Credit Assignment and the Problem of Competing Factors in Case-Based Reasoning

Edwina L. Rissland
and Kevin D. Ashley
1988

COINS Technical Report 88-62

Department of Computer and Information Science
University of Massachusetts
Amherst, Massachusetts 01003

Credit Assignment and the Problem of Competing Factors in Case-Based Reasoning [1]

Edwina L. Rissland and Kevin D. Ashley [2]
Department of Computer and Information Science
University of Massachusetts
Amherst, Massachusetts 01003

Abstract

In this paper we describe an approach to the problem of weighting and credit assignment for various factors that contribute to an analysis or outcome of a problem situation and discuss issues about weighting as they touch upon our case-based reasoner HYPO. In HYPO, we take the approach of delaying for as long as possible any assignment of weights and of symbolically comparing the competing factors. We call this approach a *least commitment weighting* scheme.

1. Introduction

Problem analysis and solution can depend on many factors, some of which are more important than others and some of which may compete with and contradict each other. Further, the importance and contribution of a factor can be highly dependent upon the context defined by the problem situation and also the other factors present in it. Rarely are all factors of equal weight or is a problem decomposable in a linear factor-by-factor manner. Experts in domains like the law and tactical planning know this. Nonetheless, they often approach the problem in a manner that at first glance might lead one to believe they are neglecting such complexities. Upon closer inspection, however, one can see that they are pursuing an approach that postpones for as long as feasible any commitment to

[1] This work was supported (in part) by: the Advanced Research Projects Agency of the Department of Defense, monitored by the Office of Naval Research under contract no. N00014-84-K-0017; the University Research Initiative, award no. N00014-86-K-0764; and an IBM Graduate Student Fellowship.

[2] Copyright ©1988. Edwina L. Rissland & Kevin D. Ashley All rights reserved.

assign weights or to select a combining function for factors. Experts do this for several reasons:

1. Such a commitment might cut off certain possibly fruitful lines of reasoning and thereby limit their problem solving performance;

2. Reduction to numerical weights, in particular, makes it difficult to recover symbolic information needed for certain reasoning methods like case-based justification and contrast-and-compare discussion of alternatives.

3. Assigning actual "weights" and predicting interactions among the factors is highly problematic and dependent on individual problem situations.

4. Experts in domains like the law simply do not reason in terms of weighting schemes. In fact in the legal domain, any reasoner that based an opinion or course of action upon a purely numerical scheme would be highly suspect.

Nonetheless, reasoning in case-based domains like the law does present the need to deal with factors which both interact and contribute to an overall analysis of a case and which may not be of equal importance. Thus, at some point in the reasoning, the reasoner must resort to some sort of balancing and trading off between the factors. That is, one could say that there must be some sort of consideration of credit assessment and some attempt at a method – symbolic or numerical – to weight the competing factors

In this paper we describe an approach to the problem of weighting various factors that contribute to an analysis or outcome and discuss issues about weighting as they touch upon our case-based reasoner HYPO. In HYPO, we take the approach of delaying for as long as possible any assignment of weights and of symbolically comparing the "weights" of competing factors. We call this approach a *symbolic least commitment weighting* scheme.

2. Background on Weighting

2.1 The Weighting Game in Law

In the legal domain, attorneys do know what factors are important in a particular legal claim. Although they may be willing to say in the abstract that a certain factor is more important than other factors, they almost never will venture numerical weights to distinguish the factors' importance. They are keenly aware that there might be some combinations of facts in which a particular factor, though normally more important than a competing factor, may not be so. Lawyers must also be prepared to justify an assertion

that in a particular fact situation one factor is more important than a competing factor and such justification cannot be made in terms of numbers or statistics. It must be a symbolic justification using precedent-based methods such as comparing and contrasting cases [Ashley and Rissland, 1987].

What the lawyer is grappling with is essentially a problem of credit assignment [Samuel, 1963]. While he knows that it is most likely not the case that all factors contribute equally, it is exceedingly difficult to come up with an overall "score" for the case or to assign credit ("weights") to the individual factors. The doctrine of precedent – that similar cases should be decided similarly – is some help in this regard.

For instance, one can try to find a similar past case and through analogical reasoning "map over" the analysis to the new case using the doctrine of precedent by arguing that it should be evaluated similarly. Of course there can be many things which can go wrong in such an approach. For instance, when there exist two precedents with the same cluster of factors but they point to opposite conclusions (e.g., in one case the plaintiff won and in the other, the defendant), one is not sure what "score" to use. In fact, this situation may indicate that one is dealing with a "hard" case [Gardner, 1987], that is, a case about which there is substantial disagreement among the courts.

To assign credit to an individual factor is even more difficult. For one thing, courts seldom make this assignment explicit even though they might provide some indication of importance. One way to clinch the issue of which factors are more important is to find another precedent with exactly the same combinations of factors and to argue that the same cluster of factors should be important in the new case. To assess the contribution of a particular factor, one tries to find cases that have exactly the same factors except for the one of interest and to infer how the absence/presence of the factor affected the outcomes of the cases. This is akin to the componentwise credit assignment strategy discussed by [Subramanian and Feigenbaum, 1986]. See also [Rissland, 1988]. It also is similar to the dropping of an antecedent condition to test its necessity as is done in mathematics and the use of the near miss in machine learning [Winston, 1975].

2.2 Relation to Other Work

The problem of assigning weights to factors and defining functions to combine their contributions and produce an overall evaluation of a problem situation appears in many different areas of AI, particularly machine learning.

In machine learning, the first, and perhaps still the best, discussion of the credit assignment problem was by Samuel in his two landmark papers [Samuel, 1963; Samuel, 1967]. In the experiments reported in the first paper, Samuel approached the problem by using a

linear polynomial evaluation function to assess a checkers board position (in conjunction with alpha-beta pruning). From a basic set of terms (e.g., piece advantage, king center control, total mobility) he constructed evaluation polynomials. These factors captured key features of checkers, for instance, that "it is usually to one's advantage to trade pieces when one is ahead and to avoid trades when behind....kings are more valuable than pieces" [p.75]. In experiments on "rote-learning", he used a linear combination of four terms; in a second set of experiments on "learning-by-generalization", where his program, itself, selected the terms included to be included in the polynomial and the weights given them, there were 16 terms, including cross terms to model interactions between factors. In the early experiments, the weights varied over a wide range, (from -2^{18} to 2^{18}); in later experiments they were of more equal magnitude. Gross differences in magnitudes of the weights, in effect, allowed his program control through what might be likened to a big switch that selected which terms to use. For instance, king center control (KCENT) and a cross term (MOC2) involving "total mobility" and "denial of occupancy" had weights of approximately 2^{16} and -2^{18}, respectively, and KCENT ranged between 0 and 8 while MOC2 was binary. Thus, in the appropriate circumstances one could completely dominate the decision or they could cancel each other out and force evaluation of a board in terms of lower order terms. Such cancelling out is in fact an example of a situation where one factor argues strongly towards one conclusion and the other equally strongly to the opposite. In such a situation Samuel's program defers to an analysis of the factors with lesser weights. Conversely, when there is no cancelling out, the evaluation essentially depends on one factor.

It is interesting to note that in learning terms and weights for the evaluation polynomial in one early version of the program, "at least 20 different terms were assigned the largest coefficient at some time or other" [p. 89]. Thus, too frequently a new term (factor) was assigned inordinate credit, so that the program was changing its model of what factor was important in a "fickle" manner. Later versions of the program remedied such instabilities by delaying the rate at which new terms could be selected, that is, by increasing the integration time with respect to moves (and cases) considered.

Also, the program could be fooled by bad play on the part of its opponent and overly dazzled by spectacular moves which "resulted in the misassignment of credit to those board positions which permitted spectacular moves when credit rightfully belonged to earlier board positions which had permitted the necessary ground-laying moves." Again, Samuel corrected this by increasing the span of moves over which the evaluation was computed.

Such problems are not confined to game playing. Although Samuel was able to remedy them to some extent, they still suggest to us deep difficulties in assessing the contributions of competing factors and coming up with weightings. If there are such problems even in a clean domain like checkers, the problem is daunting in blatantly scruffy domains like law

and tactical planning.

It is interesting to note that the rote-learning version of the program was better than the learning version in certain contexts, like opening games, when much expertise is "book knowledge"; that is, for some phases of the game, a case-based approach is reasonable. The early program had a case memory containing "something over 53,000 board positions" [p.82], an admirably large case base by any standard. His later program used a case-base approximately five times that large.

In later experiments, Samuel introduced his notion of "signature", a vector whose component entries were restricted to a small range of 3-7 discrete values. These were combined in a hierarchical way to come up with a composite signature and ultimately a score for the board position which could be manipulated and "backed up" in the same way as scores from the polynomial evaluation functions. HYPO's dimensions bear some resemblance to his signatures since both cluster features into a larger structure and both can measure whether a situation is strong, weak or indifferent with respect to it. He used the signatures to index a library of master play (containing approximately 250,000 board situations) and thereby assigned to the current situation the move from the indexed known one. Samuel's experiments showed that his program performed better (by about a factor of two) with signatures than with learning-by-generalization. One major difference between Samuel's program and HYPO is that Samuel's signatures are used to *evaluate* a current board position whereas HYPO's dimensions are used as a retrieval and comparison mechanism only.

In summary, lessons to be learned from Samuel include: (1) one needs to have a rich language for factors to assess strengths and weaknesses and to be able to handle combinations of them; (2) one needs to be able to accomodate situations in which one factor can completely overwhelm another or two competing factors can cancel each other out; (3) one needs to be able to change evaluation functions to suit the requirements of different problem solving contexts (4) a case-based approach can enhance performance, even in domains like game-playing traditionally handled by heuristic search techniques.

In work on expert systems, particularly, on mechanisms affecting control through the use of mechanisms to assess belief and certainty, there is much discussion of how to reason with weights and to combine them [Howe and Cohen, 1987; Cohen *et al.*, 1987; Mostow and Swartout, 1986]. For instance, Cohen *et al.* discuss the trade offs between using tabular or "modifiable" functions for specifying and combining the contributions of factors. The tabular approach in effect defines a multi-variable function by listing its values domain vector by domain vector; Mostow calls this the "compiling out" approach; Samuel's signatures were of this type. The other approach is to build a combining function by specifying a procedure (e.g., certainty factor calculus or probability calculus). Both approaches greatly

affect the control of reasoning and its explicitness. At the heart of both is the problem of what to do with weights.

In case-based reasoning research, other projects have had to face the problems of weights, particularly, when they assess factors in a problem situation to assess similarity and index memory. For instance, the CBR systems CYRUS [Kolodner, 1983a; Kolodner, 1983b] and MEDIATOR [Simpson, 1985; Kolodner *et al.*, 1985]. use a "reminding" process that assesses closeness of fit by considering a set of selected features assigned an *a priori* ranking. Their more contextual, dynamic indexing is done with the "E-mop" mechanism, which organizes memory according to the aspects of an event that differ from norms of the conceptual category of the event (e.g., by violating expectations). These systems do not allow for changing assessment of similarity – that is, weighting factors – based on the case at hand. However, by keeping track of successes and failures they are able to generate new factors to consider.

2.3 Overview of Weighting in HYPO

What perhaps makes the situation in our work in case-based reasoning different from past approaches to the problem, such as by Samuel or Kolodner *et al.*, is that the choice of weights and methods for combining them is influenced by the context of the case, specifically:

1. The side or position one is advocating, for instance, whether one represents the plaintiff or defendant, and what legal doctrine one is considering the problem situation to fall under. [3]

2. What cases are relevant or on-point and which side they support. This of course is highly dependent upon the state of the Case Base.

3. The specific path through the space of possible arguments one actually chooses.

Each of these choices "cascades": the choice at 2. depends upon 1. and that at 3. upon 2. There is no single evaluation function that will serve across all cases or all stages of the problem solving or states of the case knowledge base. [4] In fact the same case taken in the context of a different case base very likely would be treated differently. In short,

[3] A given case can usually be approached with claims from diverse doctrines. For instance, a misappropriation case might also be approached from tort, contract or criminal law perspectives.

[4] This need for different evaluation functions at different stages of the problem solving was also recognized by Samuel who settled on six different types of signatures and evaluation polynomials to span the game playing from opening to end game.

at the same time, one contemplates problem solutions, one must also contemplate ways to evaluate them.

In the game-playing metaphor, problem solving for a legal situation depends on a knowledge base of past book moves (cases), the stage of the game, as well as projections gained from look ahead (argument paths). As in game playing, each adversary wishes to retain the ability to choose another approach, for instance, in order to respond to a potentially damaging response by one's opponent.

In this paper, we describe a flexible, least commitment approach to weighting which we have used in the context of a case-based reasoning system. In the same spirit in which least committment planning [Sacerdoti, 1975] postpones for as long as possible any commitment to a particular sequence of operator actions, our method postpones for as long as possible any commitment to a particular set of factors, supporting cases or argument steps. In that the method relies on a self-critical phase, it is a generate-and-test method and philosophically, is in the spirit of "proofs and refutations" [Lakatos, 1976]. The approach has three phases:

1. **Clustering** applicable factors according to how they appear in prior cases most-on-point to the problem situation.

2. **Interpreting** the effect of the clustered factors by examining the outcomes of the most-on-point prior cases.

3. **Criticizing and Testing** interpretations in light of salient differences among the most-on-point cases and the problem situation and by heuristically, hypothetically changing magnitudes and combinations of factors.

In HYPO's three-phase model, the determinations of weights among competing factors is deferred. The program waits on weighting competing factors until the preferences can be determined in light of the context of the particular facts of the problem situation and the possible justifications that can be offered for the different outcomes. At the conclusion of phase 3, HYPO would be in a position to assign weights if we felt that were appropriate. However, since we are concerned with case-based advocacy and *not* adjudication, we do not take that step. However, for a case-based reasoner in another domain (e.g., tactical planning), such a decision-making step might be appropriate and we would advocate waiting for the completion of phase 3 before making the commitment to a weighting of factors and the ultimate combination of them into a final, decision-making "score".

3. HYPO's Approach to Weighting

HYPO is a computer program that analyzes legal problem situations in domains like trade secrets and tax law. Inputs to the program are a description of the problem situation. Outputs are arguments in favor of either side to a legal dispute, plaintiff or defendant, concerning various legal claims to which the facts give rise. HYPO justifies those arguments as an attorney would by citing and distinguishing legal case precedents from its own **Case Knowledge Base** (CKB) of cases. For a complete description of HYPO, see [Ashley, 1987; Rissland and Ashley, 1986; Ashley and Rissland, 1987].

The factors that matter in HYPO's legal domain are represented with dimensions. A **dimension** is a knowledge structure that identifies a factual feature that links operative facts to known legal approaches to those facts, specifies which are the most important for the approach, and specifies how a legal position's strength or weakness can be compared to that of other cases. For each dimension, there is at least one real legal case where the court decided the case because, or in spite, of the features associated with the dimension. That case can be cited in a legal argument to justify that a similar fact situation should be decided in the same way.

In any given case, some factors may favor one side while other factors favor the opponent. In addition, a factor may favor a side more or less strongly. The magnitude or strength of a factor in a case is represented by its position along the range of the dimension. The ranges may be numeric intervals, or ordered sets, including binary and partially ordered sets.

HYPO's task in analyzing a problem situation is to combine the competing factors to develop as robust an argument as possible. HYPO manipulates relevantly similar, different and most-on-point cases in proceeding through its three phases of clustering, interpreting, and criticizing/testing. A case is **relevantly similar** if it shares a factor in common with the problem situation. The most relevantly similar cases, called **most-on-point** cases (or "mop-cases"), have the maximal overlap of factors in common with the problem situation. A case is **relevantly different** from a problem situation if it differs with respect to the magnitudes of a shared factor or it differs because there are additional, unshared factors.

3.1 Phase 1. Clustering the Factors

HYPO clusters factors that apply to a problem situation in the process of generating a lattice – called a claim-lattice – of all the cases in its Case Knowledge Base that are relevantly similar. A claim-lattice defines equivalence classes of cases having the same subset of factors in common with the problem situation. Cases having a maximal subset of factors in common with the problem situation are the most-on-point cases; these are

immediate children of the root node which represents the problem situation.

For the purposes of illustrating HYPO's least commitment approach to weighting, consider the fact situation and its derived claim-lattice shown in Figures 1 and 2. For details on how HYPO produces such an analysis, see [Ashley and Rissland, 1987] which uses a similar example fact situation.

To produce initial clusters of factors, HYPO employs three simplifying heuristics:

C-1 Consider only those combinations of factors for which there is at least one most-on-point, real precedent case that has that combination.

C-2 Temporarily ignore the fact that the most-on-point cases, associated with a particular combination of factors, may differ among themselves as to other factors that they do not share with the problem situation.

C-3 Temporarily ignore differences in magnitudes of the shared factors among the most-on-point cases and the problem situation.

The first heuristic, C-1, means that HYPO only considers cases from the immediate children nodes of the problem situation root node in the claim-lattice. C-2 means that relevantly similar cases are projected onto the space spanned by the dimensions applicable to the problem situation. C-3 means that each dimensional factor is "normalized" to be of equal strength. In Figure 2, for example, HYPO uses C-1 to cluster the factors into three groups corresponding to each group of equivalent most-on-point cases: Node [1] has (a, e), Node [2] has (a, b, c), and Node [3] has (d). Using C-2 and C-3, HYPO temporarily ignores the fact that in Node [3] the *Crown Industries*, *Midland Ross* and *Data General* cases each involve other factors not shared with the problem situation and that, since they each involved different numbers of disclosures to outsiders, they all differ from the problem situation in terms of the magnitude of factor (d).

3.2 Phase 2. Interpreting the Combined Effect of a Cluster

For each cluster of factors associated with most-on-point cases, HYPO interprets their combined effect according to the outcomes of those cases. If all of the mop-cases in the claim-lattice node were won by the same side, then the cluster of factors is treated as warranting a decision of the problem situation for that side. The justification is that every past decision that presented that particular combination of factors has favored that side. Unfortunately, things frequently are not that simple.

If the equivalence class of most-on-point cases is split between those favoring the plaintiff and defendant, then there are two as yet equally justified competing interpretations of

the effect of the cluster of factors. Further steps are taken in an attempt to resolve the tie between the competing interpretations.

In Figure 2, the *Analogic* case of Node [1] supports interpreting clustered factors (a, e) for the plaintiff. Likewise, the *Amoco* case of Node [2] favors interpreting clustered factors (a, b, c) for the defendant. Things are a bit more complicated for Node [3]. While the *Crown* and *Midland Ross* cases support interpreting clustered factor (d) for defendant, *Data General* supports interpreting it for plaintiff.[5]

HYPO has three heuristic methods for showing how to resolve ties among two competing interpretations of a particular cluster of factors. These methods discredit an interpretation through discrediting the most-on-point cases justifying the interpretation. HYPO uses them to attempt to show that the clustered factors do not warrant a given result by pointing out salient distinctions between the problem situation and the most-on-point cases. The three interpretation heuristics are:

I-1 Show that alternative clusterings of factors in the problem situation justify a result inconsistent with one of two competing interpretations of a cluster.

I-2 Show that alternative clusterings of factors in the most-on-point cases favoring one of the interpretations can be used to explain away the result in those cases, and that these alternatives do not apply to the problem situation.

I-3 Show that certain of the clustered factors were not as strong in the problem situation as they were in the most-on-point cases and thus that the mop-cases do not support the interpretation.

These interpretation strategies focus on the previously ignored effects of the other clustered factors and of the relevant differences between the most-on-point cases and the problem situation, differences represented by unshared factors that favor different outcomes and differences in the magnitudes of shared dimensions. **I-1** points out distinguishing factors, not shared by a most-on-point case supporting the interpretation, which favor coming to an opposite outcome. In other words, **I-1** causes HYPO to consider sibling nodes (equivalence classes) in the claim-lattice to counter the effect of the clustering strategy of **C-1**. For each most-on-point case favoring the interpretation, **I-2** points out distinguishing

[5]Note that one possible way of resolving the tie is to count the number of cases favoring each interpretation. Although one does see this sort of numerical tie-breaking in legal argument from time to time (e.g., attorneys and judges sometimes speak of the majority and minority rules comparing not the number of cases but the number of states that would hold one way or the other on an issue) it is not generally accepted as an appropriate rationale for decision. Another way of breaking the tie is to compare the pedigrees of the courts on either side of the issue and award the decision to the highest court.

factors, not shared with the problem situation, that can be used to explain why the problem situation should have a contrary outcome. In other words, **I-2** is a "lifting" strategy to counter the "projection" strategy of **C-2**. **I-3** is an "unnormalizing" strategy to counter the effects of **C-3**.

The goal of phase 2 is to determine if one of the two "tied" sets of otherwise equivalent most-on-point cases is "less distinguishable" than the other. If so, then the clustered factors are interpreted consistently with the outcomes of that set since they are closer to the problem situation. Otherwise, as is usually the case, HYPO cannot resolve the tie but can only make case-citing arguments favoring each interpretation. In Figure 2, for example, the heuristics do not allow HYPO to resolve the tie in interpreting the effect of the cluster in Node [3]. **I-1** does not avail because the other clustered factors from Nodes [1] and [2], (a, e) and (a, b, c), seem to pull equally in favor of plaintiff and defendant. Although **I-2** allows HYPO to distinguish *Data General*, it also allows HYPO to distinguish *Crown*: Unlike the problem situation, *Data General* involves factor (f) favoring the plaintiff because all of the disclosures were subject to confidentiality agreements and *Crown* has a factor favoring the defendant that the problem situation does not have (it involved disclosures in negotiations with the defendant). **I-3** allows HYPO to distinguish *Midland-Ross*, also, because it involved more disclosures to outsiders than the problem situation (i.e., 100 disclosures as opposed to 50 in the problem situation.)

3.3 Phase 3. Criticizing and Testing

The methods of the final phase are used to criticize and test the results of the first two phases. They are based upon the use of counter-example cases, both real and hypothetical. With them, HYPO attempts to produce counter-examples to the interpretations from phase 2. The types of counter-examples used in phase 3 are:

Boundary – a case in which one of the clustered factors was far more extreme than in either the problem situation or the most-on-point case and yet the factor did not lead to the same outcome as in the mop-case.

More-on-point (or Trumping) – a case won by the opposing side whose cluster of factors shared with the problem situation overlaps, and strictly contains as a subset, the cluster of factors in the most-on-point case.

Overlapping – a case won by the opponent whose cluster of factors overlaps, but does not strictly contain, the cluster of factors in the most-on-point case.

Potentially more-on-point - a case won by the opposing side that would be a most-on-point case if certain factors, currently "near-misses" in the problem situation, were actually present. A factor is a near-miss if the problem situation contains all the information needed to tell if the factor (i.e., dimension) applies except the information about magnitude that determines where the situation should lie on the dimension.

The three phase 3 methods are:

C&T-1 Use "boundary" counter-examples to show that certain of the clustered factors favoring the outcome are not important as justifications.

C&T-2 Use trumping counter-examples to show that the cluster of factors as a whole is not important as a justification.

C&T-3 Use hypotheticals based on potentially more-on-point counter-examples to show that certain of the clustered factors or the cluster taken as a whole are not important as justifications.

The point of **C&T-1** is to show that even extreme examples of particular factors do not warrant the result in the most-on-point case. For example, Figure 3 shows that the *Data General* case is an extreme example of factor (d) in which the plaintiff still won even though it had disclosed to 6000 outsiders. HYPO uses **C&T-1** to attack the assertion that clustered factor (d) of Node [3] necessarily favors defendant by citing *Data General* as a boundary counter-example.

Given a most-on-point case that supports an interpretation of a cluster of factors, the goal of **C&T-2** is to find a more-on-point counter-example that strictly contains the cluster but had the contrary outcome. If all the factors that apply to the problem situation are taken as given, by definition there can be no such trumping counter-example. But there may be other factors that apply to the problem situation that the user has not told HYPO about because he does not know they are relevant. HYPO uses **C&T-2** to probe the user about additional factors in the problem situation that may be relevant. HYPO is guided heuristically by those cases in the claim-lattice that are potentially more-on-point counter-examples. For example, factor (f), which applies to the *Data General* case where all disclosures were restricted by confidentiality agreements, is a near-miss with respect to the problem situation. By hypothetically modifying the problem situation so that all 50 disclosures became restricted, *Data General* would become more-on-point than either of the other cases in Node [3] of Figure 2. The newly applicable factor would be incorporated into a "super" cluster (d, f) which would be interpreted as favoring the plaintiff.

C&T-3 also involves posing hypotheticals, but posing hypothetical variations of most-on-point cases, rather than of the problem situation. Where the program cannot find real boundary or trumping counter-examples, it makes them up. That is, using the most-on-point cases as seeds, it creates extreme cases by exaggerating magnitudes of factors and combinations of factors to create hypothetical cases that are extremely strong for a side, thus overwhelming any contravening factors. These hypotheticals, though cited rhetorically, are useful in obtaining concessions from a side that even though a particular factor may favor an outcome in some contexts, it does not always favor that outcome.

One can view these three criticizing and testing strategies as performing a sensitivity analysis or heuristic search through the space of cases and the space of clusters of factors. **C&T-1** varies magnitudes of factors found in both the problem situation and the most-on-point cases. **C&T-2** varies the problem situation while holding the most-on-point cases constant. **C&T-3** varies the most-on-point cases while holding the problem situation constant.

4. Discussion and Conclusion

Having performed its criticize and test phase, HYPO does not assign weights to competing factors. It does not need to. The outputs of the 3-phase process are ideal for assisting attorneys to make or anticipate reasonable legal arguments about the significance of the factors in the problem situation. HYPO's competing interpretations and concomitant critiques, complete with most-on-point cases, distinctions and counter-examples, map out possible reasonable argument paths for attorneys to follow in arguing about the problem situation.

Since HYPO does not play judge, it does not actually have to decide the problem situation and thus does not actually have to venture assignment of weights. Furthermore as was pointed out by legal realists like Holmes and Llewellyn, a judge's decision is seldom determined solely by the precedents but also upon social and psychological factors and this is outside the ken of HYPO.

Since the comparisons among the cases are not *ceteris parabus* – that is, all other things are not equal – there almost always are grounds for criticizing a case-based interpretation of clustered factors. Inevitably, more than one reasonable interpretation can be assigned for each of the various clusters of factors in the problem situation. HYPO generates such competing interpretations but does not resolve the argument.

HYPO's method illustrates one way of dealing with the central dilemma of weighting: delay and wait for as long as possible. If weights were to be assigned to competing factors, it could be done meaningfully only at the end of this 3-phase process. Only after the

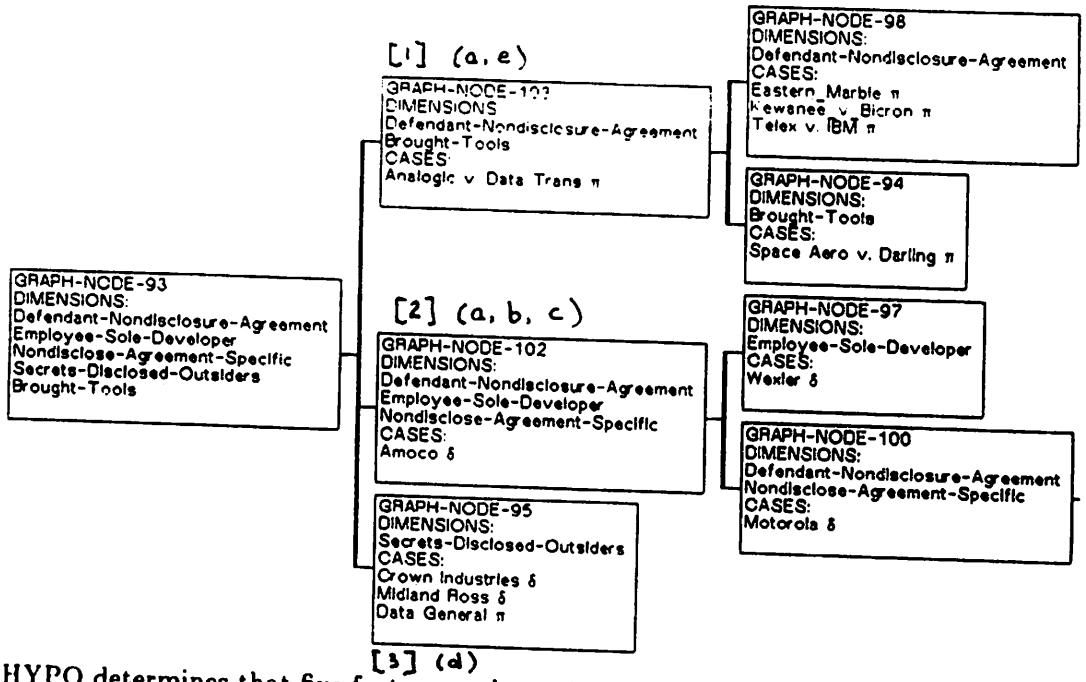
criticize and test phase could the a weighting scheme take into account the specific context of the adversarial position one is defending, the combinations of factors and magnitudes presented in the problem situation and the precedent cases that can be used as justifications in arguments, and the possible paths through the space of arguments. But then the weights are so contextual that they might lack utility. Furthermore, collapsing all of this into a number would handicap HYPO's, or any case-based reasoner's, ability to reason about how the weight should be changed as the context changes.

In conclusion, we have discussed how HYPO determines important clusters of factors and evidence pro and con various interpretations of them and how HYPO defers determination of their relative importance. Through the last phase of the 3-phase process, HYPO has not committed to any weighting scheme. HYPO does not venture an assignment of weights in which the relative importance of factors are set down for all future contexts since a normally unimportant factor may yet make a rhetorical difference if it allows an advocate in the context of a particular problem situation to distinguish between otherwise equally most-on-point precedents. Since the determination of the relative importance of factors is not carried across problem solving episodes, HYPO can change it over time as new case decisions are made and added to the CKB.

Although HYPO searches through the space of possible combinations of factors and magnitudes unassisted by an *a priori* weighting scheme, the search is heuristically guided by the combinations that actually have appeared in real precedent cases. This provides some important advantages in terms of search efficiency and justification. By focusing initially on only the combinations of factors that have historical precedent, a potentially enormous search space is enormously reduced. Actual legal cases tend to involve only small collections of factors. Moreover, the pruning of the search space is performed in a justifiable way. Howsoever HYPO combines factors, there is always an actual case to cite in support of the cluster of factors. HYPO interprets the importance of factors only to the extent that its interpreations are justified by the precedents, and since the interpretations are justified, they can also be explained in terms of the precedents.

In April, 1974, the plaintiff SDRC Corp. ("SDRC") began marketing NIESA, a computer program to perform structural analysis, that SDRC had been developing for some time. The employee-defendant named Smith worked for SDRC until January, 1973 as a computer projects leader. Smith generated the idea of the NIESA program and was completely responsible for its development. On beginning his employment, Smith entered into an Employee Confidential Information Agreement in which he agreed not to divulge or use any confidential information developed by him at SDRC. Immediately upon leaving SDRC, Smith was employed by the corporate-defendant EMRC Corp. ("EMRC") as a vice-president of engineering. In February, 1974, EMRC began marketing a structural analysis program called NISA that it had taken eleven months to develop. Smith had used his development notes for SDRC's NIESA program in building EMRC's NISA program. In connection with sales of the NIESA program, SDRC had disclosed parts of the NIESA source code to some fifty customers.

Figure 1: Problem Situation

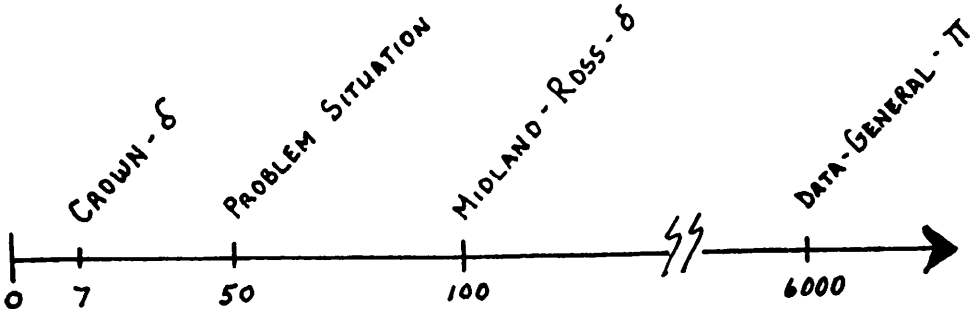


HYPO determines that five factors apply to the problem situation represented by the root node of the claim-lattice and one is a near-miss:

(a) the employee-defendant entered into a nondisclosure agreement
(b) the employee-defendant was the sole-developer of plaintiff's product
(c) whether or not the nondisclosure agreement specifically applied to the product
(d) plaintiff disclosed its product secrets to outsiders
(e) the employee-defendant brought plaintiff's product development tools to his new employer, the corporate-defendant
(f) NEAR-MISS: outside disclosees agreed to maintain confidentiality of plaintiff's product secrets.

Of those factors, (a), (c), and (e) favor the plaintiff; (b) and (d) favor the defendants. If (f) applied, it would favor the plaintiff. The claim-lattice organizes cases from the CKB in terms of overlap of factors shared with problem situation. Each case indicates whether it was won by plaintiff (π) or defendant (δ). There are three groups of equivalent most-on-point cases located in Nodes [1], [2] and [3].

Figure 2: Claim-lattice for Problem Situation



Cases are shown in order of number of plaintiff's disclosures of secrets to outsiders. Case indicates if plaintiff (π) or defendant (δ) won. Plaintiffs in cases toward the right disclosed secrets to more outsiders and are weaker for plaintiff. *Data General* is the weakest case in terms of numbers of disclosures but was still won by a plaintiff.

Figure 3: Cases in Order of Magnitude of Factor (d): Plaintiff's Disclosure of Products Secrets to Outsiders

REFERENCES

[Ashley, 1987] Kevin D. Ashley. *Modelling Legal Argument: Reasoning with Cases and Hypotheticals.* PhD thesis, Department of Computer and Information Science, University of Massachusetts, 1987.

[Ashley and Rissland, 1987] Kevin D. Ashley and Edwina L. Rissland. Compare and Contrast, A Test of Expertise. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, American Association for Artificial Intelligence, August 1987. Seattle.

[Cohen *et al.*, 1987] Paul R. Cohen, Glenn Shafer, and Prakash P. Shenoy. Modifiable Combining Functions. *AI EDAM*, 1(1):47–57, 1987.

[Gardner, 1987] A. vdL. Gardner. *An Artificial Intelligence Approach to Legal Reasoning.* MIT Press, Cambridge, MA, 1987.

[Howe and Cohen, 1987] Adele E. Howe and Paul R. Cohen. *Steps Toward Automating Decision Making.* COINS Technical Report 87–82, Department of Computer and Information Science, University of Massachusetts, 1987.

[Kolodner, 1983a] Janet L. Kolodner. Maintaining Organization in a Dynamic Long-Term Memory. *Cognitive Science*, 7(4):243–280, 1983.

[Kolodner, 1983b] Janet L. Kolodner. Reconstructive Memory: A Computer Model. *Cognitive Science*, 7(4):281–328, 1983.

[Kolodner *et al.*, 1985] Janet L. Kolodner, Robert L. Simpson, and Katia Sycara-Cyranski. A Process Model of Case-Based Reasoning in Problem Solving. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, International Joint Conferences on Artificial Intelligence, Inc., Los Angeles, CA, August 1985.

[Lakatos, 1976] I. Lakatos. *Proofs and Refutations.* Cambridge University Press, London, 1976.

[Mostow and Swartout, 1986] John Mostow and William Swartout. Towards Explicit Integration of Knowledge in Expert Systems: An Analysis of MYCIN's Therapy Selection Algorithm. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, American Association for Artificial Intelligence, August 1986. Philadelphia, PA.

[Rissland, 1988] Edwina L. Rissland. Example Selection: The Underlying Issues. To appear in International Journal of Man-Machine Studies, 1988.

[Rissland and Ashley, 1986] Edwina L. Rissland and Kevin D. Ashley. Hypotheticals as Heuristic Device. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, American Association for Artificial Intelligence, August 1986. Philadelphia, PA.

[Sacerdoti, 1975] E.D. Sacerdoti. *A Structure for Plans and Behavior.* Technical Report Tech. Note 109, SRI International, Inc., 1975.

[Samuel, 1963] A. Samuel. Some Studies in Machine Learning Using the Game of Checkers. In Feigenbaum and Feldman, editors, *Computers and Thought*, pages 71–105, McGraw-Hill, 1963.

[Samuel, 1967] A. Samuel. Some Studies in Machine Learning Using the Game of Checkers. II – Recent Progress. *IBM Journal*, November 1967.

[Simpson, 1985] Robert L. Simpson. *A Computer Model of Case-Based Reasoning in Problem Solving: An Investigation in the Domain of Dispute Mediation.* Technical Report GIT–ICS–85/18, School of Information and Computer Science, Georgia Institute of Technology, 1985.

[Subramanian and Feigenbaum, 1986] Devika Subramanian and Joan Feigenbaum. Factorization in Experiment Generation. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, American Association for Artificial Intelligence, August 1986. Philadelphia, PA.

[Winston, 1975] Patrick H. Winston. Learning Structural Descriptions from Examples. In Patrick H. Winston, editor, *The Psychology of Computer Vision*, McGraw-Hill, New York, 1975.