Mixed Decision Trees: Minimizing Knowledge Representation Bias in LCS

Xavier Llorà[1] and Stewart W. Wilson[2]

[1] Illinois Genetic Algorithms Laboratory (IlliGAL),
National Center for Supercomputing Applications,
University of Illinois at Urbana-Champaign, Urbana, IL 61801.
xllora@illigal.ge.uiuc.edu

[2] Prediction Dynamics, Concord, MA,
Department of General Engineering,
University of Illinois at Urbana-Champaign, Urbana, IL 61801.
wilson@prediction-dynamics.com

Abstract. Learning classifier systems tend to inherit—*a priori*—a given knowledge representation language for expressing the concepts to learn. Hence, even before getting started, this choice biases what can be learned, becoming critical for some real-world applications like data mining. However, such bias may be minimized by hybridizing different knowledge representations via evolutionary mixing. This paper presents a first attempt to produce an evolutionary framework that evolves mixed decision trees of heterogeneous knowledge representations.

1 Introduction

The genetic algorithms (GAs) community, as well as researchers coming from the genetic programming (GP) field and learning classifier systems (LCS) practitioners, have been using evolutionary algorithms for building (or inducing depending on the authors' emphasis) decision trees. Initially, researchers used GAs and GP for building decision trees by means of optimizing the classification error of a forest of decision trees [1,2,3]. Later on, researchers took this effort even further building frameworks that challenge the traditional methods for decision tree induction [4,5,6,7,8]. Other researchers introduced GAs for improving the efficiency of traditional tree builders [9].

Nevertheless, the majority, with a few exceptions that emphasize the relevance of the knowledge representation language [6], ignored the relevance of the knowledge representation in biasing the concepts that can be learned. Most of this research focused on evolving decision trees based on axis-parallel—let's call them orthogonal—classification boundaries. A few of them worked with oblique decision trees trying to enrich the set of concepts that can be learned, using GAs to beat the computational complexity of hyperplane split computation [9]—an NP-Hard problem. However, all these approaches evolve homogeneous decision trees, in which all the individuals of the population are encoded using the same knowledge representation language.

Evolutionary computation is a powerful tool, not only for addressing NP problems like the one mentioned above, but also for adapting the knowledge representation to the specific problem to be solved. Given this potential, we decided to investigate an evolutionary *mixing* of different kinds of trees–or different knowledge representations. This approach, besides using evolutionary algorithms to tackle NP problems, should also minimize the bias introduced by the knowledge representation, producing what we call *mixed decision trees.*

Section 2 briefly introduces an evolutionary LCS framework used for evolving decision trees. This framework already evolves different kinds of homogeneous decision trees, as shown in section 3. In this paper we took this framework one step further allowing it to produce mixed trees. Section 4 reviews the experimentation conducted using mixed trees, as well as summarizes the results obtained. Finally, section 5 presents some conclusions.

2 GALE

This section briefly reviews an algorithm of the so-called *Pittsburgh* approach to LCS. GALE (*Genetic and Artificial Life Environment*) is a fine grain parallel model for knowledge discovery that integrates elements of the *Pittsburgh* approach and cellular automata. The rest of this section presents an overview of the key elements of GALE—a detailed description can be found in [10].

2.1 Topology and Knowledge Representation

GALE uses a 2D grid (board $\mathcal{T}$) formed by $m \times n$ cells for spreading the evolving population spatially. Each cell ($\mathcal{T}_{ij}$) of the grid contains either one ($\zeta\left(\mathcal{T}_{ij}\right) = 1$) or zero individuals ($\zeta\left(\mathcal{T}_{ij}\right) = 0$); thus, for instance, a 32×32 grid can contain up to 1024 individuals, each one placed on a different cell. Each individual ($\mathcal{T}_{ij}^{I}$) is a complete solution to the classification problem; in fact, each individual codifies the knowledge that describes the mined data set. Genetic operators are restricted to the immediate neighborhood ($\mathcal{T}_{ij}^{\nu}$) of the cell in the grid. The size of the neighborhood is r. Given a cell $\mathcal{T}_{ij}$ and $r = 1$, the neighborhood $\mathcal{T}_{ij}^{\nu}$ of $\mathcal{T}_{ij}$ is defined by the 8 adjacent cells to $\mathcal{T}_{ij}$ (being $\zeta\left(\mathcal{T}_{ij}^{\nu}\right)$ the number of occupied cells in $\mathcal{T}_{ij}^{\nu}$). Thus, r is the number of hops that defines the neighborhood, and p_ζ is the probability that a cell $\mathcal{T}_{ij}$ contains an individual after initialization.

The evolutionary model of GALE coevolves different knowledge representations. GALE can perform homogeneous runs where only one type of knowledge representation is used [11,10], or heterogeneous runs where different knowledge representations compete in the same board [12]. Coevolving individuals expressed using different knowledge representations in the same board $\mathcal{T}$, helps GALE minimizing the bias introduced by the knowledge representation [13]. However, this approach is based on restricted mating policies, not allowing the mixing of different knowledge representation.

GALE mainly deals with three different knowledge representations [10]: (1) sets of fully-defined instances—or prototypes—[14,15], (2) orthogonal (or axis-parallel) decision trees [16], (3) oblique decision trees [17,18], and (4) rules sets.

GALE can perform homogeneous [14,11] and heterogeneous runs [12]. However, even in the heterogeneous runs, the individual maintain their initial knowledge representation. That means that no mixing of different knowledge representations takes place in the original GALE approach.

2.2 Mapping

GALE works as a supervised learning model. Hence, one key point is how the training instances Σ are spread over the board $\mathcal{T}$. In other words, each individual $\mathcal{T}_{ij}^{I}$ is evaluated using a set of examples mapped to the cell $\mu(\mathcal{T}_{ij}, \Sigma)$. The fitness function used is $fit(I) = \left(\frac{I^c}{l}\right)^2$ [19], being I^c the number of correctly classified instances and l the number of instances of the $\mu(\mathcal{T}_{ij}, \Sigma)$ data set.

The easiest mapping approach is to allocate all the training instances in each cell of the board ($\mu_u(\mathcal{T}_{ij}, \Sigma) = \Sigma$), or uniform mapping. Hence, all the cells in the board $\mathcal{T}$ contain the same training set. Therefore, the same environmental conditions are obtained in each cell. However, a mapping can perform non-uniform instance allocation. The pyramidal mapping $\mu_p(\mathcal{T}_{ij}, \Sigma)$ spreads the examples using a pyramid shape. Central cells contain all the available instances in Σ. The rest of the cells contains only a subset of Σ. Hence, the pyramidal mapping does not guarantee the same environment for each cell $\mathcal{T}_{ij}$. Instead, it guarantees that just few changes are introduced in adjacent cells of the neighborhood $\mathcal{T}_{ij}^{\nu}$. This means that few instances are removed when moving toward the cells in the outer part of the board $\mathcal{T}$. Under $\mu(\mathcal{T}_{ij}, \Sigma)$ assumptions, the classification complexity should grow when moving toward the inner cells of the board $\mathcal{T}$. Please refer to [10] for further details.

```
GALE($\mathcal{T}$,$\Sigma$)
   FOR-EACH $\mathcal{T}_{ij} \in \mathcal{T}$
   DO IN PARALLEL
      t ← 0
      initialize $\mathcal{T}_{ij}$
      evaluate the accuracy of individual in $\mathcal{T}_{ij}$ using $\mu(\mathcal{T}_{ij}, \Sigma)$
      REPEAT
            t ← t+1
            merge individual in $\mathcal{T}_{ij}$ among $\mathcal{T}_{ij}^{\nu}$
            split individual in $\mathcal{T}_{ij}$ among $\mathcal{T}_{ij}^{\nu}$
            evaluate the accuracy of individual in $\mathcal{T}_{ij}$ using $\mu(\mathcal{T}_{ij}, \Sigma)$
            survival of $\mathcal{T}_{ij}$ among $\mathcal{T}_{ij}^{\nu}$
      UNTIL $\Omega(\mathcal{T}_{ij}, t)$
   DONE
RETURN $\mathcal{T}$
```

Fig. 1. Pseudo-code of the algorithm running in each of the cells that compose GALE.

2.3 Control Flow

The evolutionary model proposed by GALE is based on the local interactions among cells. Each cell $\mathcal{T}_{ij}$ can only interact with the cells in the neighborhood $\mathcal{T}_{ij}^{\nu}$. Figure 1 presents the evolutionary process performed in each cell of GALE.

After the initialization phase, briefly explained in section 2.1, *merge* in GALE crosses the individual in the cell $\mathcal{T}_{ij}^{I}$ with one individual $I_{ij}^{\mathcal{M}}$ randomly chosen among its neighborhood $\mathcal{T}_{ij}^{\nu}$. The *merge* process occurs with a given probability $p_{\mathcal{M}}$. *Merge* generates only one individual $\mathcal{D}$ that replaces the individual in the cell $\mathcal{T}_{ij}$. The operator used for the genetic material recombination depends on the knowledge representation used for encoding $\mathcal{T}_{ij}^{I}$. Figure 2 shows this process schematically.

Then, *split* is applied with a given probability $p_s(\mathcal{T}_{ij}^{I}) = k_{sp} \cdot fit(\mathcal{T}_{ij}^{I})$, being $k_{sp} \in [0,1]$ the maximum splitting rate. *Split* clones and mutates the individual in the cell. The new individual is placed in the empty cell $\mathcal{T}_{kl}$ of the neighborhood $\mathcal{T}_{kl} \in \mathcal{T}_{ij}^{\nu}$ with higher number of occupied cells in its neighborhood—$\max\left(\zeta\left(\mathcal{T}_{kl}^{\nu}\right)\right)$. If all the cells of the neighborhood $\mathcal{T}_{ij}$ are occupied $\left(\zeta\left(\mathcal{T}_{ij}^{\nu}\right) = 8\right)$, the new individual is placed in the cell of the neighborhood $\mathcal{T}_{ij}^{\nu}$ that contains the worst individual—lowest fitness. Figure 3 illustrates this process graphically.

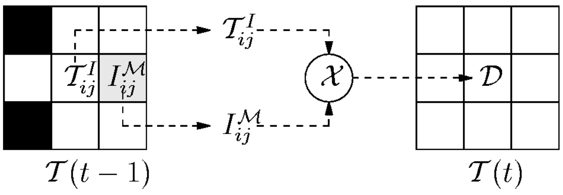
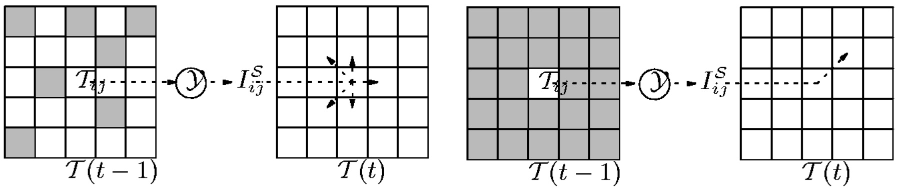


Fig. 2. Schematic representation of the merging process implemented by GALE.

The last step in the evolutionary cycle—*survival*—decides if the $\mathcal{T}_{ij}^{I}$ individual is kept for the next cycle or not. This process uses the neighborhood information. If a cell has up to one neighbor $(\zeta(\mathcal{T}_{ij}^{\nu}) \leq 1)$, then the probability of survival of the individual is $p_{sr}^{\zeta(\mathcal{T}_{ij}^{\nu})\leq 1}(\mathcal{T}_{ij}) = fit(\mathcal{T}_{ij}^{I})$. If a cell has seven or eight neighbors $\zeta(\mathcal{T}_{ij}^{\nu}) \geq 7$ then $p_{sr}^{\zeta(\mathcal{T}_{ij}^{\nu})\geq 7}(\mathcal{T}_{ij}) = 0$, where the individual is replaced by the best neighbor in $\mathcal{T}_{ij}^{\nu}$. On the other neighborhood configurations $(1 < \zeta(\mathcal{T}_{ij}^{\nu}) < 7)$, an individual survives if and only if $fit(\mathcal{T}_{ij}^{I}) \geq \overline{\mu}_{nei}^{\nu} + k_{sr} \times \sigma_{nei}^{\nu}$; $\overline{\mu}_{nei}^{\nu}$ is the average fitness value of the occupied neighbor cells $\mathcal{T}_{ij}^{\nu}$, and σ_{nei}^{ν} their standard deviation. k_{sr} is a parameter that controls the survival pressure over the current cell. For further details about GALE model, please see [6,14,11,10,12].

3 Heterogeneous Knowledge Representations

This section describes a first approach to break the constraint that individuals only use one kind knowledge representation. In order to test the feasibility of



(a) The new individual is placed in the empty cell $\mathcal{T}_{kl}$ of the neighborhood $\mathcal{T}_{kl} \in \mathcal{T}_{ij}^{\nu}$ with higher number of occupied cells in its neighborhood—$\max\left(\zeta\left(\mathcal{T}_{kl}^{\nu}\right)\right)$.

(b) If all the cells in the neighborhood $\mathcal{T}_{ij}^{\nu}$ are occupied, then the new individual I_{ij}^{S} replaces the worst neighbor in $\mathcal{T}_{ij}^{\nu}$.

Fig. 3. The location of the new splitted individual is decided based on the current cell $\mathcal{T}_{ij}$ neighborhood $\mathcal{T}_{ij}^{\nu}$.

allowing the evolutionary mixing of knowledge representations, we focused on tree based representations for GALE. Our goal was to allow individuals to represent a solution using different types of knowledge representation for evolutionary tuning. The rest of this section briefly reviews the available tree representations in GALE, as well as the modifications introduced to allow the evolution of mixed trees.

3.1 Previous Homogeneous Tree Representations

GALE originally evolved three different tree knowledge representations [11,10, 12]. The simplest type of decision tree evolved by GALE is the one termed *orthogonal* (or axis-parallel) [16,20]. The internal nodes are a simple test over one attribute of the classification problem. This test is usually presented as:

$$a_i \leq \alpha \tag{1}$$

where a_i is a problem attribute and α is a numeric constant. The leaves of the tree are labeled with the class of the set of instances represented by the path between the root of the tree and the leaf itself. The classification boundaries defined by equation 1 are parallel to the axis of the instance space. This kind of tree has been effectively built, in the machine learning community, using heuristic algorithms based, for instance, on the *information gain* concept [16,20].

In order to overcome the limitations of parallel-axis boundaries, some authors proposed a more elaborate test for the internal nodes of the decision trees. *Oblique* decision trees [17,18] define the internal nodes using the following equation:

$$\sum_{i=1}^{d} \omega_i a_i + \omega_{d+1} > 0 \tag{2}$$

This test is based on an orientable hyperplane that can be adjusted via the vector of coefficients defined by $\omega = \langle \omega_1, \omega_2 \ldots \omega_{d+1} \rangle$. Thus, non parallel-axis boundaries can be easily defined choosing the right values for the ω vector. Unfortunately, finding w^* (the optimal data set split) is *NP-Hard* [21]. Therefore, the algorithms used for building this kind of tree use several heuristics to obtain suboptimal trees [18].

The last kind of decision tree evolved by GALE is a *multivariate* decision trees. This kind of tree defines non-linear boundaries on the instance space. In order to achieve this goal, each internal node contains a prototype δ [22,23] (e.g. an artificially defined instance) and an activation threshold θ. A node is *active* if the following equation is satisfied

$$\sqrt{\sum_{i=1}^{d} (\delta_i - e_i)^2} \leq \theta \tag{3}$$

where e_i is the value of a_i in the instance e to classify. This selective activation defines hyperspheric boundaries across the instance space. Next, if the node is *active*, the instances e is given to the *nearest child* using the *nearest neighbor* algorithm [24]. Thus, children define non parallel-axis hyperplane splits inside the parent hyperspheres. Detailed descriptions of these decision trees can be found in [10].

3.2 Heterogeneous Tree Representations

The work presented in this paper explored the combination of different knowledge representations in a single individual. We focused on mixing two different kinds of decision trees (orthogonal and obliques.) This combination can be easily achieved due to the structural and functional similarity of the test nodes. For achieving such a goal, GALE needs to be slightly modified in two different places: (1) the initialization phase, (2) the *merge* phase.

The initialization phase, for a given cell $\mathcal{T}_{ij}$, now produces an orthogonal and oblique homogeneous tree. This choice is done at random, ideally filling the population with half orthogonal and half oblique homogeneous trees. The *merge* phase cuts and exchanges subtrees of two parent trees, regardless of whether they are orthogonal or oblique. Thus, *merge* mixes the different tree representations. Later on, these trees are evaluated using the original *survival* phase of GALE. It is important to mention here that the classification performed by such heterogeneous trees maintains the same hierarchical process. At any given node, the classification process relies on its type, performing orthogonal- (equation 1) or oblique-based (equation 2) classification accordingly.

4 Experiments

We conducted two different kinds of experiments for testing the mixed trees presented in the previous section. The first set of experiments used artificially-generated data sets. The goal was to test the implementation of orthogonal, oblique, and mixed trees. These experiments also let us perform a first comparison between GALE and traditional orthogonal and oblique decision tree inducers [17,25,26]. The second set of experiments focused on studying the behavior of GALE and other tree learners mainly on data sets provided by the UCI repository [27].

4.1 Classifier Schemes

Besides GALE, tree other non-evolutionary decision tree inducers were tested:

- **C4.5 revision 8** [16,20]
- **CART-LC** [17]
- **OC1** [26]

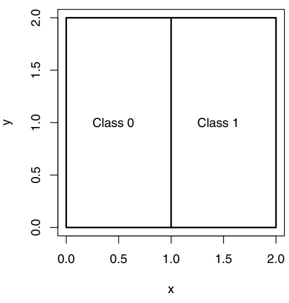
A detailed description of these algorithms is beyond the scope of this paper. However, it is important to mention here that C4.5 is a well-know inducer of orthogonal decision trees. CART-LC is an oblique decision tree inducer, as is OC1. We also tested OC1 in its orthogonal inducer facet.

4.2 Artificially-Generated Data Sets

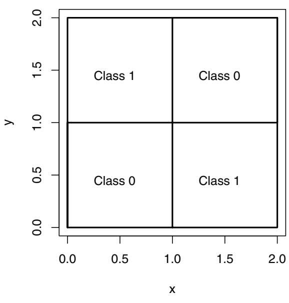
We prepared different data sets for this first test. All the artificially-generated data sets are defined on a bidimensional space for binary classification problems. Figure 4 displays each of these data sets. The data set instances were obtained sampling the given classification space using an uniformly distributed rectangular grid. Three axis-parallel classification data sets (`ORT-1`, `ORT-2`, and `ORT-3`), three oblique classification data sets (`OBL-1`, `OBL-2`, and `OBL-3`), and one mixed data set (`MIXED`) were generated. `ORT-1`, `ORT-2`, `OBL-1`, and `OBL-2` contain 400 instances each, whereas `ORT-3`, `OBL-3`, and `MIXED` the number of artificial instances raises till 1600.

The ultimate goal of the data sets presented in figure 4 was to test the performance of GALE on problems designed by axis-parallel, oblique, and mixed classification boundaries. Such tests used the three different tree knowledge representation presented in section 3.2. In order to compare the quality of the results produced by GALE, the outputted tree was compared to the one produce by OC1. For `ORT-1`, `ORT-2`, and `ORT-3` GALE evolved homogeneous orthogonal trees, and OC1 was set to induce orthogonal decision trees. In a similar way, for `OBL-1`, `OBL-2`, and `OBL-3` GALE and OC1 produced oblique trees. Finally for the `MIXED` data set, GALE produce heterogeneous trees, where as OC1 was tested in both its orthogonal and oblique tree facets.

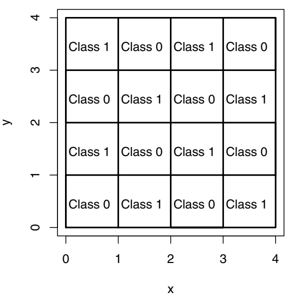
Both algorithms were run using the parameters described in GALE [11] and OC1 [26] original papers. Since the goal of this test was to learn more about the behavior of these algorithms in the artificially-generated problems, we used the whole data sets for training, and inspected the outputted trees. Both algorithms discovered the overall classification structure correctly approximating the classification boundaries presented in figure 4. For each kind of data set,



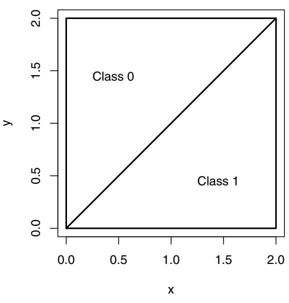
(a) ORT-1



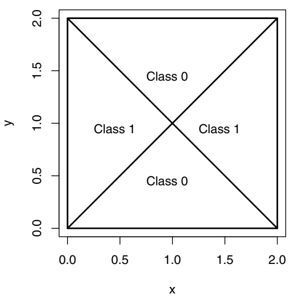
(b) ORT-2



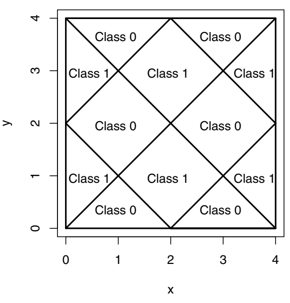
(c) ORT-3



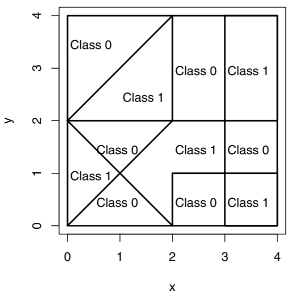
(d) OBL-1



(e) OBL-2



(f) OBL-3



(g) MIXED

Fig. 4. Artificially generated data sets for initial testing.

the appropriate knowledge representation was chosen, as explained above. An special case is OC1 and the MIXED data set, where we ran it for the two available knowledge representations. Table 1 summarizes the results achieved.

Table 1. GALE and OC1 results (classification accuracy and number of leaves) on artificially-generated data sets. For each data set, the appropriate knowledge representation was used. OC1 induced both orthogonal and oblique trees for the MIXED data set.

	ORT-1	ORT-2	ORT-3	OBL-1	OBL-2	OBL-3	MIXED
GALE	100.00(2)	100.00(4)	100.00(16)	100.00(2)	100.00(6)	98.84(20)	98.88(25)
OC1	100.00(2)	100.00(6)	100.00(20)	100.00(2)	99.00(19)	97.75(46)	98.38(73)/97.69(29)

The classification mistakes made by OC1 and GALE were done at intersection points of classification boundaries, or at the extremes of the hyperplanes. OC1 presented a strong tendency to produce overfitted trees—even after pruning them. Such tendency is clearly stated in the orthogonal data sets, where GALE was the only one able to evolve a perfect solution (minimal orthogonal tree that matches the classification boundaries.) This behavior was also observed on the oblique data sets runs. Finally, in the `MIXED` data set GALE also took advantage of evolving mixed trees, whereas when OC1 induced orthogonal trees it displayed a *staircase effect* on the oblique decision boundaries. The *staircase effect* [28] emerges in parallel-axis learners if there are non-axis-parallel classification boundaries in the problem. To define such boundaries the learner is forced to produce large subtrees.

4.3 UCI Repository Data Sets

The second kind of experiment used data sets from the UCI repository [27]. The data sets used were: (1) *Wisconsin breast cancer* (`bre`), (2) *glass identification* (`gls`), (3) *heart disease from statlog project* (`h-s`), (4) *ionosphere* (`ion`), (5) *iris* (`irs`), (6) *Pima-indian diabetes* (`dia`), (7) *sonar* (`son`), and (8) *vehicle from statlog project* (`veh`). A detailed explanation of these data sets is beyond the scope of this paper. For further details, please refer to [27]. Another non-UCI data set was also used in these experiments. The *tao* (`tao`) data set, although being artificially-generated, presents non-linear classification boundaries. This problem was firstly introduced by Llorà & Garrell [14]. The criteria for selecting these data sets was to explore problems with a wide range of dimensions, as well as different numbers of classes.

In order to compare the performance of the different algorithms in terms of classification accuracy, the following methodology was used. Classification accuracy was estimated using stratified ten-fold cross-validation runs. To estimate the difference in performance between the proposed framework for mixed tree evolution and the previous algorithms presented in sections 2 and 4.1, a paired t-test was used [29]. Table 3 summarizes the results achieved using this methodology.

The experiments revealed several interesting insights about the algorithms and the selected problems. If we just take a plain look at the results presented in table 3, the results of GALE evolving mixed trees confirm the viability of the in-

Table 2. Experimental results: percentage of correct classifications and standard deviation from stratified ten-fold cross-validation runs. Results are also marked with a ∘ if they show a significant improvement (1% significant level on paired two-sided *t*-test) over the corresponding GALE-mix results, and with a • if they show a significant degradation.

DS	C4.5r8	OC1-ort	GALE-ort	CART-LC	OC1-obl	GALE-obl	GALE-mix
`brs`	95.42±1.69	94.79±1.22	94.42±1.88	95.86±2.37	95.46±3.42	91.70±3.24	95.10±2.10
`gls`	65.89±10.47	65.19±10.89	65.42±11.89	65.45±14.52	59.81±9.55	49.07±9.20$^\bullet$	65.19±7.27
`h-s`	76.30±5.85	77.41±8.27	82.22±7.11	76.30±6.34	78.15±8.98	71.11±7.35	79.64±9.11
`ion`	89.74±5.23	89.18±7.69	94.02±3.27	84.57±3.61$^\bullet$	90.01±4.11	90.31±3.57	91.52±5.63
`irs`	95.33±3.26	93.33±8.32	96.00±3.46	94.00±6.63	95.33±6.33	98.67±2.98°	95.33±3.05
`pmi`	73.05±5.32	74.57±4.67	75.78±4.01	72.86±4.25	72.00±5.52	69.40±3.24$^\bullet$	73.60±5.88
`son`	71.15±8.54	72.57±13.04	74.52±7.42	68.14±5.94	64.79±17.24	68.27±10.03	71.57±11.32
`tao`	95.07±2.11°	95.06±1.57°	97.03±2.52°	96.23±1.48°	89.78±2.29$^\bullet$	91.74±2.65	91.31±1.53
`veh`	73.64±5.42°	71.23±5.32°	68.32±6.01°	68.33±6.48°	71.97±4.09°	58.87±5.37$^\bullet$	63.84±3.04
`Average`	80.04	81.56	81.24	80.19	80.51	75.82	80.79

Table 3. Experimental results: percentage of correct classifications and standard deviation from stratified ten-fold cross-validation runs. Results are also marked with a ∘ if they show a significant improvement (1% significant level on paired two-sided *t*-test) over the corresponding GALE-mix results, and with a • if they show a significant degradation.

DS	C4.5r8	OC1-ort	GALE-ort	CART-LC	OC1-obl	GALE-obl	GALE-mix
`brs`	95.42±1.69	94.79±1.22	94.42±1.88	95.86±2.37	95.46±3.42	91.70±3.24	95.10±2.10
`gls`	65.89±10.47	65.19±10.89	65.42±11.89	65.45±14.52	59.81±9.55	49.07±9.20$^\bullet$	65.19±7.27
`h-s`	76.30±5.85	77.41±8.27	82.22±7.11	76.30±6.34	78.15±8.98	71.11±7.35	79.64±9.11
`ion`	89.74±5.23	89.18±7.69	94.02±3.27	84.57±3.61$^\bullet$	90.01±4.11	90.31±3.57	91.52±5.63
`irs`	95.33±3.26	93.33±8.32	96.00±3.46	94.00±6.63	95.33±6.33	98.67±2.98°	95.33±3.05
`pmi`	73.05±5.32	74.57±4.67	75.78±4.01	72.86±4.25	72.00±5.52	69.40±3.24$^\bullet$	73.60±5.88
`son`	71.15±8.54	72.57±13.04	74.52±7.42	68.14±5.94	64.79±17.24	68.27±10.03	71.57±11.32
`tao`	95.07±2.11°	95.06±1.57°	97.03±2.52°	96.23±1.48°	89.78±2.29$^\bullet$	91.74±2.65	91.31±1.53
`veh`	73.64±5.42°	71.23±5.32°	68.32±6.01°	68.33±6.48°	71.97±4.09°	58.87±5.37$^\bullet$	63.84±3.04
`Average`	80.04	81.56	81.24	80.19	80.51	75.82	80.79

tuition presented in section 4.2. Results showed that for some problems parallel-axis (`gls`) or oblique (`irs`) boundaries were preferable. In other problems, there was no clear preference for any of the available knowledge representations (`brs`.)

However, the results became more interesting when we took a look at the evolved mixed trees. GALE required a little longer to evolve competent mixed trees—proved in informal experiments. Hence, the accuracy achieved by GALE evolving mixed trees was slightly lower However, we maintained a common configuration among all the data set in favor of a fair comparison. Nevertheless, we are already addressing this issue as part of our further work. Mixed trees, as result of the evolutionary guidance, adapted the knowledge representation to the needs of each of the different data sets explored. A clear tendency to obtain the right solution, in terms of knowledge representation, always emerged in GALE-mix runs. This result is encouraging, since it confirms our intuition that the bias introduced by the knowledge representation can be minimized by mixing different kinds of trees, under the guidance of evolution. This tendency also encourages us to pursue a better understanding of GALE-mix behavior using the Illinois decomposition methodology [30,31].

5 Conclusions

The knowledge representation language has been traditionally chosen beforehand, tailoring the evolutionary algorithm around it. Such a decision may constraint—*a priori*—the concepts that can be learned and the possible usage of such frameworks in real-world problems. In order to minimize this bias, we proposed the evolutionary mixing of different knowledge representations. The work presented in this paper modified an existing LCS framework, allowing the mixing of different knowledge representations in the individuals of the population.

The experiments showed promising initial results. Besides showing that mixing different knowledge representations under the guidance of evolution is possible, results also showed that mixing helped minimize the bias introduced a priori. The experiments showed how the population adapted the knowledge representations used in its individuals to fit the problem to be solved, with little extra evolutionary machinery. This adaptive behavior encourages us to conduct further research on combining other kinds of representations, as well as introducing specialized mechanisms for achieving this purpose. Further research will also include a theoretical insight into GALE using the Illinois decomposition methodology.

Acknowledgments. This work was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant F49620-03-1-0129, and by the Technology Research, Education, and Commercialization Center (TRECC), at University of Illinois at Urbana-Champaign, administered by the National Center for Supercomputing Applications (NCSA) and funded by the Office of Naval Research under grant N00014-01-1-0175. The US Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research, the Technology Research, Education, and Commercialization Center, the Office of Naval Research, or the U.S. Government.

References

1. Janikow, C.Z.: A genetic algorithm for optimizing fuzzy decision trees. Proceedings of the Sixth International Conference on Genetic Algorithms (1995) 421–428
2. Kennedy, H., Chinniah, C., Bradbeer, P.V.G., Morss, L.: The construction and evaluation of decision trees: a comparison of evolutionary and concept learning methods. In: Selected Papers from AISB Workshop on Evolutionary Computing, Springer-Verlag (1997) 147–162
3. Ryan, M.D., Rayward-Smith, V.J.: The evolution of decision trees. Genetic Programming 98 (1998) 350–358

4. Folino, G., Pizzuti, C. Spezzano, G.: Genetic programming and simulated annealing: A hybrid method to evolve decision trees. Genetic Programming: Third European Conference (2000) 294–303
5. Tanigawa, T., Zhao, Q.: A study on efficient generation of decision trees using genetic programming. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2000) (2000) 1047–1052
6. Llorà, X., Garrell, J.M.: Evolution of Decision Trees. Forth Catalan Conference on Artificial Intelligence (CCIA'2001) (2001) 115–122
7. Papagelis, A., Kalles, D.: GA Tree: genetically evolved decision trees. 12th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'00) (2000) 203–206
8. Fu, Z., Golden, B., Lele, S., Wasil, E.: A Genetic Algorithm-based Approach for Building Accurate Decision Trees. INFORMS Journal on Computing **15** (2003) 3–22
9. Cantu-Paz, E., Kamath, C.: Using evolutionary algorithms to induce oblique decision trees. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000) (2000) 1053–1060
10. Llorà, X.: Genetic Based Machine Learning using Fine-grained Parallelism for Data Mining. PhD thesis, Enginyeria i Arquitectura La Salle. Ramon Llull University, Barcelona (February, 2002)
11. Llorà, X., Garrell, J.M.: Knowledge-Independent Data Mining with Fine-Grained Parallel Evolutionary Algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001), Morgan Kaufmann Publishers (2001) 461–468
12. Llorà, X., Garrell, J.M.: Coevolving different knowledge representations with fine-grained parallel learning classifier systems. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2002), New York, Morgan Kaufmann Publishers (2002) 934–941
13. Brodley, C.E.: Addressing the seletive superiority problem: Automatic algorithm/model class selection. In: Proceedings of the 10th International Conference on Machine Learning, Morgan Kaufmann (1993) 17–24
14. Llorà, X., Garrell, J.M.: Evolving Partially-Defined instances with Evolutionary Algorithms. In: Proceedings of the 18th International Conference on Machine Learning (ICML'2001), Morgan Kauffmann (2001) 337–344
15. Llorà, X., Garrell, J.M.: Prototype induction and attribute selection via evolutionary algorithms. Intelligent Data Analysis **7** (2003) 193–208
16. Quinlan, R.: Induction of decission trees. Machine Learning, Vol. 1, No. 1 (1986) 81–106
17. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth International Group (1984)
18. Van de Merckt, T.: Decision trees in numerical attribute spaces. In: Proceedings of the 13th Intenational Joint Conference on Artificial Intelligece, Morgan Kaufmann (1993) 1016–1021
19. Jong, K.A.D., Spears, W.M.: Learning Concept Classification Rules using Genetic Algorithms. In: Proceedings of the Twelfth International Conference on Artificial Intelligence IJCAI-91. Volume 2., Morgan Kaufmann (1991) 651–656
20. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann (1993)
21. Heath, D., Kasif, S., Salzberg, S.: Learning oblique decision trees. In: Proceedings of the 13th Intenational Joint Conference on Artificial Intelligece, Morgan Kaufmann (1993) 1002–1007

22. Domingos, P.: Rule Induction and Instance-based Learning: A Unified Approach. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, Morgan Kaufmann (1995) 1226–1232
23. Wettschereck, D.: A hybrid Nearest-Neighbor and Nearest-Hyperrectangle Algorithm. In: Proceedings of the 7th European Conference on Machine Learning, LNAI. Volume 784. (1994) 323–335
24. Aha, D.W., Kibler, D., Albert, M.K.: Instance-based Learning Algorithms. Machine Learning **6** (1991) 37–66
25. Quinlan, J.R.: Induction of decision trees. Machine Learning **1** (1986) 81–106
26. Murthy, S.K., Kasif, S., Salzberg, S.: A system for induction of oblique decision trees. Journal of Artificial Intelligence Research **2** (1994) 1–32
27. Merz, C.J., Murphy, P.M.: UCI Repository for Machine Learning Data-Bases [http://www.ics.uci.edu/∼mlearn/MLRepository. html]. Irvine, CA: University of California, Department of Information and Computer Science (1998)
28. Abbott, D.W.: Combining models to improve classifier accuracy and robustness. In: Proceedings of Second International Conference on Information Fusion. (1999) 289–295
29. Dietterich, T.G.: Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. Neural Computation **10** (1998) 1895–1924
30. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading, Mass. (1989)
31. Goldberg, D.E.: The Design of Innovation: Lessons from and for Competent Genetic Algorithms. Kluwer Academic Publishers, Norwell, MA (2002)