GALOIS : An order-theoretic approach to conceptual clustering

Claudio Carpineto and **Giovanni Romano**
Fondazione Ugo Bordoni
Via B. Castiglione 59, 00142 Rome (Italy)
fubdpt5@itcaspur.bitnet

Abstract

The theory of concept (or Galois) lattices provides a natural and formal setting in which to discover and represent concept hierarchies. In this paper we present a system, GALOIS, which is able to determine the concept lattice corresponding to a given set of objects. GALOIS is incremental and relatively efficient, the time complexity of each update ranging from O(n) to $O(n^2)$ where n is the number of concepts in the lattice. Unlike most approaches to conceptual clustering, GALOIS represents and updates all possible classes in a restricted concept space. Therefore the concept hierarchies it finds are always justified and are not sensitive to object ordering. We experimentally demonstrate, using several machine learning data sets, that GALOIS can be successfully used for class discovery and class prediction. We also point out applications of GALOIS in fields related to machine learning (i.e., information retrieval and databases).

1 INTRODUCTION

Much of recent work on conceptual clustering has focused on incremental construction of concept hierarchies from a collection of unlabelled objects (e.g., Lebowitz 1986, Fisher 1987, Hanson and Bauer 1987, Hadzikadic&Yun 1989, Langley 1991). Although these approaches differ in many respects (e.g., the concept representation, the hierarchy evaluation criteria, the use of the hierarchy for learning), they usually tackle the problem in the same manner, i.e. defining a set of hierarchy change operators and carrying out a hill-climbing search through a space of possible concept hierarchies aimed at finding a "good" hierarchy (Gennari et al 1989). The operators are applied whenever a new object is presented, and they either incorporate the new object into an existing class, or create new classes, or delete old classes, depending on evaluation of some heuristic criterion associated to the resulting hierarchies. These systems perform well in many experimental settings, but, because the criteria used to modify and evaluate the hierarchy are mostly local and/or heuristic, they have some drawbacks. One main problem is the lack of a clear semantics, by which to characterize or explain the whole hierarchy in terms of the object description. In fact, it may sometimes be difficult to understand why certain nodes have been generated and others have been omitted. Another related limitation is the sensitivity to order of presentation of objects. To illustrate these points consider the following collection of four objects (a, b, c, d), each described by four binary attributes:

a = (0 0 0 1)
b = (0 0 1 1)
c = (1 1 0 0)
d = (1 0 0 0)

If Fisher's COBWEB (Fisher, 1987) - taken as a representative of the hierarchical conceptual clustering systems - were presented with the sequence (a, b, c, d), it would produce, as could be easily checked, the hierarchy: (root (a b (c&d (c d)))).
This hierarchy, however, does not seem to reflect the actual object description, in that while it contains the class c&d it does not contain the perfectly symmetrical class a&b. Furthermore, if COBWEB were given the reversed sequence (d, c, b, a) it would produce a different asymmetrical hierarchy : (root (d c (b&a (b a)))).

In this paper we present an approach to conceptual clustering that helps overcome such limitations. Rather than finding and updating a particular subset (i.e., the output hierarchy) of the concepts, we keep and update *all* the classes that can be generated in a restricted concept language. This strategy is similar to the version space approach (Mitchell 1982) in concept induction from labelled examples. Our approach relies on a theory - the theory of concept (or Galois) lattices - which has received little attention in artificial intelligence approaches to clustering (Carpineto&Romano 1992). We believe it offers a formal and natural tool for restricting, representing, and ordering the set of concepts that can be induced over a collection of unlabelled objects. Its best features are that the concept hierarchies produced are globally related to the object description and that they are not sensitive to object ordering. In the paper we build on the theory of concept lattices, tackling the problem of its

automatization and use in a performance system. We define an algorithm for incremental construction of the concept lattice, implemented in a system named GALOIS[1], and show that its time complexity ranges from O(n) to $O(n^2)$ where n is the number of concepts in the lattice. We experimentally demonstrate that the resulting lattice permits to do both class discovery and class prediction with very good results.

The rest of the paper is organized as follows. First we introduce the basics of the theory of concept lattices. Then we describe the algorithm for incrementally finding the concept lattice and discuss its computational complexity. Next we evaluate GALOIS' learning ability on several domains (soybean, congressional voting, breast cancer, iris). Finally we outline extensions and further applications of GALOIS, such as automatic keywords classification in information retrieval and inferencing of implication rules in databases.

2 THE CONCEPT LATTICE: BACKGROUND

A context is a triple (O,D,I) where O is a set of objects, D is a set of descriptors, and $I \subseteq OxD$. We write *oId*, meaning the object *o* has the descriptor *d*. We can think of the set of descriptors associated with an object as a bit vector. Each bit (descriptor) corresponds to a possible value of a particular attribute and is on or off depending on whether an object has that value for that attribute.
A concept is considered to be determined by its extent and its intent: the extent consists of all objects belonging to the concept while the intent is the collection of all descriptors shared by the objects. For $A \subseteq O$ and $B \subseteq D$, define

$$A' = \{d \in D \mid (\forall o \in A)\ oId\}$$
$$B' = \{o \in O \mid (\forall d \in B)\ oId\}$$

so A' is the set of descriptors common to all objects in A and B' is the set of objects possessing the attributes in B. Then a concept of the context (O,D,I) is defined to be a couple (A, B) where $A \subseteq O$, $B \subseteq D$, A' = B, B' = A. The extent of the concept is A while the intent is B. Therefore only some couples (A,B) - i.e., the couples that are complete with respect to I according to the given definition - represent admissible concepts. Note that a subset B of D is the intent of some concept if and only if B"=B in which case the unique concept of which B is an intent is (B',B); a dual statement holds for the extent of a concept.
The set of all concepts of the context (O, D, I) is denoted by C(O, D, I). An ordering relation ($\leq$) is easily defined on this set of concepts by

$$(A_1, B_1) \leq (A_2, B_2) \quad \leftrightarrow \quad A_1 \subseteq A_2$$

or, equivalently, by

$$(A_1, B_1) \leq (A_2, B_2) \quad \leftrightarrow \quad B_1 \supseteq B_2.$$

[1]GALOIS has been implemented in Common Lisp on a Symbolics Lisp Machine.

C(O, D, I) along with $\geq$ form an ordered set. It can be characterized in the following way (see (Davey and Priestley 1990) for the complete theorem and its proof)).

The Fundamental Theorem on Concept Lattices (C(O, D, I) ; $\leq$) is a complete lattice[2] in which the least upper bound (*Sup*) and the greatest lower bound (*Inf*) are given by

$$Sup_{j \in J}\ (A_j, B_j) = ((\cup_{j \in J} A_j)'', \cap_{j \in J} B_j)$$
$$Inf_{j \in J}\ (A_j, B_j) = (\cap_{j \in J} A_j, (\cup_{j \in J} B_j)'').$$

So for any subset of concepts there is a least upper bound whose intent is given by the intersection of the intents of the concepts in the subset (a dual statement holds for the extent of a greatest lower bound).

If we consider again the context introduced earlier, consisting of the four objects a b c d, it is straightforward to see that independently of order of presentation the corresponding concept lattice is the (symmetrical) hierarchy: (root((a&b(a b)) (c&d(c d))). In table 1 a more expressive example representing a simple context for the planets of our solar system is shown. The corresponding concept lattice is illustrated in figure 1 (ss stands for size-small, my for moon-yes,etc).

	size			distance from sun		moon	
	small	medium	large	near	far	yes	no
Mercury	x			x			x
Venus	x			x			x
Earth	x			x		x	
Mars	x			x		x	
Jupiter			x		x	x	
Saturn			x		x	x	
Uranus		x			x	x	
Neptune		x			x	x	
Pluto	x				x	x	

Table 1: A context for the planets

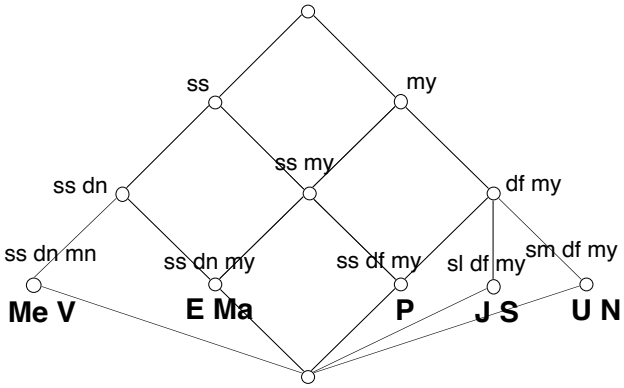

Figure 1: The concept lattice of the planet context

[2]Recall that given a non-empty ordered set P, if for all S1P there exist a least upper bound and a greatest lower bound then P is called a complete lattice.

In the graph representing the concept lattice, also called Hasse diagram, there is an edge between two concepts if they are ordered and there is no other intermediate concept in the lattice. The top concept contain all objects and is defined by their common descriptors (none in this case), the bottom concept is defined by the set of all descriptors and contains no objects. The labels in fig.1 indicate the intent of each concept; for the most specific concepts the extents are also shown. Note that many couples are not admissible in this context; the concept *df*, for instance, does not belong to the lattice in that all objects having df have also my. Note also that the intent (or, dually, the extent) of a concept in the lattice is sufficient to characterize the concept.

3 AN ALGORITHM FOR THE INCREMENTAL DETERMINATION OF THE CONCEPT LATTICE

Determining the concept lattice along with the Hasse diagram is a computationally difficult problem (Guénoche 1990). Furthermore the algorithms that have been proposed are - with one notably exception (Godin et al 1991) - non-incremental, whereas an important requirement posed by many real problems, including most conceptual clustering tasks, is incrementality. In this section we describe the incremental algorithm used by GALOIS to build the concept lattice. We start by informally discussing the effect of adding a new object to a given lattice. First of all we have to note that old concepts are never removed from the lattice. In fact, either they are incomparable to (the concept corresponding to) the new object and therefore are not affected by its introduction, or they include (or are equal to) the new object and therefore have just their extent augmented by the new object. In addition to modifying the extent of the concepts in which it is contained, the new object in general causes new concepts to be added to the lattice. This happens whenever the intersection of the new object with any set of *objects* with which it shares some attribute is not already present in the lattice. Any new concept introduced in the lattice has to be then consistently linked to the other concepts, including the other new concepts. This will in general cause the elimination of some edges (i.e., the edges between all couples of concepts C_1 and C_2 such that $C_1 > C_{new} > C_2$).

The algorithm we propose relies on two basic ideas. The first is that in order to find the concepts in the updated lattice it suffices to consider the intersections of the new object with the concepts[3] in the old lattice. The second is that the ordering of the old concept lattice can be exploited to delay the generation of each new concept and its relative edges until it is really necessary, thus avoiding much concept duplication and edge consistency checking.

[3]From now on, when we refer to a concept (sometimes called node) we usually mean its intent.

We shall explain both ideas in turn.

To generate the concepts in the updated lattice it is not necessary to intersect the new object with all possible combinations of the old *objects*. By contrast, intersecting the new object with the concepts in the old lattice is sufficient to guarantee that *all* updated concepts will be generated, including those resulting from the intersection of *new* concepts (the new concepts being those generated intersecting the new object with some old concept). This can be proved as follows.
Given any two new nodes C_1 and C_2 we have to show that there exist a (new) node C_k such that $C_k = C_1 \cap C_2$. Since C_1 and C_2 are new nodes then there exist two old nodes C_{11} and C_{22} such that
$C_1 = C_{new\text{-}ob} \cap C_{11}$, $C_2 = C_{new\text{-}ob} \cap C_{22}$,
where $C_{new\text{-}ob}$ is the node corresponding to the new object. We have
$C_1 \cap C_2 = C_{11} \cap C_{new\text{-}ob} \cap C_{22} \cap C_{new\text{-}ob} = (C_{11} \cap C_{22}) \cap C_{new\text{-}ob}$.
Since $(C_{11} \cap C_{22})$ belongs to the old lattice then $C_k = (C_{11} \cap C_{22}) \cap C_{new\text{-}ob}$.
The main loop of the algorithm therefore consists of adding the intersections of the new object with old concepts, and their relative edges, to the lattice.

At this point the second idea comes to place. If we were going to examine each node in the lattice independently of each other, then for any intersection with the new object we would have to spend much effort to check whether the intersection has already been generated. However, if we exploit the old lattice's structure we can avoid such operation. For each node in the lattice GALOIS compares the (intents of the) fathers of the node to the intersection of (the intent of) the node with (the intent of) the new object. Basically there are four possible cases:
1) there is a father < intersection;
2) there is a father = intersection;
3) there is a father > intersection;
4) all fathers are incomparable to intersection.
In case 1) GALOIS does not create the new concept, in that the same concept will be generated when we intersect the new object with the father of the node. In case 2) it does not create the new concept because it is already present in the lattice. In cases 3) and 4) it actually adds the new concept to the lattice. By doing so GALOIS is guaranteed to never generate two concepts equal. The prove goes as follows.
Consider two concepts C_1 and C_2 such that the intersection with the new object is the same:
$C_1 \cap C_{new\text{-}ob} = C_2 \cap C_{new\text{-}ob}$.
It follows:
$C_1 \cap C_2 \leq (C_1 \cap C_2) \cap (C_{new\text{-}ob} \cap C_{new\text{-}ob}) =$
$(C_1 \cap C_{new\text{-}ob}) \cap (C_2 \cap C_{new\text{-}ob}) =$
$(C_1 \cap C_{new\text{-}ob})$.
So $C_1 \cap C_2 \leq C_1 \cap C_{new\text{-}ob}$. Now, if $C_1 \cap C_2 < C_1 \cap C_{new\text{-}ob}$ then there exists a father of C_1 (i.e., C_1

$\cap C_2$) more specific than the intersection ($C_1 \cap C_{new\text{-}ob}$) and therefore the generation of the new node is ruled out by case 1); else, if $C_1 \cap C_2 = C_1 \cap C_{new\text{-}ob}$ then there exists a father of C_1 (i.e., $C_1 \cap C_2$) equal to the intersection ($C_1 \cap C_{new\text{-}ob}$) and therefore the generation of the new node is ruled out by case 2). In any case the new node is generated only once.

After a new node has been added to the lattice we have to create the relative edges. For this problem there is no simple solution because the new node has to be consistently linked also to the other new nodes, and the new nodes to be added to the lattice cannot be generated in an ordered fashion. The procedure GALOIS uses (which will be referred to as LINK) works on the current lattice, i.e. it considers also the nodes that have been added to the input lattice until that point. It determines for any new node two boundary sets, the lower boundary set S containing the most general concepts more specific than the new node, and the upper boundary set G containing the most specific concepts more general than the new node. Then it links the new node to each element in S and G, and remove the edges between S and G (if any). The sets S and G are determined considering only a subset of the whole current concept lattice; because the new node is the intersection of the new object with an old concept, the concepts to which it may be linked have necessarily to be more general than the old node or more general than the new object.

In table 2 the complete description of the algorithm is given. For each new object the algorithm is initialized adding the concept corresponding to the new object to the lattice. The function LINK realizes the procedure we described above; it takes as arguments the concept to be linked and the two concepts which determine the subset of the *current* concept lattice to be searched. To see the algorithm at work, consider again the concept lattice shown in fig. 1 and suppose to introduce the new planet X?=(size-medium,distance-near,moon-no). GALOIS returns the updated lattice shown in figure 2. The detailed working is as follows. All the concepts in the old lattice having an empty intersection with X? are processed by case a) and do not modify the lattice. There are only four remaining concepts: UN, EMa, MeV, ss-dn. Concept UN (case f) returns the node sm, the sets S=[UN, X?], G=[top] and the relative edges (UN,sm), (X?,sm), (sm,top). Concept EMa (case c) does not modify the lattice. Concept MeV (case f) returns the node mn-dn, the sets S=[MeV, X?], G=[top] and the edges (MeV,mn-dn), (X?,mn-dn), (mn-dn,top). Concept ss-dn (case f) returns the node dn, the sets S=[mn-dn, ss-dn], G=[top] and the edges (mn-dn,dn), (ss-dn,dn), (dn,top), *and eliminates the edge (mn-dn,top).*

Table 2: The algorithm for incremental update of the concept lattice

```
For each concept in lattice
    do
    intersection ¨ concept ( new-object
    select-case
        <<intersection = empty>>                          ;a
            do-nothing
        <<intersection = concept>>                        ;b
            do-nothing
        <<there is father(concept) < intersection>>       ;c
            do-nothing
        <<there is father(concept) = intersection>>       ;d
            LINK(father(concept), new-object, concept)
        <<there is father(concept) > intersection>>       ;e
            add intersection to lattice
            LINK(intersection, new-object, concept)
            for each father(concept) > intersection>
                eliminate edge (concept,father(concept))
        <<otherwise >>                                    ;f
            add intersection to lattice
            LINK(intersection, new-object, concept)
```

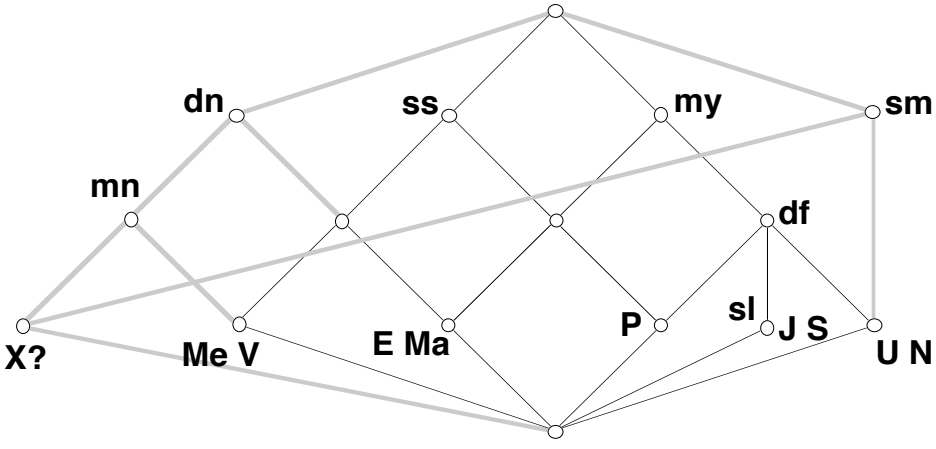

Figure 2: The concept lattice after the introduction of the new planet

4 COMPUTATIONAL COMPLEXITY

In this section we evaluate the space and time complexity of GALOIS.

4.1 SPACE COMPLEXITY

For the number of concepts present in the lattice there is a theoretical upper bound which is dependent on the number of descriptors. Assuming the attribute space used to describe the objects consists of m v-valued attributes, the conjunctive concept language on which GALOIS operates contain $(v+1)^m$ admissible concepts. For practical values of v and m this bound may be very high, so it is significant to see how the number of concepts varies with the number of objects. The experimental evaluations we have done on several machine learning datasets reveal that the growth of concepts ranges from linear to quadratic with respect to the number of objects. The lower bound was obtained with the dataset Iris (150 objects, 363 concepts), the upper bound with the dataset Soybean (47 objects, 3593 concepts).

4.2 TIME COMPLEXITY

For each object GALOIS iterates on the concepts in the lattice, which - assuming there are m binary attributes - are at most 3^m. In each iteration it may then have to apply the function LINK, which at most requires the examination of all concepts greater than the new object (2^m concepts).Therefore the worst-case time complexity is $O(3^m 2^m N) < O(3^{2m} N)$ where N is the number of objects. If the number of attributes is upper bounded, as it usually happens, the asymptotical behavior is linear in the number of objects. In practice, however, the constant factor relative to each update may be very large. While we have seen that it theoretically ranges from l to l^2 where l is the number of concepts in the lattice, it may be interesting to evaluate its experimental behavior. For this purpose we plotted the number of nodes examinated in each update by the key operation of GALOIS (i.e., determination of the lower boundary set S within the function LINK) as a function of the number of concepts in the lattice. We found (for the datasets Soybean, Congressional voting, Breast cancer, Iris) that although the complexity of each update varied between the two theoretical extremes with many oscillations, it never exceeded a small fraction of the square of the number of concepts. We have planned further experiments that may help clarify the actual behavior of the algorithm and may give clues as to reducing its overall complexity.

5 EMPIRICAL EVALUATION OF GALOIS AS A LEARNING SYSTEM

The concept lattice corresponding to a given set of objects may contain a large number of concepts. As all concepts in the lattice are formally justified they are equally good, in a sense. Also, their number monotonically grows as more objects are seen. Therefore GALOIS, like other hierarchical conceptual clustering systems, needs further information and/or concept selection mechanisms to be applied as a learning system. In this section we explore this issue, presenting the two current learning modes of GALOIS.

5.1 GALOIS FOR CLASS DISCOVERY

GALOIS can be applied to datasets containing objects which are assigned to two or more "natural" classes. When working in the discovery mode GALOIS takes as input growing samples of objects. Each sample reflects the natural distribution of the objects in the dataset, which is assumed to be known. For any sample GALOIS constructs out of the concepts in the lattice all possible partitions satisfying the given distribution. As we shall see experimentation demonstrates that as the sample grows the set of partitions may converge to the natural partition. Sometimes the system cannot find any partition with the given distribution. This kind of information is also useful because it reveals that the structure of the data set is such that the sought concepts lack logical descriptions.

5.2 GALOIS FOR CLASS PREDICTION

A typical experiment to evaluate a clustering system (e.g., Fisher 1987, Hadzikadic&Yun 1989, McKusick&Langley 1991, Anderson&Matessa 1992) is to let the system predict the class of new objects. The system is first given a training set of objects having their class labels hidden, so that concepts are determined as if the objects were unlabelled. Afterwards the system is given access to the class label of the objects stored under each concept and it may use this information to predict the class name of new unlabelled objects. We have provided GALOIS with different procedures for predicting class membership based on the properties of the dataset to which it is applied. They are illustrated in the following.

The two learning modes we have just described were evaluated on four datasets. The datasets we chose represent on the whole a difficult testbed for any inductive learning program, in that they feature most problems encountered with real data (e.g., multiple classes, missing attributes, noise, numeric attributes). We now describe in detail the result we obtained with each dataset.

5.3 SOYBEAN

We used the small Soybean data set, consisting of 47 objects (descriptions of soybean diseases) partitioned in 4 classes (disease categories). The distribution is (0.21,0.21,0.21,0.37). Each object is described by 35 attributes with no missing values. The number of partitions found by GALOIS in the discovery mode decreased from 279 to 4, including the natural partition.

We also ran GALOIS on the subsets of the small Soybean data set containing only two classes, and in these cases we obtained just the natural partitions. In the prediction mode we split the whole data set into two random halves and let GALOIS build the lattice on a half and predict the class on the other half. The procedure to assign objects to classes is as follows. After GALOIS has been trained it finds for each class the best consistent concepts (i.e., the concepts which are consistent and maximally complete) in the lattice. Then for any test object GALOIS computes the similarity[4] between the object and each best consistent concept and assign the object to the class with the highest score. We ran GALOIS on ten random samples. Usually the predictive accuracy on the test set reached its asymptote (100%) after few training objects. Only in one run it did not raise to 100% quickly (i.e., until the 21st training object).

5.4 CONGRESSIONAL VOTING

This data set consists of 435 objects described by 16 binary attributes with missing values (the votes of members of Congress on key issues in 1984). Objects are partitioned into two classes (Democrats and Republicans). When there is an attribute with missing value, as in this case, we give each occurrence of the missing value a unique, unmatchable value. We made the experiments on a subset containing 100 objects, with 50 random objects for each class. In the discovery mode, GALOIS became unable to find any partition with the given distribution after seeing the first small samples[5]. Therefore we relaxed the constraint on the degree of coverage of each concept in the partition introducing a tolerance value (Δ). We found that for small values of Δ GALOIS converged to a very limited number of partitions which were good approximation of the natural partition. To use GALOIS in the predicton mode we defined another mechanism for assigning objects to classes. In fact, the performances of the procedure used for the Soybean dataset degrade as the partitions found by Galois diverge from the natural partition (this is a case in point, although the results for this data set are still good). The new procedure is as follows. For any test object GALOIS finds all concepts in the lattice which are consistent *and more general than the object*. Next it assigns the object to the more numerous class in the previous set. The experiments were made presenting GALOIS with a random set of 100 objects (50 for training and 50 for test). We repeated the experiments ten times. The predictive accuracy on the test set ranged from 88% to 100% ($\sigma = 4.4$).

5.5 BREAST CANCER

This data set consists of 286 objects (clinical records) described by 9 attributes with missing values Objects are partitioned into two classes (recurrence and no-recurrence of breast cancer). The data set is intrinsically noisy, in that there are six pairs of objects with equal description and different class label. When we ran GALOIS in discovery mode on this data set the results were poor. Even introducing the tolerance value for the coverage degree, the results did not improve much, in that the partitions we found bore a little resemblance to the natural partition. By contrast, the prediction mode, which does not rely on existence of concepts consistent and complete, worked fairly well. We used a slightly modified version of the procedure we introduced above[6], and ran GALOIS on a random sets of 100 objects (50 for training and 50 for test) ten times. The predictive accuracies we obtained ranged from 72% to 86% ($\sigma = 5.3$).

5.6 IRIS

This data set consists of 150 objects (exemplars of Iris) described by 4 numeric attributes. The objects are partitioned into three classes (species of Iris). If we consider the numeric values in the object description as if they were nominal values then the concept lattice corresponding to the data set contains few objects, none of which cover more than 28 objects. Therefore it was impossible for the discovery mode of GALOIS to find partitions containing concepts with a sufficient degree of coverage.We tried to overcome this difficulty discretizing the attributes (thus increasing the object similarity). As a result, however, the concepts in the lattice did not increase.Neither did the results for class discovery improve. We found that strong discretizations (e.g., binary attributes) have the effect of reducing the concept space too much, whereas weak discretizations do not help increase object similarity. By contrast, discretizing (we partitioned the numeric range of each attribute in 10 intervals) turned out to be useful when we ran GALOIS in prediction mode. The training and test set used in the experiments were generated by randomly selecting 100 instances as training set and using the remaining 50 instances as test set. To predict the class membership we applied the same procedure used for the congressional voting data base. The predictive accuracy we obtained varied between 88% and 100% ($\sigma = 3.5$).

In figure 3 the complete results obtained for each data set are depicted. The results are averaged over ten runs The predictive accuracies we obtained are usually comparable to the best accuracies reported in the literature, including probabilistic clustering systems (see (Anderson&Matessa

---

[4]The similarity between an object and a concept is given by the number of concept descriptors equal to or more general than the object descriptors.

[5] Consider that a concept lattice is such that if there is no partition with a given distribution for a sample S, then there is no partition with the same distribution for any sample $S' \supset S$.

[6]In this data set, unlike the congressional voting data base, there are some inconsistently labelled objects. To account for this, when GALOIS searches the lattice to collect the concepts more general than the object it also considers partially inconsistent concepts, i.e., concepts that contain a small percentage (10%) of objects with a different class label.

1992) for a detailed comparison).

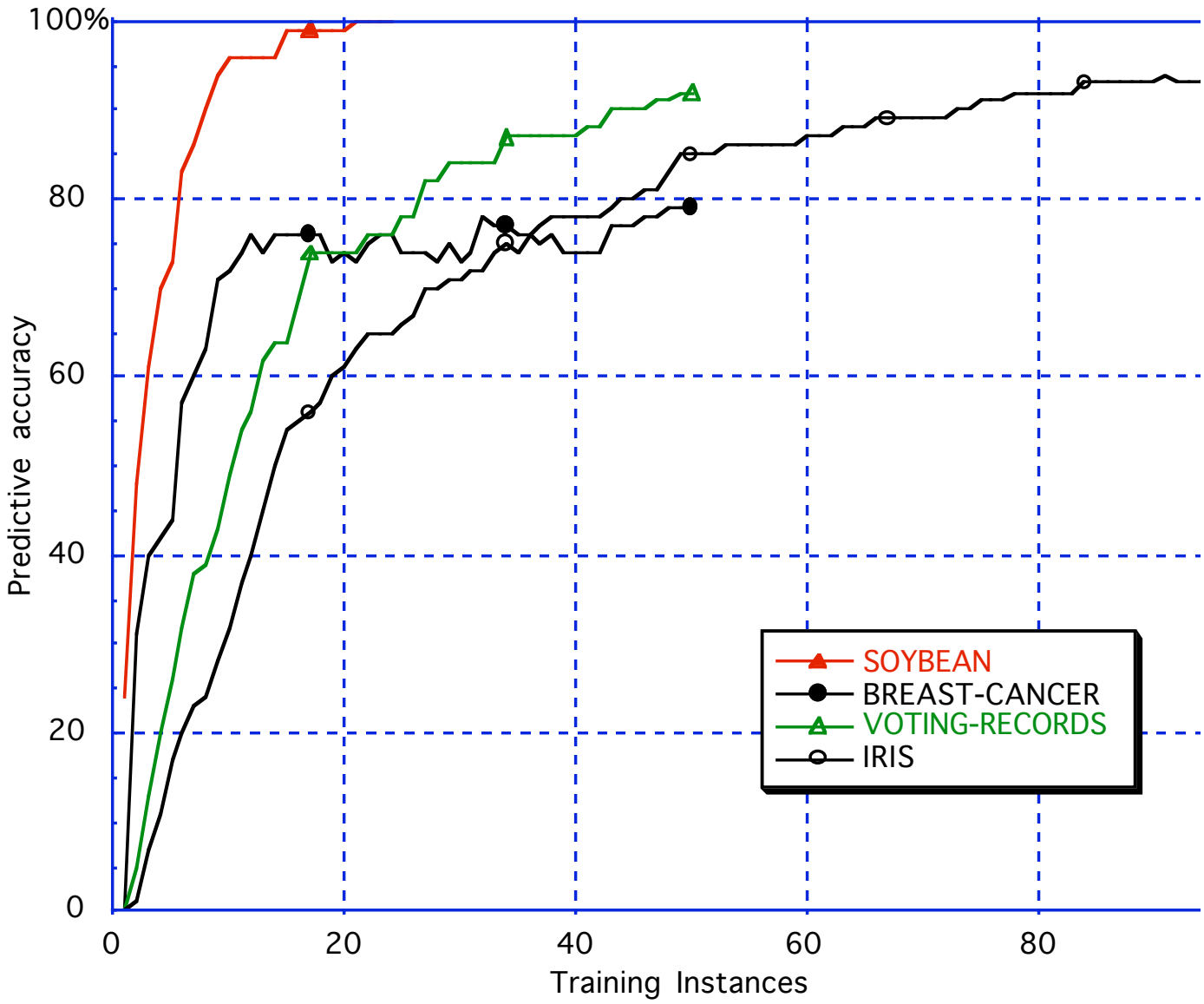

Fig.3. Prediction performance of GALOIS on four data sets

6 RELATED WORK

The only other algorithm for incremental determination of concept lattices we are aware of is reported in (Godin et al 1991). It is substantially different from GALOIS. It is based on partitioning the concepts in the lattice according to the cardinality of their intents, and subsequently processing the obtained subsets (at most m, where m is the number of attributes) in ascending cardinality order. While the asymptotic behavior of its time complexity is the same as GALOIS, in practice it may be less efficient. In fact, on one hand the sets in the partition are in general larger than the lattice subset explored by the function LINK in GALOIS:

$$(\max \|S_i\| = \max_i \binom{m}{i} 2^{m-i} > 2^m);$$

on the other hand, for each concept the algorithm in (Godin et al 1991) has to examine all elements in such set twice, once to check concept duplication and once to properly set the edges. By contrast, GALOIS applies the

function LINK at most once for each concept. Another major difference between this work and the work by Godin et al is application of concept lattices. They apply concept lattices to the information retrieval domain, where objects are documents, descriptors are keywords, and a context is the indexing relationship between documents and keywords. In this context the concept lattice is seen as an automatic keyword classification structure supporting a browsing interface. Another suggested application is discovery of implication rules between object attributes, which turns out to be closely related to the theory of functional dependencies developed in databases. However, they do not explore the potentials of concept lattices for inductive learning tasks, as we do.

The major advantages of GALOIS over other incremental hierarchical conceptual clustering systems (e.g., Lebowitz 1986, Fisher 1987, Hanson and Bauer 1987, Hadzikadic&Yun 1989, Langley 1991) are clarity and generality. A concept lattice represent a virtually "unbiased" set of clusters - i.e., all complete classes of objects having a conjunctive description - and has therefore a clear and simple semantics. Unlike the concept hierarchies produced by other systems, it can be easily explained and manipulated, and it lends itself to several applications. The applications include, but are not limited to, inductive learning tasks. The major disadvantage of GALOIS is its relative efficiency, especially when, as for the Soybean data set, the number of concepts in the lattice become very large. There are also other differences. While GALOIS' concepts are strict conjunctive logical descriptions, other systems employ more flexible formalisms, such as probabilistic concepts (Fisher 1987, McKusick&Langley 1991), contingency tables (Hanson&Bauer 1989), prototype schemata (Hadzikadic&Yun 1989). Also, constraints on the classification structure are different: while most clustering systems (one exception is Lebowitz's UNIMEM) can only find disjoint classes, GALOIS can also generate overlapping classes. This may be an important requirement for many applications, even though data sets with non-disjoint classes are rare in the machine learning field (in fact, we regret we could get none).

7 CONCLUSION AND FUTURE WORK

We presented GALOIS, a system for conceptual classification based on the theory of concept lattices. GALOIS implements a new incremental algorithm for determining the concept lattice corresponding to a set of objects. For each update the algorithm takes time ranging from O(n) to $O(n^2)$ where n is the number of concepts in the lattice. In the paper we focused on the application of concept lattices to conceptual clustering tasks, presenting several experimental results.

This work can be extended in several directions. We are currently investigating the following.

- Generalizing Galois lattices to include richer concept representations (e.g., conjunctive languages on tree-structured attributes).

- Adding domain knowledge to constrain the growth of the lattice or to improve the learning ability.

- Characterizing the relationships between the performances of the different prediction methods used by GALOIS and the properties of the data sets.

- Factorizing the concept language to reduce the complexity of the algorithm.

- Applying GALOIS to other tasks and/or domains.

Acknowledgements

We would like to thank Attilio Giordana and Lorenza Saitta for providing us with the data sets. We would also like to thank Jan Zytkow for his comments on an earlier version of this paper. This work was carried out in the framework of the agreement between the Italian PT Administration and the Fondazione Ugo Bordoni.

References

Anderson, J., Matessa, M. (1992). Exploration of an Incremental, Bayesian Algorithm for Categorization. *Machine Learning*, 9, 275-308.

Carpineto, C., Romano, G. (1992). Concept Lattices and Conceptual Clustering. In *Proc. of GAA-92 (Italian Workshop on Machine Learning)*, Rome.

Davey, B., Priestley, H. (1990). *Introduction to Lattices and Order*. Cambridge University Press.

Fisher, D. (1987). Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2, 139-172.

Gennari, J., Langley, P., Fisher, D. (1989). Model of Incremental Concept Formation. *Artificial Intelligence*, 40, 12-61.

Godin, R., Missaoui, R., Alaoui, H. (1991). Learning Algorithms using a Galois Lattice Structure. In *Proc. of the 1991 IEEE International Conference on Tools for AI.* San Jose, CA.

Guénoche, A. (1990). Construction du Treillis de Galois d'une Relation Binaire. *Mathématiques et Sciences Humaines*, 109, 41-53.

Hadzikadic, M., Yun, D. (1989). Concept Formation by Incremental Conceptual Clustering. In *Proc of 11th IJCAI*, Detroit.

Hanson, S., Bauer, M. (1989). Conceptual Clustering, Categorization, and Polymorphy. *Machine Learning*, 3, 343-372

Lebowitz, M. (1986). Concept learning in a rich input domain: Generalization-based memory. In R. Michalski et al. (Eds.), *Machine Learning: An Artificial Intelligence*

Approach 2. Morgan Kaufmann.

McKusick, K, Langley, P. (1991). Constraints on Tree Structure in Concept Formation. In *Proc.of 12th IJCAI*, Sidney.

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