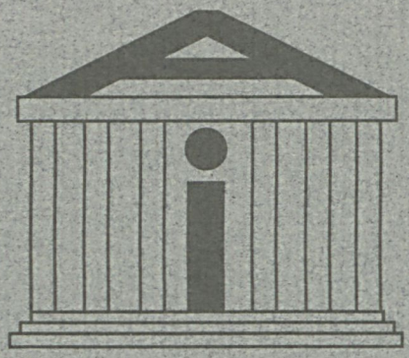
The Fifth International Conference on

Artificial Intelligence and Law



Proceedings of the Conference

May 21-24, 1995
University of Maryland
College Park, MD USA

Sponsored By:
The International Association for Artificial Intelligence and Law
University of Maryland Institute for Advanced Computer Studies
National Center for Automated Information Research

In Cooperation With:
ACM SIGART
American Association for Artificial Intelligence

Detecting Change in Legal Concepts*

Edwina L. Rissland and M. Timur Friedman
Department of Computer Science
University of Massachusetts
Amherst, MA 01003

Abstract

In this paper we discuss a general approach to detecting conceptual change developed in our on-going work on concept drift [Rissland et al., 1994]. We illustrate our approach on an actual, still evolving, legal example, the "good faith" concept in the law of personal bankruptcy. In our approach, we detect that a concept is changing by measuring changes in a structural representation of it, such as a decision tree.

1 Introduction

Concepts often change over time. For example, prior to the energy crisis in 1973 a desirable car was one that had a smooth ride and a roomy interior, whereas in the 80's and early 90's a desirable car was one that gets high gas mileage; more recently there is evidence that the concept is creeping back to a meaning that includes good gas mileage plus certain aspects of its earlier meaning, like roominess. This type of change has been called *concept drift* [Schlimmer, 1987]. It is ubiquitous.

Legal concepts are constantly-evolving and open-textured. Legal concepts cannot be defined by tight necessary and sufficient conditions, but rather must be represented, in part, through examples, that is to say, cases [Levi, 1949; Llewellyn, 1989; Hart, 1958]. As new cases are decided, a concept changes. For instance, Levi and others have described the birth, evolution, and maturation of the concept of "inherently dangerous" in the law of product liability [Levi, 1949, Radin, 1933]. Just about any legal concept—contract, unconstitutional warrantless search, income, good faith—shares the same evolutionary arc of refinement through the process of resolving the concept on specific cases. In time, the legal system "learns" what the concept means [Rissland & Collins, 1986]. The meaning is represented in common law areas primarily in terms of case exemplars; in statutory areas, the representation is a hybrid of rules and cases [Rissland et al., 1991; Rissland, 1994]. A single exception can open the flood gates to a wave of exceptions which permanently change the concept. For instance, a single Supreme Court case created the "automobile exception" to the Fourth Amendment's warrant requirement for a constitutionally acceptable search; this case forever changed the meaning of our Fourth Amendment, which is still evolving today [Rissland, 1989; Rissland & Collins, 1986].

In this paper, we first review a general approach to detecting concept drift that we have developed in on-going work on concept drift [Rissland et al., 1995; Brodley & Rissland, 1993]. We then apply it to the evolving concept of *good faith* as it applies to the issue of approval of a debtor's plan in personal (Chapter 13) bankruptcy law, an area we have used in our BankXX project [Rissland et al., 1993, 1994, 1995].

Our general approach is quite straightforward. We use a standard inductive algorithm, like ID5R, to induce representations of a concept from a stream of examples, that is, cases. (Cases are represented as case facts plus outcome in a standard feature vector format.) We use a measure—called the *structural-instability metric*—to measure structural change in the induced representations. By monitoring trends in values of the *structural-instability metric*, we can be alerted to episodes of possible conceptual change: an increasing trend signals change and a decreasing trend indicates stability (little or no change). We use a standard statistical test to determine whether the perceived change is significant enough to trigger a reaction to it. Then depending on whether we want to take an aggressive or wait-and-see approach, we can take steps either to generate a new concept representation immediately from scratch (based on the examples occurring after the onset of change) or to change the old one incrementally (using examples drawn from a window on either side of the change).

Incremental supervised learning algorithms, such as ID5R [Utgoff, 1989] and ITI [Utgoff, 1994], adjust the concept representation in response to each observed example. Other algorithms, like Quinlan's C4.5 [Quinlan, 1993], respond inductively to a set of designated training instances. However, none can detect when a concept drifts, that is, when the stream of examples begins to represent a new or a slightly altered concept definition. Of course, detection of conceptual change is not the job of such inductive algorithms: theirs is simply to induce a conceptual representation from a set of specified exemplars.

*This research was supported in part by the Air Force Office of Sponsored Research under Contract no. 90-0359.

2 Structural Concept Change

A concept can change in many ways. It can become more general or more specific, the relevance of attributes describing it can change, or the value of attributes can change. The change can be abrupt or gradual. One way to see the impact of such changes is to consider a decision tree representation of the concept.

In a decision tree, a node is either a leaf indicating a concept class or a test (on an attribute). For each possible outcome of a test, there is a branch to a sub-decision tree or a leaf. To classify an instance, one starts at the root and follows the branches indicated by the test outcomes until a leaf node is reached. The label at the leaf indicates the resulting classification: positive (+) for membership or negative (-) for non-membership in the concept class.

The following are six examples of concept change. Figure 1 shows the corresponding changes in their decision tree representations. Note that attributes not important to classification do not appear in the trees.

1. **Adding a disjunct generalizes a concept.**

   In Figure 1a, the concept changes from AB ∨ C on the left to AB ∨ C ∨ D on the right. Adding a disjunctive clause to the concept causes one or more nodes to be added to a decision tree.

2. **Removing a disjunct restricts a concept.**

   In Figure 1a, the concept changes from AB ∨ C ∨ D on the right to AB ∨ C on the left. Removing a disjunctive clause from the concept causes one or more nodes to be removed from the decision tree.

3. **Adding a conjunct restricts a concept.**

   In Figure 1b, the concept changes from AB ∨ CD on the left to AB ∨ CDE on the right. Adding a conjunction to a clause adds one or more nodes to the tree.

4. **Deleting a conjunct generalizes a concept.**

   In Figure 1b, the concept changes from AB ∨ CDE on the right to AB ∨ CD on the left. Removing a conjunction from a clause removes one or more nodes from the tree.

5. **The relevance of attributes can change.**

   In Figure 1c, the concept changes from AB ∨ C on the left to AD ∨ C on the right. The new concept has the same number of clauses, but the change in the relevance of attributes causes nodes to be both added and deleted from the tree.

6. **An attribute's value can be inverted.**

   In Figure 1d, the concept changes from AB ∨ C on the left to AB ∨ ¬C on the right.[1] Negation of an attribute in a clause causes the subtrees of the corresponding test to switch branches.

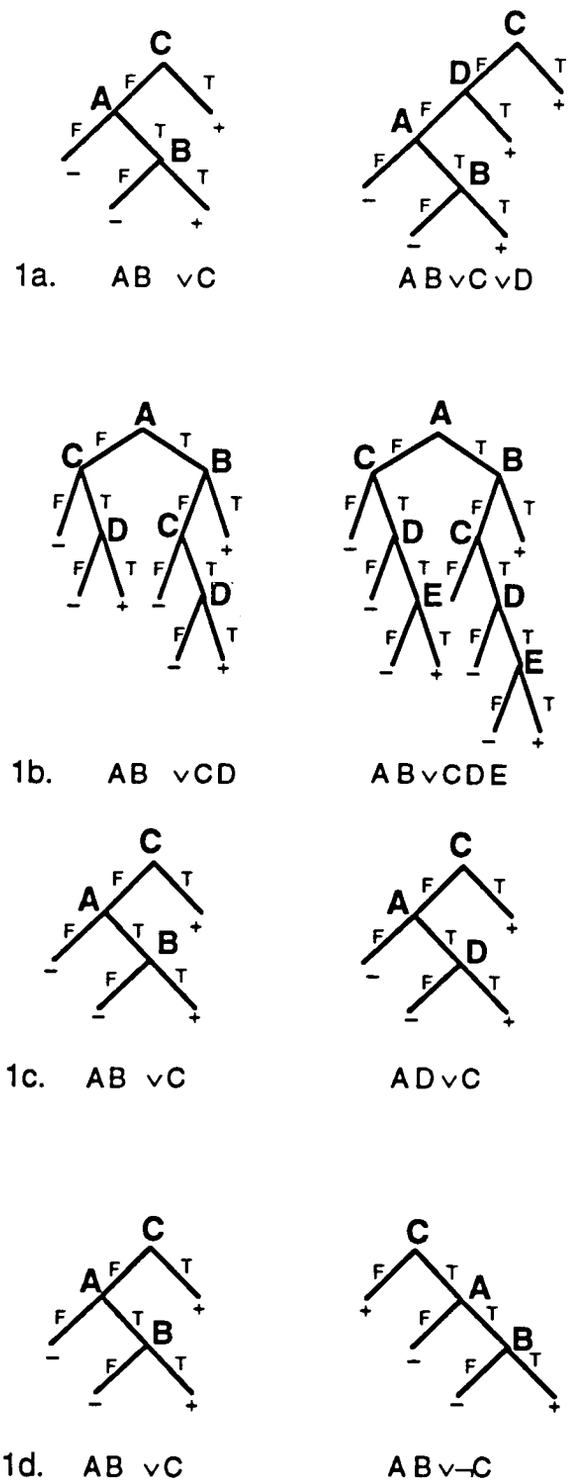


Figure 1. Decision trees before and after six types of conceptual changes.

[1] Although the tree on the right literally shows the concept as CAB ∨ ¬C, this is logically equivalent to AB ∨ ¬C.

Change in a concept can be assessed by examining changes in the structure of the concept's representation, particularly change in the organization and importance of attributes used to predict concept membership. For instance, we can examine the structural changes—the movement, appearance, and disappearance of attributes—in a decision tree representation for a concept. This can be done by examining the values that occur at each address in the old and new trees and taking note of which ones change. For example see Table 1, which presents the inventory of the changes that occur when a conjunct is negated. (This corresponds to the trees in Figure 1d.)

Address in AB ∨ C	Value at address	Address in AB ∨ ¬C	Value at address
1	C	1	C
1.1	A	1.1	+
1.2	+	1.2	A
1.1.1	–		
1.1.2	B		
		1.2.1	–
		1.2.2	B
1.1.2.1	–		
1.1.2.2	+		
		1.2.2.1	–
		1.2.2.2	+

Table 1. An inventory of the changes that occur when a conjunct is negated.

Some attributes of a concept are more indicative of concept membership than others. For example, if the concept to be learned is *chair* and each example is described by the attributes *number-of-legs* and *color*, then more than likely, the value of *number-of-legs* turns out to be more predictive of whether an object is a chair. Decision trees algorithms, for example, ID5R [Utgoff, 1989], ITI [Utgoff, 1994] and C4.5 [Quinlan, 1993], define a partial ordering of attributes according to their importance. They typically use an information theoretic measure to assess importance. In decision trees, more important attributes occur higher up in the tree.

We monitor conceptual change by measuring structural change in the concept representation, such as a decision tree representation. Monitoring changes in the ordering of the attributes by their importance for predicting concept membership leads directly to the idea of *structural instability* for a concept. Structural instability (or stability) is a measure of how much the ordering—that is, the concept's structure—changes in relation to the number of observed examples.

If the concept is *not* changing as the number of observed examples increases, then the ordering stabilizes and the concept stability increases, or, equivalently, the concept instability decreases. Note that initially an ordering will typically fluctuate quite a bit because when few examples have been observed it is difficult to ascertain which attributes are most important. See the schematic in Figure 2 (or actual plots of data shown in Figure 3.)

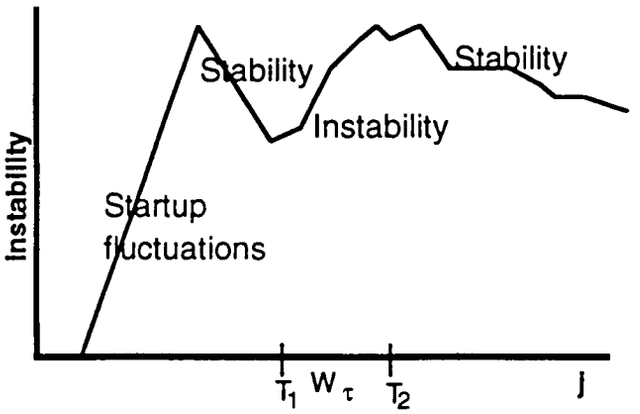


Figure 2. A typical plot of the *structural-instability metric* against, *j*, the number of examples.

Figure 2 shows the overall features of a typical graph of the *structural-instability metric* plotted against, *j*, the number of examples. T_1 is the time concept drift starts making itself felt in the structural-instability metric; its actual onset is earlier. T_2 is the time at which our algorithm detects it, based on the use of the Kendall's τ-test on a window of W_τ "time" steps (i.e., updates to the concept representation).

3 Measuring Structural Change

In this section, we present the ***structural-instability metric*** for measuring conceptual change and how to use it to detect concept drift in an ordered sequence of examples—called the *example-stream*—being presented to the learning algorithm. We tag examples by the order in which they are presented: $E_1, E_2, \ldots, E_{j-1}, E_j, \ldots$

The general scheme is this: At a certain point, a decision tree has been generated to represent the concept over a particular set of examples—say, the first *j-1* examples in the example-stream. Call this the *old* tree. Then a new example, E_j, is input and a *new* decision tree is generated based on the first *j* examples. We then measure the changes that occur between the old and new trees by comparing their structure. We do this by taking an inventory of the nodes that have appeared, disappeared, or changed labels, and computing a measure of change based on this inventory.

We keep track of these measures of change between pairs of old and new trees—the first and second, the second and third, the third and fourth, ... the *(j-1)th* and *jth*—as each new example is input, and maintain a running average of these pairwise measures. If the concept is stabilizing, the changes and hence the running average—called the *structural-instability metric*—will decrease. If the concept is becoming unstable, the values of the *structural-instability metric* will not decrease: they will stay flat, wiggle around, or even increase. An abrupt upward spike in the instability metric is a diagnostic clue that a major change is underway. See the schematic of Figure 2.

The *structural-instability metric* is defined as follows. First, we sum up all the changes at each level of the tree. Second, we compute a weighted sum of the changes over all levels of the tree. So as to give more weight to changes higher in the tree, the changes at the different levels are "discounted" according to how deep they are in the tree. Third, we maintain an average of these summations of change across each successive pair of trees spawned from the example-stream. We examine this running average for changes in its overall slope or trend.

In decision trees, since structural changes occurring higher up in the tree represent more important changes than do those lower down, we want to pay more attention to them. We do this by weighting the number of changes at depth (or level) i in the tree by a *discount factor*, f_i. We use the convention that the depth of the root node is 1. Typical discount factors are:

(1) $f_i = (N+1-i)/N$ where N is a constant

(2) $f_i = a^{(1-i)}$ where a is a constant

Note that the maximum value of these coefficients is 1 and is achieved when $i = 1$, that is, at the level of the root node. The value 0 is achieved at depth $N+1$ in the first coefficient but never in the second. ($N+1$ is the level of the leaf nodes in a tree with $N+1$ levels.) Note that if we do not wish to allow the first discounting coefficient ever to be negative, we can choose to define $f_i = 0$ for i greater than $N+1$.

When actually using these discount factor, one needs to choose the constant N for the first factor and the constant a for the second. Different choices lead to different discounting schemes: that is, how much attention is paid to changes at different levels.[2]

For instance, we can set N equal to the number of attributes, and then $N+1$ is the maximum possible depth of any leaf node. Or we could take N to be a different number such as the average depth of trees. This latter choice is very useful where the number of attributes is large—as in our bankruptcy example where there are 61 attributes—and where using that number for N would lead to a minuscule rate of decrease ($1/N$) in the discount factor as one goes down the levels of the tree. This would not allow much differentiation between the weights at the different levels.[2]

When we update an *old* tree to produce a *new* tree, we define the ***discounted-changes at depth i*** as:

$$DChange_i \text{ (old tree, new tree)} = f_i * k_i \text{ (old tree, new tree)}$$

where f_i is a discount factor and k_i *(old tree, new tree)* is the number of changes that occur at depth i between the old and the new tree. With an appropriate discount factor, the contributions of changes higher up in the tree can be given (much) greater weight than those lower down.

To measure the change in the entire tree representation, we need to sum up the discounted changes at each depth between the old and new versions of the decision tree down to a desired depth. The ***Total-change*** to depth M is defined as:

$$\textit{Total-change (old tree, new tree)} = \sum_{i=1}^{M} Dchange_i \text{ (old tree, new tree)}$$

The upper bound M of the summation specifies how far down into the tree we wish to take change into account. (Of course, the right choice of discount factor can take care of this as well by zeroing out or greatly discounting contributions below certain levels.) In a tree based on N attributes, where the maximal depth of any leaf node is $N+1$, and where we wish to consider changes throughout the tree, setting $M = N+1$ is a natural choice.

Table 2 shows how to compute the *Total-change* measure for each of the six examples of Figure 1. In these examples, there are at most 5 attributes and thus:

- the maximum depth of any test node in the trees is 5
- the maximum depth of any leaf in the trees is 6.

We use the first discount factor with N=5 and the limit M=6 in the summation.[3]

Columns 1-5 show the calculation for $DChange_i$ at each level in the tree. The last column, shows the value of the *Total-change* measure for each type of change we showed in Figure 1. The changes at level 6 are not shown, since they are all discounted to 0, and thus $Dchange_6 = 0$ for all rows.

Type of Change	level=1 (root level)	level=2 $Dchange_2$	level=3 $Dchange_3$	level=4 $Dchange_4$	level=5 $Dchange_5$	*Total-change*
Add a Clause	0	1*(6-2)/5	2*(6-3)/5	4*(6-4)/5	2*(6-5)/5	**4.0**
Delete a Clause	0	1*(6-2)/5	2*(6-3)/5	4*(6-4)/5	2*(6-5)/5	**4.0**
Add an Exception	0	0	0	1*(6-4)/5	3*(6-5)/5	**1.0**
Delete an Exception	0	0	0	1*(6-4)/5	3*(6-5)/5	**1.0**
Relevance Change	0	0	1*(6-3)/5	0	0	**0.6**
Clause Negated	0	2*(6-2)/5	4*(6-3)/5	4*(6-4)/5	0	**5.6**

Table 2. *Total-change* computed using N=5, M=6, and discount factor $f_i = (N+1-i)/N$ for the examples in Figure 1.

[2] With N=61, f_1=1, f_2=60/61, f_3=59/61, etc.

[3] Note, since $f_6 = 0$ for the first discounting factor ($6 = N+1$), this effectively sums changes for levels i=1 through i=5, that is, effectively, M=5.

For example, if a disjunctive clause is added to the concept (Figure 1a), then one node changes at level 2 and $Dchange_2$ = 1*(6-2)/5. At level 3 a leaf changes to a test and a test changes to a leaf, so $Dchange_3$ = 2*(6-3)/5. At level 4, two leaves are removed, a leaf is added and a test is added, thus $Dchange_4$ = 4*(6-4)/5. At level 5 two leaves are added, $Dchange_5$ = 2*(6-5)/5. *Total-change* for the decision tree is the sum $DChange_i$ for $i=1, \ldots, 6$, and in this case, is equal to 4.0.

To determine if a concept representation is becoming stable one looks for decreasing changes in the structural instability metric. That is, a stable concept, with respect to some interval of "time" is one where the average *Total-change* decreases over that interval.

In order to define the *structural-instability metric*, we need to rewrite the above formulae with the use of some additional subscripts in order to index which trees we are taking into consideration when calculating $DChange_i$ and *Total-change*. For instance, we write the changes between the *(j-1)th* and *jth* trees that occur at level *i* as $DChange_{i,j}$

We define $Total\text{-}change_j$ as the measure of change between the (old) tree based on *j-1* instances and the (new) tree based on the inclusion of the *jth* instance. We compute $Total\text{-}change_j$ by summing up the changes at each level we wish to consider.

$$Total\text{-}change_j\ ((j\text{-}1)th\ tree, jth\ tree) = \sum_{i=1}^{M} DChange_{i,j}\ ((j\text{-}1)th\ tree, jth\ tree)$$

To compute structural instability we take into account the amount of changes in pairs of decision trees that occur as each new example is presented. To this end we average the $Total\text{-}change_j$ in the example-stream from the first (j=1) through the t^{th} (j=t) instance:

$$Structural\text{-}instability\ [1, t] = \frac{1}{t} \sum_{j=1}^{t} Total\text{-}change_j$$

We divide by t, the total number of examples observed thus far, in order to create an average measure. Note the first calculation of the metric must wait until t instances have been seen.[4] Because the number of changes in the tree is normalized by t (the number of observed examples), as t increases the instability will decrease if the concept is not changing.

[4]N.B., $Total\text{-}change_1$—calculated when the representation goes from a trivial tree with only one node (the root labeled as +) to the first decision tree—is artificially large since everything in the representation changes. We could ignore early elements in the sum, but for simplicity, leave them in. Such big early changes contribute to the very steep rise in our plots of the *structural-instability metric*.

Note that the actual value of *structural-instability* at any one point in time may not in itself yield any information about the concept's stability since it only reflects change over one "time" step: the addition of the t^{th} example. However, there should be a trend in these values. That is, there should be a consistent sign in the first derivative. If the concept is stable, the instability metric should consistently decrease. If there is truly concept drift, it should increase. In the next section, we describe a method for spotting possible concept drift by testing for a significant increasing trend in the structural instability metric—that is, there is a consistent positive slope in the plot of *structural-instability* against *j*, the number of examples observed—with the aid of a standard statistical test for significance.

4 Detecting Conceptual Change

In this section, we describe how we use the *structural-instability metric* and a statistical test to detect whether there is a significant enough trend in the metric to warrant positing the hypothesis that there is actual change in the meaning of the concept and reacting to it.

We subject the values of *structural-instability*[1, *t*], as *t*, the number of examples, increases to a test of trend over a certain range of its values. Changes in these instability measures can be manifestations of conceptual change. A trend of decreasing values—that is, a downward slope in the graph of metric values plotted against *t*—is indicative of stability and a trend of increasing values is indicative of instability. To spot such a trend, we use a backwards-looking window of W_τ "time steps" (i.e., pairs of trees). Note, we first require that *structural-stability metric* has stabilized to avoid start-up fluctuations (see Fig 2.)

To test for a trend, we use Kendall's τ-test [Kendall, 1962]. The τ-test is a nonparametric test of correlation between two variables, x and y. Given a set data points, x_j, y_j for the variables, the τ statistic measures the degree of positive or negative correlation between x and y. Values close to +1 indicate a high positive correlation and values close to -1 indicate a high negative correlation.

In our application of the τ-test, the variable x is the value of the *structural-instability metric* and y is the number of examples observed thus far:

x_j = *structural-instability [1, t]*

$y_j = t$

To measure whether the concept instability is increasing or decreasing we compute the τ statistic for a window of size of W_τ data points. In the bankruptcy example (Section 5), we use W_τ = 12.

We say that a concept is *stable* if the τ-test gives a negative correlation with high confidence[5] between the *structural-instability-metric* and *j*, that is, a definite downward slope in the plot of the instability values. If there is a positive correlation with high confidence—that is, a definite upwards slope in the instability values—then we say that the concept is *unstable* and form the hypothesis that the concept is drifting.

We test the hypothesis that the concept is drifting in the work reported here, by (1) continuing to update the old tree, and (2) generating a possible replacement tree. Although the number of examples used to generate the replacement tree can be set at will, here we use the same window size as we do with the τ test, that is, 12 examples. After generating a possible replacement tree, we then test its accuracy on the next 12 examples. If it achieves higher accuracy we then switch to it for continued training by examples from the example-stream, and discard the old tree.

5 An Example of Change

In this section we examine drift in the concept of "good faith" as it pertains to approval of debtor plans for debtors filing for personal bankruptcy under Chapter 13 of the United States bankruptcy law (11 U.S.C. §§ 1301-1330). Chapter 13 provides a means for individual debtors to obtain relief from debts while keeping much of their property by following a court-approved plan that allocates 100% of their disposable income to paying off their debts for a period of three to five years. Successful completion of the plan discharges the entire debt regardless of the portion that is actually paid. Among other things, the law requires that the plan be "*proposed in good faith*" (§1325(a)(3)) to minimize abuse by insincere debtors. The original wording of this section, which was enacted as part of the *Bankruptcy Reform Act* in October 1979, was further amended in the *Bankruptcy Amendments and Federal Judgeship Act of 1984*. At that time, a requirement that 100% of the debtor's disposable income be used was added to §1325(b).

While the "good faith" requirement is the linchpin to approval of a Chapter 13 plan, it is nowhere defined in the statute. Rather its meaning has been defined through case law. Since the statute originally took effect, many cases have been litigated around the "good faith" issue. Courts of Appeals for most of the circuits have addressed the issue; the Supreme Court has not. Thus, what the appeals courts say and how lower courts apply their interpretations is the best and only reading—so far—on what "good faith" means.

The appellate courts have typically addressed the issue in terms of factors based on key facts of a case. These factors typically rely on tests of key attributes of a case. Examples of factors from the very important *Estus* case, *In re Estus*, 695 F.2d 311, 317 (8th Cir. 1982), include: "the amount of proposed payments and the amount of debtor's surplus," "duration of the proposed plan," "the accuracy of the plan's statements of debt, expenses and percentage repayment of unsecured debt and whether any inaccuracies are an attempt to mislead the court," "the existence of special circumstances such as inordinate medical expenses," "the motivation and sincerity of the debtor." *Estus* lists eleven such factors, which we call *Estus-factors*; many are compound, really involving more than one factor. In BankXX, we call such a set of factors a *legal theory* [Rissland et al., 1994, 1995]. Other courts, such as the *Kull*, *Kitchens*, and *Makarchuk* courts, have listed their own, often by expansion or modification of the *Estus* set. Some opinions, such as those for *Kitchens* and *Kull*, both of which were appellate cases, really used the same set of factors. Further, the so-called *Kitchens-Kull* set of factors has a great deal of overlap with the *Estus* set of factors.

We performed our experiment on concept drift by creating an example-stream of the 55 cases used in the BankXX project [Rissland et al., 1993, 1994, 1995]. All these cases addressed the "good faith" issue and little else. We ordered them by their dates. The earliest case is from 1980, the latest from 1990. Our corpus includes all of the 14 appellate cases that have addressed this issue.

BankXX uses a HYPO-style model of case-based reasoning and represents cases in a standard way using hierarchical sets of frames [Rissland & Ashley, 1987; Ashley, 1990]. The cases are represented in terms of 61 features that describe the debtor[s], their debt[s], their creditor[s], employment, special circumstances, cash flow, details of the proposed plan for paying off creditors, etc. We "flattened" the hierarchical case representation used in BankXX to a standard feature vector usable by the C4.5 algorithm. We did not alter the case representation in any other way. Using C4.5, we then proceeded with the approach described in this paper.

The parameters we used in this experiment are as follows. We experimented with two versions of both the first and second discount factors:

(1) the *linear* discount factor with N=7 and N=61:
- (1) $f_i = (8\text{-}i)/7$
- (2) $f_i = (62\text{-}i)/61$

(2) the *exponential* discount factor with a=2 and a=10:
- (2a) $f_i = 2^{(1-i)}$
- (2b) $f_i = 10^{(1-i)}$

We thus have four sets of data on our example.

We choose N=7 for the first discount factor because we knew that the maximum depth of any leaf nodes in the trees generated by C4.5 on our data-set is 8.[6] We also tried

[5] For instance, the probability of the τ-test being wrong—that is, that there is no trend—is less than 5%.

[6] Having knowledge ahead of time about the maximal or average depth of the trees generated from an example-stream is not the usual circumstance. However, we did know this here because of exploratory work. The selection of the constant N is itself a subproblem.

N=61, the number of features. We used the second type of factor to try an exponential discounting scheme, which theoretically could reach level 61 but in reality never went deeper than level 7. The average depth of our trees was closer to 5.

We used the same window size of 12 examples for the τ-test (W_τ), the generation of replacement-tree and testing of replacement tree.

As in all of our plots of the instability metric as a function of "time" (i.e., j, the index of the examples), there is an initial period of fluctuation. Structural stability is established early, around j=5 and lasts until about j=10. See the four plots in Figure 3.

All the plots in Figure 3 show a steady rise in the instability metric starting at about case 11. The presence of high values in the plot (around j=17 or so) actually lag a bit behind the onset of major changes in the concept representation (which kick in around j=15) because it takes time for them to be felt in the *structural-instability metric.*

In all of our four runs, the algorithm spotted a significant increasing trend with high confidence at or before case 18 with the τ-test.[7] The lag in detection of the increasing values of the instability metric is due to the time needed for the τ-test window, which is 12 cases wide, to start picking up the increasing values. For instance, at step j=18, the window for the τ-test covers the interval j=7 through j=18. The period of a significantly rising trend continues from case j=18 through case j=23.

At step 18, we thus posit that drift has occurred during the interval j=7 through j=18. We start spawning an alternative replacement tree starting with case j=7 because we assume that drift started at the beginning of the window in which we detected it. We continue training the replacement tree for a total of 12 more instances, that is through inclusion of case 18. We then test the two trees for the next 12 instances, that is, on cases 19 through 30.

Since the original tree performs better than the replacement tree, we do not accept the hypothesized change. Close examination of the trees shows, however, that a major change did indeed start occurring around case 11 or so. Thus, while our *structural-instability metric* and use of the τ–test was highly accurate in detecting the period of change, our concept-representation algorithm (C4.5) was not able to support the conclusion. Thus, the algorithm did not abandon the original concept representation and switch to a replacement.

Thus, even though the *structural-instability metric* caught the real episode of change, the algorithm, as a whole, passed it up. We believe that this is due to the fact that for this data set the old instances were not inconsistent with the revised concept representation; new information arrived, but it did not make the old information obsolete. In this case, throwing away old information leads to worse performance because it is still relevant to the concept.

Before case 11, the decision trees classified instances solely on the basis of what type of debtor the debtor was (e.g., student-loan debtor). In particular, the feature for the *duration of proposed plan* first appears (as the root node) when case 11 is input. This feature is a key fact for the factor addressing plan duration, announced in the 1982 *Estus* case. It was first introduced into the legal discourse in the 1981 *Kull* case. Case 11 is the important *Rimgale* case, which along with *Estus* was one the first three appellate cases to address the "good faith" issue.[8]

The changes occurring from j=15 through j=23 or so involve the introduction into the decision tree of more tests for features that are key to important legal factors. In particular, the concept representation begins to include tests for:

- *duration of proposed plan* (re-introduced at j=15; it had exited after it had been first introduced at j=11);
- *amount of payments and surplus* (introduced at j=16);
- *motivation and sincerity of debtor* (introduced at j=22).

All these attributes are used in the factors from the landmark *Estus* case (1982), which is our case j=17. By the time case 22 has been introduced, these key tests have appeared in the top two or three levels of the decision tree. In particular, a test of *motivation-sincerity* rises to the root node at j=22 and remains firmly planted there throughout the rest of the example-stream. Other attributes come and go, rise and fall, and then eventually settle into the decision tree as the meaning of "good faith" settles down over the rest of the example-stream.

Many of the cases occurring during this interval of rising instability (j=15 through j=23) are exactly those cases which have greatly influenced the treatment of the "good faith" issue:

j=10 *Kull* (1980 trial-level case)
j=11 *Rimgale* (1982 appeals case)
j=12 *Goeb* (1982 appeals case)
j=13 *Barnes* (1982 appeals case)
j=14 *Deans* (1982 appeals case)
...
j=17 Estus (1982 appeals case)
j=18 *Memphis* (1982 appeals case)
...
j=23 *Kitchens* (1983 appeals case)

Note that six of these cases are 1982 appellate cases. Since so many appellate cases were addressing the "good faith" issue in 1982-1983, the law on this issue was in great flux

[7]The algorithm run with a=2, N=7, and N=61 picked up a decreasing trend at j=18 with confidence higher than 98%. The algorithm with a=10 picked it up with equally high confidence but earlier at j=13.

[8]The other was the 1980 *Terry* case, a fairly empty case and not really in the same league as *Estus*.

during this period. ("Good faith" was essentially a new issue introduced by the 1979 Act.)

Appeals cases determine the law to be applied when cases are brought before a lower court, the trier-of-fact. Thus, changes in a legal concept announced by an appeals court can be very influential and their effect on the law—here, its concept of "good faith"—starts showing up in lower court decisions very soon thereafter. For instance, *Estus* case—the most important and one of the earliest appeals court case to address the "good faith" issue—radically changed the treatment of the "good faith" issue by announcing a legal theory of how to decide whether or not a debtor had proposed the Chapter 13 plan in "good faith." All the other appellate cases and most of the trier-of-fact level cases have used or talked about *Estus* in some way.

To summarize the example up to j=23: the *structural-instability metric* has spotted drift in the good faith concept very accurately. It spotted the big changes set in motion by the *Estus* case and those other cases that closely followed it. Although drift was detected correctly for this data set, it was not verified. In this example, the drift was actually a major refinement of the old concept instead of a replacement of it with a completely new concept that contradicted the old one.

After case j=23, the "good faith" concept is fine-tuned. In terms of the decision trees, tests for certain case attributes (e.g., *amount of debt, motivation-sincerity, size-of-surplus*) move around in the tree but at deeper and deeper levels. That is, according to our metric, the "good faith" concept is stabilizing. In fact, there are many instances where the tree does not vary at all from one step to another.

When we use our *structural-instability metric* with the exponential discount factor using a=10 or a=2, these changes cause its values to jiggle around at various points after j=23 but from about there onwards it starts a long steady decrease. See Figures 3a and 3b. The τ-test computes a clearly (high confidence) negative trend from j=24 onwards.

The interval of j=30 to about j=40 is assessed somewhat differently by the metric using the linear discount factor with N=7 or N=61. For both, the τ-test computes a high confidence positive trend in this period.[9] Thus, the *structural-instability metric* run with the first discount factor, which is more sensitive to changes lower down in the decision tree representation, picks up a period of instability here.

This period corresponds to cases from 1984 and 1985. The changes could be due to the addition of the §1325(b) requirement (for using 100% of the debtor's disposable income) that was added in the 1984 amendments to the bankruptcy code, but it seems a little early for the cases to be reflecting this. This change in the law, in effect, obviated the *Estus* factor for surplus in the plan. It takes a while, but eventually it causes the test of the *surplus* feature to exit the decision tree (at j=43).

When the *structural-instability metric* is run with the exponential discount factor, these changes did not make themselves felt. Although the change in the statute may account for the minor rise in instability around case j=34 or so (see Figures 3a and 3b), the rise is not significant enough—according to our use of the τ-test—to merit going so far as to posit a hypothesis of conceptual change.

How observable such blips are depends on the chosen discount factor. The use of a particular discount factor can greatly attenuate the noticeability of change. Note, however, that all of the graphs in Figure 3 tell basically the same story. There is major structural instability early on (j=15 to j=23 or so). They differ somewhat in the range j=30 to j=40. They agree on stability thereafter.

The algorithm run with the exponential discount factor with a=10 was the fastest to pick up the episode of drift and to do so with the most confidence (i.e., with a very small τ-test probability of doubt over a small interval). See Figure 3a. It was certain of drift—that is the τ-test produced high positive correlation with high confidence—the fastest, at j=13. It was also the fastest to observe with confidence the return of a tend of increasing stability, at j=24. The algorithm run with a=2 was a little slower and a little less sure. It picked up drift at j= 18 and subscribed to returned stability at j=26. See Figure 3b.

The algorithm run with the linear discount factor with N=7 was next slowest and sure in picking up instability at j=19 and returned stability at j=40. See Figure 3c. The algorithm run with the linear discount factor with N=61 picked up drift with confidence at j=18 but waited until j=46 to be sure of returned stability. See Figure 3d.

In summary, all versions of the *structural-instability metric* with the τ-test detected drift around j=18, where it indeed had occurred. Where they differed was in their confident detection of returned stability in the late cases and whether there existed an intermediate period of instability in the 1984-1985 cases. These differences are due solely to the various choices of discount factors, since all other parameters remained constant. How to design and instantiate discount factors is an open research question, as is the selection of the various window sizes (e.g., W_τ used in the τ–test) and other parameters used in our algorithm.

[9] With N=7, the high confidence interval of positive trend is [32, 37]. With N=61, it is a bit more spread out: [31, 38].

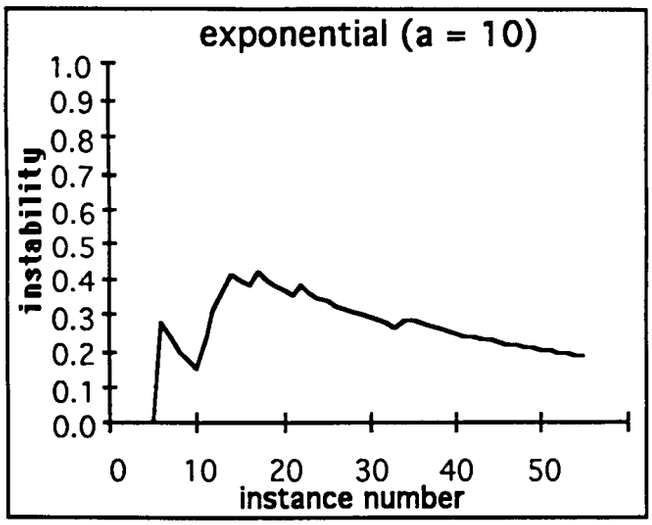


Figure 3a. Plot of *structural-instability metric* with discount factor $f_i = 10^{(1-i)}$.

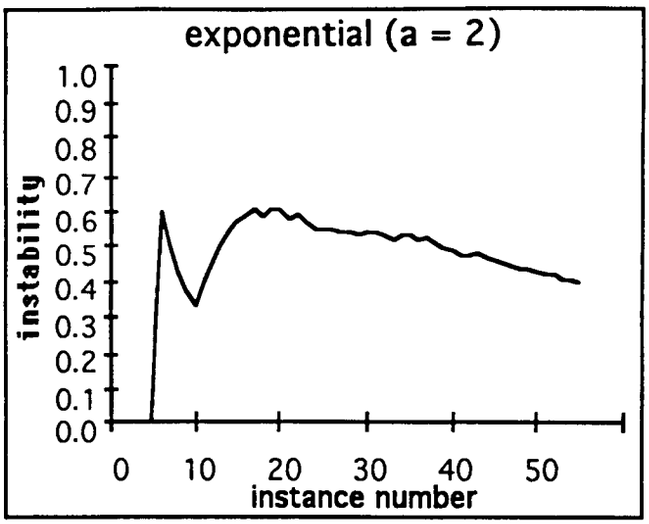


Figure 3b. Plot of *structural-instability metric* with discount factor $f_i = 2^{(1-i)}$.

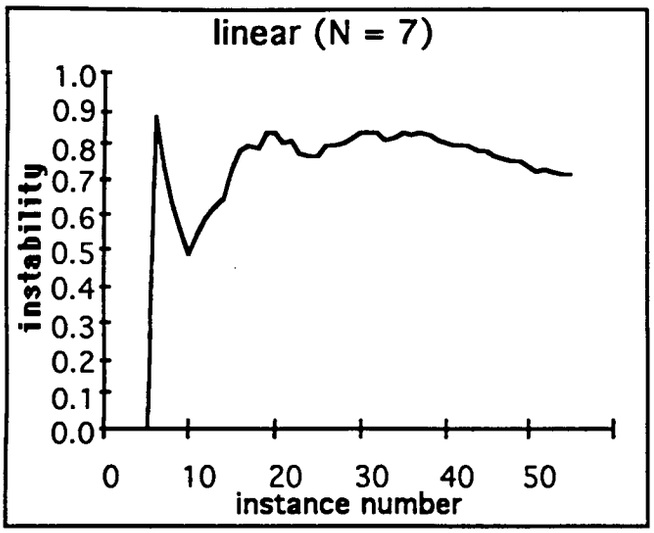


Figure 3c. Plot of *structural-instability metric* with discount factor $f_i = (8-i)/7$.

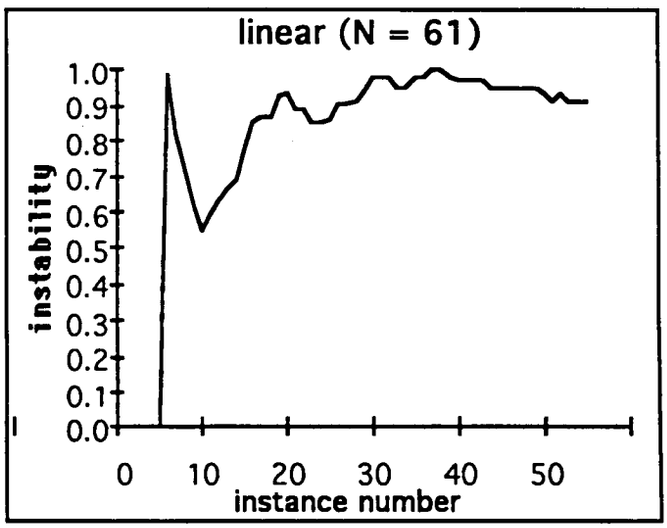


Figure 3d. Plot of *structural-instability metric* with discount factor $f_i = (62-i)/61$.

6 Conclusion

In this paper we presented our approach to detecting concept change during incremental supervised learning. Our method first detects that a concept may be shifting by (1) measuring the changes that occur as each new example is incorporated into the emerging decision-tree representation of a concept, (2) averaging these measures across several "time" steps, and (3) subjecting the average to a test of trend. If there is a significant increasing trend in the running average—that is, the *structural-instability-metric*—we form the hypothesis that there is concept drift. We can then test this hypothesis by spawning a possible new concept representation and comparing its classification accuracy with the old representation. Our method is very general and can be used in conjunction with many incremental learning algorithms, such as C4.5, ID5R, and ITI.

We demonstrated how our method works on an example of conceptual change taken from the bankruptcy domain. By applying our method to a stream of cases, ordered by their dates of decision, we were able to detect episodes of drift. Our data set of cases was taken pretty much *as is* from the corpus we had developed in our previous BankXX project; the only changes made were those needed to accommodate the feature vector representation of a case required by C4.5. We then discussed how our results compared with the actual case law.

In summary, we found that our algorithm was able to detect episodes of possible concept drift quite well. On the other hand, it was perhaps too strict in its verification of the drift hypothesis.

In future work, we plan to explore various details of our algorithm, including how to set system parameters: window sizes for the τ-test, generation and testing of replacement tree; discount factors, etc. It would also be very useful to experiment with our approach on other data sets.

7 References

Ashley, K.D. (1990). *Modeling legal argument: Reasoning with cases and hypotheticals.* M.I.T. Press.

Brodley, C. E., & Rissland, E. L. (1993). Measuring Concept Change. *Working Notes of the 1993 AAAI Spring Symposium on Training Issue in Incremental Learning*, 98-107. Stanford.

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

Hart, H. L. A. (1958). *The Concept of Law.* Clarendon Press. Oxford, U.K.

Kendall, M.G. (1962). *Rank correlation methods.* New York: Hafner Publishing Company, Inc.

Kuhn, T.S. (1970). *The Structure of Scientific Revolutions.* Second Edition. University of Chicago Press.

Lakatos, I. (1976). *Proofs and Refutations.* Cambridge University Press.

Llewellyn, K. N. (1989). *The Case Law System in America.* The University of Chicago Press.

Levi, E.H. (1949). *An introduction to legal reasoning.* University of Chicago Press.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning.* Morgan-Kaufmann.

Radin, Max (1933). Case Law and Stare Decisis: Concerning Prajudizienrecht in Amerika. *Columbia Law Review*, Volume 33, p. 199.

Rissland, E. L. (1989). Dimension-Based Analysis of Supreme Court Hypotheticals. *Proceedings of the Second International Conference on Artificial Intelligence and Law (ICAIL-89)*, 111-120. Vancouver. ACM Press.

Rissland, E. L. (1994). Cases and Rules: A Natural Synergy. *Proceedings Fifth Generation Computer Systems Conference.* Tokyo. December 1994.

Rissland, E. L., & Ashley, K. D. (1987). "A Case-Based System for Trade Secrets Law." *Proceedings First International Conference on AI and Law, (ICAIL-87)*, 60-66. Boston. ACM Press.

Rissland, E.L., & Collins, R. (1986). The Law as Learning System. *Proceedings Eighth Annual Cognitive Science Society Conference*, 511-513.

Rissland, E. L., Brodley, C. E., & Friedman, M. T. (1994). *Measuring Structural Change in Concepts.* Submitted for publication.

Rissland, E. L., Skalak, D. B. & Friedman, M. T. (1993). BankXX: A Program to Generate Argument through Case-Based Search. *Proceedings Fourth International Conference on Artificial Intelligence and Law (ICAIL-93)*, Amsterdam, The Netherlands. ACM Press.

Rissland, E. L., Skalak, D. B. & Friedman, M. T. (1994). *BankXX: Supporting Legal Arguments through Heuristic Retrieval.* TR 94-76, Dept. of Computer Science, Univeristy of Massachusetts, Amherst. Submitted for publication.

Rissland, E. L., Skalak, D. B. & Friedman, M. T. (1995). *Evaluating a Legal Argument Program: The BankXX Experiments.* TR 95-30, Dept. of Computer Science, Univeristy of Massachusetts, Amherst. Submitted for publication.

Schlimmer, J.C. (1987). *Concept acquisition through representational adjustment.* Doctoral dissertation, University of California, Irvine.

Utgoff, P.E. (1989). Incremental induction of decision trees. *Machine Learning*, 4, 161-186.

Utgoff, P.E. (1994). An Improved Algorithm for Incremental Induction of Decision Trees. *Proceedings of Eleventh Machine Learning Conference.* Rutgers University, July 1994. Morgan Kaufman.