A Case Study of Incremental Concept Induction

Jeffrey C. Schlimmer and Douglas Fisher

Department of Information and Computer Science
University of California, Irvine, 92717
ArpaNet:Schlimmer@ICS.UCI.EDU, DFisher@ICS.UCI.EDU

Abstract

Application of machine induction techniques in complex domains promises to push the computational limits of non-incremental, search intensive induction methods. Learning effectiveness in complex domains requires the development of incremental, cost effective methods. However, discussion of dimensions for comparing the utility of differing incremental methods has been lacking. In this paper we introduce 3 dimensions for characterizing incremental concept induction systems which relate to the *cost* and *quality* of learning. The dimensions are used to compare the respective merits of 4 incremental variants of Quinlan's learning from examples program, ID3. This comparison indicates that cost effective induction can be obtained, without significantly detracting from the quality of induced knowledge.

I Introduction

Work in machine learning has concentrated significantly on the problem of concept induction (e.g., *learning from examples*, *conceptual clustering*). Thus far, the majority of concept induction systems are *nonincremental*, in that they require all objects over which induction is to occur to be present from the outset of system execution, while *incremental* systems accept objects over a span of time. Motivations for incremental systems hinge primarily on the realization that as learning systems are required to deal with a greater number and diversity of observations, non-incremental systems may not be computationally usable (Michalski, 1985). Specifically, the primary motivation for incremental induction is that a knowledge store may be rapidly updated as each new instance is encountered, thus sustaining a continual basis for reacting to new stimuli; this property is becoming paramount with the development of simulated world environments (Carbonell & Hood, 1985; Sammut & Hume, 1985) which promise to push the limits of current learning systems.

Along with the *cost* advantages of incremental systems come disadvantages which emerge as a result of the constraints mandated by performing rapid update. Nonincremental concept induction systems tend to be search intensive (depth-first with backtracking, breadth-first, or version-space) which requires maintaining a frontier of hypotheses and/or a list of previously observed instances (Mitchell, 1982) until some stable hypothesis is converged on. New objects serve to expand the frontier of hypotheses which makes incorporation costly. However, exhaustive techniques generally guarantee that a correct or optimal hypothesis is obtained. Incremental systems seek to reduce the cost of update, thus precluding the luxury of keeping past instances or equivalent information (i.e., a frontier of hypotheses). Incremental systems will generally require more objects to converge on a stable hypothesis, and may sacrifice the guarantee of the correctness or optimality of a final hypothesis. A reduction in search control yields a search process which Simon (1969) has termed *satisficing*. An environment which places constraints on response time precludes searching for optimal solutions, and necessitates a search for satisfactory hypotheses. This does *not* preclude the possibility of obtaining optimal solutions, but only the explicit search for such solutions. In fact, a satisficing strategy can allow rapid memory update and hypotheses which are of high quality.

As interest in incremental learning mounts, it becomes increasingly important to make explicit the *computational properties* (e.g., cost, concept quality) of incremental induction techniques, and not limit their characterization to the *behavioral property* that objects are accepted one at a time. For instance, any exhaustive search technique (e.g., version space) can be implemented so as to accept instances incrementally, but such an implementation may have limited utility in an environment demanding incremental *computation*. In this paper we discuss several dimensions which differentiate incremental and nonincremental learners, as well as serving as a basis for evaluating competing incremental systems. These dimensions are

Proceedings of the Fifth National Conference on Artificial Intelligence, Philadelphia, PA Morgan Kaufmann. pp. 495-501. (1986)

- The *number of observations* required by a learning system to obtain a 'stable' set of concept descriptions.
- The *cost of updating* memory to accommodate an observed object.

These two factors can be combined into a single measure of cumulative cost, which reflects the amount of resources expended during learning. A last dimension for characterizing incremental induction systems is:

- The *quality of concept descriptions* derived by a concept induction system.

The remainder of the paper focuses on a case study for discussing these dimensions. Specifically, the dimensions are used to compare the behavior of several incremental variants of Quinlan's (1983, 1985) ID3 program. Each system is of the learning from examples variety, and each builds decision trees over observed objects which distinguish positive from negative instances. A formal analysis of these systems is bolstered by an empirical analysis which indicates that object incorporation can be considerably reduced without a significant decrease in the quality of derived decision trees. In general, however, reducing update cost implies an increase in the number of observed objects required to find an 'optimal' tree.

II A case study: ID3

Quinlan's (1983, 1985) ID3 constructs a discrimination tree that distinguishes between examples and nonexamples of a particular concept. The algorithm starts out with an empty decision tree and a collection of examples and nonexamples of a concept. Each object is described by a number of attributes (e.g., size, shape, color) and a value for each of those attributes (e.g., size is red). A measure is applied to each of the attributes to determine how well they discriminate between positive and negative examples. The most informative attribute is used to form the root of the decision tree with a branch for each of its values. The instances are then divided into groups according to their value for this attribute, and the process is recursively applied for each group, thus building subtrees. The process continues until all of the examples in a subtree are either positive or negative. At this point, the decision tree completely discriminates between examples and nonexamples of the concept to be acquired.

The choice of a measure for selecting discrimination tree roots is critical if good decision trees are to be obtained. ID3 uses an information theoretic measure to determine which attribute values best divide an object (sub)set at each subtree. ID3 has also been designed to accommodate occasional errors (or *noise*) in the concept instances. In a noisy situation, ID3 will attempt to discriminate between every positive and negative instance, resulting in a decision tree which may be unnecessarily large. To limit this growth, a χ^2 (chi-squared) statistical test is used to insure (with a high degree of confidence) that the sampling of instances is not due to chance. A more detailed account of ID3 and its measures can be found in Quinlan (1985).

ID3 is a nonincremental algorithm where a large number of instances are available for processing at one time. One 'brute force' method of applying ID3 in an incremental manner would be to allow objects to be presented one at a time, and simply rerun ID3 as each object is observed. However, the ID3 framework is amenable to modification so that instances may be processed one at a time in an efficient, computationally incremental manner. The heart of these modifications lies in a series of tables located at each potential decision tree root. Each table consists of entries for the values of all untested attributes and summarizes the number of positive and negative instances with each value. As a new instance is processed, the positive or negative count for each of its attribute-values is incremented according to whether this instance was an example or nonexample. Classification then proceeds down the the appropriate subtree. If a subtree is encountered which does not yet have a root test attribute, and there are both positive and negative counts for classified instances, then the information measure is used to compute the most informative of the previously unused attributes. This attribute is then evaluated using the χ^2 test; if it is unlikely to have arisen by chance, then it is chosen as the root attribute. Otherwise, this root is left empty. The process continues until either all instances at a subtree are of one type (positive or negative) or a new root cannot be reliably chosen. Table 1 depicts the steps followed in the modified version of ID3. This modification is termed ID4.

ID4 also allows changing a poorly chosen root (step 3d). Changing the root of a subtree discards information gleaned over previous instances and requires examining a number of subsequent instances before deeper subtree roots can be rechosen. This process occurs infrequently, to the credit of the information measure heuristic. As learning progresses, the important roots (higher in the tree) become more stable, and changes in subtree root choices have less of an effect. In the following analyses, ID4 was able to discard previous subtree attribute choices, and converge on the same decision tree as ID3.

III Analysis

In this section, a brute force incremental application of ID3 (henceforth, simply ID3) and ID4 are compared with respect to the dimensions discussed earlier. The formal

Inputs: *A decision tree, One instance.*
Output: *A decision tree.*

1. If this instance is positive, increment the total number of positive instances. Otherwise increment the number of negative instances.
2. If all of the instances are positive or negative then return the decision tree.
3. Else
   (a) Compute the expected information score.
   (b) For each attribute, for each value present in the instance, increment either the number of positive or negative instances.
   (c) Compute the information scores for all attributes.
   (d) If there is no root or the maximal attribute is not the root, then build a new tree.
      i. If the maximal attribute is χ^2 dependent then make it the root of this tree.
      ii. Make a test link from the root for every value of the root attribute.
   (e) Go to step 1 with the subtree found by following the link for the root attribute's value in this instance.

Table 1: Pseudo code for incremental ID4.

analysis is augmented with an empirical analysis, during which two additional incremental variants are introduced (I$\widehat{\text{D3}}$ and I$\widehat{\text{D4}}$). The introduction of these latter methods serves to refine the space of incremental techniques, and empirical analysis indicates that the cost of incorporating an instance can be significantly reduced without significantly effecting the quality of learning.

A. Number of observations to an effective characterization

An important computational measure is the number of instances required to construct an optimal tree. ID3 chooses an attribute to form the test for the root based on the information that attribute contains over the observed instances. A sample of objects (from the environment) of sufficient size, n_0, must be seen for ID3 to choose the root attribute whose values best discriminate objects of the environment as a whole. This is true in the creation of all subtree roots as well, where the required number of objects is $n_{i,j}$ for the subtree rooted at node j of level i. For an entire level, i, of the decision tree, n_i ($= \sum_j n_{i,j}$) represents the number of objects required for all nodes at that level to attain a stable discriminating attribute. A level cannot stabilize until previous levels have achieved stability, and thus $n_i \geq n_{i-1}$. Since ID3 retains all instances seen, the number of objects to construct a decision tree of depth d is

$$\max_{i=0}^{d-1} n_i = n_{d-1}$$

Again, assuming a representative object sample, ID4 must examine the same n_0 instances in order to choose the root attribute in an optimal manner. However, because ID4 does not store all instances encountered, at the next level it must examine *another* n_1 instances because the first n_0 instances are not available for inspection. Consequently, the number of instances required to construct the tree is the sum of all of the root choice points, or

$$\sum_{i=0}^{d-1} n_i$$

B. Cost of updating memory

In ID3, to update a tree is to build a new tree from scratch. Constructing each node of the tree requires that instances be examined to determine their values for previously unused attributes. The cost of constructing an entire tree is

$$\sum_{a=|A|}^{|A|-d} a \times |I| = O(|I| \times |A|^2)$$

where $|I|$ is the number of instances, $|A|$ is the number of attributes, and d is the depth of the tree (which cannot exceed $|A|$).

If a tree is built after every instance, then the above expense is incurred over a single object, over two objects, ..., for a total of

$$\sum_{i=1}^{|I|} i \times |A|^2 = O(|I|^2 \times |A|^2)$$

Asymptotically, the most important term is $|I|^2$ since the number of instances is presumably much greater than the number of attributes.*

In ID4, building a decision tree is proportional only to the number of objects times the square of the number of attributes, as shown below

$$\sum_{i=1}^{|I|} |A|^2 = O(|I| \times |A|^2)$$

C. Cost to performance

A third measure combines the previous two: what is the overall expense of constructing an effective characterization if it is done in a incremental manner? For ID3, the number

*Quinlan (1983) also presents a 'windowing' method which builds a decision tree based on a subset of the given instances. All of the remaining instances are then used to test this tree. If it fails to correctly classify each, the window is enlarged and the process repeated. This algorithm, however, is also asymptotically bounded by $|I^2|$ since the dominant cost is testing each of the instances.

of objects to an efficient characterization is n_{d-1}. When this is substituted into the cost equation for ID3 we have

$$\sum_{i=1}^{n_{d-1}} i \times |A|^2 = O\left[n_{d-1}^2 \times |A|^2\right]$$

For ID4, the number of instances to an optimal decision tree is larger: $\sum_{i=0}^{d-1} n_i$. Substituting this into the expression for object incorporation yields

$$\sum_{i=1}^{\sum_{j=0}^{d-1} n_j} |A|^2 = O(\sum_{j=0}^{d-1}(n_j) \times |A|^2)$$

Comparing total expense hinges on the number of instances required to select each subtree root attribute. If $n_{d-1}^2 > \sum_{i=0}^{d-1} n_i$ then ID3 is more expensive than ID4. This is very likely the case, since the number of instances required to construct the tree is probably greater than the depth of the decision tree, or $n_{d-1} > d$.

Our analysis has assumed a 'representative' sample of objects - lacking a rigorous discussion of regularity and distribution of objects in an environment, we now perform an empirical analysis.

IV Empirical performance

Consider the task of classifying chess endgames. Given a board position, a classifier attempts to identify the situation as a win or loss. Following Quinlan (1979), we define a concept attainment task as determining whether a black king and knight versus a white king and rook results in the safety or loss of the black knight or king in two moves with black to move. Figure 1 depicts a sample board configuration.

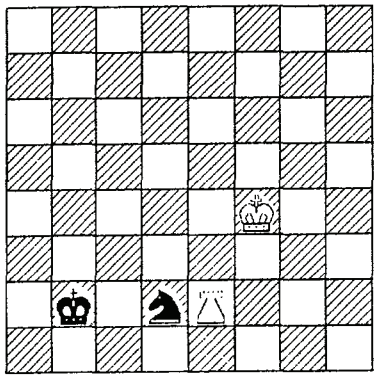

Figure 1: Example of a safe, pinned black knight.

Boards were randomly generated and described in terms of the **distance** (in squares) between each pair of pieces (6 attributes of this type), **board relationship** between every pair of pieces (i.e., whether they lie on the same rank or file, diagonal, or otherwise) (6 attributes), and the **square type** where each piece resides (i.e., corner, edge, or otherwise) (4 attributes). There are a total of sixteen attributes, each with three values. Although there are $6^3 \times 6^3 \times 4^3 = 2,985,984$ objects possible, an exhaustive enumeration of the actual 95,480 distinct knight pins indicates that there are only 3,259 actual objects in terms of these attributes.

Four behaviorally incremental variants of the ID3 algorithm are tested. The first is a brute force version of ID3 which constructs a new decision tree from scratch after each new instance is received. A smarter version, $\widehat{\text{ID3}}$, only reconstructs the decision tree when an instance has been misclassified. The third variant is ID4; the counts of positive and negative instances are updated for each new instance. Finally, the fourth variant $\widehat{\text{ID4}}$, only updates attribute counts when an error in classification is made (similar to $\widehat{\text{ID3}}$). The same randomly generated boards ($\approx 69\%$ of the instances were positive) were presented to these four variants. The decision tree formed by *all* of the variations tested is depicted in figure 2.**

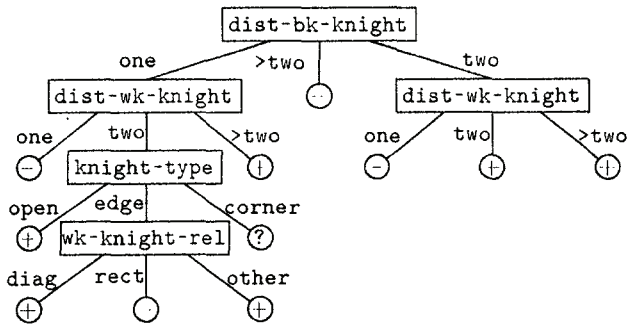

Figure 2: Decision tree for a safe knight pin.

For the three more efficient algorithms, the number of observations required to form this decision tree ranges from the least for $\widehat{\text{ID3}}$ to the greatest for $\widehat{\text{ID4}}$. Figure 3 depicts the average depth of the decision tree in figure 2 (averaged over 50 executions) built by each variant as a function of the number of instances. Depth gives a rough and simple picture of learning speed, but should not be equated with correctness. $\widehat{\text{ID3}}$ rapidly builds a complete tree, while ID4 requires substantially more instances. $\widehat{\text{ID4}}$ requires the largest, converging consistently on the complete tree after approximately 20,000 instances.

Though there is a substantial range in the time each variant constructs its decision tree, each of the variants quickly forms an effective classification for the instances. Classification performance of the three more efficient variants (averaged over 50 executions) was measured over 1000 instances. For $\widehat{\text{ID3}}$ and ID4, a 90% effective classification

**The ? in the lower right leaf corresponds to a situation where no instances have been observed. In fact, the knight cannot be in a corner and also be pinned by the rook and the black king.

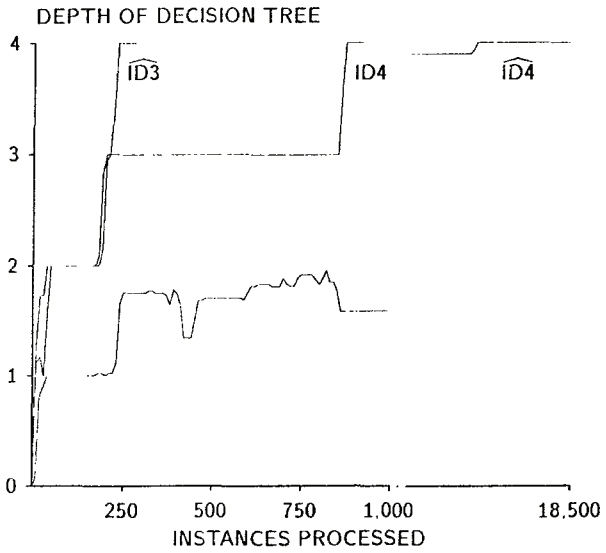

Figure 3: Average decision tree depth.

is formed after as few as 100 instances. The classification performance of ID4 takes somewhat longer to level off; it reaches 75% correct classification after 275 instances, and 90% correct classification after 750 instances. Thus, although it may take considerable time for these incremental algorithms to achieve perfect classification performance, good ($\approx 90\%$) classification was achieved relatively quickly. The apparent speed with which these algorithms form an effective classification is due to the order of the decision tree's construction; important, decisive attributes are chosen first, leaving tests of lesser value until deeper in the tree.

The cost of updating concept descriptions is measured by counting the number of comparisons made to incorporate each new instance. Figures 4 and 5 depict the number of comparisons per instance for a typical execution (among 50 executions performed) for each variant. ID3 reconstructs a new decision tree after each new instance and exhibits a steep curve. ID3' $O(|I| \times |A|^2)$ behavior arises from reexamination of the complete set of saved instances as each new instance is processed. ID3, which tests an instance before rebuilding the decision tree is less expensive than ID3, but it exhibits occasional peaks as the decision tree proves inadequate. Though the overall average expense incurred per instance is low, and it apparently asymptotes to a small value, the upper bound is still $O(|I| \times |A|^2)$.

The expense of processing an instance in ID4 is nearly constant when compared to ID3 and ID3 (note that the vertical scale for ID4 has been magnified 400 times). The $O(|A|^2)$ bound of ID4 is greater than the usually low cost of ID3, but it is always considerably less than the latter's peaks. The fourth variant, ID4, displays the least expense of the three. It asymptotes to a value as small as ID3 while remaining within the bounded expense per instance

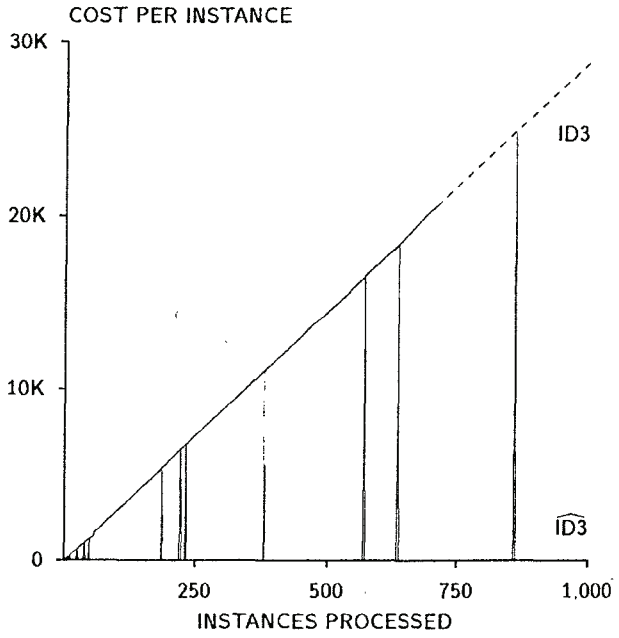

Figure 4: ID3 and ID3 cost per instance.

established by ID4. The price of this efficiency is that ID4 learns slowly.

In figure 6 the cumulative performance of each of the four variants over a typical sequence of 1000 instances is depicted. The vertical axis reflects the *cumulative* number of comparisons made by each algorithm as a function of the number of instances. The brute force version of ID3 displays the most expensive approach of the four. The geometrically accelerating curve reflects the $O(|I|^2 \times |A|^2)$ asymptotic bound. The step function nature of the ID3 curve reveals the low, intermediate expense of classifying instances and the high, intermittent expense of rebuilding the decision tree from scratch when an incorrectly classified instance is encountered. ID4 performs $O(|A|^2)$ work for each instance processed, and this is clearly reflected in its nearly linear curve. The least expensive curve results from ID4 which updates attribute counts only if the test classification is incorrect.

V Conclusion

As machine learning methods are applied in more complicated domains, the deficiencies of nonincremental, search intensive methods have become evident. This has increased interest in incremental concept induction methods which process observations as they are observed. An important point though, is that *any* nonincremental algorithm can be made to behave in an incremental fashion (i.e., process observations one at a time). In general, however, incremental *behavior* does not insure the compu-

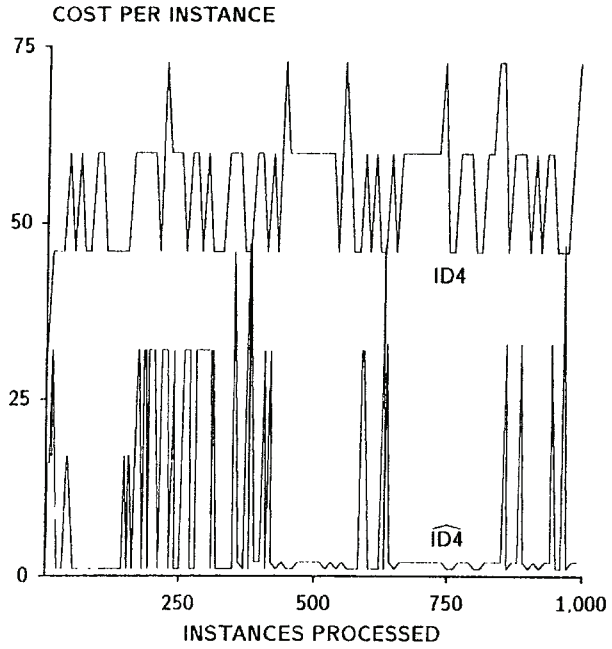

Figure 5: ID4 and ÎD4 cost per instance.

tational efficacy of such an algorithm. Three dimensions for evaluating incremental concept induction methods have been outlined. These dimensions are:

- The cost of updating memory to accommodate a new object.
- The number of objects necessary to obtain a stable concept description.
- The quality of derived concept descriptions.

These dimensions have been used to compare the behavior of 4 incremental variants of Quinlan's ID3 program. A case study in the domain of chess endgames has served as a promising indication that incremental induction methods can meet the computational constraints of complex environments, while meeting high standards of quality and correctness.

Acknowledgements

Discussions with Dennis Kibler initially raised a number of ideas expressed in this paper, specifically the criteria relating to the cost and quality of learning. This research was supported in part by the Office of Naval Research under grants N00014-84-K-0391, N00014-84-K-0345, and N00014-85-K-0854, the National Science Foundation under grants IST-81-20685 and IST-85-12419, the Army Research Institute under grant MDA903-85-C-0324, and by the Naval Ocean Systems Center under contract N66001-83-C-0255.

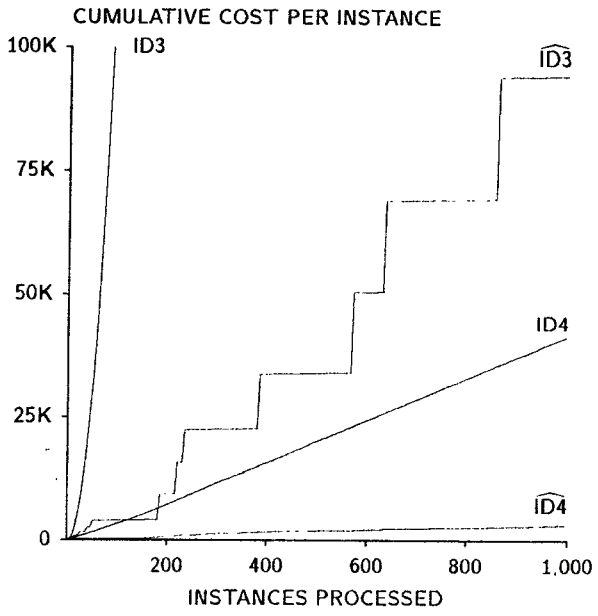

Figure 6: Cumulative cost per instance.

References

Carbonell, J. & Hood, G. (1985). The World Modelers Project: Objectives and Simulator Architecture. *Proceedings of the Third International Machine Learning Workshop* (pp. 14-16). Rutgers University.

Michalski, R. (1985). Knowledge Repair Mechanisms: Evolution Versus Revolution. *Proceedings of the Third International Machine Learning Workshop* (pp. 116-119). Rutgers University.

Mitchell, T. (1982). Generalization as Search. *Artificial Intelligence, 18.* 203-226.

Quinlan, J. R. (1979). Discovering Rules by Induction From Large Collections of Examples. In D. Michie (Ed.), *Expert systems in the micro electronic age.* Edinburgh: Edinburgh University Press.

Quinlan, J. R. (1983). Learning efficient classification procedures and their application to chess end games. In R. S. Michalski, J. G. Carbonell & T. M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach.* Palo Alto, California: Tioga Publishing Company.

Quinlan, J. R. (1985). *Induction of decision trees* (Technical report 85.6). New South Wales, Australia: The New South Wales Institute of Technology, School of Computing Sciences.

Sammut, C. & Hume, D. (1985). Learning Concepts in a Complex Robot World. *Proceedings of the Third International Machine Learning Workshop* (pp. 173-176). Rutgers University.

Simon, H. (1969). *The Sciences of the Artificial.* Cambridge, Mass.: The M.I.T. Press.