Machine Learning, 13, 229–257 (1993)

Competition-Based Induction of Decision Models from Examples

DAVID PERRY GREENE DGlV@ANDREW.CMU.EDU
STEPHEN F. SMITH SFS@ISL1.RI.CMU.EDU
The Robotics Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213

Abstract. Symbolic induction is a promising approach to constructing decision models by extracting regularities from a data set of examples. The predominant type of model is a classification rule (or set of rules) that maps a set of relevant environmental features into specific categories or values. Classifying loan risk based on borrower profiles, consumer choice from purchase data, or supply levels based on operating conditions are all examples of this type of model-building task. Although current inductive approaches, such as ID3 and CN2, perform well on certain problems, their potential is limited by the incremental nature of their search. Genetic algorithms (GA) have shown great promise on complex search domains, and hence suggest a means for overcoming these limitations. However, effective use of genetic search in this context requires a framework that promotes the fundamental model-building objectives of predictive accuracy and model simplicity. In this article we describe COGIN, a GA-based inductive system that exploits the conventions of induction from examples to provide this framework. The novelty of COGIN lies in its use of training set coverage to simultaneously promote competition in various classification niches within the model and constrain overall model complexity. Experimental comparisons with NewID and CN2 provide evidence of the effectiveness of the COGIN framework and the viability of the GA approach.

Keywords. Genetic algorithms, symbolic induction, concept learning

1. Introduction

Many practical decision-making problems involve prediction in complex, ill-understood domains where the principal source of predictive knowledge is a set of examples of observed, prior-domain behaviors. These examples typically relate the values of a set of measured situational features (or variables) to specific outcomes, and the critical prerequisite for decision-making is construction of a model that captures the regularities in these data. Estimation of loan risk based on borrower profiles, prediction of consumer preferences given past purchase data, and forecasting resource supply levels based on operating conditions are all examples of this type of problem. We can identify two desirable properties of a decision model in such contexts: (1) to accurately predict behavior in new situations and (2) to provide descriptive insight into the underlying dynamics of the problem domain by fitting the most concise model given the data.

The task of constructing useful decision models from examples in realistic environments is complicated by problem complexity. Sources of problem complexity include:

- Generating function. Often the underlying behavior comes from an unknown function that is nonlinear and that may have interacting, nonhomogeneous variables.[1]

- Noise. Further complicating the model-generation problem is the possibility of noise in the form of errors in recording the attribute values, executing the example-generating behavior, or classifying the example.
- Scale. The data set may have many examples measured along a large number of dimensions with mixed data types. This gives rise to a combinatorially large space of possible models that must be searched.

The goal of constructing decision models from examples has been approached from different perspectives. Statistical methods (e.g., regression techniques) have proved effective in generating accurate predictive models that are robust in the presence of noise. However, their effectiveness is limited to domains where underlying modeling assumptions (e.g., linear feature interactions) are reasonably satisfied, and even there, statistical models provide little descriptive insight. Alternatively, research in concept learning has emphasized mechanisms for generating symbolic decision models that map classes of situations to specific categories or values. A symbolic representation improves descriptive potential, but it also increases the combinatorics of the model-building problem. Problem size, coupled with noise and non-linear feature interactions, can create problems for many current symbolic induction approaches. Much effort has focused (often successfully) on augmenting the basic search with post-processing and other refinement techniques.

We argue that much of the difficulty in dealing with complexity stems from the bias inherent in the deterministic, stepwise search techniques exploited by these approaches. In contrast to these approaches, genetic algorithms (GAs) have shown great promise in complex search domains. A GA operates in an iterative improvement fashion where search is probabilistically concentrated toward regions of the model representation space that have been found to produce good classification behavior. The properties of genetic search match up well with the above identified characteristics of practical classification problems. However, effective use of genetic search in this context requires a framework that promotes the fundamental model-building objectives of predictive accuracy and model simplicity.

In this article we describe COGIN, a GA-based inductive system that provides such a framework. The design of COGIN is tailored to exploit the special character of problems that require model induction from a set of pre-specified training examples. The result is a fairly significant departure from standard GA-based learning paradigms. The fundamental organizing principle of COGIN is the use of training set coverage as both an explicit constraint on model complexity (size) and a basis for creating a pressure toward appropriate diversity within the model. Survival of a given individual is thus a direct function of its ability to fill a particular classification niche in the evolving model. We present comparative experimental results conducted with COGIN and two other well-known symbolic induction systems that provide evidence of the utility of this approach. More specifically, we demonstrate the leverage provided by genetic search across model-building problems of varying complexity and show performance differential trends that suggest that this leverage can increase with problem complexity.

The remainder of this article is organized as follows. In section 2, we define the symbolic induction problem to be addressed, describe the dominant current approaches to this problem, and identify their potential weaknesses. Section 3 provides background concerning our previous work in applying GAs to symbolic induction problems, tracing the evolutionary

progression to the current COGIN system. Section 4 presents the principal components of COGIN and discusses its relationship to other work in GA-based learning. In section 5, we present the results of initial experimental comparisons of COGIN, NewID, and CN2 across both a range of artificial test problems and an externally provided data set representative of an actual resource supply forecasting application. We conclude in section 6 with a brief discussion of outstanding research issues.

2. Induction of symbolic decision models from examples

2.1. The structure of the problem

In defining the symbolic induction problem of interest, we first assume the existence of a pre-classified data set. "Pre-classified" normally refers to a set of examples in which each example has been assigned a discrete class as the criterion variable. However, this may also refer to the more general case of a data set in which there is a specified dependent variable. This distinguishes the problem of constructing models from a pre-classified data set from the cluster problem, where no dependent variable is identified. More precisely, a pre-classified data set consists of a collection of individual examples or observations where each example is described by a set of features (also referred to as predictor or independent variables) and an associated classification value (also referred to as the criterion or dependent variable). Typically there will be multiple predictor variables of mixed data types and a single criterion variable. The criterion variable is often binary or discrete-valued, but might be continuous. For instance, consider the data set of business loan profiles depicted in figure 1.

An example extracted from this data base might be specified as follows:

industry = electronics, sales = 15M, growth = 15%, debt/equity = 0.6 → loan_risk = low

In this example the predictor variables are industry, sales, growth, and debt/equity ratio. The criterion variable would be "loan-risk," with each example classed as either "high," "medium," or "low" risk—possibly based on the previous judgment by an expert or an actual loan default.

Given a pre-classified data set, the induction task is one of finding a set of feature patterns that partition the elements of the data set into the desired classes and provides a basis for effective classification of future unseen examples expressed in terms of the same predictor

rec_num	industry	sales($K)	growth	debt/equity	loan_risk
...					
0032	electronics	15,000	0.15	0.6	low
0033	oil&gas	137,000	0.09	1.1	medium
0034	restaurant	3,500	0.12	0.45	medium

Figure 1. Data set of business loan profiles.

variables. Such a decision model commonly takes the form of a set of IF → THEN rules; for example,

IF [industry=oil_and_gas] and [debt/equity < 1.2] and [cash_flow=high]
THEN loan_risk ⇒ low
IF [industry=technology] and [debt/equity > 2] and [company_age < 5 yrs]
and [current_growth < 20%]
THEN loan_risk ⇒ high

In this case, each rule consists of a conjunction of specific attribute-value conditions and an associated outcome, and different rules with the same associated outcome are used to express disjunctive sets of classifying conditions. An ordering assumption might also be imposed over the rule set, in which case the specification of condition/outcome pairs depends additionally on an implicit "IF → Then, else" structure, (e.g., if ⟨condition a⟩ then class-x, else if ⟨condition b⟩ ....).

As indicated at the outset of this article, we can identify two dimensions along with the utility, or quality, of a given decision model can be measured: (1) the model's predictive accuracy and (2) the descriptive insight that the model provides. During model construction, or training, predictive ability can be estimated by how accurately the decision model classifies the available set of training examples. Ultimately, the performance of the system will be based on the predictive accuracy of the final model when applied to a holdout sample of examples not seen during training. Insight is difficult to operationalize, but a common surrogate is to use "simplicity"—that is, prefer smaller models, or, in the case of a rule-based model representation, prefer models with fewer rules.

With respect to the size of a model, one additional distinction can be made regarding "ordered" versus "unordered" rule sets. Ordered rule sets are based (and constructed) on an explicit assumption of sequential interpretation (i.e., the rule set has an implicit If-Then-Else structure). Within unordered rule sets, alternatively, each rule fully specifies the circumstances under which it should be applied, and an ordering is imposed only as needed to resolve situations of conflicting matches. Because of the explicit dependence on context in specifying rules, ordered models will require fewer rules than unordered models. However, this additional compactness does not necessarily translate to increased descriptive insight or model comprehensibility. It is difficult to evaluate an individual rule of an ordered model out of context of the rules that precede it, while in the case of an unordered model the rules themselves have been validated independently.

2.2. Prototypical approaches

For the problem of learning decision models from examples under realistic conditions, the most successful and widely applied approaches are those which incrementally generate decision trees and decision lists (MacDonald & Witten, 1989). The prototypical examples of these approaches are ID3 (Quinlan, 1984; Quinlan, 1986) and CN2 (Clark & Niblett, 1989), respectively. Under noise-free conditions, these systems attempt to induce a generalized description that is both complete, by including all positive examples of the concept,

and consistent, by excluding all negative examples. For noisy environments, CN2 relaxes the consistency requirement, while the C4 varient of the ID3 relies on tree pruning and other post-processing techniques.

Both ID3 and CN2 conduct a stepwise search for a solution in their respective problem spaces. In the case of ID3, a decision tree is developed by first identifying the individual feature value condition that maximally separate the positive and negative examples, and then recursively finding the next best split for each branch that leaves one or more examples unclassified. CN2 (an extension of Michalski's AQ algorithm (Michalski & Chilauski, (1980)) alternatively constructs a decision list model, represented as a sequence of rules (called complexes) that denote alternative conjunctive conditions of feature values. CN2 has an *ordered* and *unordered* variant, each of which uses a pruned, general-to-specific beam search to construct complexes incrementally. In the ordered case, the training examples matched by a generated complex are removed from further consideration; in the unordered case, all examples are considered during generation of each complex.

Reliance on incremental search methods enables systems like ID3 and CN2 to effectively manage the combinatorics of the model inference problem. At the same time, the utility of the decision models generated using these methods depends on the nature of the modeling problem. Stepwise search methods assume that the problem features are independent, additive, and non-epistatic[2] in their impact on decisions. In more complex problem domains, where these assumptions are not met (e.g., noisy data, feature interactions), the best path at each step can lead the search astray. The search procedure is locally optimal, but it is not necessarily globally optimal.

3. Induction of decision models using genetic algorithms

In contrast to the deterministic, stepwise search mechanisms described above, the GA is a probabilistic, competition-based search technique based on the concept of adaptive efficiency in natural organisms (Holland, 1975). A GA maintains a collection of candidate solution structures (the population) and proceeds by repeatedly selecting structures (individuals) from the current population according to relative performance (fitness) and applying idealized reproductive operators (principally recombination) to produce new structures (offspring) for insertion into the population and testing. GAs gain their power from an ability to implicitly exploit performance information about the exponentially large number of solution "building blocks" (e.g., rules, rule clauses) present in the structures composing the current population in order to focus the generation of new candidate solutions. Specific building blocks are propagated through the population over time in direct proportion to the observed performance of the structures containing them. From a sampling perspective, this implicit parallelism leads to increasing concentration of trials to those regions of the search space observed to be most profitable. The potential of genetic search has been demonstrated in a variety of complex learning contexts (Booker, 1982; Goldberg, 1985; Grefenstette, 1990; Smith, 1983; Wilson, 1987). By their iterative and probabilistic nature, GAs are relatively insensitive to both noise and the manner in which generalizing data are presented. The reader is referred to Goldberg (1989) for an introduction to the theory and practice of GAs.

3.1. Basic choices

Approaches to GA-based inductive learning can be categorized along two dimensions, corresponding to two principal design decisions that must be taken. The first concerns the representation of the decision model to be learned. Given the simple syntactic character of the basic recombination operator (i.e., crossover), its straightforward application to complex model representations is problematic from the standpoint of consistently generating new rule structures that are semantically meaningful (which is fundamental to furthering the search). Historically and still predominantly, this representation problem has been solved through the use of simple, fixed-length condition/action rule formats defined with respect to an ordered, binary encoding of situational features and possible actions. Such rule-based model representations permit direct application of standard crossover operators with assurance that all generated offspring will be viable rule structures. Some more recent efforts (e.g., Grefenstette, 1991; Koza, 1991) have alternatively emphasized higher-level symbolic rule representations and concentrated on the design of knowledge recombination and mutation operators for manipulating these representations. Our approach adopts the more commonly used fixed-format approach to representing decision models.

Along a second dimension, learning of rule models with GAs has been considered within two basic paradigms: the so-called Classifier Systems or "Michigan" approach (e.g., Holland, 1986), and the so-called "Pitt" approach (e.g., Smith, 1983). At an operational level, these two paradigms can be distinguished by the role given to the GA in the learning process. The Michigan approach assumes a population of individual rules (classifiers) that collectively constitute the current decision model. The GA is given responsibility for generating new rules; the integration of rules into a coherent model is handled by other mechanisms. The Pitt approach, in contrast, is more directly based on the standard GA-based optimization framework in which each individual in the population represents a complete candidate model (i.e., a complete set of rules). Here the scope of the GA is extended to include rule integration. Under the Michigan approach, the GA is typically applied in a controlled fashion, reflecting a bias toward incremental, on-line model development and refinement. A small subset of rules (individuals in the population) are periodically selected for reproduction based on measures of prior rule performance in the task domain, and existing low-performance rules are replaced by the newly generated offspring. Pitt approaches reflect a more off-line training perspective, and have generally relied much more extensively on genetic search. Larger percentages of the population are usually selected (in this case, based on measures of overall model performance), recombined, and replaced on each cycle. Michigan approaches have assumed decision models with a fixed number of rules (the size of the population) whereas Pitt approaches have typically allowed individuals with variable numbers of rules and thus have allowed the final model size to be determined within the search.

The approach to learning rule-based decision models advocated in this article represents a unique synthesis of aspects of both paradigms. Specifically, it combines the off-line and rule integration perspectives of the Pitt approach with the "population-is-model" assumption of the Michigan approach. Our framework has itself "evolved" as a result of our prior experiences in attacking learning from examples problems. Thus, we first summarize the progression of systems that led us to our approach and highlight some of the difficulties that were encountered.

3.2. ADAM

ADAM (Adapative Decision Acquisition Model) was an initial approach to using a GA for inductive classification (Greene, 1987; Greene & Smith, 1987). The problem addressed in this work was a binary discrimination task: to produce a model that reflects consumer buying preference, given a set of examples of past consumer choices (to buy or not buy) in the context of a specific set of product features (e.g., size, color, cost, etc.). One important issue in consumer choice modeling (in addition to producing a model that predicts well) is recovery of the underlying functional form of the consumers' decision model (e.g., is the decision dominated by the value of a specific product feature, or are tradeoffs being made among the values of several feature values?). Since such information is of obvious interest to marketing research, consumer choice modeling is a problem well suited for symbolic induction.

The ADAM system represented a fairly standard adaptation of the Pitt approach to rule learning, with the exception that the special structure of the binary classification problem was exploited to simplify the representation of candidate models. Within ADAM, a model (or individual in the population) was represented as a set of disjunctive clauses that collectively specify the circumstances under which a positive classification (e.g., "buy") should be made. Each clause expressed a particular conjunctive pattern over the entire feature set using a variant of a pattern specification language commonly employed in GA learning research. Specifically, clauses were encoded as fixed-length strings over the alphabet {0, 1, #}, (where, as usual, # designates "don't care") to be matched against a correspondingly ordered binary encoding of the product feature values associated with any given example. For example, in the case of a problem involving three boolean feature detectors d_1, d_2, d_3, the condition *1#0* would be interpreted as

If d_1 *AND* (*NOT* d_3) *Then buy*

If the problem alternatively involved a single eight-valued feature detector d_1, then the interpretation of the same condition might be

If (d_1 = *value*4) *OR* (d_1 = *value*6) *Then buy*

Thus, outcomes were not explicitly represented. If a matching clause was found during application of the model to a given example, then the model's response was interpreted as positive; if no clause was found to match, the response was a negative classification. A similar perspective on model representation, which can be seen equivalently as a collection of "implicit-positive" classification rules, has been exploited by (DeJong & Spears, 1991) in more recent work in GA-based concept learning.

Within the training phase search conducted by ADAM, candidate models were exposed to distinct, overlapping subsets of the overall set of training examples on each generation. This overlapping sample technique was designed to encourage models with appropriately general clauses (since overly specific and overly general models failed to respond properly on successive subsets over time) as well as to provide a computationally efficient evaluation step. The overall fitness of a given model was defined as a weighted sum of predictive

accuracy (percentage of correct responses) and model simplicity (number of clauses). Given the sensitivity of these weights to the unknown form of the data-generating function, a simple mechanism for dynamically shifting this bias as the search proceeded was employed. Specifically, increased weight was shifted to model simplicity if and when the number of clauses in the best individual in the population fell below the average number of clauses in the individuals of the randomly generated initial population. A single-point crossover operator, parameterized to operate at either the inter-clause or intro-clause level according to a specified probability, was used as the principal means of generating new candidate models for testing: a clause was randomly selected from each selected parent, the two structures were aligned, and the structures were then crossed at a randomly selected point (at the appropriate level) common to both. The probability of crossing within a clause or at clause boundaries was varied according to a predefined schedule that encouraged more intra-clause exchange early on in the search and more inter-clause manipulation at later stages. The search was terminated after a pre-specified number of generations, and the best model found was then evaluated with respect to a separate holdout set of examples.

In a comparison of symbolic and statistical approaches applied to consumer choice data sets, the ADAM system was found to have superior performance versus a symbolic induction system called CLS (Currim, 1986) (a variant of the ID3 decision tree-building approach) and equivalent performance versus a statistical logit model (the current standard in practice). This comparison was carried out in a factorial design encompassing different noise levels, data-set sizes and generative models. Results demonstrated that ADAM was able to construct decision models that not only predicted well but also captured the functional form of the choice model.

One of the keys to ADAM's success was the strong coupling between its specialized, two-class model representation, which permitted a focused search for the conditions relevant to positive classification only, and the fitness function, which effectively reflected this "optimization" goal and required no finer-level search bias. The cost of these assumptions, on the other hand, was their inability to extend to multi-class problems.

3.3. GARGLE

To explore the application of GAs in more complex, multi-class and continuous valued modeling domains, an experimental testbed called GARGLE (GA-Based Rule Generation Learning Environment) was created (Greene, 1992). While retaining the Pitt approach of manipulating a population of a complete decision models, a new model representation was adopted within GARGLE that replaced ADAM's multi-clause model with a set of ⟨single-clause outcome⟩ rules. This model representation is similar to representations typically exploited in classifier system research (e.g., Wilson, 1987). However, in contrast to the approaches taken in this work, rule outcomes were not explicitly manipulated by the GA in GARGLE. Exploiting the presence of the data set of training examples, a newly created rule was instead assigned an outcome based on the predominant outcome of the set of examples that its condition matches. This approach, referred to as *ex-post assignment*, is illustrated in figure 2.

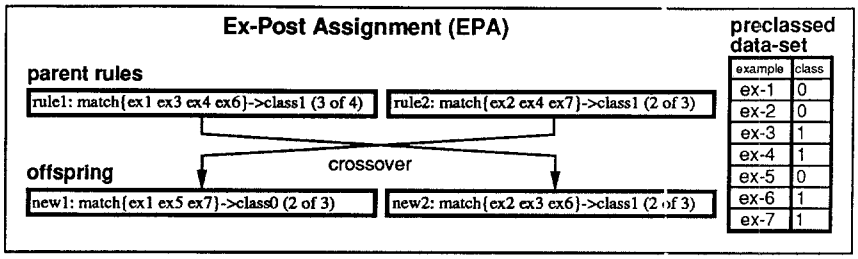

Figure 2. Assigning an outcome to a newly created rule.

Ex-post assignment offers advantages in the context of learning decision models from examples:

- it reduces and simplifies the size of the space to be searched;
- it creates a useful search bias, maximizing the potential utility of any condition that is generated during the search; and
- it provides a common basis for addressing continuous valued as well as discrete valued multi-class decision problems. In the former case, for example, the mean value of the outcomes in the matched example set can serve as the rule's outcome, with the variance providing a measure of rule strength.

While providing the flexibility to address this broader class of decision models, movement to a multi-outcome representation introduced several thorny issues that could be effectively avoided in the simpler two-class approach (many of which are not unknown to research in GA-based learning (Wilson, 1989)) All of these issues related to the problem of creating a search bias that adequately promoted coverage of the training examples while constraining the size of rule sets that were generated. In specifying an appropriate conflict resolution strategy, there was no clearly compatible resolution metric (e.g., accuracy, total matches, specificity, etc.) or combination of metrics from the standpoint of global model-building objectives, nor was it obvious how best to apply resolution criteria. A variety of dominance and consensus strategies (choosing the outcome based on the "best" matching rule or the collective influence of all matching rules, respectively) were tested with mixed results. For example, consensus strategies, in the absence of other biases, tended to promote model growth. Establishment of an appropriate bias toward smaller models within the global fitness measure also proved problematic. While effective in ADAM, the simple weighted use of predictive accuracy and model size as a measure of model fitness invariably led to either unacceptable model growth (too little size bias) or an inability to maintain sufficient diversity to get good coverage of the training set (too much size bias). To compensate, more sophisticated, multi-point crossover operators were explored, which utilized rule coverage similarity measures to bias crossover point selection. While some system configurations did produce fairly reasonable search performance, and several promising mechanisms were developed, the continuing difficulty of simultaneously managing model complexity (size) and maintaining sufficient diversity for good performance within the Pitt approach finally led us to seek an alternative search framework.

4. Coverage-based genetic induction

By starting from the problem and exploiting its inherent structure, a different system organization emerges. This perspective led to the development of COGIN (COverage-based Genetic INduction), a system for symbolic model induction from a set of pre-specified examples. The fundamental organizing principal of the COGIN search framework is the concept of a *coverage-based constraint*. The basic idea applied to modeling a set of examples is to think of the examples as corresponding to niches in an ecology, where the exact number of niches is unknown but presumed to be less than the example set size (i.e., one or more examples per niche). Treating the population as a model and each rule in the model as an individual representing a possible niche, we impose a specific bias in preferring the ecological model, which classifies accurately and requires the fewest niches. This implies a system that will simultaneously search for the ideal number of niches as well as the best individual within each niche.

Consider first the problem of identifying the best individual. To get the highest accuracy with the smallest model, the system needs individuals with high discriminability, that is, individuals that differentiate many examples correctly. By ordering on discriminability, we can identify the "best" individuals. To identify the fewest number of niches, we can move sequentially through this ordering and "allocate" correctly matched examples to each rule along the way. If there are no examples available to a rule when its turn to receive examples comes, it has no unique niche and does not survive. When all the examples have been "consumed," the ecology supports no more rules. The resultant model provides a minimal number of niches with the best available individual designating each niche. Evolution suggests that survival of the fittest will occur where a limited environment causes competitive pressure—it is this coverage-based constraint that provides just such an environment.

4.1. Overview of COGIN approach

Figure 3 depicts the basic search cycle carried out by COGIN to construct a decision model from a pre-specified set of training examples. The system's current model at any point

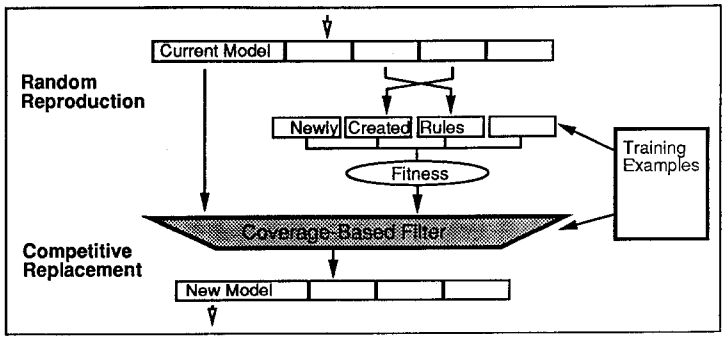

Figure 3. The COGIN search cycle.

during the search is represented as a population of fixed length rules. However, in contrast to typical classifier system approaches, the population *size* (i.e., the number of rules in the model) will vary from cycle to cycle as a function of the coverage constraint that is applied. At the start of each cycle, a percentage of the rules constituting the current model are randomly paired and recombined to produce a set of new candidate rules. Each of these new rules is assigned a fitness measure, based on discriminability with respect to the training set, and then pooled with the original rules of the current model for competitive configuration of a new model. The heart of the cycle is the competition step, which determines the rules that survive to the next generation. This is carried out through application of a *coverage-based filter*, which "allocates" training examples to rules that correctly match them in fitness order. The output of the filter, which constitutes the new decision model, is the set of rules receiving at least one example, and all other candidate rules are discarded. After the new model has been determined, its predictive accuracy relative to the training set is evaluated. If the new model yields a higher predictive accuracy than the best model evaluated thus far in the search, or performs at the same level and contains fewer rules than the previous best model, then a copy is retained off-line as the new best for future comparison and the cycle repeats. The overall search process terminates after a pre-specified number of iterations, and the best model produced during the search is taken as the final model.

4.1.1. An example

Figure 4 provides a representative example of the dynamics of this search process in the context of a two-class problem, assuming a training set of seven examples, with examples {1, 2, and 5} of class 0 and examples {3, 4, 6, and 7} of class 1. Notationally, each rule in the figure is depicted by number, followed by the set of matching examples, its action, and its fitness. For instance, [r1: {1 3 4 6 } → 1(.089)] indicates that rule r1 matches examples 1, 3, 4, and 6, assigns class 1, and has a fitness of .089. For our purposes here, entropy is used as a fitness measure; the formulation is provided in Quinlan (1986). (The specific fitness measure employed in COGIN is described later in this section.) At generation N the current model consists of four rules {r1, r2, r3, r4} that correctly predict 6 of the 7 examples (86%) in the training set and have an average rule fitness of .070. Three new rules are created in this cycle (r5, r6, r7), and the "allocation" of training examples within the filter (depicted) leaves no examples for rules r2, r3, or r6. These rules are discarded, and the remaining rules (r7, r5, r1, r4) become the generation $N + 1$ model. Notice that rule r4 survives despite its low fitness, by being the only rule that matched example 7 *correctly*. Notice, too, that this model has the same number of rules as the generation N model but actually a lower predictive accuracy. At the same time, the average rule fitness has increased, suggesting the emergence of better model components. After generating new rules and applying the filter, at generation $N + 2$ the new model is in fact seen to correctly classify all training set examples. Although this example was created for illustrative purposes, this behavior is quite representative of actual system performance. We will return to the issue of search dynamics after summarizing the principal elements of the COGIN inductive framework.

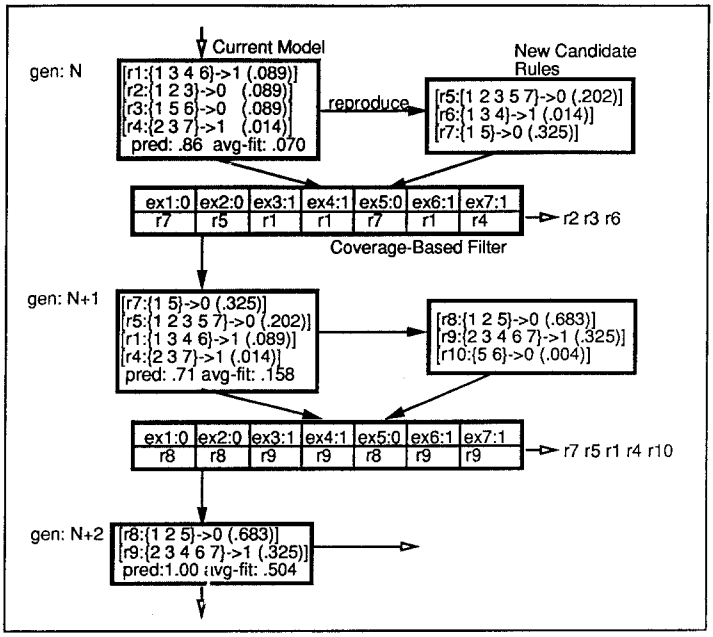

Figure 4. An example of COGIN's search procedure.

4.1.2. Model representation and interpretation

As previously stated, decision models are represented within COGIN as a set of fixed-length rules (the number of which is determined during the search process). The COGIN rule representation is a simple extension of the rule representation employed in GARGLE. Each rule similarly specifies a single conjunctive pattern in the ternary alphabet {0, 1, #} over an ordered binary encoding of the feature set. The extension is the additional association of a negation bit with each feature subpattern, which, if set, designates the complement of the specified value set (the negation bit is ignored in the case where the feature pattern matches all possible values, i.e., "#. . #"). The system is parameterized to run with or without this extension. Thus, for example, if we assumed use of the negation extension in the case of a four-class problem with ordered feature detectors d_1 and d_2 taking on eight and four possible values, respectively, the COGIN rule

$$0\ 11\#\ 1\ 01 \rightarrow 11$$

would be interpreted as

$$(d_1 = value6\ OR\ value7)\ AND\ (NOT\ (d_2 = value1))\ Then\ class3$$

During the application of a model to classify a given example, rule fitness (discussed below) is used as a basis for conflict resolution. In situations where no matching rules are found (i.e., in classifying examples not seen during training), a default classification is made based on the majority class of the training set examples.

4.1.3. Initializing the search

An initial model is constructed by randomly generating rules (using 50% don't-care bits) until all examples in the training set have been correctly matched. Thus, the search is initialized with a random model of the domain in which all niches are covered. The competitive replacement step of the search cycle will enforce this constraint throughout the search, ensuring that no niche description will be lost until replaced by better niche description (i.e., a superior rule).

4.1.4. Generating new candidate rules

New candidate rules are generated on any given cycle of the search by randomly recombining rules in the current model (population). The current rule set is duplicated, and a fixed percentage of parent rules are randomly selected from each copy and paired for reproduction—since all offspring and parents are retained, it is not clear that there are any advantages in using less than 100%. Rules are chosen for participation without replacement, and pairings are made to ensure distinct parents. For each set of paired rules, a single crossover point is randomly selected among the bits in the condition strings (i.e., the crossover point may fall within or between feature patterns), and the condition segments to the right of the crossover point are swapped to produce two new rule condition patterns.[3] An outcome is assigned to each newly created rule using the *ex-post assignment* scheme discussed in section 3. That is, the set of training examples matched by each new rule condition is collected, and the majority outcome of this match set is assigned (choosing randomly in the case of a tie). This match set is recorded with the new rule for subsequent use in assigning fitness.

4.1.5. Assigning rule fitness

As indicated above, the overall goal of producing the smallest possible model with the best possible predictive accuracy requires rules with high discriminability. One well-supported measure of discriminability is the information gain or mutual information (Quinlan, 1986). The information gain tells us how many bits of information about the correct classifications of the examples are provided by a given rule. Consider a set S of examples. Let p_i be the proportion of examples in S belonging to class i. Then the *entropy*, $H(S)$, is generally defined as $-\Sigma_{i=1}^{n} (p_i \, ln(p_i))$. Now consider a rule R. Let *Matched* be the set of examples that R matches, and *Unmatched* be the set of examples that R does not match. With these definitions, the information gain of rule R is

$$Info(R) = H(S) - \frac{[|Matched|\ H(Matched) + |Unmatched|\ H(Unmatched)]}{|S|}$$

Entropy-based measures have been frequently utilized as a search bias in symbolic learning contexts (e.g., serving as the basis for tree splitting in ID3). *Info*(*R*) is similarly adopted as a basic measure of rule fitness in COGIN.

However, unlike our earlier simplified example (figure 4), the fitness measure assigned to a rule within COGIN is not based solely on its entropy value. In practice, the use of entropy alone creates a bias toward quantity in the sense that it will favor a rule with many matches as long as the number of incorrect classifications is proportionately small, and this characteristic can adversely affect the model-building goal of predictive accuracy. To mediate this bias toward overly general rules, we formulate rule fitness in COGIN as *lexicographically-screened entropy*, defined as

$$Fitness(R) = Lex(R) + Info(R)$$

where *Info*(*R*) was described previously and takes on values between 0 and 1, and *Lex*(*R*) is an integer-valued screening function that measures classification accuracy. A user-specified "interval" parameter i is used to establish misclassification error levels (or bins) that are considered equally fit, and *Lex*(*R*) is defined as

$$Lex(R) = \left\lceil \frac{N}{i} \right\rceil - \left\lceil \frac{Misclass(R)}{i} \right\rceil$$

where N is the total number of examples in the training set, and *Misclass*(*R*) is the number of misclassifications made by R. For example, if $i = 2$, then the fitness scores assigned to rules r1 through r7 in our previous example of figure 4 (where $N = 7$) are as shown in table 1. The principal effect of the *Lex* term in this case is to lower the competitive priority of r5, making it less likely to survive into the next generation model.

Table 1. Fitness assignment under lexicographically screened entropy.

Rule	Info(R)	Misclass(R)	Fitness(R)
r7: {1 5} → 0	.325	0	3.325
r1: {1 3 4 6} → 1	.089	1	3.089
r2: {1 2 3} → 0	.089	1	3.089
r3: {1 5 6} → 0	.089	1	3.089
r4: {2 3 7} → 1	.014	1	3.014
r6: {1 3 4} → 1	.014	1	3.014
r5: {1 2 3 5 7} → 0	.202	2	2.202

There are several more general points to make about the approach to fitness assignment represented by the above calculation.

- The goal of fitness assignment in COGIN is to establish a rank ordering of candidates; for example, the fact that *fitness*($R6$) > *fitness*($R5$) in table 1 is important, not the distance between *fitness*($R6$) and *fitness*($R5$).
- The approach taken here deviates from the approach previously taken in ADAM and GARGLE, where weighted compensatory fitness functions were utilized. The above described "lexicographic" approach does not presume a smooth tradeoff between contributing fitness measures, but instead is structured so that the influence of *Lex*(R) will dominate (or impose constraints on) the influence of *Info*(R). This choice was made to allow the effects of search bias to be more easily controlled and understood. Within *Lex*(R), increasing values of i define increasingly broader (looser) constraints on the influence of *Info*(R).
- A number of alternative measures have been proposed as a means of counterbalancing the undesirable aspects of a search bias based solely on entropy (e.g., statistical tests of significance). We have chosen misclassification error for purposes of simplicity in our current implementation, and make no claims that this is the best choice.

4.1.6. Competititve replacement

While reproduction is random, replacement is not. Newly created rules compete with the original rules in the current model for entry into the population. This is accomplished by a filtering process that utilizes the set of training examples to impose a coverage-based constraint on the size of the new model that is being configured for the next generation of the search. The filtering process proceeds as follows. The entire pool of candidate rules is ordered by fitness. Next, the set of training examples is passed over this list, removing those examples correctly matched by the current rule from further consideration at each step. The subset of rules remaining when the set of training examples becomes empty are eliminated, and the new model is established. Given the use of entropy as a determinant of fitness, the model's rules naturally form a hierarchy with generalist rules dominating and more specific "exception" rules filling in remaining gaps. This suggests an interesting parallel to the "default hierarchies" sought in classifier system research.

4.2. Search bias

At first glance, it would appear that the commitment to random selection of rules for reproduction would disrupt the ability of the search to efficiently allocate trials to high-performance building blocks as suggested by the classical GA's theoretical foundation. However, this does not seem to be the case. In terms of the classical select-reproduce-evaluate GA search framework, we can see application of the coverage-based filter as a form of rank selection (Baker, 1985), in this case with a dynamically determined cutoff on the individuals allowed to participate in influencing the next generation. Selection, in the classical

sense, is going on with respect to the larger pool of current and newly created rules on any given cycle. In Eshelman (1991), a very similar search paradigm is defined and analyzed in the context of function optimization and is shown to retain the fundamental property of implicit parallelism.

Moreover, it is not clear what further useful search bias could be introduced to guide reproduction, since rules exist in the model because they fill different niches. One possibility that could be explored is a pairing strategy based on overlap of training examples matched. However, given that replacement will provide competitive pressure, reproduction should therefore emphasize diversity, and this is already promoted through random pairing. One useful bias with respect to uncovering or better filling specific niches is the use of ex-post assignment. As mentioned earlier, this rule-building heuristic maximizes the potential of any rule condition that is generated.

Figure 5 gives some insight the dynamics of COGIN's search. It tracks four values over time, averaged over five runs of one of the conjunctive four-class problem considered in the experimental study described in section 5: the accuracy on the training set, the accuracy on the holdout set (of course unavailable to the system during the search), the average fitness of the rules in the current model, and the number of rules in the current model (the curves of the last two values are scaled to the maximum value seen). We can see that 100% accuracy on the training set is achieved very early on in the search, with the size of the current model generally increasing as well (a function of the dominant accuracy bias). At this point, the entropy bias takes over and focuses the search toward more general rules and correspondingly smaller models (rule sets).

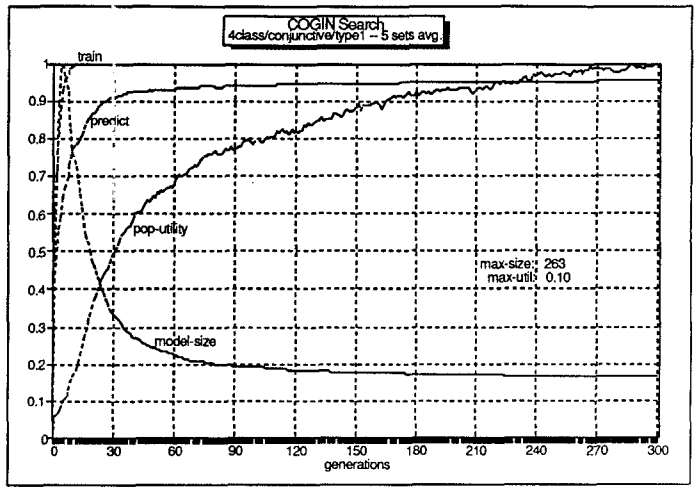

Figure 5. Performance curves for simple conjunctive four-class problems.

4.3. Relationship to other approaches

The use of coverage as an explicit constraint on model (population) size represents a fairly significant departure from previous approaches to the problem of maintaining the diversity in model components (rules) necessary for robust decision-making. Research with Michigan-style classifier systems (the closest match to the COGIN framework) has typically proceeded from the base assumption of a fixed model size, large enough to accommodate the range of decision-making circumstances that the problem solver must face. The generation of higher-performance rules is treated as a separable issue from niche building and is typically handled by separate mechanisms. Use of various measures of individual rule strength to focus selection of rules for recombination (the typical approach) promotes improvement of model performance in niches currently filled by the model. However, this selection bias does little to address the problem of uncovering new niches. Several additional mechanisms have been utilized to provide this latter search bias. The use of partial matching techniques during model interpretation (Booker, 1982) provides the opportunity to accumulate rule strength with respect to a wider range of situations than the rule explicitly implies and thus to focus search toward appropriate expansion of model coverage. A more explicit approach to building new niches has been the use of so-called "triggered operators" (Wilson, 1987; Robertson & Riolo, 1988; Booker, 1989), specialized procedures that are invoked when a non-matched example (or situation) is encountered and that act to construct a new rule through some amount of generalization from this example. Despite the addition of these mechanisms, however, selection according to a rule performance still provides only indirect pressure toward niche building, since rules paired for recombination may be playing distinctly different roles in the model. Recognizing this problem Booker (1989) has advocated the use of restricted mating policies, where only rules that are simultaneously active (i.e., simultaneously match a given situation) during model interpretation are allowed to be paired for recombination, and experimental results have demonstrated the desired effect in the context of concept learning.

What these approaches fail to emphasize, in contrast to the approach taken by COGIN, is the relationship between model complexity and the underlying behavior or decision process that is being modeled. There is an explicit global constraint (the population size), but no finer pressure to cover niches in the most economical fashion. Admittedly, this may be reasonable in the context of developing intelligent agents (where available "brain cells" are the principal constraint). However, as we have previously observed, model simplicity is an important concern in the context of constructing comprehensible decision models from data. Research with Pitt-style rule learning approaches, which has typically manipulated variable-length models, has had to face this issue out of computational necessity. The most common approach has been to simply introduce a bias for smaller models in the fitness function used to estimate the utility of candidate solutions, typically with an underlying global constraint on model complexity (i.e., a maximum allowable number of rules (Smith, 1980). Our prior research with ADAM (Greene, 1987) has demonstrated the sufficiency of this approach relative to binary classification problems. However, as previously discussed in section 3, the problem of diversity and the insufficiency of a size bias to properly address the model performance/complexity tradeoff really only emerge in multi-class contexts. One approach taken to promote diversity within the Pitt approach has been the use of mutli-

valued fitness functions (Schaffer, 1984), designed to estimate model performance with respect to each individual class. Other approaches (e.g., Grefenstette, 1990; our explorations with GARGLE) have resorted to similar types of mechanisms as those investigated in classifier system research (e.g., triggered operators, incorporation of individual rule discriminability information into recombination operators). However, neither of these types of extensions directly impact the object of simpler models.

In contrast to this work, COGIN's use of coverage as a global constraint on model complexity, applied in conjunction with a discriminability-based selection/survival bias, provides a conceptual framework that establishes exactly the desired search dynamics for effective induction from examples. Model complexity expands or contracts as necessary to accommodate the data, while fitness based on discriminability of individual rules biases search toward a model of appropriate complexity. Taking the analogy from nature, individuals in the ecology are allowed to intermingle, creating wide variations; but their environment is constrained, and survival depends on being more fit than competitors for the same resource niche.

5. Experimental study

To understand the performance of COGIN in constructing decision models from examples, a two-phase experimental study was designed and carried out. During the first phase, experiments were run to determine appropriate settings for two parameters in COGIN that impact the necessary tradeoff between model predictive accuracy and model size. During the second phase, the results obtained were used as a basis for conducting a comparative performance analysis of COGIN and two symbolic induction systems—CN2 and NewID (a current version of the ID3 methodology developed at the Turing Institute)—across a range of model induction problems. Each problem was represented as a data set of 800 examples, with each example having eight to ten four-valued features and either a two-class or four-class dependent variable (outcome). The data sets were randomly split into 400 training examples (the training set) and 400 prediction examples (the holdout set). To provide a basis for contrasting performance across problems of different complexity, data sets were varied along two dimensions of complexity: noise in the training examples, and the data-generating function.

Along the noise dimension, two levels were considered for each data set: "clean" (i.e., 0%) and "noisy," with 20% random distortion among the features in the training data. Because we were interested in how well each system would recover the true underlying model in the presence of noise (as opposed to evaluating performance on recovery of the "related" underlying model resulting from introduction of noise), the holdout samples were kept clean. With respect to the data-generating function, we considered two types: conjunctive (generally lower complexity) and compensatory (higher complexity). Conjunctive problems are defined by a lexicographic decision rule (e.g., consider feature-X first, then feature-Y, and so on); thus the classification may be determined by a single feature or specific sequence. Compensatory problems are weighted summations of the relevant features; thus a poor value on one feature can be "compensated" by a better value on another. Linear compensatory problems are the raison d'etre for statistical methods but can be problematic

for symbolic induction because of the number of tradeoffs that are possible. Conjunctive problems, on the other hand, are difficult for regression approaches but quite consistent with stepwise search. Through experimentation, several conjunctive and compensatory functions were simulated to create instances of varying complexity within both types: from simple two-class to multiple-interaction four-class conjunctive and simpler two-class to multiple feature four-class compensatory. For each data set a copy was then made with and without noise.

As an additional problem, a data set supplied by a sponsoring agency, the U.S. Army Human Engineering Laboratory (HEL), was also used. This HEL data set was of interest because the underlying generating function was based on an important resource resupply problem that had proven difficult in practice for other forecasting methods. It exhibited both conjunctive and compensatory problem characteristics. While reflecting a primarily conjunctive decision process, the training examples contained real-valued situational features, resulting in a pseudo-compensatory behavior. The HEL data set consisted of approximately 1600 examples with seven features of multiple values and a continuous valued outcome, which, for purposes of the experiments described below, was discretized into four classes. Two additional data sets were then generated from this base data set: one with 15% noise in the features only and the other with 15% noise among the discretized outcomes.

5.1. Phase 1: Analysis of COGIN performance characteristics

In the first phase, we evaluated two specific parameters in COGIN expected to be particularly relevant to the goal of generating compact, high-performance models:

- the use of a negation within the individual feature patterns of a rule, and
- the size of the error intervals used in the fitness assignment

The association of a negation bit with each feature pattern provides a richer framework for expressing feature interactions, and thus suggests the possibility of simpler rule models. The error interval in the fitness function acts to tolerate more or less misclassification when selecting the covering set of rules to constitute the population for the next generation. Since it was expected that overfitting might occur under some problem conditions, error interval levels of 1, 2, and 3 were evaluated in combination with the presence or absence of negation in the representation. The results of multiple experiments over several of the above problems indicated a fair amount of insensitivity to these parameter settings. As expected, the use of negation resulted in somewhat shorter rules. Larger error intervals yielded slightly better predictive performance on noisy data, while the smaller intervals performed better in most other cases. It appears that the width of the first interval has the strongest potential influence on the performance. For purposes of assessing the comparative performance of COGIN, CN2, and NewID in phase 2, we selected the parameter values that were most consistent across all problem conditions: the use of a negation bit and a error interval level of 1.

5.2. Phase 2: Comparative performance analysis

The goal of the second set of experiments was to determine the effectiveness of COGIN on problems of varying complexity relative to inductive approaches rooted in a stepwise

search framework. CN2 provides two search variants, one designed to construct an ordered rule set and the other operating under the assumption of an unordered set of rules. Since, in either case, CN2 searches a similar space to that searched by COGIN (albeit differently), it provides an interesting opportunity to evaluate how performance varies among these three search orientations, and both versions were included in the evaluation. NewID constructs ordered, decision tree models in a stepwise fashion, and was included in the evaluation to provide an additional point of comparison. Both CN2 and NewID were used with their respective default settings. Because of the possible influence of beam size on CN2, several settings were evaluated. Increasing the beam size had a negligible and, in a number of cases, detrimental effect on performance (possibly due to overfitting). It was decided that the default size of 5 was probably the default for a reason, and hence it was retained. No post-processing of generated models was done for any of the three systems.

As a first-order indicator of effectiveness, the absolute performance levels attained by COGIN, CN2 and NewID relative to each of the problem categories described above can be measured, where performance is defined as predictive accuracy (i.e., the percentage of holdout examples correctly classified.) Our basic conjecture was that GA-based search could effectively address conditions of noise and problem complexity that inhibit stepwise search performance. Since CN2 and NewID have already proven effective under simple, noise-free conditions, our expectation was that as problems became more complex COGIN should begin to outperform them. The range of problem sets was intended to find the conditions where this transition would occur.

Hence, the first hypothesis to be tested was the following:

> Hypothesis 1: COGIN should produce decision models with superior predictive accuracy to those produced by CN2 and NewID on problems with greater complexity.

This hypothesis implies that COGIN should do better on at least some of the problems associated with greater complexity. We might also expect that COGIN should perform better (on a relative basis) as the problem became more complex. Therefore, additional insight into the effectiveness of COGIN can be gained by measuring the trend in the performance differential between systems over increasingly more complex problem categories. We would generally expect absolute performance levels to decline with noise and increaisng function complexity (regardless of approach). However, given the apparent greater susceptibility of incremental approaches to interacting features in more complex problems, it was anticipated that, with increasing problem complexity, their relative performance would decrease in comparison to COGIN's.

> Hypothesis 2: The relative performance differential between COGIN and CN2 and NewID would become increasingly larger as complexity increases.

As noted in section 2, ordered decision models will tend to be smaller than unordered models because of the inherent if-then-else relationship (which may reduce comprehensibility). While unordered models may rely on an ordered interpretation for resolving conflicts, since their construction does not discourage overlapping coverage of examples, they tend to be larger. Therefore, from the standpoint of the size of models generated by CN2, the

most direct comparison to be made is between COGIN and unordered CN2. Although ID3 forms decision trees, they can be roughly interpreted as unordered rule models based on the number of nodes.

In conducting the analysis, 13 problems were defined (each pre-labelled for later reference):

1. [cp0.2cl] compensatory, 2-class (involving 8 features), no noise
2. [cp1.4cl] compensatory, 4-class (involving 6 of 8 features with 2 irrelevant features), no noise
3. [cp1n.4cl] compensatory, 4-class (involving 6 of 8 features), 20% noise
4. [cp2.4cl] compensatory, 4-class (involving 8 features), no noise
5. [cp2n.4cl] compensatory, 4-class (involving 8 features), 20% noise
6. [cj0.2cl] conjunctive, 2-class (involving 8 features), no noise
7. [cj1.4cl] conjunctive, 4-class (involving 8 features), no noise
8. [cj1n.4cl] conjunctive, 4-class (involving 8 features), 20% noise
9. [cj2.4cl] conjunctive, 4-class (involving 10 features), no noise
10. [cj2n.4cl] conjunctive, 4-class (involving 10 features), 20% noise
11. [hel0] hel, 4-class, 8 features, no noise
12. [hel0n1] hel, 4-class, 8 features, 15% noise in features
13. [hel0n2] hel, 4-class, 8 features, 15% noise in classification

Since problems 1 to 10 were represented by simulated data, five instances of each problem were created yielding 50 simulated data sets plus the three hel sets for a total of 53 experiments. Because both CN2 and NewID are deterministic procedures, each was run once per data set using their standard default settings (for CN2, once each for its ordered and unordered search variants). COGIN, using negation and the one-error interval setting, was run four times on each data set using different starting seeds The other principal parameter, the crossover rate, was set at 0.6. Each run was terminated after 300 generations, and the best rule found (based on training performance) was selected and its predictive performance calculated. COGIN's performance on each data set was the average of the multiple runs.[4] Performance on all 53 data sets for all four systems (COGIN, ordered CN2, unordered CN2, NewID) were collected, and composite performances were aggregated for each system based on the 13 problems, the three generation-function types (conjunctive, compensatory, and hel), the two noise levels (0%, 20%), and the six combinations of noise and function type. These results are presented below in tables 2, 3, 4 and 5 (in each table, starred entries indicate the best value for the corresponding problem category). Table 6 indicates the average size of the decision models generated by each system across noise and function type problem categories. In the case of COGIN and CN2, the values given indicate number of rules. In NewID's case, the values indicate the number of decision tree nodes. Because of different hardware platforms and the different developmental stages of the systems tested, no direct time comparisons were made. However, in a companion study (Greene & Smith, 1992), COGIN was found to scale up in roughly the same ratio as ID4 (an automatic pruning version of ID3).

Table 2. Predictive performance of COGIN, CN2, and NewID by problem set: (a) conjunctive problem sets, (b) compensatory problem sets, (c) HEL problem sets.

(a) Predictive Accuracy—Conjunctive Data Sets

	Clean cj0.2cl	Clean cj1.4cl	Noise cj1n.4cl	Clean cj2.4cl	Noise cj2n.4cl
COGIN	0.98*	0.96*	0.91*	0.87*	0.80
Ordered CN2	0.93	0.93	0.89	0.87*	0.82*
Unordered CN2	0.95	0.88	0.85	0.74	0.71
NewID	0.82	0.88	0.84	0.66	0.64

(b) Predictive Accuracy—Compensatory Data Sets

	Clean cp0.2cl	Clean cp1.4cl	Noise cp1n.4cl	Clean cp2.4cl	Noise cp2n.4cl
COGIN	0.83*	0.64	0.59*	0.52*	0.53*
Ordered CN2	0.78	0.56	0.53	0.45	0.44
Unordered CN2	0.78	0.56	0.53	0.45	0.44
NewID	0.64	0.68*	0.58	0.38	0.38

(c) Predictive Accuracy—HEL Data Sets

	Clean hel0	Noise hel0n1	Noise hel0n2
COGIN	0.83*	0.76*	0.66*
Ordered CN2	0.72	0.71	0.60
Unordered CN2	0.74	0.70	0.59
NewID	0.74	0.67	0.55

Table 3. Average performance of COGIN, CN2, and NewID by data generating function type.

	Conjunctive	HEL	Compensatory
COGIN	0.90*	0.75*	0.62*
Ordered CN2	0.89	0.68	0.55
Unordered CN2	0.83	0.68	0.55
NewID	0.77	0.65	0.53

Table 4. Average performance of COGIN, CN2, and NewID on clean vs. noisy training data.

	Clean Examples	Noisy Examples
COGIN	0.80*	0.71*
Ordered CN2	0.75	0.67
Unordered CN2	0.73	0.63
NewID	0.68	0.61

Table 5. Average performance of COGIN, CN2, and NewID by noise and generating function type.

	Clean Conj	Clean HEL	Clean Comp	Noisy Conj	Noisy HEL	Noisy Comp
COGIN	0.93*	0.83*	0.66*	0.85	0.71*	0.56*
Ordered CN2	0.91	0.72	0.60	0.87*	0.66	0.48
Unordered CN2	0.86	0.74	0.60	0.78	0.65	0.49
NewID	0.78	0.74	0.57	0.74	0.61	0.48

Table 6. Average size of COGIN, CN2, and NewID models.

	Clean Conj	Clean HEL	Clean Comp	Noisy Conj	Noisy HEL	Noisy Comp
COGIN	50	209	157	119	275	195
Ordered CN2	33	103	70	38	109	84
Unordered CN2	60	198	117	81	240	134
NewID	159	380	266	221	490	325

5.3. Interpretation of results

Tables 2 through 5 support our initial hypotheses (Hypothesis 1) that COGIN can provide superior performance on more complex problem sets. Less expected was its superior performance on the simpler problems as well. While some of the one-to-one comparisons are not large enough to be significant, the overall consistency is significant and compelling. Further, most of the aggregated comparisons and specifically those between COGIN and unordered CN2 significantly favor COGIN.

As indicated in table 6, the number of rules in the models generated by COGIN and unordered CN2 is similar across all problems, although unordered CN2 models are somewhat more compact in most cases. Also, as indicated and expected, the ordered CN2 models are significantly smaller in most conditions.

Consideration of the second hypothesis stated above, whether genetic search becomes more effective as problem complexity increases, requires an appropriate ordering of the test problems. One difficulty in this respect is the lack of a specifically calibrated complexity measure. A conjunctive data set with considerable interaction may be more complex than a simple compensatory. The addition of noise complicates matters further. Finally, although the hel data share characteristics of both conjunctive and compensatory problems, there are obviously additional dimensions of complexity (e.g., size of the problem) that do not enable us to reliably place it along the conjunctive-compensatory continuum. Overall, the selection of the data sets was intended to provide comparisons on a variety of conditions, not specifically to rank them according to complexity. We therefore use the average performance of all systems tested as a basis for ordering the set of test problems. Computing the average performance of all systems in each function/noise problem category, we get the pattern of problem complexity depicted in figure 6.

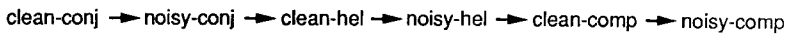

Figure 6. Problem categories ordered in increasing complexity.

Table 7. Relative performance differences between COGIN and CN2/NewID by increasing problem complexity.

	Clean Conj	Noisy Conj	Clean HEL	Noisy HEL	Clean Comp	Noisy Comp
COGIN	1.07	1.06	1.10	1.08	1.09	1.11
Ordered CN2	−.03	−.00	−.15	−.08	−.11	−.15
Unordered CN2	−.09	−.09	−.12	−.10	−.11	−.15
NewID	−.17	−.14	−.12	−.15	−.16	−.16

This ordering suggests that the generating function has the strongest influence on complexity, with the presence of noise a secondary contribution. In general, this pattern of categories seems to be a rough approximation of increasing problem difficulty. Table 7 gives an indication of the relative performance differences between systems across this pattern of problem categories. When comparing performance differences, the values shown are normalized within the given category to avoid scaling effects, so entries for COGIN in this table represent normalized predictive performance percentages based on the mean performance of all systems tested. Entries for other systems are the normalized point differences from the COGIN values.

Figure 7 shows a simple regression of CN2 and New-ID values from table 7. Given the caveats on ordering complexity, the data show a solid correlation of 0.57 for ordered CN2 and 0.62 for unordered CN2, with the slope of the regression implying a positive relationship between increasing complexity and the relative performance difference (although we do not presuppose a linear relationship). For NewID the correlation is 0.002, suggesting no relationship. As noted, both the hel data set and the influence of noise confound a reliable ordering of complexity. To evaluate the possibility of a performance trend, we separate the noise and data-function conditions and examine the relative difference in performance

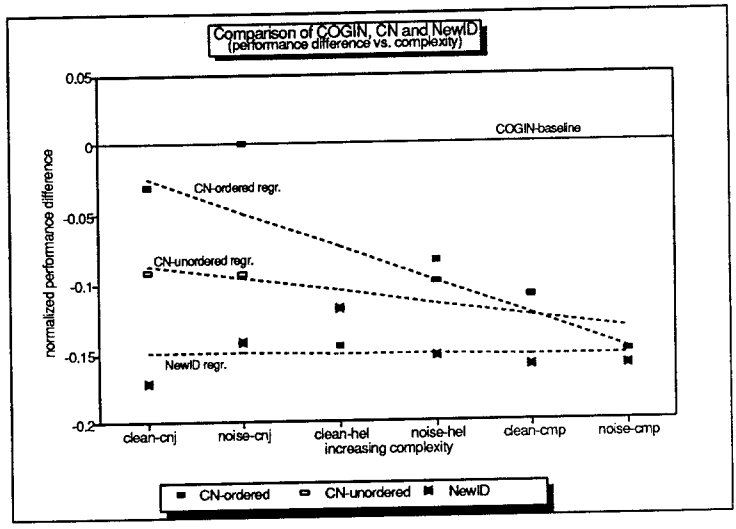

Figure 7. Trend analysis.

Table 8. Relative performance differences between COGIN and CN2 by clean vs. noisy training data.

	Clean	Noisy
COGIN	1.09	1.09
Ordered CN2	−.08	−.05
Unordered CN2	−.10	−.09
NewID	−.16	−.14

between the noise and data-function conditions and examine the relative difference in performance between COGIN and the other systems. The first comparison, table 8, looks at whether COGIN's performance advantage increases as noise increases.

From the table, it is obvious that although the performance is significantly better in both conditions, there is no apparent trend advantage with respect to noise. Examining pairwise comparisons of problem sets confirmed that results were mixed. That is, the performance difference with respect to noise increased for some problems and systems and decreased or remained the same for others—with no obvious pattern. One probable factor is that the dominant influence of the data-generating function overshadows the effect of noise. As noted with the regression, the number of data points does not support breaking the categories further. Given that the data function is a stronger correlate of complexity, to best test the hypothesis of a performance *trend* we examine the relative performance differences between COGIN and the alternative systems based on conjunctive versus compensatory problems as shown in table 9 below.

Given that the dominant complexity effect is the data-generating function, the clearest comparison we can use with confidence is the aggregates of the compensatory and conjunctive experiments, allowing us to formulate hypothesis 2 as a T-test of the difference between two means. We consider the case of ordered CN2 first. If M1 is the mean of the difference between COGIN and ordered CN2 on all conjunctive problem sets, and M2 similarly for all compensatory problem sets, we have the null hypothesis, [H_0: M2 − M1 = 0] and the alternate hypothesis, [H_1: M2 − M1 > 0]. Given 10 problem sets and 8 degrees of freedom, we determine a T-value of 4.309, which rejects H_0 and supports the positive difference at .9999% confidence—in strong support of our second hypothesis. With the data available and using the same tests on unordered CN2 and NewID, the T-statistics are 0.43 and −.25, respectively. We cannot reject the null that there is no clear trend versus

Table 9. Relative performance differences between COGIN and CN2 by data generating function type.

	Conjunctive	HEL	Compensatory
COGIN	1.07	1.09	1.10
Ordered CN2	−.02	−.11	−.10
Unordered CN2	−.09	−.11	−.09
NewID	−.16	−.14	−.14

complexity. Nonetheless, the results suggest an overall normalized advantage of COGIN's competitive-style search over the stepwise feature searches of unordered CN2 and NewID of 10% and 15%, respectively, on this problem set.

5.4. Summary

The purpose of this exploratory study was to evaluate the performance of COGIN over a range of problem conditions emphasizing complexity. Phase 1 examined several parameter variations for COGIN and, based on consistency, settled on the use of a negation bit and a one-error interval level. Phase 2 contrasted how three systems, COGIN, CN2, and New-ID, performed across a variety of problems with different combinations of noise and complexity. Results not only confirmed our initial hypothesis that COGIN would perform better on more complex problems, but also indicated that the GA-based system performed better on simpler problems as well. We then examined whether the relative performance gap between COGIN and the other tested systems actually widened with increasing problem complexity. Significance tests supported this claim in the case of the ordered search variant of CN2, but no significant trends relative to unordered CN2 and New-ID were found.

While these results are promising, three points are worth noting. First, a significant amount of research has investigated the use of various pre- and post-processing procedures to tune and enhance the performance of both CN2 and New-ID. In conducting the above experimental study, both systems were evaluated in their default states, and none of the systems tested (including COGIN) was permitted the luxury of such auxiliary mechanisms. Second, the experimental design executed above was exploratory in nature, and more comprehensive attempts to characterize problem complexity and analyze comparative performance along this dimension are clearly required. One important dimension of complexity that was not emphasized in the above study and clearly needs to be investigated is size in terms of the problem, the sample, and scalability. In a companion study, however, we have shown the basic COGIN system to perform favorably versus extensively post-processed decision-tree models with respect to both prediction and scalability (Green & Smith, 1992). Finally, this study has focused on evaluating the effectiveness of COGIN in relation to traditional induction techniques. Further studies are needed to examine the comparative performance of COGIN and other GA-based learning frameworks.

6. Discussion

In this article, we presented the COGIN framework for induction of decision models from examples. In contrast to previous research in GA-based learning, the COGIN framework exploits the special character of this class of problems, and was designed to directly address the model performance and model complexity tradeoff that this problem presents. Given this emphasis, COGIN represents a fairly significant departure from standard GA-based learning paradigms. Key in this regard is the shift from a "reproduction of the fittest" to a "survival of the fittest" perspective, based more directly on an ecological model where the environment imposes a capacity constraint and survival depends on an ability to exploit

a particular niche. Viewing the set of training examples as being representative of specific niches to be described by model rules, coverage is used as an explicit basis for constraining model complexity (size) while at the same time promoting the diversity required to produce good model performance.

COGIN is still very much an evolving system, and several aspects of its design are under active investigation. One issue involves gaining a better understanding of the criteria that should drive both the ordering of rules during competitive replacement and the resolution of conflicts during model interpretation. In the current implementation, a lexicographic fitness function is used to combine entropy and accuracy metrics, relying on an error interval to mediate the potentially harmful effects of strict reliance on entropy. However, the COGIN search framework can flexibly accommodate a variety of strategies, both lexicographic and compensatory, for combining various fitness-related measures (including those commonly applied in other research in GA-based learning). The tradeoffs between various ordering strategies within this framework are still very much an open question. A related area of interest is the introduction of dynamic control of the influence of various parameters during the search. This concept of shifting bias was effectively (albeit simplistically) exploited in the earlier ADAM system, and our current system offers far richer opportunities.

Another principal direction of our current research concerns generalization of the COGIN framework to accommodate modeling problems that require continuous-valued classification. The COGIN approach to dynamic niche allocation based on coverage appears to provide a much stronger framework for exploiting the concept of ex-post assignment than did the search architecture of our earlier GARGLE system. For example, variance in the outcomes of matched examples provides a natural analog to accuracy in the current lexicographic approach to fitness assignment. Recent work reported in Packard (1990) also suggests several interesting ways of measuring and incorporating continuous values, which could work very effectively in COGIN. Such continuous-valued model-building capabilities are an important extension of current symbolic induction systems.

The goal of this research was to develop an induction system that could exploit the potential of a genetic algorithm to overcome the limitations inherent in stepwise search techniques. While preliminary, the experimental results reported in this article clearly indicate the comparative robustness of COGIN's search framework. The development of accurate, comprehensible decision models for real-world problems requires systems that can effectively navigate complex problem spaces. In this regard, we believe COGIN offers a viable alternative to current inductive systems.

Acknowledgments

This work was supported in part by the US ARMY Human Engineering Laboratory under contract DAAD05-90-R-0354, and the Robotics Institute. The authors would like to thank the Turing Institute for providing us with the CN2 and NewID software and Robin Boswell for his assistance in their use. We would also like to thank Rick Riolo, John Grefenstette, and two anonymous reviewers for helpful comments on an earlier draft of this article.

Notes

1. Nonhomogeneity refers to the variables having different relationships in different parts of the measurement space (Breiman, 1984).
2. In genetics, this means a nonreciprocal interaction between nonalternative forms of a gene in which one gene suppresses the expression of another affecting the same part of an organism.
3. Our decision to employ a single-point crossover operator was motivated strictly by reasons of historical bias and computational simplicity. Multi-point and uniform (Syswerda, 1989) crossover operators are equally applicable within the COGIN framework, and we plan to evaluate their comparative utility.
4. The actual variance in performance across runs was very low, usually less than 1%.

References

Baker, J.E. (1985). Adaptive selection methods for genetic algorithms. *Proceedings First International Conference on Genetic Algorithms and their Applications*. Pittsburgh, PA: Morgan Kaufmann, pp. 101–111.

Booker, L. (1989). *Intelligent behavior as an adaptation to the task environment*. Doctoral dissertation, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI.

Booker, L. (1989). Triggered rule discovery in classifier systems. *Proceedings 3rd International Conference on Genetic Algorithms*. Fairfax, VA: Morgan Kaufmann, pp. 265–274.

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification and regression trees*. Monterey, CA: Wadsworth.

Clark, P., & Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning*, *3*(4), 261–283.

Currim, I., Meyer, R.J., & Le, N. (1986). *A concept learning system for the inference of production models of consumer choice*. Working paper, UCLA.

DeJong, K.A., & Spears, W.M. (1991). Learning concept classification rules using genetic algorithms. *Proceedings 12th International Conference on Artificial Intelligence*. San Mateo, CA: Morgan Kaufmann.

Eshelman, L. (1991). The CHC Adaptive Search algorithm: How to have safe search when engaging in nontraditional genetic recombination. In G.J.E. Rawlins (Ed.), *Foundations of genetic algorithms*. San Mateo, CA: Morgan Kaufmann.

Goldberg, D. (1985). Dynamic systems control using rule learning and genetic algorithms. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence (Los Angeles, California)*. Morgan Kaufmann, pp. 588–592.

Goldberg, D.E. (1989). *Genetic algorithms in search, optimization and machine learning*. Reading, MA: Addison-Wesley.

Greene, D.P. (1987). Automated knowledge acquisition: Overcoming the expert systems bottleneck. *Proceedings of the Seventh International Conference on Information Systems*. Pittsburgh, PA: Lawrence Erlbaum Assoc., pp. 107–117.

Greene, D.P. (1992). *Inductive knowledge acquisition using genetic adaptive search*. Doctoral dissertation, The Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh, PA.

Greene, D.P., & Smith, S.F. (1987). A genetic system for learning models of consumer choice. *Proceedings of the Second International Conference on Genetic Algorithms and their Applications*. Boston, MA: Morgan Kaufmann, pp. 217–223.

Greene, D.P., & S.F. Smith. (1992). COGIN: Symbolic induction with genetic algorithms. *Proceedings Tenth National Conference on Artificial Intelligence*. San Mateo, CA: Morgan Kaufmann.

Grefenstette, J.J., Ramsey, C.L., & Schultz, A.C. (1990). Learning sequential decision rules using simulation models and competition. *Machine Learning*, *5*(4), 355–381.

Grefenstette, J.J. (1991). Lamarkian learning in multi-agent environments. *Proceedings 4th International Conference on Genetic Algorithms*. San Mateo, CA: Morgan Kaufmann, pp. 303–310.

Holland, J.H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor, MI: University of Michigan Press.

Holland, J.H. (1986). Escaping brittleness: The possibilities of general purpose learning algorithms applied to parallel rule-based systems. In R. Michalski, J. Carbonell, and T. Mitchell (Eds.), *Machine learning: An artificial intelligence approach* (Vol. 2). San Mateo, CA: Morgan Kaufmann.

Koza, J. (1991). Evolving a computer program to generate random numbers using the genetic programming paradigm. *Proceedings of the Fourth International Conference on Genetic Algorithms and their Applications*. San Diego, CA: Morgan Kaufmann, pp. 37–44.

MacDonald, B.A., & Witten, I.H. (1989). A framework for knowledge acquisition through techniques of concept learning. *IEEE Transactions on Systems Man and Cybernetics, 19*(3), 499–512.

Michalski, R.S., & Chilauski, R. (1980). Learning by being told and learning from examples. *International Journal of Policy Analysis and Information Systems, 4*, 210–225.

Packard, N.H. (1990). A genetic learning system for the analysis of complex data. *Complex Systems, 4*. 543–572.

Quinlan, J.R. (1984). Inductive inference as a tool for the construction of high-performance programs. In R. Michalski, J. Carbonell, and T. Mitchell (Eds.), *Machine learning: An artificial intelligence approach* (vol. 1). Palo Alto, CA: Tioga Press.

Quinlan, J.R. (1986). The effect of noise on concept learning. In R. Michalski, J. Carbonell, and T. Mitchell (Eds.), *Machine learning: An artificial intelligence approach* (Vol. 2). San Mateo, CA: Morgan Kaufmann.

Robertson, G.C., & Riolo, R.L. (1988). A tale of two classifier systems. *Machine Learning, 3*, 139–159.

Schaffer, J.D. (1984). *Some experiments in machine learning using vector evaluated genetic algorithms*. Doctoral dissertation, Department of Electrical Engineering, Vanderbilt University, Nashville, TN.

Smith, S.F. (1980). *A learning system based on genetic adaptive algorithms*. Doctoral dissertation, Department of Computer Science, University of Pittsburgh, Pittsburgh, PA.

Smith, S.F. (1983). Flexible learning of problem solving heuristics via adaptive search. *Proceedings 8th International Joint Conference on AI*.

Syswerda, G. (June 1989). Uniform crossover in genetic algorithms. *Proceedings of the Third International Conference on Genetic Algorithms and their Applications*. Fairfax, VA: Morgan Kaufman.

Wilson, S.W. (1987). Classifier systems and the animat problem. *Machine Learning, 2*, 199–228.

Wilson, S.W., & Goldberg, D.E. (1989). A critical review of classifier systems. *Proceedings 3rd International Conference on Genetic Algorithms*. Fairfax, VA: Morgan Kaufmann, pp. 244–255.

Received November 5, 1991
Accepted January 23, 1992
Final Manuscript October 23, 1992