Models of Incremental Concept Learning:
A coupled research proposal

Douglas H. Fisher

Jeffrey C. Schlimmer

Computer Science Department
Vanderbilt University
Nashville, TN 37235

Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213

February 9, 1997
Contents

1 Project Summary .................................................. 1

2 Project Description .............................................. 2
  2.1 Focused Research Issues ...................................... 2
  2.2 Related Work .................................................. 5
    2.2.1 Incremental learning from examples ...................... 5
    2.2.2 Incremental conceptual clustering ...................... 10
  2.3 Incremental Concept Learning: Our Approach ............ 11
    2.3.1 Search control strategies for incremental learning ... 11
    2.3.2 Concept and knowledge base organization ............ 12
  2.4 Plan for Continued Research ............................... 15
    2.4.1 Domain-independent incremental methods ................ 15
    2.4.2 Application areas for incremental learning .......... 18
    2.4.3 Frameworks for understanding incremental learning ... 19
    2.4.4 Methodological Approaches ............................. 21

3 References ....................................................... 23
1 Project Summary

Artificial Intelligence systems must be adaptive if they are to operate effectively in complex, real-world domains. In many real-world situations, knowledge must be frequently and sporadically accessed; it may be impossible to make large numbers of observations prior to decision making. This proposal addresses the problem of incremental learning, which assumes that environmental observations are assimilated as they become available. A desirable characteristic of such systems is that they maintain an accurate knowledge base that can be efficiently accessed and updated. However, efficient assimilation of new observations may require that we be satisfied with a knowledge base of lower overall ‘quality’. Thus, incremental learning systems trade quality against cost, hopefully not to the significant detriment of the former. We begin by reviewing a number of incremental concept learning systems with special attention to their management of the cost/quality tradeoff. From these, we abstract control strategies and knowledge representation schemes that support efficient, but accurate incremental induction. Our analysis motivates proposals for improved incremental learning systems and draws from the literature of cognitive psychology, as well as belief revision. Our primary goal is to investigate domain-independent learning methods, but we have concrete plans to test our systems in areas such as intelligent tutoring, game-playing, and expert knowledge acquisition.
2 Project Description

2.1 Focused Research Issues

Building successful Artificial Intelligence (AI) systems for real-world domains might only be possible by incorporating a flexibility that is realized through learning. As a result, machine learning studies in real-world environments are gaining prominence (Carbonell & Hood, 1985; Langley, Kibler, & Granger, 1985; Sammut & Hume, 1985). These studies relax many of the simplifying assumptions that enabled earlier work to progress. In particular, this proposal is concerned with incremental concept learning systems, which drop the commonly-made assumption that all environmental observations can be processed simultaneously (Dietterich & Michalski, 1983; Hayes-Roth & McDermott, 1978; Quinlan, 1986; Vere, 1980). Instead, observations are assimilated as they become available. This incremental assumption has important computational ramifications, for it bounds the amount of processing that can be expended on new observations.

A simple model of intelligent processing is depicted in Figure 1 (Dietterich, 1982). Learning organizes environmental observations to improve
performance on some task(s). Assumptions about the environment, knowledge base, and performance task impact the formulation of the learning process, but it is the environment that primarily plays the role of the ‘independent variable.’ The environmental assumptions that underlie nonincremental learning have crept into all aspects of the intelligent processing model. In particular, nonincremental systems assume that all ‘required’ information is available from the outset and conceptual knowledge is induced ‘all-at-once’. Since nonincremental systems do not assume that knowledge is intermittently updated and applied, the learning element may extensively search for appropriate concepts with little concern for efficiency. Moreover, induced concepts need not be organized for efficient retrieval – the result is a simplified ‘knowledge base’ that is little more than a list of disparate (often a single) concept descriptions. Finally, the extensive search invariably carried out by nonincremental systems insures that performance (e.g., prediction accuracy) tends toward optimality.

While nonincremental systems may lead to near-optimal performance, their environmental assumptions and processing characteristics become increasingly untenable as the number and variety of observations grow. We will study the cost/performance tradeoffs implied by incremental learning. This study will focus on a number of behavioral, computational, and representational issues.

- **Environmental Dimensions**: The main environmental assumption that drives our study is that observations are not simultaneously available for inspection. This allows the environment to change over time and requires that the learning element and knowledge base track this change; we report on an existing system that tracks environmental changes and propose further studies. We review and propose studies of incremental learning from examples, which assume that a ‘teacher’ preclassifies observations, but we also drop this assumption and extend our analysis to incremental conceptual clustering.

- **Learning Element Dimensions**: One way to control the cost of knowledge base update is to limit the number of alternative hypotheses explored during search. However, this also limits a system’s ability to backtrack or otherwise recover from unproductive search paths. Our analysis illustrates that incremental learning methods can be effectively
supported by coupling a constrained search control strategy with belief revision techniques such as dependency-directed backtracking.

- **Knowledge Base Dimensions:** Efficient access of knowledge is best facilitated by structuring and indexing it appropriately. Thus, there is a need to move beyond the unstructured knowledge base commonly found in learning systems. We propose hierarchical knowledge bases and mechanisms for identifying and exploiting default values (Brachman, 1985) to aid efficient retrieval. In addition, we suggest the use of probabilistic representations and saved exemplar observations to increase knowledge base pliability. We draw upon psychological studies of human typicality and basic level effects for guidance in selecting observations and generalizations to retain.

- **Performance Dimensions:** Our previous work has characterized incremental learning systems along four dimensions that reflect the inherent tradeoffs between learning cost and quality: cost of knowledge base update, the number of observations that are required to converge on a knowledge base with given characteristics, the total effort (as a function of the first two dimensions) that a system exerts, and the quality of the final knowledge base. We discuss computational characteristics of prior learning systems and motivate new systems along these dimensions.

In summary, our primary goal is to investigate domain-independent mechanisms and representations that efficiently and robustly support incremental learning. However, the applicability of our investigations promises to reach beyond the context of incremental learning. For example, guidelines for identifying and maintaining default generalizations during learning may have important implications for belief revision. Further, recent studies demonstrate that induced knowledge may ‘over-fit’ the data and actually detract from performance. We will explore the apparent tradeoff between knowledge ‘complexity’ and performance ability and its implications for incremental learning.

Second, we will test our general methods on real-world tasks such as intelligent tutoring where intermittent, accurate, and timely feedback is invaluable.

Finally, we have and will continue to empirically evaluate our learning systems by methods addressed under the research plan. Importantly, our
methodology downplays anecdotal evidence and insists that AI systems be characterized across a range of possible scenarios. Apparently, our approach has already had some impact on the methodological biases of the machine learning community (Langley, 1987b).

2.2 Related Work

Incremental learning is motivated by a need to rapidly and frequently exploit knowledge during learning. Theoretically, it is possible to rerun inherently nonincremental methods with each new observation (plus all previous ones) and thus produce an entirely new knowledge base from scratch. Michalski (1985) has called this a revolutionary approach to knowledge base update. However, as the number and variety of observations grows, the cost of revolution becomes prohibitive. An alternative is an evolutionary strategy that changes only ‘faulty’ parts of a knowledge base to accommodate new observations (Michalski, 1985). One disadvantage of this approach is that localized changes may result in a knowledge base of lower ‘quality.’ Thus, incremental learning methods trade quality against cost, hopefully not to the significant detriment of the former. We now turn our attention to a number of previous studies of incremental learning with special attention to their management of the cost/quality tradeoff.

2.2.1 Incremental learning from examples

Hunt, Martin, and Stone (1966) were among the first to study machine concept learning from examples. Their Concept Learning System (CLS) nonincrementally builds decision trees that discriminate observations of different classes. CLS first divides the observations by their values along the ‘best’ descriptive attribute; it uses a primitive frequency measure to determine the attribute whose values are most uniquely associated with different classes. The values of this divisive attribute are used to label arcs from the decision tree root and segregate observations into disjoint subsets. Each subset is treated as a child of the root, and CLS recursively builds a subtree for each. Decision tree expansion terminates when all observations at a (sub)node are members of the same class.

A new observation is classified by following the labeled arcs that correspond to the observation’s values. When a leaf is reached, the class of the
observations residing there is asserted as the class of the new observation. If this prediction is correct, the new observation is saved (along with the original set). If erroneous, the tree is reconstructed using previous observations and the misclassified one. This exemplifies a revolutionary approach to learning – a revolution occurs with each misclassification.

There is a sense in which the revolutionary procedure appears to be incremental, since observations are processed serially. However, each misclassification incurs subsequently larger amounts of processing. Hunt et al. explored variants of CLS that restricted the memory of past observations to a constant by randomly replacing prior observations with new ones. Limiting the number of saved observations also limits tree reconstruction costs, but experiments showed that it slowed learning rates as well; that is, more observations were required to form a decision tree that perfectly discriminated the observations.

In a similar vein, Michalski and Larson (1978) investigated the utility of restricting the observations used during learning. The basis of their strategy is AQ (Michalski, 1973), a nonincremental learning from examples system that does not build decision trees, but a ‘flat’ set of logical (i.e., DNF) concept descriptions. Michalski and Larson adapt AQ to behave incrementally by limiting the number of observations used to reconstruct a faulty knowledge base. Unlike CLS, retained observations are not selected randomly, but on the basis of a Euclidean distance-like measure that identifies ‘good’ concept representatives. In addition, incremental AQ limits reconstruction to those portions of the knowledge base (i.e., individual concept descriptions) that led to a misclassification. This ‘locality’ constraint reduces computational costs and sets it further apart from CLS, which constructs the entire knowledge base (i.e., decision tree) after each incorrect prediction.

Reinke and Michalski (1986) carried the locality constraint a step further. Instead of recomputing a complete concept with each misclassification, their GEM system rederives only a small part of the concept. As in AQ, concepts are in DNF; if new observations are inconsistent with a concept, the faults are traced to individual conjunctive terms within the concept’s description. Each faulty term is submitted to a generalization procedure along with the observations it currently covers and those that triggered inconsistency. However, unlike AQ, GEM saves and may reuse all previous observations.

Reinke and Michalski empirically compared their method with a nonincremental version of AQ in three domains. Each experiment measured con-
cept description complexity, classification performance, and computational expense. The results tentatively indicate that: (a) the incremental method yields more complex concept descriptions than the nonincremental method, (b) the incrementally formed concept descriptions classify novel observations almost as well as the nonincremental ones, and (c) knowledge base update is less expensive using the incremental method than with the nonincremental method.

CLS, incremental AQ, and GEM differ in the extent to which they repair a faulty knowledge base: the entire knowledge base, entire concept descriptions, or partial concept descriptions, respectively. By reducing the scope of repair, these systems gain computational advantages over their nonincremental counterparts. In addition, each of these systems forms logical concept descriptions (CLS decision trees are tree-structured DNF concepts), and not coincidentally they insist on perfect consistency between the knowledge base and the environment. To maintain flexibility during incremental learning, they retain observations so that inconsistent portions of the knowledge base can be recomputed following each misclassification. Similarly, a system by Michalski (1985) retains observations that are misclassified by a knowledge base of production rules. In contrast, Winston’s (1975) well-known system incrementally learns conjunctive concepts without retaining observations. However, Mitchell (1982) and Vere (1980) point out that Winston’s system cannot insure consistency between a learned concept and observations.

Schlimmer’s (1987a) STAGGER system (also Schlimmer and Granger, 1986a, b) is a recent addition to the line of incremental learning from examples systems. Like these previous systems (except CLS), STAGGER builds a knowledge base of ‘flat’ concept descriptions and makes local knowledge base repairs. However, STAGGER departs in significant ways from these earlier systems. In part, these differences are motivated by the fact that real-world systems must be resistant to ‘noise’ (i.e., incorrectly described observations due to faulty perception). As such, the system does not insist on perfect consistency between the knowledge base and the environment, nor does it make abrupt repairs following each misclassification. Rather, repair is triggered by a variable amount of inconsistency. To implement this strategy, STAGGER represents concepts as a probabilistic summary of important concept subcomponents. Like GEM, it is these components that are subject to repair, but repairs are made conservatively – only after a num-
ber of misclassifications indicate that revision is appropriate. Repairs are
made by chunking primitive components, an ability that adds new terms to
the concept language and improves learning (Schlimmer, 1987b). Even after
making a repair though, the revised knowledge competes with the previous
representation and is retracted if new observations prove the repair unwar-
ranted. STAGGER does not reuse observations to effect repairs as do earlier
systems. Rather, probabilistic information summarizes the training set and
guides repair.

Computer experiments demonstrate that probabilistic representations and
a conservative revision strategy enables STAGGER to deal effectively with
noise. Furthermore, these characteristics enable the system to discern and re-
respond to long-term environmental changes. STAGGER is relatively novel in
addressing the problem of tracking environmental drift. Not coincidentally,
other systems that address this problem (Hampson & Kibler, 1983; Holland,
1975; Langley, 1987a) also rely on the flexibility afforded by probabilistic
representations.

A system that appears quite different than STAGGER at a cursory level,
but that draws important principles from it, is Schlimmer and Fisher’s (1986)
ID4. ID4 descends from CLS by way of Quinlan’s (1986) ID3, and like CLS,
ID4 constructs decision trees. Its control structure is similar to CLS’s, but it
uses a more sophisticated evaluation measure for selecting the ‘best’ divisive
attribute: the ‘best’ attribute maximizes the expected information gained
from the attribute’s values. Intuitively, this reflects how confidently one can
predict an observation’s class by knowing an attribute’s value.

ID4 is incremental and updates a decision tree with each new observation.
At the core of this incremental ability is the observation that the information-
theoretic evaluation measure need not be computed directly from the set of
observations, but a probabilistic summary of the observations is sufficient
(i.e., co-occurrence counts for all attribute-value/class membership combina-
tions). As with STAGGER, the use of a probabilistic representation frees
ID4 from saving observations. For initial division at the root, the values
of each new observation are used to update the counts necessary for com-
puting the information measure. When a statistically significant comparison
between the attributes can be made (based on the chi-square measure), a
root attribute is chosen. The new subtrees are not constructed immediately
because no observations have been saved. Instead, after each subsequent
observation has updated counts at the root, it is routed to the appropri-
ate subtree to update the counts there before being discarded. Subtrees are gradually grown in this manner. Over time a new attribute may come to be preferred at the root of a subtree. In this case the current root is supplanted by the attribute with the superior information gain, and the nodes under the original subtree are discarded and regrown. Subtree repair (versus full-tree reconstruction as in CLS) illustrates the same locality principles as GEM and STAGGER do for flat representations.

Schlimmer and Fisher used formal and empirical methods to characterize ID4 in terms of learning quality and cost. Their general findings are that ID4 converges on decision trees equivalent in quality to ID3’s. The cost of updating a decision tree in ID4 is at worst logarithmic in the number of previous observations, while revolutionary application of ID3 (similar to full-memory CLS) requires polynomial time. Finally, ID4 may require more observations to converge on the same decision tree as the revolutionary application of ID3, but the ‘total work’ (number of observations × cost per observation) required to reach equivalent trees is considerably less for ID4.

Incremental learning systems exhibit great variety in the strategies they use for balancing the cost/quality tradeoff. CLS is a nonincremental system forced into frequently rebuilding a decision tree from scratch. Update costs are controlled by limiting the number of retained observations, but only at the expense of slowing learning speed. Incremental AQ and GEM lessen costs further by localizing knowledge base repairs, again based on saved observations. Work with AQ is novel in its use of ‘good’ concept representatives in knowledge base update. Schlimmer’s STAGGER localizes knowledge base revision as well, but drops the assumption that perfect consistency is desired or even possible. As a result, repairs are not triggered after each classification. ID4 is a descendant of CLS, but shares STAGGER’s assumptions and strategies: probabilistic evidence and conservative knowledge base repair are the main sources of learning robustness.

Section 2.3 imposes further structure on these different strategies and suggests some directions for future work. However, we first turn to conceptual clustering, which drops the requirement of a ‘teacher’ that preclassifies observations. Instead, the system must identify ‘useful’ classes. This new specification has some important implications for performance, knowledge base structure, and learning strategy.
2.2.2 Incremental conceptual clustering

Michalski (1980) proposed conceptual clustering as a means of discovering ‘understandable’ patterns in data. However, this definition does not specify a performance task that improves with learning (cf. Figure 1). Fisher (1987b, c) proposes that one such task is the prediction of unobserved properties. As such, Fisher’s COBWEB system forms classification trees that are intended to yield ‘good’ prediction along many attributes, rather than optimal prediction along a single ‘teacher’-defined attribute as in learning from examples. Despite COBWEB’s reduced expectations, Fisher (1987a, b) shows that in many cases prediction with a single COBWEB classification tree approximates the accuracy obtained from multiple, special-purpose ID3 decision trees.

An important difference between COBWEB and earlier conceptual clustering systems is that it is incremental – COBWEB integrates an observation into an existing classification tree by classifying the observation along a path of ‘best’ matching nodes. Like ID4, probabilistic summaries of previous observations are stored at each node, but the matching functions and the criteria used for subtree revision differ considerably. COBWEB uses the category utility function (Gluck & Corter, 1985) to guide classification and tree formation. Category utility bases its evaluation on all of the observation’s attribute-values rather than a single one, making COBWEB a polythetic classifier as opposed to a monothetic classifier (e.g., CLS and ID4). Similarly, COBWEB’s subtree revisions are triggered by considering prediction ability over all attributes, but concern for multiple attributes complicates subtree revision. In ID4 a subtree is simply deleted, but in COBWEB a deletion that benefits one attribute may be inappropriate for others. In response, the system identifies points in the tree for cost-effective prediction of individual attributes. These points are marked by normative (Kolodner, 1983) or default values that COBWEB dynamically maintains during incremental clustering.

COBWEB’s main contribution to the present discussion is that it maintains a knowledge base that coordinates many prediction tasks, one for each attribute. This is in sharp contrast to learning from examples systems where a knowledge base need only support one task. However, despite COBWEB’s greater knowledge base complexity, Fisher (1987c) argues that its tree structure is still too restrictive; more general structures like directed acyclic graphs
(DAGs) would yield better prediction. These structures are used by COBWEB’s main precursors, UNIMEM (Lebowitz, 1982) and CYRUS (Kolodner, 1983). Regrettably, neither UNIMEM and CYRUS is characterized in terms of prediction accuracy, the increased costs that maintaining a DAG is likely to incur, or the fundamental tradeoff that must exist between these dimensions.

2.3 Incremental Concept Learning: Our Approach

Induction identifies rules that classify environmental observations. Inevitably, there are problems of inconsistency – trivially, a new observation cannot be classified as both \( C \) and \( \neg C \). However, over the course of induction, inconsistencies are likely to arise. If a system simultaneously confronts multiple tasks in complex domains it must efficiently resolve inconsistencies and refine its knowledge. Thus, a computational framework for incremental learning must identify control strategies and knowledge base organizations that afford efficient and accurate update. The following framework addresses these issues and draws from the computational studies of Section 2.2.

2.3.1 Search control strategies for incremental learning

With the exception of CLS, the evolutionary strategy advocated by Michalski (1985) and others (e.g., Schlimmer & Fisher, 1986; Simon, 1969) is reflected in the learning systems of Section 2.2. Each system limits computational requirements by making local changes to the knowledge base in response to misclassified observations. Over a sequence of observations, the knowledge base evolves into a set of concepts that consistently (or optimally) classify observations.

It is informative to view incremental learning in terms of a dominant AI paradigm: search. In particular, Mitchell (1982) has characterized a number of learning systems in terms of their search strategies (e.g., breadth-first, depth-first). Viewed as a search task, each of the systems of Section 2.2 maintains exactly one copy of the knowledge base throughout learning. As such, they eliminate the possibility of chronological backtracking as a mechanism for resolving inconsistencies. Rather, they rely on a control strategy much like hill climbing, which is computationally economical, but without chronological backtracking this strategy can lead to well-known problems (Rich, 1983). To avoid these problems, a belief revision-like process of dependency-
directed backtracking (Doyle, 1979) is employed to modify only those portions of the knowledge base that cause apparent problems.

CLS, incremental AQ, and GEM insist on perfect consistency between the environment and knowledge base. As such, they invoke dependency-directed backtracking with each classification, albeit inefficiently by TMS standards. In contrast, STAGGER and ID4 are motivated by the assumption that perfect consistency may not be realizable. Rather, optimal (perhaps perfect) prediction accuracy is the goal. Backtracking is not invoked following each misclassification, but only after some body of evidence suggests that such a move is warranted. STAGGER demonstrates that this strategy also allows the system to distinguish and track environmental changes. Viewed from a larger context, ID4 and STAGGER try to minimize the backtracking requirements of a problem-solving system that would depend on their learned knowledge (cf. Figure 1). This view also fits COBWEB, which dynamically identifies default values.

Limited experimentation (Reinke & Michalski, 1986; Schlimmer & Fisher, 1986) suggests that a hill-climbing/local-revision strategy approximates search-intensive systems in terms of concept quality, without the associated overhead of maintaining multiple copies of the knowledge base or of having to ‘recompute’ consistent portions of the knowledge base. Fisher (1987b, c) and Schlimmer (1987a) explicitly point to the effectiveness of this incremental strategy on computational grounds. Langley, Gennari, and Iba (1987) and Shrager (1987) imply that similar strategies model many aspects of human learning.

2.3.2 Concept and knowledge base organization

An effective control strategy must be coupled with an appropriate knowledge base organization. The knowledge base should facilitate the identification of sources of inconsistency and limit the cost of revision once inconsistencies are found. Attention to these issues is necessary or the cost advantages of the above control scheme may be negated. The systems of Section 2.2 illustrate two distinct biases towards knowledge base representation that, in turn, reflect three views of concept representation (Smith & Medin, 1981). The classical view assumes that concepts are represented by logical expressions of the necessary and/or sufficient properties of concept members. Probabilistic representations assume that concepts list important concept properties,
but qualify their individual importance with probabilities or other confidence measures. Last, *exemplar* representations are composed of specific observations – there may be no summary description at all. In large part, the latter two strategies are motivated by the inability of classical concepts to account for psychological findings that concept members differ in their perceived *typicality* (e.g., a *robin* is a more typical *bird* than a *penguin*).

In part, CLS, AQ, and GEM reflect the classical view towards concept representation. Each assumes that perfectly-consistent logical rules can be learned. However, by themselves, classical representations have problems dealing with the frequent inconsistencies that arise during incremental induction. Thus, these systems augment classical concepts with saved observations that can be used to recompute inconsistent portions of the knowledge base. With this hybrid classical/exemplar representation, a learning system can incrementally maintain consistency. In a trivial sense, revolutionary applications of nonincremental systems can be viewed as using this type of hybrid representation. Typically, as in AQ, incremental systems further limit the scope of repair and reduce costs by reusing only representative observations.

Despite the feasibility of using hybrid representations, we believe that such a strategy is less than optimal. Except for CLS, the flat structure of a classically-represented knowledge base probably makes recognition and repair computationally intolerable for all but the smallest knowledge bases. This also applies to strict exemplar models, which dispense with a summary concept representation entirely (Kibler & Aha, 1987). A second objection to the classical/exemplar hybrid is that it is motivated by an assumption that perfect consistency is possible and paramount, encouraging dependency-directed backtracking to be invoked following each misclassification. In real-world domains it may be preferable to alter the knowledge base conservatively based on evidence collected over a span of time. Of course, saved exemplars could be used to dynamically compute such evidence, but a cheaper alternative is to maintain summary evidence throughout learning.

Schlimmer’s STAGGER system rejects the classical/exemplar hybrid representation in favor of probabilistic concepts that allow cheap assessments of ‘average’ rule consistency. Probabilistic concepts allow aspects of an observation to be recorded without insisting on revision after every misclassification. This flexibility is highly desirable in an incremental learning system, and it accounts for STAGGER’s abilities to tolerate noise and track environmental changes. In this, STAGGER addresses one objection to classical/exemplar
hybrids, but its knowledge base remains ‘flat,’ and thus problems of inefficient recognition remain for all but the smallest knowledge bases.

Combining aspects of CLS and STAGGER, Schlimmer and Fisher’s ID4 incrementally builds decision trees. Probabilistic representations at decision tree nodes allow knowledge base repair to be conservatively applied. Furthermore, the decision tree enables an efficient identification of nodes (subtrees) where modification optimally affects decision tree quality. ID4 disposes of a subtree when it discovers a predictive rule that is superior to the current one. In this manner, the system attempts to maximize the correct number of classifications, while recognizing that it may not be possible to achieve perfect accuracy. A second reason to discard a subtree is if the current rule losses statistical significance (according to chi-square) in response to new observations – if there is no significant rule then classification should cease and a prediction can be made.

ID4 illustrates several principles of knowledge base organization, which when coupled with the control strategy of the previous section, promise to effectively support incremental learning. The tree structure of the ID4 knowledge base makes sources (rules) of inconsistency easy to locate. Once found, the implications of faulty rules are easily isolated. A tree structure is also used in Fisher’s COBWEB system, while DAG’s are used in Lebowitz’ (1982) UNIMEM and Kolodner’s (1983) CYRUS. Furthermore, COBWEB, UNIMEM, and CYRUS each identify nodes in the tree (or DAG) where certain predictions can be ‘optimally’ made. Fisher (1987c) illustrates how these points, labeled by normative values, serve roughly the same function as chi-square cutoff in ID4; normative values can be viewed as default values (Brachman, 1985) with probabilistic qualifiers that dynamically demarcate where predictions are justified.

STAGGER, ID4, and COBWEB indicate that hierarchically-structured probabilistic concepts are an effective knowledge base organization for incremental learning. Incremental systems that do not explicitly maintain such an organization will need to ‘compute’ aspects of the structure on demand. For example, CLS, AQ, and GEM recompute many of the statistical and logical dependencies that are continuously maintained in a hierarchical knowledge base. A good knowledge base organization is one that ‘hard-wires’ those dependencies that are most likely and/or most important. In this light, hierarchically-organized probabilistic concepts are efficient implementations of the classical/exemplar hybrids of earlier systems, rather than alternatives
to them (cf. Fisher, 1987c).

2.4 Plan for Continued Research

We are continuing our investigations of incremental learning in several directions: (1) combining domain-independent aspects of COBWEB, STAGGER, ID4, and other closely-related systems, (2) exploring incremental learning’s utility in application areas such as intelligent tutoring, and (3) analyzing incremental concept learning in terms of other theoretical frameworks such as belief revision. We elaborate specific projects below, listing journal articles to benchmark our progress when appropriate. We also expect conference (AAAI, IJCAI, Cognitive Science) publications to precede our journal submissions, but do not list them here.

2.4.1 Domain-independent incremental methods

One problem with ID4 is that under certain conditions it may discard subtrees and thus much of the information gleaned from past training. New observations will eventually allow ID4 to ‘fill in the gaps,’ but these additional costs may be avoidable. A recent extension to ID4, called ID4S, is similar to CLS in that a number of observations can be retained and reused. This represents a partial compromise with the classical/exemplar hybrid. However, ID4S reuses observations to occasionally rebuild subtrees, whereas CLS rebuilds the entire decision tree following each misclassification. Computer experiments with ID4S indicate that in the domain of chess end games (several thousand observations), few observations need to be saved (on the order of 25 – 100) to match the learning rate of revolutionary ID3. Recently, Utgoff (1987) has devised an extension to ID4, called ID5, that saves all observations. It shows how this store of observations can be exploited so that a faulty subtree can be reorganized and not simply rebuilt as with ID4S.

Work in the first year will extend ID4 in directions suggested by ID4S and ID5. These extensions will identify and save only the most ‘useful’ observations. This general strategy is inspired by Michalski and Larson (1978), but our work draws from a significant body of psychological research that indicates humans do not treat all observations as equally representative or typical (Mervis & Rosch, 1981; Smith & Medin, 1981). Typical instances tend to share many properties with members of the same class and have few
properties in common with members of different classes. To identify useful observations, ID4S will use collocation (Jones, 1983), a probabilistic measure that captures the within- and between-concept factors that effect typicality. In addition, we have verified (Fisher, 1987c; 1988) the consistency of collocation as a predictor of typicality with respect to human data (Rosch & Mervis, 1975).

Closely related to the problem of ‘remembering’ is the problem of ‘forgetting.’ ID3 uses the chi-square measure to prune portions of the knowledge base that do not help classification. This strategy can lead to a simpler knowledge base with little ill or even positive effect on prediction accuracy (Michalski, 1987; Quinlan, 1988). However, these results are in the context of nonincremental systems where many observations can be examined simultaneously. We will explore the utility of forgetting during incremental learning, where overly liberal strategies can have dilatory effects on learning speed. In fact, Fisher and Schlimmer (1988) have already found that a forgetting strategy based on chi-square is detrimental early in learning, though its benefits increase with training.

Efficient Induction of Decision Trees (Computational Intelligence, 1989)

This paper will describe results using an extension of ID4S that saves and reuses observations based on typicality. The major dimensions of a full-factorial experimental study include the relative utility of typical as well as atypical observations and the efficacy of concept simplification over the duration of learning. Revolutionary versions of ID3 will provide the baseline performance and cost characteristics against which ID4S will be compared.

A second project is to extend ID4S to track environmental changes. In particular, STAGGER modifies a concept definition to the extent that is minimally required to retain consistency after an environmental shift. Unfortunately, ID4S may be forced to recompute the entire decision tree should the importance of the root attribute be lessoned following changes in the environment. In this case, the utility of tree reorganization (cf. Utgoff’s ID5) becomes an attractive alternative to regeneration.

Tracking Environmental Change During Learning (Machine Learning, 1989)
The abilities of STAGGER and an ID4S/ID5 hybrid to track environmental changes will be reported. The relevant dimensions of this study include the ability of each system to minimize knowledge base modifications in response to change and the time to reconverge on an accurate knowledge base. The ability of each system to distinguish ‘noise’ from long-term changes will also be analyzed.

In addition to ID4 and other systems that learn from examples, we will continue our investigations of conceptual clustering in the first year. In this context, Fisher’s COBWEB will serve as the initial model; it builds probabilistic concept trees that support good prediction along many (versus a single) attributes. In fact, prediction with a single COBWEB classification tree approximates the accuracy of multiple, special-purpose ID3 decision trees (Fisher, 1987a, c). COBWEB demonstrates how default or normative values can be dynamically identified to facilitate cost-effective prediction. In many ways, the use of normative values is analogous to the forgetting strategy of ID4. Despite COBWEB’s performance, a DAG would allow more attribute-to-attribute dependencies to be directly encoded in the knowledge structure. In general, a DAG increases the flexibility that is important during incremental learning, but at the cost of maintaining multiple classification paths. DAG’s are employed by Kolodner (1983) and Lebowitz (1982, 1987) in CYRUS and UNIMEM, respectively. However, neither system is characterized along dimensions such as learning speed, cost, or prediction accuracy, nor is there any indication that the dependencies that they capture are any way ‘optimal.’

Critical Reviews of Incremental Concept Formation Systems (Artificial Intelligence, 1989)

This paper will review reimplemented concept formation systems, including COBWEB, UNIMEM, CYRUS, and EPAM (Feigenbaum & Simon, 1984). Important dimensions for comparing these systems include the prediction accuracies along all descriptive attributes, the time required to converge on stable classifications, and the cost of assimilating a single observation. The relative costs and benefits of DAG versus tree structured knowledge bases will be addressed and will motivate a second version of COBWEB, which forms DAG-structured classifications.
The empirical studies above will shed light on a number of fundamental issues of knowledge representation and learning.

An Analysis of Knowledge Representation Schemes (Cognitive Science, 1990)

This paper will analyze the inherent tradeoffs in different knowledge representation schemes, including the classical, probabilistic, and exemplar schemes, as well as more sophisticated approaches like decision trees and probabilistic concept trees and DAGs. The paper will characterize these representation schemes in terms of pliability during incremental learning, prediction accuracy under ideal and noisy conditions, and the understandability of learned knowledge. This analysis will rely on the empirical studies reported in earlier papers. Concept simplification, default value identification, and tracking will emerge as important secondary issues. In addition, the paper will address the psychological (e.g., typicality effects) plausibility of these schemes. We have intentionally downplayed the cognitive modeling aspects of our work, but it has nonetheless played an important role in our respective dissertations (Fisher, 1987c; Schlimmer, 1987a).

2.4.2 Application areas for incremental learning

As we proceed, we will test our general learning methods in a number of application areas. For example, STAGGER’s ability to incrementally introduce new terms through chunking is important for adaptive problem solving. Quinlan (1983) recognized the need for this capability in learning to recognize chess-end games. We plan to highlight and extend STAGGER’s ability in the context of Samuel’s (1967) Checker’s program.

Augmenting the Concept Language During Game-Playing (Man-Machine Studies, 1989)

Samuel’s original Checker’s program learned by adjusting the weights of a linear equation of primitive board features. Later versions overcame the fundamental weaknesses of this approach by using a signature table, which explicitly ‘hard-wired’ feature combinations. This paper will report on STAGGER’s ability to incrementally add new terms (i.e., chunks) to the feature description language, thus overcoming the need to manually derive these feature combinations.
Intelligent tutoring (Wenger, 1987) is another area in which we will test our systems. While an effective tutor must adapt along a number of fronts, we are currently focusing on the problem of incremental student modeling.

**Incremental Student Modeling (Man-Machine Studies, 1990)**

Langley, Ohlsson, and Sage (1984) used ID3 to construct a student model from buggy behavior. Their system accounted for many of the subtraction "bugs" cataloged by VanLehn (1982). Unfortunately, ID3 is nonincremental, and as observations were accumulated lag times became significant. We plan to incorporate ID4S into the student modeling process. The primary aim is to reduce lag time without significantly affecting the model’s accuracy.

Finally, interactive knowledge acquisition is an area that would benefit from incremental learning techniques. For example, Bareiss and Porter (1987) have developed an expert 'apprentice' in the area of clinical audiology. Their PROTOS system represents a novel approach to building expert systems, as it combines aspects of traditional knowledge acquisition techniques and automatic induction approaches (Michalski & Chilausky, 1981; Quinlan, 1986). Abstracting somewhat, PROTOS is an incremental learner of a DAG-structured knowledge base. Currently PROTOS relies heavily on the expert to define the organization of knowledge, though there are a few internalized principles of knowledge base design that allow it to operate in domains where expertise is harder to come by. Of pragmatic and psychological interest is the evolution of the basic level during an expert’s training. Psychological studies indicate that there is a basic level of classification where performance on certain inference tasks is maximized. Basic level studies motivated many of the design decisions of COBWEB (e.g., category utility) and they may suggest important principles for classification-based expert systems. Our research on extending COBWEB to construct DAG knowledge bases will enable us to address this issue.

**2.4.3 Frameworks for understanding incremental learning**

We have viewed incremental learning in terms of search and belief revision. Our analysis has been cursory given the simplistic nature of knowledge bases constructed by most inductive programs. While systems like COBWEB,
UNIMEM, CYRUS, and PROTOS are introducing organizations of greater complexity, the important characteristics and tradeoffs associated with them are not well understood. Paralleling our empirical studies, we will flesh out the relationship between inductive learning and belief revision techniques. Undoubtedly, there will be advantages for each field.

Belief revision techniques can suggest how to scale up inductive learning methods. For example, while STAGGER and a few other systems can track environmental drift, they do so by ‘destroying’ previously-learned conceptualizations. However, this strategy can undo useful information – we do not want to have to relearn important lessons about driving as we cycle between Summer and Winter. To our knowledge, no incremental learning system can track change without destroying prior knowledge.

*Concept Learning, Belief Revision, and Concept Drift (Machine Learning, 1990)*

Research into the problem of tracking concept drift will look for alternates to the justification-based belief revision techniques (Doyle, 1979) in which our discussions have been implicitly framed. In particular, DeKleer’s (1986) assumption-based truth maintenance system (ATMS) allows the simultaneous existence of multiple ‘world views’ and suggests a viable framework in which to explore sophisticated tracking mechanisms. The abilities of STAGGER and ID4S will be extended to partition knowledge into distinct and internally-consistent conceptual blocks. Tracking will also be extended to COBWEB, which must simultaneously support multiple prediction tasks.

Finally, our work on inductive learning assumes that perfect consistency may not be attainable. In theory, truth maintenance systems may be able to deal with noisy domains, but not in an efficient manner. Knowledge base organizations that exploit probabilistic representations and conservative revision strategies may lead to parsimonious and efficient belief revision methods. A unification of induction and belief revision mechanisms will also shed light on the continuum between data-driven and knowledge-driven (e.g., explanation-based) learning methods. Work on these issues will undoubtedly be well underway by the end of the second year, but will most likely not reach fruition until afterwards. Thus, they will be a natural basis for extensions to this proposal.
2.4.4 Methodological Approaches

Before concluding, we want to explicitly address the methodological biases that have guided the research proposed here. This is an important aspect of our research plan since it conveys the extent to which our results can be generalized and applied.

The surest way to characterize system behavior is through formal analysis (e.g., proofs of correctness), and these exist for a number of search-intensive concept learning systems (Hayes-Roth & McDermott, 1978; Vere, 1980); if a concept space is exhaustively searched, then it is not surprising that concepts of appropriate forms will be found. However, incremental processing is complicated by a cost/performance tradeoff. Recently, analytic techniques for characterizing probably approximately correct learning have been developed (Haussler, 1987; Valiant, 1985) that may be more amenable in characterizing incremental methods. Generally though, analytic methods result in statements of worst case and global performance that cover a wide range (often all) of the possible inputs. In many cases they hide fluctuations in behavior over subranges of input. For these reasons, our studies of incremental learning have been, and will continue to be, predominantly empirical.

While analytic statements are sure, empirical characterizations can differ radically in the accuracy with which they portray a system’s performance. Testing a system in very few domains may leave doubt as to the generality of the system’s apparent success. To understand the limitations and applicability of a method, we must determine the source of the method’s effectiveness. Like Simon’s (1969) metaphorical ant, it may be that most of the interesting computation occurs as a simple method interacts with a complex environment; despite a researcher’s claim, there may be very little ‘power’ in the system alone. For these reasons, it is not sufficient to test a system in a number (even many) domains; it is necessary to show that the domains are qualitatively different in important respects. Only in this manner can we fully characterize the spectrum of a system’s ability.

Fisher (1987a, c) and Schlimmer (1987a) have employed two techniques for empirically characterizing incremental learning systems. One technique computes a measure of ‘domain complexity’ (or regularity); in (Fisher, 1987a; Fisher & Schlimmer, 1988; Schlimmer, 1987a) this is a function of statistical interdependencies within a domain. A second approach is to construct artificial domains to delimit the conditions of system success and failure.
While it is important that systems effectively operate in natural domains, it is a formalized representation of the domain that is presented to the learning system, and this representation can be made arbitrarily complex or trivial. Artificial domains can be used to sweep away any confusion as to just how difficult a task is.

The use of ‘domain complexity’ measures, natural domains, and artificial domains can be combined into an effective empirical methodology. We will continue to use these techniques to characterize incremental learning cost and quality. This empirical methodology is relatively novel with respect to current practices in machine learning, and we believe it is a contribution in itself.
3 References


Workshop on Machine Learning (pp. 24–30). Irvine, CA: Morgan Kaufmann.


Schlimmer J. C., & Granger, R. H., Jr. (1986a). Beyond incremental processing: Tracking concept drift. *Proceedings of the Fifth National


