Bounding the Effect of Noise in Multiobjective Learning Classifier Systems

Xavier Llora
Illinois Genetic Algorithms Laboratory, National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign, Urbana, 61801 IL, USA

David E. Goldberg
deg@illigal.ge.uiuc.edu
Illinois Genetic Algorithms Laboratory, Department of General Engineering, University of Illinois at Urbana-Champaign, Urbana, 61801 IL, USA

Abstract
This paper analyzes the impact of using noisy data sets in Pittsburgh-style learning classifier systems. This study was done using a particular kind of learning classifier system based on multiobjective selection. Our goal was to characterize the behavior of this kind of algorithms when dealing with noisy domains. For this reason, we developed a theoretical model for predicting the minimal achievable error in noisy domains. Combining this theoretical model for crisp learners with graphical representations of the evolved hypotheses through multiobjective techniques, we are able to bound the behavior of a learning classifier system. This kind of modeling lets us identify relevant characteristics of the evolved hypotheses, such as overfitting conditions that lead to hypotheses that poorly generalize the concept to be learned.

Keywords
Noise analysis, multiobjective learning classifier systems, minimal achievable error.

1 Introduction
The goal of this paper is to analyze the impact of noisy data sets on a particular kind of learning classifier systems (LCS) (Holland, 1975; Goldberg, 1989; Wilson, 1995). Our work is focused on Pittsburgh-style LCS (Smith, 1983; De Jong & Spears, 1991; Janikow, 1991; Michalewicz, 1992). The presence of noise in the data set usually misleads the behavior of learning algorithms (Mitchell, 1997). As we will show in this paper, introducing some multiobjective techniques in Pittsburgh LCSs, combined with theoretical models, we can bound the effect of noise on the population of evolved hypotheses. These bounds can easily help us to identify overfitted hypotheses in the evolved population.

Pittsburgh-style LCSs evolve a population of hypotheses for a given classification tasks. In order to evolve accurate hypotheses, this kind of LCS uses a set of instances that describe the knowledge to be learned. Each individual in the population codifies an hypothesis, usually represented by a set of rules. Recently, efforts in improving the efficiency of Pittsburgh-style LCSs have turned to multiobjective techniques (Llorà, Goldberg, Traus, & Bernadó, 2003). Balancing accuracy and generality, multiobjective LCSs evolve a tradeoff of hypotheses which are useful for understanding the underlying knowledge that produced the data set. Moreover, the graphical representation of the population of hypotheses using Pareto fronts (Pareto, 1896) is the basis of the bounding process presented in this paper for noisy data sets.

©2003 by the Massachusetts Institute of Technology
Although Pareto fronts are going to be the basis for computing the boundaries in noisy problems, this process also requires some theoretical insights. For this reason, we focused on the LED problem (Breiman, Friedman, Olshen, & Stone, 1984). This is a well-known problem in the machine learning community because its main characteristic is that it was designed for the analysis of the behavior of learning algorithms on noisy domains. Using the LED problem, we develop the minimal achievable error model (MAE). The aim of this model is to predict, using statistics, which is the minimal error that a crisp learning algorithm can achieve given a noise ratio $\epsilon$ in the data set. Using the MAE model, together with multiobjective LCS, we can describe the behavior of the learning process. Moreover, we can easily identify overfitted hypotheses evolved by the LCS.

The paper is structured as follows. Section 2 briefly reviews some relevant multiobjective definitions used throughout the rest of the paper. It also describes how multiobjective ideas can be introduced in LCS. Then, Section 3 introduces the LED problem, and the procedure of noise addition used for generating noisy data sets. Section 4 presents the minimal achievable error model for the LED problem and its theoretical grounds. The theoretical model proposed in this paper is empirically validated in Section 5. The validation of the model also includes the validation of the noise boundaries identified using the theoretical model. Finally, Section 6 offers a summary of our main conclusions.

2 Multiobjective Evolution and Learning Classifier Systems

Since multiobjective optimization plays a central role in this paper, this section summarizes some relevant issues. First, we briefly summarize general multiobjective optimization definitions in Subsection 2.1. Then, Section 2.2 presents how multiobjective ideas can be used for defining the evaluation of the hypotheses in an LCS. Section 2.3 describes a simple Pittsburgh LCS based on multiobjective ideas. Finally, Section 2.4 details the multiobjective evaluation introduced in the LCS.

2.1 Multiobjective Optimization

In a multiobjective optimization problem (MOP) (Van Veldhuizen & Lamont, 2000) a solution $\mathbf{x} \in \Omega$ is represented as a vector of $n$ decision variables $\mathbf{x} = (x_1, \ldots, x_n)$, where $\Omega$ is the decision variable space. We want to optimize $k$ objectives, which are defined as $f_i(\mathbf{x})$, with $i = 1 \ldots k$. These objectives are grouped in a vector function denoted as $F(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x}))$, where $F(\mathbf{x}) \in \Lambda$. $F$ is a function which maps points from the decision variable space $\Omega$ to the objective function space $\Lambda$:

\[
F : \Omega \rightarrow \Lambda \\
\mathbf{x} \mapsto \mathbf{y} = F(\mathbf{x})
\]  

Without loss of generality, we can define a MOP as the problem of minimizing a set of objectives $F(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x}))$, subject to some constraints $g_i(\mathbf{x}) \leq 0$, $i = 1, \ldots, m$. These constraints are necessary for problems where there are invalid solutions in $\Omega$. Although the definition of a MOP addresses a minimization problem, an MOP is not limited exclusively to minimization. A MOP can be applied to maximization problems as well as to problems where some objectives must be minimized and some others are maximized. Nevertheless, in the rest of this section we assume a minimization MOP.

In an MOP, the solution that minimizes all the objectives and satisfies all the constraints may not exist. Sometimes, the minimization of a certain objective implies a
degradation in some other objective. Then, there is not a global optimum that mini-
mizes all the objectives simultaneously. In this context, the concept of optimality must
be redefined. Pareto (1896) introduced the concept of dominance and multiobjective
optimality to deal with this issue.

In general terms, a vector \( \vec{u} \) dominates another vector \( \vec{v} \), written as \( \vec{u} \preceq \vec{v} \), if and only
if every component \( u_i \) is less or equal than \( v_i \), and at least there is one component in \( \vec{u} \)
which is strictly less than the corresponding component in \( \vec{v} \). This can be formulated
as follows:

\[
\vec{u} \preceq \vec{v} \iff \forall i \in 1, \ldots, k \colon u_i \leq v_i \land \exists i \in 1, \ldots, k : u_i < v_i
\]  

(2)

For example, given a MOP with three objectives and the vectors \( \vec{u} = F(\bar{x}_1) = (1, 1, 2) \) and \( \vec{v} = F(\bar{x}_2) = (1, 2, 2) \), we notice that \( \vec{u} \preceq \vec{v} \). However, if \( \vec{u} = (1, 1, 2) \) and \( \vec{v} = (1, 2, 1) \), neither \( \vec{u} \) dominates \( \vec{v} \) nor \( \vec{v} \) dominates \( \vec{u} \).

The concept of a Pareto optimum is based on the dominance definition. Thus, a
solution \( \bar{x}_2 \) is a Pareto optimum if there is not any other solution \( \bar{x}_0 \) whose
objective vector \( \vec{u}_0 = F(\bar{x}_0) \) dominates \( \vec{u} = F(\bar{x}) \). In other words, a Pareto optimum is a
solution whose objectives can not be improved simultaneously by any other solution.
The set of all solutions whose objective vectors are not dominated by any other
objective vector is called the Pareto optimal set \( \mathcal{P}^* \):

\[
\mathcal{P}^* := \{ \bar{x} \mid \exists \bar{x}_2 : F(\bar{x}_2) \preceq F(\bar{x}) \}
\]  

(3)

Analogously, the set of all vectors \( \vec{u} = F(\bar{x}) \) where \( \bar{x} \) belongs to the Pareto optimal
set is called the Pareto front \( \mathcal{P}^F \):

\[
\mathcal{P}^F := \{ \vec{u} = F(\bar{x}) = (f_1(\bar{x}), \ldots, f_k(\bar{x})) \mid \bar{x} \in \mathcal{P}^* \}
\]  

(4)

2.2 Classification and Multiobjective Optimization

The goal of our multiobjective approach, introduced elsewhere (Llorà, Goldberg, Traus,
& Bernadó, 2003), is to trade off two objectives: (1) the accuracy of a hypothesis, and
(2) its size. If we inspect each objective separately, we are interested in evolving accu-
rate hypotheses (minimizing the classification error). Moreover, we are also interested
in evolving general solutions to the classification problem. This means that we pre-
fer maximally general solutions (Wilson, 1995) describing the knowledge behind that
classification problem. In a LCS where an hypothesis is a complete solution to the
classification problem, this can be achieved by biasing the evolution toward compact
hypotheses. Therefore, we have two different objectives to optimize at the same time,
accuracy and size.

Define \( \bar{x} \) as an hypothesis that is a complete solution to the classification problem;
\( \mathcal{D} \) the training data set for the given problem; \( |\mathcal{D}| \) number of instances in \( \mathcal{D} \); \( \text{miss}(\bar{x}, \mathcal{D}) \)
the number of incorrectly classified instances of \( \mathcal{D} \) performed by \( \bar{x} \); and finally, \( \text{size}(\bar{x}) \)
a measure of the current size of \( \bar{x} \) (e.g. the number of rules contained in \( \mathcal{P} \)). Using this
notation, a multiobjective approach can be achieved as follows:

\[
F(\bar{x}) = (f_e(\bar{x}), f_s(\bar{x}))
\]  

(5)

\[
f_e(\bar{x}) = \frac{\text{miss}(\bar{x}, \mathcal{D})}{|\mathcal{D}|}
\]  

(6)

\[
f_s(\bar{x}) = \text{size}(\bar{x})
\]  

(7)

Thus, our multiobjective approach minimizes \( F(\bar{x}) \). With this simple multiob-

\[\footnote{We used error instead of accuracy for simplicity reasons.}\]  

Evolutionary Computation Volume 11, Number 3

281
jective definition, the evolution is biased toward the hypotheses that form the Pareto optimal set. Therefore, the population may evolve hypotheses with different tradeoffs between accuracy and generality. The main purpose of the evolved Pareto front (or classification front) is two fold. First we seek solutions with different tradeoffs in their accuracy and size. Second we wish to visualize the distribution of the population along the evolutionary learning process.

Moreover, coevolving these compromise solutions, we can delay the need of choosing a solution with the desired tradeoff between the different objectives in order to take the best solution to the problem. This decision can be made once the evolution is over and the classification front is provided. Then, the decision maker has several hypotheses among which to choose, all provided by the classification front. Moreover, the decision maker can be a human or an expert system with some knowledge about the problem. However, there are other approaches already explored in the machine learning (ML) and the genetic-based machine learning (GBML) communities. Among others, some interesting approaches are based on the bagging technique propose by Breiman (1996). The goal in bagging is to combine different hypotheses (e.g. the ones that form the classification front) into a new single classification solution. The hope is to obtain a new combined hypothesis that reduces the impact of overfitting, producing a high quality general hypothesis when tested using unseen data. This technique tends to reduce the deviation among runs, as well as it often improves the generalization capability of the combined solution. A detailed discussion of this issue is beyond the scope of this paper. For further information, the reader is referred to (Llorà, 2002).

2.3 Multiobjective-Evaluated Learning Classifier Systems

There are several approaches to LCSs (Goldberg, 1989; Michalewicz, 1992; Wilson, 1995). Among the different kinds of LCS, our work is based on the Pittsburgh approach. For more detailed information, please refer to (Smith, 1983; Goldberg, 1989; De Jong & Spears, 1991; Janikow, 1991; Michalewicz, 1992; Llorà, 2002). Recently, some work has been done introducing multiobjective evaluations on the Michigan LCS (Bernadó & Garrell, 2001) and the Pittsburgh LCS (Llorà, Goldberg, Traus, & Bernadó, 2003). Our work is based on a Pittsburgh LCS based on genetic algorithms (MOLCS-GA) introduced in (Llorà, Goldberg, Traus, & Bernadó, 2003). This section describes briefly MOLCS-GA, and afterwards it focuses on the evaluation phase where multiobjective techniques are used.

MOLCS-GA is a Pittsburgh-style LCS based on genetic algorithms. If the problem’s attributes are nominal, MOLCS-GA uses rule sets, represented by the ternary alphabet (0, 1, #) often used in other LCSs (Holland, 1975). Thus, each hypothesis is a set of rules. The genetic algorithm learning cycle works as follows. First, the fitness of each hypothesis in the population is computed. This is done on a multiobjective basis, taking into account the misclassification error and the size of each hypothesis. This phase is explained in detail in Section 2.4. Then, selection is applied using a tournament selection algorithm (Oei, Goldberg, & Chang, 1991; Bäck, 1995; Miller & Goldberg, 1995) with elitism. Elitism is often applied in evolutionary multiobjective optimization algorithms and it usually consists in keeping the solutions of the Pareto front evolved in each generation (Van Veldhuizen & Lamont, 2000). Moreover elitism also plays the role of maintaining the solutions at the ends of the Pareto front, preventing a collapse of the front on its central compromise solutions (Deb, 2001; Coello, Van Veldhuizen, & Lamont, 2002). MOLCS-GA performs similarly: it keeps all distinct solutions of the evolved Pareto front, as well as 30% of the hypotheses with the lowest error. This
Figure 1: Example of sorted population fronts at a given iteration of MOLCS-GA in the 11-input multiplexer problem.

guarantees that the best compromise solutions evolved so far are not lost, and the best low-error solutions are also kept, driving the evolution toward more accurate solutions.

After selection, crossover and mutation are applied. The crossover operator is based on the operator described in De Jong and Spears (1991). It is a variant of the classical two-point crossover, adapted to deal with variable-size hypotheses. It works in the following way. The crossover point can occur anywhere (i.e., both on the rule/instance boundaries as well as within a rule/instance). The only requirement is that the crossover points in the two parents must be equivalent in order to produce valid solutions. That is, if one parent is cut on a rule/instance boundary, then the other parent must also be cut on a rule/instance boundary. Similarly, if one parent is cut within a rule/instance, then the other parent must be cut in a similar spot. Finally, the mutation consists of generating a random new gene value. This process is based on the niche mutation technique used in XCS (Wilson, 1995; Kovacs, 1999).

2.4 Multiobjective Fitness in MOLCS-GA

The multiobjective fitness scheme used by MOLCS-GA is inspired by NSGA (Srinivas & Deb, 1995) and NSGA-II (Deb, Agrawal, Pratab, & Meyarivan, 2000) and it works as follows. The hypotheses of the population are sorted in equivalent classes. These classes are defined by the Pareto fronts that can be defined among the population. That is, given a population of hypotheses \( I \), the first equivalence class \( I^0 \) is the set of hypotheses which belongs to the evolved Pareto optimal set \( I^0 = \mathcal{P}^* (I) \). The next equivalence class \( I^1 \) is computed without considering the hypotheses in \( I^0 \), as \( I^1 = \mathcal{P}^* (I \backslash I^0) \), and so forth. Figure 1 shows an example of the different equivalence classes, presented using the fronts that appear in a population at a given iteration. This plot was obtained solving the 11-input multiplexer problem (Llorà, Goldberg, Traus, & Bernadó, 2003).
The population is classified into nine different fronts. The left front is $I^0$, which corresponds to the non-dominated vectors of the population. The next front to the right represents $I^1$ and so on.

Once the population of hypotheses $I$ is sorted, fitness values are assigned. Since the evolution must bias the population toward non-dominated solutions, we impose the constraint:

$$fitness(I_i) > fitness(I_{i+1})$$

(8)

Thus, the evolution will guide the population toward $I^0$, the Pareto front of the population. The fitness of each hypothesis depends on the front where the hypothesis belongs. That is, all the hypotheses of the same equivalence class $I_i$ receive the same constant $n_i$ base value, where $n$ is the number of equivalence classes in $I$ and $\delta$ is a constant. Moreover, in order to spread the population along the Pareto front, a sharing function is applied. Thus, the final fitness of an hypothesis $j$ in a given equivalence class $I_i$ is:

$$fitness(I_i^j) = \delta \left( \frac{1}{\sum_{k \in I_i} \phi(d_I^j I_k)} + n - i - 1 \right)$$

(9)

where $\phi(d_I^j I_k)$ is the sharing function (Goldberg & Richardson, 1987). The sharing function is computed using the phenotypical distance between the hypotheses; that is, the Euclidean distance between their multiobjective vectors. The radius of the sharing function was set to $\sigma_{sh} = 0.1$.

3 The LED Problem

The LED problem is a simple classification task that consists of, given the active leds on a seven segment display, identifying the digit that the display is representing. It was introduced by Breiman, Friedman, Olshen, and Stone (1984). A simple scheme of the seven segments used is shown in Figure 2. Thus, the classification task to be solved is the one described by seven binary attributes ($X = \{s_0, s_1, s_2, s_3, s_4, s_5, s_6\}$) over the ten available classes ($C = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$). A 1 in the $s_i$ attribute indicates that the $i$ led is active, and vice-versa. Figure 3 lists the minimal subset of instances $O$ that describe the classification problem.
### 4 Minimal Achievable Error

This section presents a theoretical model for predicting the minimal achievable error (MAE) (Llorá & Goldberg, 2002), given a noise ratio $\epsilon$ in the LED data set problem described in Section 3. The motivation for developing this theoretical model is to understand and to explain some of the results that different systems achieve when solving the LED problem. Moreover, when MAE is combined with multiobjective LCS, it provides useful insights and explanations about the learning behavior. The model is developed in two steps. The first one, Section 4.1, models a simplified LED problem using only one diode. Then, the second step, Section 4.2, generalize the previous model to the seven segments LED problem.

#### 4.1 The Simplified 1-LED Problem

Before building the MAE model for the previously presented LED problem, we present the MAE model obtained for a simplified version of the LED problem. The simplified 1-LED problem only uses 1 led (bit in the antecedent) and 2 classes (0,1). The optimal solution $O$ for the 1-LED is:

<table>
<thead>
<tr>
<th>$s_0$</th>
<th>$s_1$</th>
<th>$s_2$</th>
<th>$s_3$</th>
<th>$s_4$</th>
<th>$s_5$</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

A program that implements this method can be found in Merz and Murphy (1998). Each instance $y$ is picked at random from $O$ with a uniform probability $p = \frac{1}{|O|} = \frac{1}{10}$. Then, for each attribute in $y$, its value is swapped with a given probability equal to the noise ratio $\epsilon$ introduced into the data set.
Given a noise rate $\epsilon$, the goal is to build an statistical model that predicts the MAE. Since this problem only has two possible antecedents, 0 and 1, there are only four possible scenarios for noise perturbation. These scenarios are:

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>The perturbed antecedent is 0 and the result is 0</td>
</tr>
<tr>
<td>01</td>
<td>The perturbed antecedent is 0 and the result is 1</td>
</tr>
<tr>
<td>10</td>
<td>The perturbed antecedent is 1 and the result is 0</td>
</tr>
<tr>
<td>11</td>
<td>The perturbed antecedent is 1 and the result is 1</td>
</tr>
</tbody>
</table>

Therefore, we can write the jumping probabilities between antecedents as a matrix. Let us call this matrix $J$. Rows represent the original antecedent (either 0 or 1), whereas the columns show the final antecedent after noise perturbation. This matrix can be written as:

$$
J = \begin{bmatrix}
1 - \epsilon & \epsilon \\
\epsilon & 1 - \epsilon
\end{bmatrix}
$$

(10)

From $J$ we can compute two different things:

1. The probability of the appearance of a given antecedent.
2. The minimal classification error for a given antecedent.

In order to compute the likelihood vector of appearance of the antecedents, we need to impose the constrain that $\sum J_{ij} = 1$. The normalized matrix $J^N$ is achieved as follows:

$$
J^N = \frac{1}{\sum J_{ij}} \cdot J = \begin{bmatrix}
\frac{1-\epsilon}{2} & \frac{\epsilon}{2} \\
\frac{\epsilon}{2} & \frac{1-\epsilon}{2}
\end{bmatrix}
$$

(11)

Using $J^N$, the appearance vector $\alpha$ for the two possible antecedents after noise perturbation, is defined as:

$$
\alpha_i = \sum_j J^N_{ji}
$$

(12)

One antecedent may appear for two different reasons. The first one is that the antecedent has not been perturbed by the noise addition. The second one is due to the noise perturbation. Computing $\alpha$ for the 1-LED problem we obtain that:

$$
\alpha = \left( \frac{1-\epsilon}{2} + \frac{\epsilon}{2} \cdot \frac{\epsilon}{2} + \frac{1-\epsilon}{2} \right) = \left( \frac{1}{2}, \frac{1}{2} \right)
$$

(13)

The second step in the model building process is to compute the minimal classification error for a given instance. Since a given antecedent $a$ is the result of a noise-free instance or a noisy perturbation of an instance, the introduction of noise causes inconsistencies in the data set $D$ (i.e the same antecedent has different consequents). Therefore, the best performance that any crisp learning algorithm can achieve solving the noisy data set $D$ is bounded. This boundary is reached when the algorithm classifies the antecedent $a$ as belonging to the majority class that appears in the class distribution for the given antecedent $a$. The class distribution for a given antecedent $a$ can be computed using $J$. Each column contains this distribution information.
In order to compute the class distribution vector of the inconsistencies for a given antecedent $a$, we need to impose the restriction that $\sum_j J_{ja} = 1$. The class distribution vector $p$ for a given antecedent $a$ is achieved as follows:

$$p_j = \frac{1}{\sum_k J_{ka}} J_{ja} \quad (14)$$

where $j$ is the class. Therefore, the class distribution $p(a)$ vectors for the 1-LED problem are:

$$p(0) = \left( \frac{1}{1 - \epsilon + \epsilon} \cdot (1 - \epsilon), \frac{1}{1 - \epsilon + \epsilon} \cdot \epsilon \right) = (1 - \epsilon, \epsilon) \quad (15)$$

$$p(1) = \left( \frac{1}{1 - \epsilon + \epsilon} \cdot \epsilon, \frac{1}{1 - \epsilon + \epsilon} \cdot (1 - \epsilon) \right) = (\epsilon, 1 - \epsilon) \quad (16)$$

Given a probability distribution $p(a)$ of the class inconsistencies of the antecedent $a$, the best that any crisp learning algorithm can do on the given antecedent is to assign the majority class to the antecedent $a$. Therefore, the minimal classification error $e$ vector for a given antecedent $a$ is:

$$e_a = 1 - \max\{p(a)\} \quad (17)$$

Thus, the errors for the 1-LED problem are:

$$e_0 = 1 - \max\{1 - \epsilon, \epsilon\} \quad (18)$$

$$e_1 = 1 - \max\{\epsilon, 1 - \epsilon\} \quad (19)$$

MAE is the weighted sum of these $e_a$ errors, where the weights are provided by the likelihood vector of instance appearance $\alpha$. Therefore, the MAE is computed as:

$$MAE = \sum_i \alpha_i e_i \quad (20)$$

The final MAE value for the 1-LED problem is:

$$MAE = \sum_i \alpha_i e_i = 1 - \max\{1 - \epsilon, \epsilon\} \quad (21)$$

In order to validate the MAE model for the 1-LED problem empirically, we conducted a simple experiment. For different noise ratios ($\epsilon = \{0, 0.02, 0.04, 0.06 \ldots 1\}$) we generated a 1-LED noisy data set $D$. This data set contains 10,000 noisy instances. Then, we computed the error based on the degree of inconsistencies, as previously explained, for each $D$, data set. Figure 4 shows the validity of the MAE model showing both the theoretical MAE and the empirical one computed using the $D$, data sets.

One property of the MAE is its specular symmetry related to $\epsilon = 0.5$, as the result of the $1 - \max\{1 - \epsilon, \epsilon\}$ term. This property can be intuitively understood analyzing the extreme cases $\epsilon = 0$ and $\epsilon = 1$. In both cases, the MAE value is equal to 0 and in both cases it is solving equivalent problems. When $\epsilon = 0$, no instance is perturbed, whereas when $\epsilon = 1$, the problem that is being solved is the negation of 1-LED, since all the bits in the antecedent are always swapped. Therefore, when $\epsilon = 1$, the problem that is being solved is $\neg$(1-LED).
4.2 Seven Segments LED Problem

In order to generalize the 1-LED MAE model to the original seven segments LED problem, only few changes must be introduced to the 1-LED MAE model. All these changes only take place in matrix $J$.

The optimal subset of instances $O$ for the LED problem was shown in Figure 3. The number of possible antecedents is $2^7 = 128$. In the noise-free LED problem, only ten of all the antecedents are part of $O$, and thus part of the data set $D$. The remaining 118 antecedents only appear in $D$ as the effect of noise addition. Having this in mind, we can now define the jump matrix $\mathcal{J}$. As in the 1-LED problem, rows represent the original antecedent (indexed by the class of the instance $\chi(i)$), whereas the columns show the resulting antecedent after noise perturbation (128 possible antecedents). Thus, $\mathcal{J}$ is a $10 \times 128$ matrix in the LED problem.

In order to keep the notation simple we assume that each antecedent is referred by the number that it codifies (binary number representation). Moreover, $d_{ij}$ is the Hamming distance between antecedents $i$ and $j$, and $\epsilon$ is the noise ratio added to the data set. Thus, the jump matrix $\mathcal{J}$ is defined as:

$$\mathcal{J}_{\chi(i)j} = \epsilon^{d_{ij}} \cdot (1 - \epsilon)^{(7-d_{ij})}$$

(22)

This is not a binomial distribution because only one of the possible changes of $d_{ij}$ bits leads to the target antecedent $j$. This is the reason why equation 22 is missing the combinatorial factor. Once we have the $\mathcal{J}$ matrix, we can compute $\mathcal{J}^N$, $\alpha$, $p(a)$ and $\epsilon$ in the same way as described in the 1-LED problem.

Thus, the normalized $\mathcal{J}^N$ matrix is defined as:

$$\mathcal{J}_{\chi(i)j}^N = \frac{1}{\sum_k \mathcal{J}_{kl}} \cdot \mathcal{J}_{\chi(i)j}$$

(23)
Figure 5: Empirical validation of the MAE model for the LED problem using different sizes of the $D_c$ noisy data set.

Once we have computed the normalized $\mathcal{J}_N$ matrix, we can move forward and compute the appearance vector $\alpha$ for all the antecedents (128 antecedents for the LED problem) as:

$$\alpha_i = \sum_{\chi(j)} \mathcal{J}_N^{(j)}_{i}$$

(24)

Before we can compute the MAE for the LED problem, we need to compute the class distribution $p(a)_j$ for a given antecedent $a$, and its minimal classification error. The $p(a)_j$ vector is computed as shown in equation 14 using $\mathcal{J}_N^{(j)}_{i}$ matrix. Then, we can use $p(a)_j$ to compute the minimal classification error $e_i$ as shown in equation 17.

Once we have all these values, the MAE for the LED problem is simply computed using the same formula used for the 1-LED problem. Thus, the MAE for the seven segment LED problem is computed as:

$$MAE = \sum_i \alpha_i e_i$$

(25)

Figure 5 shows the empirical validation of the MAE model for the LED problem. The empirical data was obtained using different noisy $D_c$ data set sizes (500, 1,000, 2,000, 5,000, and 500,000 noisy instances) and computing the error based on the degree of inconsistencies. There are two interesting observations to be made from the data shown in Figure 5. The first one is that, when enough instances are provided to the $D_c$ data set, the theoretical model and the experimental results match perfectly (e.g. 500,000 instances). The second interesting observation appears when we analyze the results achieved using the sizes of the $D_c$ data set: 500, 1,000, 2,000, and 5,000. These sizes are usually used for measuring the performance of learning algorithms (see Table 1) when they solve the LED problem. The empirical results obtained using small data sizes...
Table 1: Some previous results obtained in the noisy LED data set.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Source</th>
<th>Noise ratio ($\epsilon$)</th>
<th>Accuracy (%)</th>
<th>Training/Test Instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>CART</td>
<td>(Breiman, Friedman, Olshen, &amp; Stone, 1984)</td>
<td>0.1</td>
<td>71.0</td>
<td>200/5000</td>
</tr>
<tr>
<td>Bayes</td>
<td>(Breiman, Friedman, Olshen, &amp; Stone, 1984)</td>
<td>0.1</td>
<td>74.0</td>
<td>200/5000</td>
</tr>
<tr>
<td>C4</td>
<td>(Quinlan, 1987)</td>
<td>0.1</td>
<td>72.6</td>
<td>2000/500</td>
</tr>
<tr>
<td>IWN</td>
<td>(Tan &amp; Eshelman, 1988)</td>
<td>0.1</td>
<td>73.3</td>
<td>400/500</td>
</tr>
<tr>
<td>PART</td>
<td>(Bernadó, Llora, &amp; Garrell, 2001)</td>
<td>0.1</td>
<td>74.9</td>
<td>2000/5000</td>
</tr>
<tr>
<td>SMO</td>
<td>(Bernadó, Llora, &amp; Garrell, 2001)</td>
<td>0.1</td>
<td>75.1</td>
<td>2000/5000</td>
</tr>
<tr>
<td>XCS</td>
<td>(Bernadó, Llora, &amp; Garrell, 2001)</td>
<td>0.1</td>
<td>74.5</td>
<td>2000/5000</td>
</tr>
<tr>
<td>GALE</td>
<td>(Llora, 2002)</td>
<td>0.1</td>
<td>75.0</td>
<td>2000/5000</td>
</tr>
</tbody>
</table>

Figure 6 presents a zoom of Figure 5. The main interest of this zoom is to show the MAE results achieved for $\epsilon = 0.1$. This value is often used for comparing algorithms using the LED problem (see Table 1). This zoom shows that a comparison between algorithms using such small data sets can be misleading. This is even worse if the comparison is done using different randomly generated $D_e$ data sets, because the empirical MAE for the different data sets (since they are not big enough) may be different, and consequently incomparable. Moreover, since the empirical MAE is different from the theoretical one, the generalization capabilities of the learning algorithm are misguided, overfitting the final hypothesis to the training data set provided. This can be proved easily providing another randomly generated $D_e$ data set (test set) that may have a dif-

sets show smaller MAE values than the ones theoretically predicted, showing some interesting deviations. These are the result of the random number generator bias and the resulting instance distribution. Therefore, the experimental MAE for a given noise ratio $\epsilon$ is different than the theoretically expected MAE, since not enough instances are generated. This fact leads to data sets that maintain some regularities that reduce the amount of inconsistencies in $D_e$.
Using this optimal solution for the LED problem, presented in Figure 3, we can draw the optimal Pareto front. This front is built removing one rule at a time.

Different experimental MAE value due to its small size. This fact is critical when using small data sets like the ones in Table 1.

5 Visualizing Learning: Pareto Fronts and Overfitting

This section analyses the learning process performed by MOLCS-GA. This study is based on two different concepts. The first one is the visualization of the population through Pareto fronts. When this visualization is mixed with the theoretical MAE model developed in Section 4, we gain useful insights of the learning process.

In order to allow the replication of the results presented in this section, we briefly summarize the parameter setting used in MOLCS-GA. The parameter values used were: $\sigma_{sh}=0.1$, $\delta=1000$, $\text{pop\_size}=285$, crossover probability $p_c=0.4$, probability of mutation of an hypothesis $p_{mut}=0.02$, and the gene perturbation probability $p_{gen}=0.25$. The maximum number of iterations allowed in MOLCS-GA was set to 500. This is an overdimensioned value. The reason for using this value is to show how this multiobjective approach controls the size of the evolved hypotheses. One of the main problems that arises with the evolution of variable-size hypotheses is the bloat phenomenon (Tackett, 1994). Bloat is usually defined (in genetic programming terms) as the code growth (i.e. number of rules) of hypotheses without any fitness improvement (i.e. accuracy). MOLCS-GA runs along 500 iterations proved the ability of the multiobjective approach to control the bloat phenomenon (Llorà, Goldberg, Traus, & Bernadó, 2003).

The usage of a multiobjective evaluation technique, like the one presented in Section 2, in a LCS splits the population of hypotheses along the Pareto front. Since the work done deals with the LED problem, we need to know its optimal Pareto front. This front can be computed using the optimal solution $O$, shown in Figure 3. This front is obtained by removing one rule and computing the accuracy of the resulting set of
rules. This process is repeated until only one rule is left. Thus, using this computed
pair (accuracy and number of rules) the optimal Pareto front is shown in Figure 7.

Using the theoretical results presented, in Section 4, we can explain the Pareto
front achieved in the noisy LED problem using MOLCS-GA. We run MOLCS-GA using
a data set containing 2,000 instances. These instances were generated using a noise ratio
ε = 0.1 (see Section 3), that theoretically leads to a MAE equal to 0.26 (see Figure 5).
We also compute the empirical MAE of the randomly generated data set. However, the
empirical MAE obtained from the $D_{r}$ used was 0.23. Figure 8 presents the Pareto front
achieved by MOLCS-GA. The figure also overlaps the theoretical and empirical MAE
boundaries, as well as the optimal non-noisy Pareto front, and the size of the optimal
solution to the LED problem $O$.

Figure 8 shows some interesting characteristics pointed out by the MAE model.
The first one is the fact that the theoretical and the empirical values of MAE do not
match due to the noisy sample process of the LED problem. The second one is the
existence of a rupture point in the Pareto front; the point is defined as (MAE($\epsilon =
0.1), |O|) = (0.26, 10). The rupture point indicates the place where the evolved Pareto front
changes its slope abruptly. The front that appears on the left-hand side of the rupture
point is the result of the deviation of the empirical MAE from its theoretical value. All
the hypotheses that define this left-hand side segment of the front are over-fitted solu-
tions. They are overadapted to the hidden noisy patterns introduced by the data
set generator. This means that they are learning some misleading noisy pattern as the
result of the deviation of the MAE value. Therefore, if any of the hypotheses in the
left-hand side front is tested using a different randomly generated $D_{r}$ data set, then
the hypothesis may not maintain its accuracy. With newly generated data sets, accu-
dacy dropped around 2-4%. These overfitted hypotheses did not generalize the learned
concept against unseen data efficiently. Moreover, these solutions were on the edge of

Figure 8: Pareto fronts achieved in LED problem. The data set contains 2000 instances
perturbed with a noise ratio $\epsilon = 0.1$. 
Bounding the Effect of Noise in MOLCS

the *bloat* phenomenon because very small (misleading) improvements require a large growth of the hypotheses size. This overfitting situation would disappear if we could force the theoretical and the empirical MAE value to be the same. This constraint would remove the part of the front that appears at the left-hand side of the *rupture point*.

One solution is to bound the evolved Pareto front. The bounds are defined by the area delimited by the intersection of four measures: (1) the theoretical MAE, (2) the empirical MAE, (3) the optimal Pareto front, and (4) the *rupture front*. The *rupture front* is defined by the *rupture point* and the *random guess point*. The *random guess point* is defined by the majority rule that describes solutions like \( F(\bar{x}) = (0.9, 1) \), whereas the *rupture point* is theoretically defined by the hypotheses \( F(\bar{x}) = (\text{MAE}(\epsilon), |\Omega|) = (0.26, 10) \).

In order to validate previous reasonings, we repeated the experiments using four different newly generated data sets. These data sets were generated using four different noise ratios, \( \epsilon = \{0.2, 0.3, 0.4, 0.5\} \). Figures 9 and 10 present the Pareto fronts obtained using these data sets. As we can see, the *rupture points* appear at the point predicted by the theoretical computation of MAE. The Pareto fronts are also correctly fitted by the boundaries early presented. Thus, we can use the MAE to identify a point beyond which the hypotheses become overfitted.

MOLCS-GA has also been applied to some real-world problems showing its efficiency when compared to other LCS and learning systems. Llorà, Goldberg, Traus, and Bernadó (2003) presented a detailed explanation of the obtained results on several real-world problems.

6 Conclusions

This paper analyzed the effect of noisy data sets on a *Pittsburgh-style* learning classifier system (LCS). We focused on a particular LCS (MOLCS-GA) based on multiobjective selection techniques. MOLCS-GA balances accuracy and parsimony explicitly as two different objectives. In order to achieve this goal, MOLCS-GA introduces multiobjective selection techniques in the evolutionary learning process for maintaining this tradeoff across the population. Moreover, this tradeoff can be graphically represented using Pareto fronts which became the basis for the bounding analysis presented.

However, the bounding analysis required some theoretical modeling of the effects of the noise in the data sets. Therefore, we computed a theoretical model of the *minimal achievable error*. This model assumes a crisp learner, and it shows how accuracy is degraded as the result of noise in the training data. The accuracy degradation is the result of the inconsistencies introduced by noise in the antecedents of the instances of the training data set. The MAE model was computed for the *LED* problem, a well-known artificial problem designed for studying the effects of noise in learning algorithms.

We conducted experiments using MOLCS-GA to validate the theoretical assumptions on the MAE model for the *LED* problem. Results reveal several interesting issues. The first one was that the Pareto front of the evolved population of hypotheses could be bound using the MAE model. Moreover, results suggested the presence of a *rupture point* in the evolved front. This point could be predicted theoretically in the *LED* problem using the MAE model. The *rupture point* was not only the spot where the Pareto front presents a significant change in slope, it was the point beyond which overfitted hypotheses were trapped. These overfitted hypotheses improved their accuracy only slightly as the result of the overadaptation to the given data set, harming their generalization capabilities and size. This fact could be easily seen in the graphical representation of the Pareto front. These results were also obtained when solving real-world
Figure 9: Pareto fronts achieved in the 1ed problem. The data sets contain 2000 instances perturbed with noise ratios $\epsilon = \{0.2, 0.3\}$. 
Figure 10: Pareto fronts achieved in the led problem. The data sets contain 2000 instances perturbed with noise ratios $\epsilon = \{0.4, 0.5\}$. 
problems, although only the empirical value of the MAE model could be computed—see (Llorà, Goldberg, Traus, & Bernadó, 2003). Thus, the decision maker now has an extra aid for deciding which hypothesis represents a better generalization of the target concept to be learned. This decision is crucial for solving unseen data efficiently, and the results of this study should be useful in practical settings.

Acknowledgments

Thanks to Ivan Traus and Ester Bernadó for useful discussions related to multiobjective learning classifier systems. This work was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant F49620-00-0163, and by the Technology Research Center (TRECC), a program of the 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. Research funding for this work was also provided by a grant from the National Science Foundation under grant DMI-9908252. 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 National Science Foundation, or the U.S. Government.

References


