Evolving Balanced Decision Trees with a Multi-Population Genetic Algorithm

Vili Podgorelec, Sašo Karakatič
University of Maribor
FERI, Institute of Informatics
SI-2000 Maribor, Slovenia
{vili.podgorelec, saso.karakatic}@um.si

Rodrigo C. Barros
Pontifícia Universidade Católica do Rio Grande do Sul
Faculdade de Informática
Av. Ipiranga, 6681, 90619-900, Porto Alegre, RS, Brazil
rodrigo.barros@pucrs.br

Márcio P. Basgalupp
Universidade Federal de São Paulo
Instituto de Ciencia e Tecnologia
Av. Cesare Mansueto Giulio Lattes, 1021, 12247-280, São José dos Campos, SP, Brazil
basgalupp@unifesp.br

***Abstract*—Multi-population genetic algorithms have been used with success for several multi-objective optimization problems. In this paper, we present a new general multi-population genetic algorithm for evolving decision trees. It was designed to improve the possibility of evolving balanced decision trees, simultaneously optimized for the predictions of each class. Single-population genetic algorithms namely tend to construct decision trees with great variance in single class accuracies. The proposed approach is tested over 10 UCI datasets, and it is compared with a single-population genetic algorithm as well as with traditional decision-tree induction algorithms. Results show that the designed multi-population approach provides classification results comparable to C4.5 and CART in terms of accuracy and tree size, while outperforming them regarding balanced solutions (in terms of average class accuracy and range of single-class accuracies).**

Keywords—genetic algorithms; machine learning; classification; decision trees; multi-population genetic algorithm

I. Introduction

Decision trees (DT) are one of the most widely used data classification method. They are reliable and provide classification performance comparable with other classification approaches. DTs are especially popular because of their simple, transparent and straightforward representation, resulting in a classification process that one can easily interpret, understand, validate and learn from. Indeed, DTs are directed acyclic graphs that can be easily read as a disjunction of conjunctions in the form of `if-then` classification rules. DTs are the preferred model in application domains in which understanding the reasons that lead to a certain prediction is as important as the prediction itself (e.g., medical diagnosis [1] and protein function prediction [2]).

Most DT induction algorithms employ the top-down greedy strategy for building the trees (e.g., C4.5 [3] and CART [4]). Nonetheless, the top-down approach is prone to falling into local-optima since it locally optimizes a criterion for each node of the tree in a recursive fashion. For addressing this issue, researchers turned to evolutionary algorithms (EAs) as a means to avoid local-optima by performing a robust global search on the space of candidate DTs, usually achieving enhanced predictive performance at the expense of increased computational cost [5, 6]. As EAs evolve solutions in accordance with the given fitness function, they can generally optimize solutions regarding a single criterion, which is usually the predictive performance (such as accuracy or F-measure). In the case of DTs, however, there are several other criteria which could (or should) also be optimized, like complexity (DT size) and attributes cost, just to name a few. To address this issue, researchers usually use advanced fitness calculation methods, like the weighted sum of various measures [7] or the lexicographic approach [8] in order to obtain better DTs.

EAs have been extensively applied to supervised learning in the past decades, specially for the problem of data classification. Classification can be regarded as an optimization problem, where the goal is to find a function f that better approximates the unknown true function f responsible for mapping attribute values from the input space into discrete categories (decision classes), $f: X \rightarrow Y$. It has been already demonstrated in many applications that EAs can improve the classification performance of DTs [5].

In this paper, we address the challenge of improving the balance of evolutionary induced DTs by using a multi-

This work was produced in part within the framework of the operation entitled "Centre of Open innovation and ResEarch UM". The operation is co-funded by the European Regional Development Fund and conducted within the framework of the Operational Programme for Strengthening Regional Development Potentials for the period 2007 – 2013, development priority 1: "Competitiveness of companies and research excellence", priority axis 1.1: "Encouraging competitive potential of enterprises and research excellence". The authors would also like to thank *Coordenacão de Aperfeiçoamento de Pessoal de Nível Superior* (CAPES), *Conselho Nacional de Desenvolvimento Científico e Tecnológico* (CNPq) and Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) for funding this research.

978-1-4799-7492-4/15/$31.00 ©2015 IEEE

population genetic algorithm (MPGA). In our previous work, we have designed an ad hoc MPGA for prediction of churning telecommunication customers [9]. Encouraged by the results, here we generalize the approach by designing a new MPGA for DT induction. It consists of two co-evolving populations of DTs, where each population is being evolved regarding a different criterion, and the information exchange between two populations provides the means to generate one single balanced solution.

The remaining of this paper is organized as follows. Section 2 presents an overview of multi-population genetic algorithms, and introduces our novel approach for evolving balanced decision trees. Section 3 details the methodology employed for performing the experimental analysis, whose results are presented in Section 4. Finally, Section 5 concludes the paper with our final thoughts and lists some future work possibilities.

II. Multi-Population Genetic Algorithm for Inducing Decision-Trees

A. Multi-Population Genetic Algorithms

Genetic algorithms (GAs), one of the most important representatives of EAs, are capable of exploring a wide range of search space when the selection pressure is properly controlled, while crossover and mutation evolve solutions towards local optima, keeping the needed genetic diversity. The evolutionary search for the solution is directed towards the optimal solution based on a predefined fitness function.

In this paper, the optimal solution is the best DT for a given dataset. However, the evaluation of a DT is generally a multi-objective problem (MOP), since different criteria – for instance, the overall accuracy, F-measure, average class accuracy, and size – influence the quality of a DT. Usually, the fitness function in such cases is defined as a weighed sum of all of the above single objectives. The problem with the weighted formula approach is that the improvement of one single parameter generally leads to the worsening of others. As the solution in the evolutionary search needs to build up before it can reach an adequate level of quality, the problem of such aggregated multi-objective fitness functions is that the solution becomes biased towards only a few objectives.

To overcome this problem, the evolutionary search should have the capability of preserving the building blocks needed to optimize the solution towards several objectives. In other words, the occurrence of premature convergence should be avoided. Various techniques such as the objective aggregation technique, the objective alternate technique, and the Pareto-based technique have been proposed to address this problem, all with their own disadvantages [10]. One way to overcome some issues of alternate objectives and standard aggregation is by using a multi-objective approach, where several solutions are considered as best (not being dominated by others), each one dominating others in (at least) one of the objectives, forming the Pareto front [11]. However, when using the multi-objective EAs to solve MOP, the problem on how to select good individuals for the next generation arises [10]. Since a MOP has multiple objectives that often contradict with each other, it is difficult to say whether one individual is better than another if it is better at one objective but it is worse at another objective [12].

Another way to address this problem is to improve the fine-tuning capability of a simple GA, for which MPGAs have been developed in many applications [13], including data mining [14]. MPGA is an extension of a simple GA by dividing a population into several isolated subpopulations within which the evolution proceeds and individuals are allowed to migrate from one subpopulation to another. In recent years, MPGAs have been recognized as being more effective both in speed and solution quality than single-population GAs [14].

B. GA for Decision-Tree Induction

For the induction of balanced DTs, we propose in this paper a new general MPGA that extends the *genTrees* algorithm [15, 6]. But first, let us summarize the basic details of *genTrees* (GT).

GT represents individual DTs as binary tree data structures, where each node has five values (Fig. 1). `Node type` can be either class or test node. Depending on the `node type`, `index` represents either a decision class number (in this case further values are ignored) or a test attribute index. In case of continuous test nodes, `split value` represents a splitting threshold *th*: instances with values lower than *th* proceed to the `left child` and the rest proceed to the `right child`. In case of discrete test nodes, only the integer part of `split value` int(*th*) is used as an index into the set of discrete values of the corresponding attribute: instances with values equal to int(*th*) proceed to the `left child` whereas instances which values differ from int(*th*) proceed to the `right child` until a class (leaf) node is reached.

For the selection the binary tournament is used. For crossover, a randomly chosen training instance is first used to determine a path (from root to a class node) in both selected DTs. Then one attribute node is randomly chosen on defined paths in both selected DTs. Finally, the two corresponding sub-trees from the chosen attribute nodes are exchanged. Mutation can change one of five possible things in a randomly chosen node within a DT: class number, attribute index, split value, attribute node into class node (consecutively deleting both sub-trees), or class node into attribute node (consecutively constructing both sub-trees). Fitness function will be discussed later.

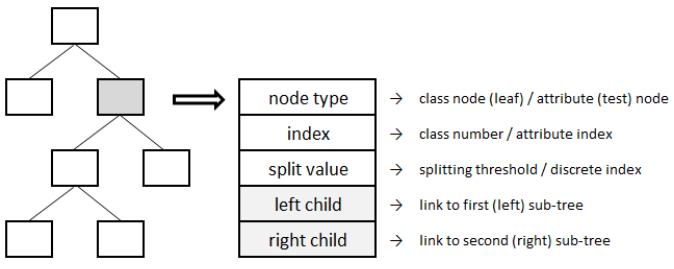


Fig. 1. A binary tree-based representation of a DT; a node is a quintuple of values: node type (class/test attribute), index, split value, and links to both child nodes (when node is a test attribute).

C. MPGA for Decision-Tree Induction

The proposed MPGA for the induction of balanced DTs, hereby called MPGT (multi-population *genTrees*), consists of two co-evolving subpopulations which employ the same initialization, tournament selection, crossover and mutation operators. Each subpopulation, however, has a different fitness function, which optimizes a different objective. In a regular cycle, after a predefined number of generations (migrate interval) the exchange of DTs between the two subpopulations occurs according to a predefined parameter (migrate rate). MPGT is outlined in Fig. 2.

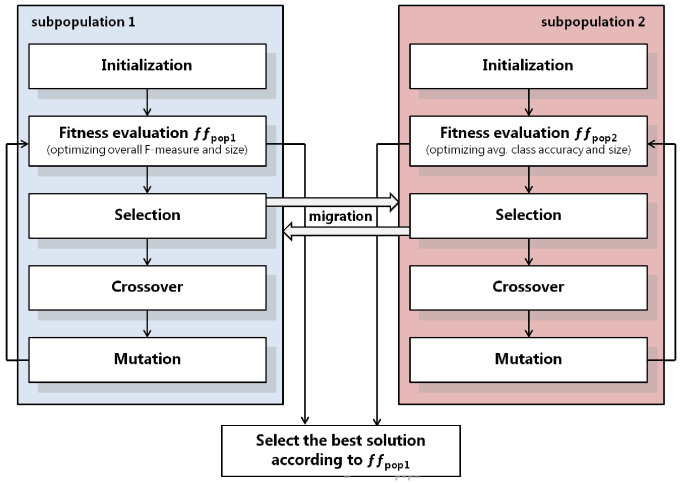


Fig. 2. General outline of the multi-population genetic decision-tree induction algorithm MPGT.

From the viewpoint of a single subpopulation, the migration of DTs is the main difference to a standard single-population GA for DT induction. For implementing the migration process, we used the best-worst migration policy [14]. A set of best-fit DTs from one subpopulation is selected (using the normal regular selection), which then replaces a set of worst-fit DTs from the other subpopulation (using the inverse selection). The literature is not consistent in defining parameters such as migrate rate and migrate interval. Whereas some researchers perform migration in every generation [11], others prefer to evolve subpopulations for a certain amount of generations before migration reoccurs [16]. Since we wanted some robust setting for all the experiments, we set and fixed the migrate interval to 100 generations and migrate rate to 1/8 of the population size, which were used throughout experiments as default values.

The use of two different fitness functions serves as a means to generate balanced DTs. It is our goal to build accurate DTs where all single-class accuracies are balanced. Thus, the first subpopulation will optimize solutions regarding the overall predictive performance, while the second will optimize solutions regarding the balanced single-class accuracies. As the evolution tends to produce introns (unreachable nodes because of contradictory test conditions), the same penalty for larger trees is added in both cases. Since the F-Measure is regarded as a more suitable performance prediction measure than accuracy [17], the fitness function for the first subpopulation is defined as:

$$ff_{pop_1} = (1 - fsc) + \frac{1}{s} \cdot \frac{n}{sf} \quad (1)$$

where S is the total number of instances, n is the number of nodes in the tree, sf is a size factor that defines how many additional nodes outweigh one misclassified instance (we set it to 10), and fm is the F-Measure criterion (harmonic mean of the precision and recall values). The fitness function for the second subpopulation replaces the F-Measure by the weighted single-class accuracy, though keeping the same size penalty, as shown in (2).

$$ff_{pop_2} = (1 - \sum_{i=1}^{K} w_i \cdot acc_i) + \frac{1}{s} \cdot \frac{n}{sf} \quad (2)$$

In (2), K is the number of decision classes, acc_i is the accuracy of the i^{th} class, and w_i is the weight associated to the i^{th} class. By setting different weights, the search can be driven towards the desired outcome. Fig. 3 represents three possible scenarios (defined by different weight values and represented by different lines) of choosing different solutions (represented by triangle markers) as optimal for a 2-class classification problem. In general, if a balanced solution is sought, in which each class is equally important, the class accuracy weights can be set to $w_i=1/K$ (represented by solid line in Fig. 3), transforming (2) into (3), where there is no need for choosing weights.

$$ff_{pop_2} = \left(1 - \frac{1}{K} \sum_{i=1}^{K} acc_i\right) + \frac{1}{s} \cdot \frac{n}{sf} \quad (3)$$

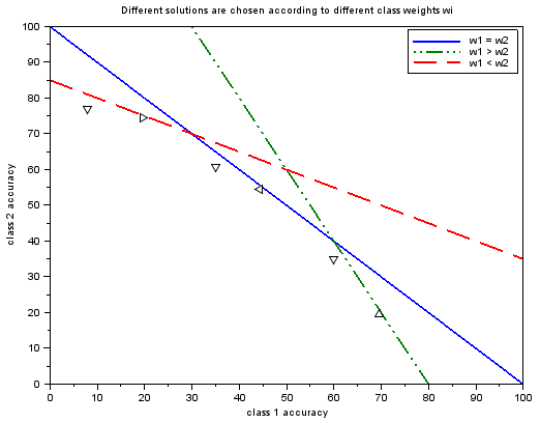


Fig. 3. By setting different weights w_i different solutions from the Pareto front can be chosen as the best ones (example for a dataset with two classes). Trinagle markers represent several found solutions (a marker nearest to a single line represents best found solution according to the criteria).

By employing the described procedure, the balanced DTs from the second subpopulation will regularly feed the first subpopulation (and vice-versa) with building blocks needed to optimize the global solution, which could have dropped out during the evolution of a single subpopulation towards one direction within the search space.

Fig. 4 shows an example of the evolution of all the individuals within MPGT (several stages of a single evolutionary run for the 2-class *churn* dataset problem [18] at generations 0, 50, 150, 250, 500, and 550.). It can be seen how different subpopulations optimize solutions towards different directions. When each single subpopulation begins to converge, the migration introduces new genetic material, thus preventing the premature convergence and influencing the remaining of the evolutionary cycle. It can be seen how solutions converge until generation 500 and then, after information is exchanged between the two populations, diverge again in the next 50 generations.

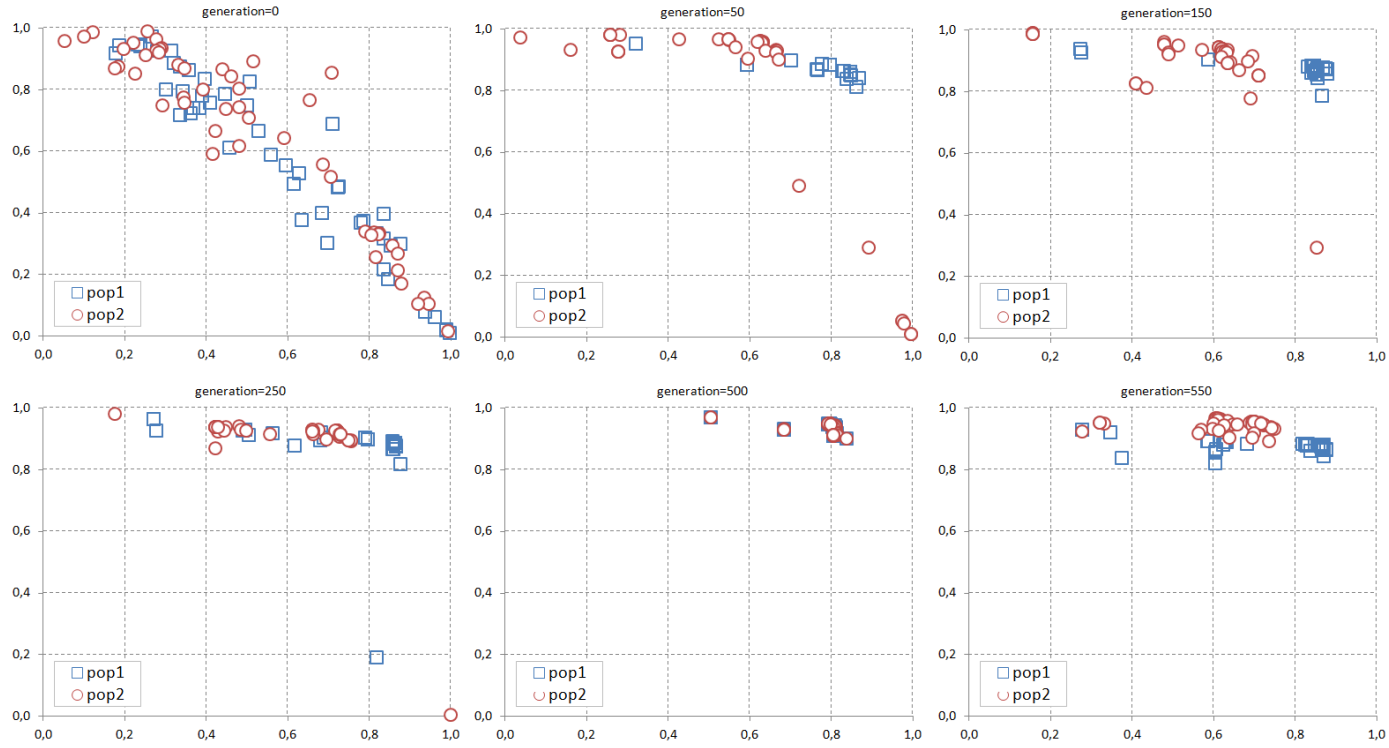
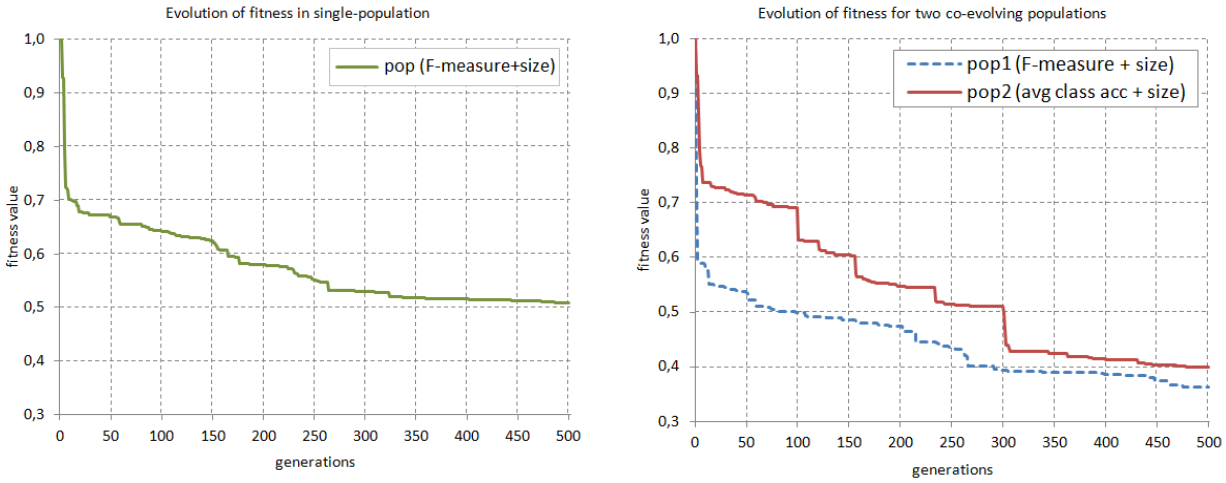


Fig. 4. The co-evolution of two populations of decision trees for the *churn* dataset; the x and y axis represent class-1 and class-2 acuracy, respectively (note how solutions converge until generation 500 and then, after information is exchanged between the two populations, diverge again in the next 50 generations).



a) single-population

b) two co-evolving populations

Fig. 5. Comparison of fitness evolution for the churn dataset for a) single-population GT, and b) multi-population GT with two co-evolving populations; the fitness improvements after information exchange in generations 100 and 300 can be easily seen (note that the same amount of computer processing is used per generation in both cases as the single-population GT contains twice the amount of individuals as one subpopulation in the multi-population GT).

The influence of migration between subpopulations can be easily seen when comparing the evolution of a single-population GT (Fig. 5a) with a MPGT (Fig. 5b). It drives the individuals to possibly improve after each migration; the improvement of fitness is most obvious at generations 100 and 300 (migration interval is set to 100 generations).

III. Experimental framework and settings

To assess the performance of the MPGT algorithm, we performed an experiment over 10 well-known public datasets from the UCI machine learning repository [19], presented in Table 1. First, we compared the resulting DTs with two variants of a single-population GA: one that uses the overall F-measure as a fitness function, and another that uses the averaged class accuracy as a fitness function (all class accuracy weights are set equally as w_i=1/*K*). We used our *genTrees* [6, 15] algorithm (GT) as the baseline single-population EA for DT induction. In a second moment, we compared MPGT with the two most widely used traditional top-down DT induction algorithms: C4.5 [3] and CART [4], both with their default settings.

TABLE I. UCI DATASETS USED IN THE EXPERIMENT.

	# classes	*# attributes*	*# instances*	*imbalance ratio*
breast_cancer	2	9	286	2.36
car	4	6	1,728	11.90
colic	2	22	368	1.71
eye_movements	3	27	10,936	2.81
glass	7	9	214	8.73
heart_statlog	2	13	270	1.25
iris	3	4	150	2.06
segment	7	19	2,310	1.34
sick	2	29	3,772	15.33
sonar	2	60	208	1.14

Adopting the 5-fold cross-validation, all datasets were pre-divided into five subsets in such a manner that preserved very similar (practically equal) class distribution. Each subset has been used as a test set once, while the other four subsets combined constituted a training set. In this manner, all the classification algorithms were operating on exactly the same dataset divisions. Each method was tested on each fold (a pre-divided training/test set combination) on all the datasets. All results are based on the average of all five folds, whereas the results of all the evolutionary methods are additionally averaged from 10 evolutionary runs per fold (giving 50 evolutionary runs per dataset).

We used the same settings of genetic parameters for all the evolutionary runs: binary tournament selection, 100% probability of crossover, 10% probability of mutation, 2000 generations, only the best fit solution as the elite, information exchange between populations after every 100 generations. The population size was 500 individuals in GT and 250 in MPGT – as there are two co-evolving populations in MPGT, the same amount of processing was used in both cases to ensure a fair comparison. These settings can be regarded as default for MPGT and were fixed for all the experiments, as we wanted some robust setting (for the same reason, we are also using default parameters for C4.5 and CART).

To provide valid conclusions regarding the performance of the algorithms that were used in the experiments, statistical tests were applied by following the approach proposed by Demšar [20]. These tests seek to compare multiple algorithms on multiple datasets, and are based on the use of the Friedman test with the corresponding Nemenyi post-hoc test.

Let R_i^j be the rank of the j^{th} of k algorithms on the i^{th} of N datasets, the Friedman test compares the average ranks of algorithms, $R_j = \frac{1}{N}\sum_i R_i^j$. The original Friedman statistic and its adjusted less-conservative version are given by:

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right] \quad (4)$$

$$F_f = \frac{(N-1)\times\chi_F^2}{N\times(k-1)-\chi_F^2} \quad (5)$$

where the adjusted version is distributed according to the *F*-distribution with *k*-1 and *(k*-1*)(N*-1*)* degrees of freedom.

If the null hypothesis of similar performance is rejected, then we proceed with the Nemenyi post-hoc test for pairwise comparisons. The performance of two classifiers is significantly different if their corresponding average ranks differ by at least the critical difference

$$CD = q_\alpha\sqrt{\frac{k(k+1)}{6N}} \quad (6)$$

where critical values q_α are based on the Studentized range statistic divided by $\sqrt{2}$.

IV. Results

A. Comparing the evolutionary methods

First, we compared the classification performance results of MPGT with two single-population variants of GT, each having its fitness function equal to one subpopulation of MPGT: in GT1 the solutions were optimized for F-measure and tree size (1), while in GT2 the solutions were optimized regarding the averaged class accuracy and tree size (3). Results on the test sets are presented in Table 2. For each dataset, we report the following data: classification accuracy (acc), F-measure (fm), averaged class accuracy (aca), range of single class accuracies (rca), and the size of the generated tree (size). The range of single class accuracies is the difference between the class with the highest accuracy and the class with lowest accuracy – the lower range is better as it means more balanced predictions. The best obtained results for each measure and each dataset are written in bold.

TABLE II. RESULTS FOR THE COMPARISON BETWEEN MPGT AND BOTH SINGLE-POPULATION GT VARIANTS.

	MPGT					**GT1 (F-measure + size)**					**GT2 (avg. class acc + size)**				
	acc	***fm***	***aca***	***rca***	***size***	***acc***	***fm***	***aca***	***rca***	***size***	***acc***	***fm***	***aca***	***rca***	***size***
breast_cancer	.726	**.727**	**.669**	.264	32.6	**.727**	.698	.606	.565	**10.0**	.688	.698	.657	**.141**	12.2
car	**.945**	**.946**	**.886**	**.199**	50.0	.844	.821	.519	.922	**11.0**	.841	.852	.870	.280	26.2
colic	.843	**.843**	**.836**	**.103**	17.0	**.844**	.841	.820	.203	**7.0**	.836	.835	.820	.140	8.6
eye_movements	**.557**	**.556**	**.557**	**.082**	323.0	.487	.483	.490	.205	**8.2**	.473	.427	.502	.500	10.2
glass	**.760**	**.762**	**.724**	**.589**	64.3	.693	.655	.455	.904	**13.0**	.595	.576	.574	1.00	16.3
heart_statlog	**.862**	**.859**	**.850**	.182	33.8	.858	.855	.846	.188	**13.8**	.858	.856	.848	**.158**	14.6
iris	.947	.947	.944	.117	**6.4**	.951	.951	.951	**.094**	7.0	**.962**	**.961**	**.959**	.104	8.6
segment	**.958**	**.958**	**.961**	**.092**	65.0	.763	.761	.777	.523	23.7	.798	.789	.808	.588	**20.3**
sick	**.982**	**.983**	**.968**	**.032**	20.4	.933	.900	.500	1.00	**3.0**	.950	.956	.965	.033	8.6
sonar	**.787**	**.787**	**.788**	**.073**	40.6	.774	.771	.778	.210	**17.8**	.749	.745	.750	.198	20.6

Results show that MPGT evolved better DTs in terms of predictive accuracy in 7 out of 10 datasets, GT1 in two datasets and GT2 in only one dataset. In terms of F-measure and averaged class accuracy, MPGT evolved better solutions in 9 out of 10 datasets, and GT2 in one dataset. Regarding class accuracy variance (expressed through the range of class accuracies), MPGT evolved better trees in 7 out of 10 datasets, GT1 in one and GT2 in two datasets. When analyzing the tree sizes, we can observe that the smallest trees are evolved by GT1 in 8 out of 10 datasets, whereas both MPGT and GT2 evolved the smallest tree in only one dataset.

To evaluate the statistical significance of these results, we first applied the Friedman test by calculating the average Friedman ranks, χ_F^2, F_f, and Friedman asymptotic significance for MPGT, GT1, and GT2, for all 5 measures (acc, fm, aca, rca, and size). The statistical results are summarized in Table 3. The critical values here are: asymptotic significance α=0.05, χ_F^2=5.99 for α=0.05, and $F(k\text{-}1,(k\text{-}1)(n\text{-}1)) = F(2,18)$=3.55 also for α=0.05. We can see that there is a significant difference among the methods for all five measures.

TABLE III. THE STATISTICAL ANALYSIS RESULTS FOR THE COMPARISON BETWEEN MPGT, GT1, AND GT2

	acc	***fm***	***aca***	***rca***	***size***
χ_F^2	6.2	9.6	12.359	6.2	12.8
F_f	4.043	8.308	14.557	4.043	16.000
asymp. sig.	**0.045**	**0.008**	**0.002**	**0.045**	**0.002**

As the null hypothesis of similar performance is rejected, we proceeded with the Nemenyi post-hoc test for pairwise comparisons. Since we have three methods and 10 datasets, the critical difference is *CD*=0.74 (for standard α=0.05). If the average ranks of two methods differ by at least *CD*, they can be considered significantly different (the better being the method with the lower average rank). The results of the Nemenyi test are presented in Table 4: a positive difference means a better result for the first method in the pairwise comparison. All the significant differences are written in bold. Notice that MPGT outperformed GT1 in all measures but tree size; MPGT outperformed also GT2 in terms of accuracy, F-measure, and averaged class accuracy. Both GT1 and GT2 outperformed MPGT in terms of tree size. GT1 outperformed GT2 in terms of tree size, whereas GT2 outperformed GT1 in terms of averaged class accuracy.

TABLE IV. THE DIFFERENCE BETWEEN AVERAGE RANKS (THE NEMENYI TEST) FOR MPGT, GT1, AND GT2 (CRITICAL DIFFERENCE IS CD = 0.74)

	acc	***fm***	***aca***	***rca***	***size***
MPGT vs. GT1	**+0.7**	**+1.2**	**+1.6**	**+1.1**	**-1.6**
MPGT vs. GT2	**+1.1**	**+1.2**	**+0.8**	+0.7	**-0.8**
GT1 vs. GT2	+0.4	0	**-0.8**	-0.4	**+0.8**

B. Comparing MPGT with C4.5 and CART

The next step was to compare MPGT with two traditional greedy top-down DT induction algorithms: C4.5 and CART. Results of performance over the test sets are presented in Table 5. They show that MPGT evolved better DTs in terms of predictive performance (both the accuracy and F-measure) in 7 out of 10 datasets, C4.5 in one dataset and CART in two datasets. In terms of averaged class accuracy, MPGT evolved better solutions in 7 out of 10 datasets, C4.5 in two datasets and CART in one dataset. In terms of range of class accuracies, MPGT evolved better trees in 8 out of 10 datasets, whereas both C4.5 and CART evolved better trees in only one dataset. When analyzing the size of trees, we can observe that the smallest trees are evolved by CART in 6 out of 10 datasets, while MPGT evolved the smallest tree in the remaining 4 datasets.

The statistical analysis is summarized in Table 6. We can see that there is a significant difference among the three methods in terms of range of class accuracies and tree size. As the null hypothesis of similar performance is rejected in terms of the range of class accuracies and tree size, we proceeded with the Nemenyi post-hoc test for pairwise comparisons. Even though the Friedman test did not indicate a significant difference among the methods regarding the remaining measures (accuracy, F-measure, and averaged class accuracy), we can still proceed to the pairwise Nemenyi test, which is known to be less conservative than Friedman's.

TABLE V. RESULTS FOR THE COMPARISON AMONG MPGT, C4.5, AND CART.

	MPGT					C4.5					CART				
	acc	***fm***	***aca***	***rca***	***size***	***acc***	***fm***	***aca***	***rca***	***size***	***acc***	***fm***	***aca***	***rca***	***size***
breast_cancer	**.726**	**.727**	**.669**	**.264**	32.6	.726	.664	.553	.799	11.8	.720	.675	.571	.692	**6.2**
car	.945	.946	.886	.199	**50.0**	.895	.895	.770	.443	165.0	**.973**	**.973**	**.935**	**.149**	101.0
colic	**.843**	**.843**	**.836**	**.103**	17.0	.806	.805	.790	.143	10.0	.827	.825	.805	.185	**5.8**
eye_movements	.557	.556	.557	**.082**	**323.0**	**.660**	**.660**	**.666**	.103	2399.4	.589	.589	.595	.097	709.8
glass	**.760**	**.762**	**.724**	**.589**	64.3	.722	.714	.700	.822	42.3	.742	.725	.672	.878	**16.3**
heart_statlog	**.862**	**.859**	**.850**	.182	33.8	.823	.820	.812	**.178**	31.0	.821	.817	.807	.232	**14.6**
iris	**.947**	**.947**	**.944**	**.117**	6.4	.907	.907	.903	.184	7.8	.907	.907	.903	.184	**6.2**
segment	**.958**	**.958**	**.961**	**.092**	**65.0**	.954	.954	.958	.145	76.3	.955	.955	.958	.132	75.7
sick	.982	.983	.968	**.032**	**20.4**	.993	.993	**.969**	.055	47.6	**.993**	**.993**	.959	.080	44.2
sonar	**.787**	**.787**	**.788**	**.073**	40.6	.684	.678	.683	.213	23.4	.647	.630	.649	.405	**5.8**

The results of the Nemenyi post-hoc test for pairwise comparisons between MPGT, C4.5 and CART are presented in Table 7 (the critical difference is CD=0.74). Results show that MPGT outperformed C4.5 in all measures but tree size (note though that the Friedman test indicated a significant difference only in terms of range of class accuracies); MPGT outperformed CART in terms of averaged class accuracy and range of class accuracies (the latter being indicated as significantly different also by the Friedman test). The only significant difference between C4.5 and CART is in terms of tree size, in which CART outperformed C4.5.

TABLE VI. STATISTICAL ANALYSIS RESULTS FOR THE COMPARISON AMONG MPGT, C4.5, AND CART.

	acc	***fm***	***aca***	***rca***	***size***
χ_F^2	4.051	4.667	5.59	9.897	6.2
F_f	2.286	2.739	3.491	8.816	4.043
asymp. sig.	0.132	0.097	0.061	**0.007**	**0.045**

TABLE VII. THE DIFFERENCE BETWEEN AVERAGE RANKS (THE NEMENYI TEST) FOR MPGT, C4.5, AND CART (CRITICAL DIFFERENCE IS CD = 0.74)

	acc	***fm***	***aca***	***rca***	***size***
MPGT vs. C4.5	**+0.85**	**+0.95**	**+0.85**	**+1.15**	+0.4
MPGT vs. CART	+0.65	+0.55	**+0.95**	**+1.25**	-0.7
C4.5 vs. CART	-0.2	-0.4	+0.1	+0.1	**-1.1**

V. CONCLUSIONS

This work presented a new general multi-population GA for evolving DTs called MPGT. It is a multi-population extension of a standard single-population GA that achieved promising results in generating DTs with good predictive performance and low complexity.

First, we compared MPGT with two variants of single-population GA: GT1, which optimizes the overall predictive performance in terms of F-measure; and GT2, which optimizes the averaged class accuracy. Results show that MPGT is capable of inducing DTs with significantly better predictive performance than GT1 and GT2, and which are also more balanced regarding the single-class accuracies. On the other hand, MPGT produced on average somewhat larger trees. In a second moment, we compared MPGT with the two most traditional greedy top-down DT induction algorithms: C4.5 and CART. Results show that MPGT is capable of inducing significantly more balanced DTs with competitive predictive performance and size. The less conservative post-hoc statistical test suggests the advantage of MPGT over C4.5 in all measures but tree size, and also the advantage of MPGT over CART in averaged class accuracy and balanced single-class accuracies. In terms of size, there were no significant differences between MPGT and the two baselines.

As future work, we intend to extend our research towards employing different general multi-objective optimization techniques, such as NSGA-II [12] and also more recent co-evolutionary techniques, such as multiple populations for multiple objectives [10]. Additionally, we believe that the multi-population approach may also leverage our work on evolving DT induction algorithms with multi-objective hyper-heuristics [17, 21, 22].

REFERENCES

[1] V. Podgorelec, P. Kokol, P., B. Stiglic, and I. Rozman, "Decision Trees: An Overview and Their Use in Medicine," Journal of Medical Systems vol. 26, pp. 445–463, 2002.

[2] R. Cerri, R.C. Barros, and A.P.L.F. de Carvalho, "Hierarchical multi-label classification using local neural networks," Journal of Computer and System Sciences, vol. 80, pp. 39-56, 2014.

[3] J.R. Quinlan, C4.5: programs for machine learning. San Francisco, CA: Morgan Kaufmann, 1993.

[4] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone, Classification and Regression Trees. Wadsworth, 1984.

[5] R.C. Barros, M.P. Basgalupp, A.C.P.L.F. de Carvalho, and A.A. Freitas, "A Survey of Evolutionary Algorithms for Decision-Tree Induction," IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, vol. 42(3), pp. 291–312, 2012.

[6] V. Podgorelec, M. Sprogar, and S. Pohorec, "Evolutionary design of decision trees," WIREs Data Mining and Knowledge Discovery, vol. 3(2), pp. 63–82, 2013.

[7] A. Papagelis and D. Kalles, "Breeding decision trees using evolutionary techniques," In Proceedings of the 18th International Conference on Machine Learning, pp. 393–400, 2001.

[8] M.P. Basgalupp, A.C. de Carvalho, R.C. Barros, and D.D. Ruiz, “Lexicographic multi-objective evolutionary induction of decision trees,” International Journal of Bio-Inspired Computation, vol. 1(1), pp. 105–117, 2009.

[9] V. Podgorelec and S. Karakatic, “A multi-population genetic algorithm for inducing balanced decision trees on telecommunications churn data,” Electronics and Electrical Engineering, vol. 19(6), pp. 121-124, 2013.

[10] Z.-H. Zhan, J. Li, J. Cao, J. Zhang, H.S.-H. Chung, and Y.-H. Shi, “Multiple Populations for Multiple Objectives: A Coevolutionary Technique for Solving Multiobjective Optimization Problems,” IEEE Transactions on Cybernetics, vol. 43(2), pp. 445–463, 2013.

[11] C.A.C. Coello, G.B. Lamont, and D.A.V. Veldhuizen, “Evolutionary Algorithms for Solving Multi-Objective Problems,” Genetic and Evolutionary Computation. New York: Springer-Verlag, 2006.

[12] K. Deb, S. Agrawal, A. Pratap, and T. Meyariva, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” IEEE Trans. Evol. Comput., vol. 6(2), pp. 182–197, 2002.

[13] W.-Y. Lin, T.-P. Hong, and S.-M. Liu, “On adapting migration parameters for multi-population genetic algorithms,” In The IEEE International Conference on Systems, Man, and Cybernetics, pp. 5731–5735, 2004.

[14] H. Zhu, J. Licheng, and P. Jin, “Multi-population genetic algorithm for feature selection.” In 2nd International Conference on Natural Computation, pp. 480–487, 2006.

[15] V. Podgorelec and P. Kokol, “Evolutionary induced decision trees for dangerous software modules prediction,” Information Processing Letters, vol. 82(1), pp. 31–38, 2002.

[16] P.-C. Chang, S.-H. Chen, and K.-L. Lin, “Two-phase sub population genetic algorithm for parallel machine-scheduling problem,” Expert Systems with Applications, vol. 29, pp. 705–712, 2005.

[17] R.C. Barros, M.P. Basgalupp, A.A. Freitas, and A.C.P.L.F. de Carvalho, “Evolutionary design of decision.tree algorithms tailored to microarray gene expression data sets,” IEEE Transactions on Evolutionary Computation, vol. 18(6), pp. 873-892, 2014.

[18] D.T. Larose, Discovering Knowledge in Data: An Introduction to Data Mining. Wiley, 2005.

[19] K. Bache and M. Lichman, UCI Machine Learning Repository. University of California, School of Information and Computer Science, `http://archive.ics.uci.edu/ml`

[20] J. Demšar, “Statistical Comparisons of Classifiers over Multiple Data Sets,” J. Mach. Learn. Res., vol. 7, pp. 1–30, 2006

[21] R.C. Barros, M.P. Basgalupp, A.C.P.L.F. de Carvalho, and A.A. Freitas, “A hyper-heuristic evolutionary algorithm for automatically designing decision-tree algorithms,” In 14th Genetic and Evolutionary Computation Conference GECCO-2012, pp. 1237-1244, 2012.

[22] R.C. Barros, M.P. Basgalupp, A.C.P.L.F. de Carvalho, and A.A. Freitas, “Automatic design of decision-tree algorithms with evolutionary algorithms,” Evolutionary Computation, vol. 21(4), 2013.