Pan, S.: Mining competent case bases for case-based reasoning. Artificial Intelligence 171(16-17), 1039–1068 (2007)
19. Smyth, B., Keane, M.: Remembering to forget - a competence-preserving case deletion policy for case-based reasoning systems. In: International Joint Conference on Artificial Intelligence, IJCAI 1995, pp. 377–382 (1995)
20. Smyth, B., McKenna, E.: Competence guided incremental footprint-based retrieval. Knowledge-Based Systems 14(3-4), 155–161 (2001)
21. Wilson, D.: Asymptotic properties of nearest neighbor rules using edited data. IEEE Transactions on Systems Man and Cybernetics SMC 2(3), 408 (1972)
22. Wilson, D., Martinez, T.: Reduction techniques for instance-based learning algorithms. Machine Learning 38(3), 257–286 (2000)
23. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. IEEE Transactions on Evolutionary Computation 3(4), 257–271 (1999)