Explanation-Based Indexing of Cases

Ralph Barletta and William Mark
Lockheed AI Center
2710 Sand Hill Rd.
Menlo Park, CA 94025
(415) 354-5236
Mark@VAXA.ISI.EDU

Abstract

Proper indexing of cases is critically important to the functioning of a case-based reasoner. In real domains such as fault recovery, a body of domain knowledge exists that can be captured and brought to bear on the indexing problem–even though the knowledge is incomplete. Modified explanation-based learning techniques allow the use of the incomplete domain theory to justify the actions of a case with respect to the facts known when the case was originally executed. Demonstrably relevant facts are generalized to form primary indices for the case. Inconsistencies between the domain theory and the actual case can also be used to determine facts that are demonstrably irrelevant to the case. The remaining facts are treated as secondary indices, subject to refinement via similarity based inductive techniques.

1 Introduction

Case-based reasoning is an approach to problem-solving based on retrieving and applying stored solution examples or "cases" (e.g., see [Schank 82], [Kolodner 88]). This problem-solving methodology brings up a variety of research issues–How are new cases acquired over time? What happens if the chosen case fails to accomplish the goal? What knowledge is needed to adapt a case to a new problem? How should case memory be organized in order to select cases relevant to new problems?

Our research focuses on the last of these issues: how to determine the set of storage indices that enable a case to be retrieved "most appropriately" in the future. The system must determine a set of index predicates (called simply "indices" from now on) whose values differentially select cases in memory when applied to incoming problem descriptions. The goal is to determine indices that select exactly those cases that are applicable to—i.e., will result in a solution of—the new problem.

For example, in our domain of fault recovery in automated machinery, the system must infer relevant indices for recovery procedures based on observed fault recovery cases. The system records the series of actions that were used to recover from a fault, indexing it in memory according to features that are relevant to retrieving it again. The indexing problem is to determine which of the observed features were really relevant to performing the particular series of actions that make up the case. Was the time since the last tool change relevant? The current machine operator? Time of day? Temperature and humidity?

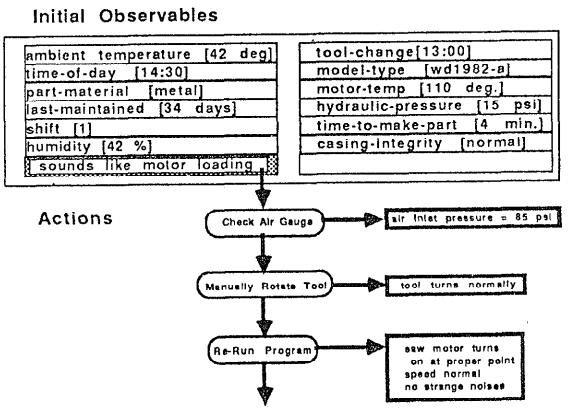

Figure 1: Problem Situation (The Case)

Automatically determining appropriate indices is a learning task: the system must infer relevant indices for a case by generalizing the initial conditions of this specific problem-solving instance. The indexing mechanism has available to it:

- The **problem situation**: the given case and the circumstances of the case's application. The case consists of the series of actions that led to recovery from a particular fault; the circumstances of the case's application are the values of certain observables when the series of actions was initiated. For example (see Figure 1), the operator of a robotic fabrication cell hears the motor "loading up" (i.e., straining) as it is cutting a part. He shuts down the cell and performs a series of diagnostic and recovery actions that result in the problem being fixed (only the first three actions–checking the air inlet gauge, turning the tool by hand, and re-running the machine's program–are shown in Figure 1; the actual case continues).
- A **domain theory**: for fault recovery, a description of cause-effect relationships within the machine. E.g., "low motor temperature is indicative of lubricant that is too cool, indicating that the lubricant viscosity is too high, causing the motor to slow down". As in most real domains, the available cause-effect theory is incomplete.
- **Actions**: The set of actions the operator can perform and their preconditions and results.

This paper discusses our current work in combining these sources of information to guide the indexing of cases. The focus is on adapting explanation-based learning techniques [Mitchell 86; DeJong 86] to identify relevant indices from a set of features.

2 Indexing

The success of a case-based reasoning system relies on the selection of the most applicable stored case. Selection of the wrong case can be very expensive; much more so than selecting the wrong rule to fire in a rule-based system. It is therefore most important for the system to determine indices that most effectively indicate (or contra-indicate) the applicability of a stored case (*cf.* [Bareiss and Porter 87]).

Indices must be selected from knowledge about the state of the world when the case occurred. Unfortunately, everything the system knows about the state of the world when the case occurred *might* be relevant to its applicability. Thus, if a fault recovery case occurred when the ambient temperature was 44 degrees F, that temperature might somehow be relevant.

Second, indices must be generalized [DeJong 81; Kedar-Cabelli 87]. Otherwise, only an exact match can be the criterion for case applicability. For example, if ambient temperature really is a relevant indicator for a case, we would want to index the case on a range of temperature values. Without this generalization, the case would only be deemed applicable when it happened to be exactly 44 degrees again. On the other hand, indices should not be over-generalized. A case that is a good choice at 44 degrees may be a very poor choice at 34 degrees.

What is needed, then, is a method for indexing that wades through the large quantity of world state knowledge, most of which is irrelevant to the case's applicability, to find the truly relevant initial conditions–and then generalizes these conditions just enough but not too much.

Previous approaches have relied on inductive methods [Kolodner 83; Schank 82] to reduce the index set and generalize the resulting indices. This approach has achieved a rudimentary level of success in early case-based reasoning systems [Kolodner and Simpson 88]. There are, however, some major problems that inductive methods have with respect to indexing. Many cases must be observed before the relevant indices can be induced. Also, induction allows *irrelevant* indices to be put into memory, and remain there, until they are incrementally weeded out by the induction process.

Additionally, coincidental occurrences can cause the generation of erroneous indices. For example, if the system sees five fault recovery cases where the same problem occurred and the operator on duty was the same for all five, then the name of the operator might be chosen as a relevant index. Although this could be an appropriate index, it is more likely just a coincidence.

This leads to the final problem with any inductive method: induction can use only the evidence available to the system. It would be much better if index selection could be based on theories of the domain that are derived from widely held principles of how the world works. This seems like a very reasonable goal to attain given the fact that most real world domains are in fact understood (at least at some level) in terms of physical properties, cause-effect, etc. Of course, this is the fundamental insight of explanation-based learning approaches.

3 Explanation-Based Indexing

The task in case indexing is to find relevant features to look for in future problems to decide whether or not the stored case is applicable. This sounds like an explanation-based learning problem: create an explanation of what makes a case applicable to a problem description, and consider as relevant those features required for the explanation (*cf.* Koton 1988]). In the domain of fault recovery, we will be looking at explanations that answer the question: what was it about that state of the world that prompted the given sequence of diagnostic and repair actions? Thus the "goal concept" [Mitchell 86] in explanation-based learning terms is the sequence of actions in the case, and the reasoning is aimed at justification [DeJong 83], not validation [Hammond 86; Keller 87] or classification [Bareiss and Porter 87]. In other words, we are not attempting to explain the final diagnosis of the case with respect to the initial conditions (validation), but instead in justifying the choice of the actions taken in the case with respect to those conditions.

3.1 Using an Incomplete Domain Theory

One consequence of an incomplete domain theory is that the system will not be able to show that every feature is either relevant or irrelevant. This means that in addition to deriving explanations for which features are relevant, the system must also derive explanations for which features are irrelevant (*cf.* [Hammond and Hurwitz 1988]). And, the system will have to do something with those features that cannot be explained as either relevant or irrelevant.

In our Explanation-Based Indexing (EBI) process, the system uses the case and the domain theory to categorize features as "known to be relevant", "known to be irrelevant", and "possibly relevant" (i.e., neither relevant nor irrelevant according to the domain theory). "Known to be relevant" features may be either indicative or contra-indicative with respect to applying the case.

These features are then made into indices to store the case in memory. Known to be relevant features are generalized to form the main organization of primary indices–necessary conditions for case applicability. Known to be irrelevant features are dropped from consideration as indices. Possibly relevant features are used as secondary indices. When a new problem comes in, the primary indices are used to determine a set of applicable cases, and the secondary indices are used to choose among those cases. Secondary indices are refined by induction as more and more cases are observed.

3.2 Justification

The EBI justification process used to determine the relevance of features must be driven by the purpose of the problem solver (*cf.* [Kedar-Cabelli 1987]). In the fault recovery domain, the purpose is some form of "fix the machine", which may include "find the problem", "make the

machine safe for investigation", etc. Cases are triggered by a "presenting symptom", i.e., observable evidence that something is wrong. The goal of the EBI process is to justify all actions in the case in terms of their role in relating symptoms and observables to "causes", i.e., correctable problems, and/or their role in correcting those problems. (Actions whose effect is to establish prerequisites for later actions do not require this justification; they are considered to be "covered" if the later actions can be justified.) Thus, if the effect of an action is to provide additional observables, the system will examine its domain theory to see how the additional information must have been used to reduce the number of possible causes, increase certainty about a cause, etc. Deriving what the actual cause turned out to be is not necessary for this justification. In fact, an important aspect of this work is that the system can justify the actions even if it *cannot* validate the result.

3.3 Assumptions

In developing the justification process, we have made several simplifying assumptions concerning the fault recovery activity. One is that the goal of the case is to recover from the fault, not to do exhaustive testing to find all possible causes of the fault. This is what allows us to reason about relevancy based on reducing the number of possible causes, etc. Second, we assume that the actions in the case are there solely for the purpose of fault recovery. Thus, all actions are justified strictly in terms of their contribution to the fault recovery goal (though their contribution may consist of setting up prerequisites for later actions). Third, we assume that each action in the case really *does* contribute to fault recovery–even if the system cannot see how. This means that in case of doubt, we will believe the operator; this is important in determining irrelevant features, as we will see.

Finally, we have a set of assumptions about the completeness of the system's knowledge. Knowledge of operator actions is expected to be "complete" in the sense that the operator is not doing things to affect the machine that are not included in the case (a reasonable assumption in this domain). The system's domain theory of causal knowledge is expected to be incomplete. If the system is unaware of relevant observables (e.g., if the the system is unaware that the operator makes decisions based on whether or not there is a burning smell), indexing will be be incomplete, but it will still be effective with respect to the set of observables that are known. Similarly, if the system is unaware of certain cause-effect relationships, an index that should have been primary may end up as secondary, and certain irrelevancies will go undetected, but the system will still be effective in finding primary indices in the context of its known theory.

3.4 Available Knowledge

The knowledge available to the system consists of the case, the set of observable facts, the domain theory and the operator actions. The observable facts consist of those available initially, including the presenting symptom (top of Figure 1), and those "discovered" via actions in the case–e.g., the fact that air inlet pressure = 85 psi (bottom of Figure 1). The domain theory consists of some causal knowledge relating functions and states of the machine. For example, the theory in Figure 2 shows (at the top) a basic functional structure consisting of an air compressor that drives a motor, which turns a tool, which cuts a part. In addition, the theory defines some state information that is relevant to the functioning and malfunctioning of the machine. For example the figure shows that "high internal friction" indicates "slow turning of the motor"; that "low quantity of lubricant" indicates "high internal friction"; and so on.

The theory also relates observables–both those available initially and those discovered via actions–with the states they indicate. In Figure 2 the observables are shown in boxes, with discovered observables in thicker boxes. The discovered observables are the results of the actions that discover them, as shown at the bottom of Figure 2. (Remember that the system has available to it all of the actions the operator can perform.) Note that different values of an observable may be connected to different states: e.g., "tool resists steadily" indicates the state of high internal friction, while "tool turns normally" indicates normal internal friction; similarly only motor temperatures less than 150 degrees indicate the state of the lubricant being too cold.

3.5 The EBI Process

Given the assumptions and the available knowledge described above, the EBI process constructs indices from the initial observables. Only the initial observables are used for indexing because they will be all that is available the next time a similar problem arises.

The EBI process consists of three steps. Step 1 identifies all of the initial observables that can *possibly* be relevant to justifying the actions. These will be the observables that connect to all of the hypotheses (causal explanations) that can *possibly* explain the behavioral states relevant to the case. The domain theory represents all known hypotheses that explain all known behavioral states. The goal of Step 1 is therefore to find that subset of the domain theory that applies to the case at hand. By assumption, this part must be the set of hypotheses that explain the behavioral states indicated by:

- the presenting symptoms (because the "why" of performing the case has been defined to be "to recover from the presenting symptoms"); or
- the observables that can be discovered by any of the actions in the case (because, due to the incompleteness of the theory, the system may not be able to understand how some of the actions help to recover from the presenting symptoms).

The procedure for Step 1 is therefore to find all subnetworks of the domain theory that contain either a presenting symptom or an observable that can be discovered by one of the case actions. These sub-networks are in fact trees (*explanation trees*) whose roots are behavioral states and whose leaves are observables (see Figure 3). Explanation trees represent all known causal explanations of the behavioral states that are known to be important for the case.

For example for the case in Figure 1 all initial observables that can be connected (through any number of "indicative of" links in Figure 2) to the state "motor turns slow" are marked as possibly relevant, because this is the

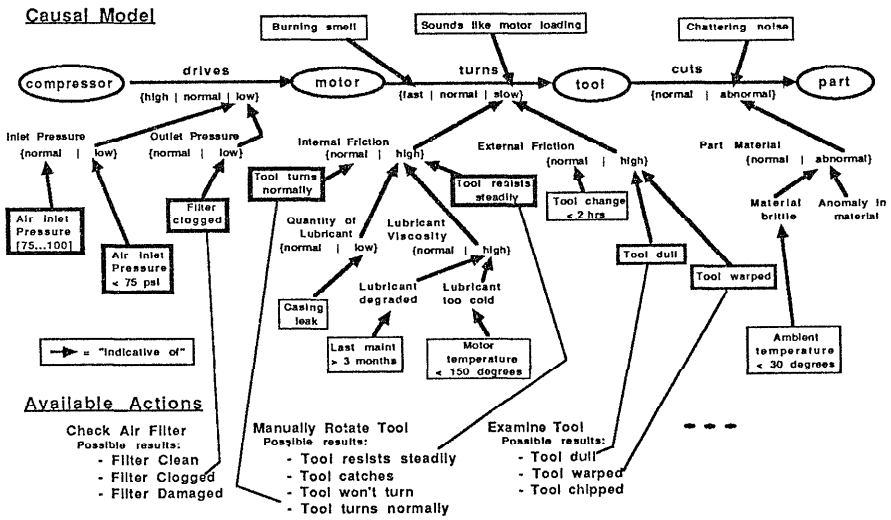

Figure 2: Domain Knowledge

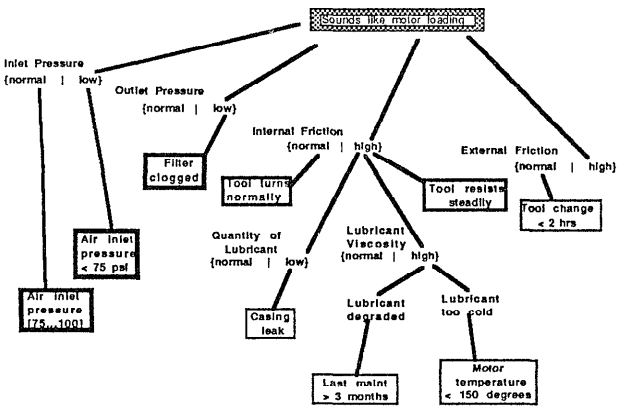

Figure 3: Explanation Tree

root of the (single) presenting symptom "sounds like motor loading". Thus, "motor temperature less than 150 degrees" is included in the round-up because it indicates lubricant too cold, which indicates "high lubricant viscosity", which indicates "high internal friction", which indicates "motor turns slow". Figure 3 shows the explanation tree for the case of Figure 1 and the domain theory of Figure 2. The computational demand on producing this tree can be reduced by caching pre-computed sub-trees or by using parallel computation; we are currently looking at both alternatives.

It is important to note that because of the incompleteness of the domain theory, more than one explanation tree can be created. This happens when there is no known relationship between the behavioral states at the root nodes of the separate trees. The existence of these disjoint trees prevents validation of the final result of the case, because for validation, all actions in the case must be understood with respect to their role in recovering from the presenting symptoms. This can only be accomplished if all actions are understood in terms of a connected causal explanation, i.e., a single explanation tree. Justification is still possible because subsets of actions in a case can be understood in terms of their role in eliminating branches of a single (disjoint) explanation tree.

Step 2 of the EBI process is to reason with the explanation trees in order to determine the relevance or irrelevance of each observable in the trees. Following our assumptions, the only reason to perform an action is to "prune" the tree by eliminating one or more of its hypothesis-branches. As was previously stated, if there are more than one explanation trees, each can be treated separately for this justification reasoning.

Once the trees have been formed, the process can proceed to determine which observables are relevant and irrelevant to the action to be justified. Relevance and irrelevance are determined according to the following rules:

- An observable is relevant and indicative with respect to an action if:
  - The presence of the observable removes some of the possible *competing* hypotheses.
- An observable is relevant and contra-indicative with respect to an action if:
  - A different value of the observable would have eliminated the hypotheses that would have been resolved by taking the action.
  - Some other value of the observable would have determined a different hypothesis as the cause of the presenting symptom.
- An observable is irrelevant if *all* values of the observable have no bearing on taking or not taking *all* actions of the case. Because of the inherent incompleteness of the model this condition is very hard to show. However, we can demonstrate the irrelevance of an observable if:
  - the observable suggested *not* taking the action, and it was taken anyway. Since (by assump-

tion), we believe that the operator acted correctly, the domain theory must be missing the causal paths that would render that observable irrelevant. Thus, in addition to showing the observable to be irrelevant, this rule also pinpoints places in the theory in which there is missing information.

Note that all actions must be justified in the context in which they were originally initiated (see [Keller 1987]). That is, knowledge about the problem being solved changes as the case progresses, because actions discover new information, which influences future actions. So, as each action in the case is justified, the actual result of that action may cause branches of the explanation tree to be pruned. This incremental updating of the explanation tree with information from the actual case allows us to maintain the appropriate context for justifying each action in the case. For example, after we justify the **Check Air Gauge** action, we use the fact that air pressure equals 85 psi to prune the "low inlet pressure hypothesis" from the explanation tree, and then proceed to justifying the next action in the case, **Manually Rotate Tool.**

Step 3 of the EBI process is to make the features into indices. All features not included in the explanation trees become secondary indices. Irrelevant features are elided. The relevant features are generalized using the ranges supplied in the theory: since the justification reasoning has been solely in terms of the ranges specified in the theory, we can certainly generalize to that level. These generalized relevant features become the primary indices of the case–necessary conditions for retrieving the case. The secondary indices are sufficient conditions for retrieving the case. In general there will be a "family" of cases indexed under any set of primary indices; they are differentiated within the family by their secondary indices.

4 Example

To clarify these ideas, we describe an example of the EBI process for the case in Figure 1. For simplicity, we will focus on the single action **Manually Rotate Tool.** So, for our example, the justification question is: "why is it reasonable to perform the action **Manually Rotate Tool,** given the currently known observables (i.e., including the fact that the air gauge has been checked and the inlet pressure found to be 85 psi) ?"

The explanation tree has already been constructed, as shown in Figure 3. The EBI process is therefore at Step 2. The system examines **Manually Rotate Tool** to see which hypotheses are impacted by taking the action. For **Manually Rotate Tool,** these are the hypothesis that confirm or deny "high internal friction".

For the **Manually Rotate Tool** action, tool change is relevant and indicative. We know that the tool was last changed at 1 o'clock and the model tells us that if the tool was changed within two hours we can conclude that external friction was normal. This situation matches the relevant and indicative rule which says an observable is relevant if it eliminates a competing hypothesis.

"Casing leak" is relevant and contra-indicative. Because, if the casing *were* leaking, we would conclude that the quantity of lubricant is low and therefore that internal friction is high. This would make it unnecessary to perform **Manually Rotate Tool.** So, knowing there is *no* leak justifies (in part) taking the action.

Motor temperature is irrelevant. The actual temperature was 110 degrees. According to the theory, this indicates that the lubricant is too cold, making the lubricant viscosity too high, which indicates high internal friction. This means that it is not necessary to perform the **Manually Rotate Tool** action. But we know this action was taken anyway. Since we assume that the operator acted correctly, it must be that the theory is missing the information that would show that motor temperature is in fact irrelevant. Note that motor temperature has so far been shown to be irrelevant only to **Manually Rotate Tool.** In order to be shown irrelevant to the entire case, it must be shown to be irrelevant to all of the actions.

The final result of the EBI process applied to all of the actions of the case is the following set of indices:

- **irrelevant**
  - Motor temperature
- **primary indices**
  - Sounds like motor loading (indicative)
  - Tool changed less than 2 hours ago (indicative)
  - Casing leak (contra-indicative)
  - Maintained more than 3 months ago (contra-indicative)
- **secondary indices**
  - all other features at the values shown in Figure 1

5 Discussion

We have tried to show that explanation-based learning techniques can be applied to the indexing problem, even in the face of an incomplete domain theory. The existence of the case turns the learning problem into one of justification, which we believe is more tractable when domain knowledge is incomplete. The EBI process differs from previous explanation-based approaches primarily in terms of what happens *after* the explanation trees have been formed. As we have seen, relevance and irrelevance must be determined by reasoning about eliminating branches, etc., not by simply examining the leaf nodes of the tree.

The system described in this paper is currently under construction. In addition to the many unexpected issues that will arise during the implementation, we have already noted several places for required extensions of our method. In particular, we must have a mechanism for dealing with uncertainty–i.e., for domain theory links that express "might indicate" rather than "indicates". We must also weaken some of our assumptions to allow the fact that any diagnostic process contains actions that have more to do with "standard procedure" than with confirming or denying any particular hypothesis. Our goal is to explore these issues in the context of a building a working fault recovery system.

6 Acknowledgements

We would like to thank Linda Cook for reviewing and commenting on this paper as well as providing input that helped us understand and solidify our approach.

7 References

Bareiss E.R., Porter B.W., Wier C.C., (1987) "Protos: An Exemplar-based Learning Apprentice", *Proceedings of the Fourth International Workshop on Machine Learning*, Irvine, CA, 1987.

Carbonell J., Velose M., (1988) "Integrating Derivational Analogy into a General Problem Solving Architecture", *Proceedings of the 1988 Case-based Reasoning Workshop*, Clearwater, FL.

DeJong G.F., (1981) "Generalizations Based on Explanations", *Proceedings of the Seventh International Joint Conference on Artificial Intelligence* (pp. 67-70), Vancouver, Canada.

DeJong G.F., (1983) "Acquiring Schemata Through Understanding and Generalizing Plans", *Proceedings of the Eight International Joint Conference on Artificial Intelligence*, Karlsruhe, Germany.

DeJong G., Mooney R., (1986) "Explanation-based Learning: An Alternative View", *Machine Learning Journal*, Vol. 1, Number 2, Pg. 145.

Hammond, K.J., (1986) *Case-based Planning: An Integrated Theory of Planning, Learning and Memory* PhD. Dissertation, Yale University.

Hammond, K.J., (1987) "The Structure of a Diagnostic Case", Technical Report: University of Chicago.

Hammond K.J., Hurwitz N., (1988) "Extracting Diagnostic Features from Explanations", *Proceedings of the 1988 Case-based Reasoning Workshop* Clearwater, FL.

Kedar-Cabelli S.T., (1987) "Formulating Concepts According to Purpose", *Proceedings of AAAI-87*, Seattle, WA.

Keller R.M., (1987) "Concept Learning in Context", *Proceedings of the Fourth International Workshop on Machine Learning*, Irvine, CA, 1987.

Kolodner, J.L., (1983) "Maintaining Organization in a Dynamic Long-term Memory", *Cognitive Science*, Vol. 7, Pg. 243.

Kolodner, J.L., (1988) "Retrieving Events from a Case Memory: A Parallel Implementation", *Proceedings of the 1988 Case-based Reasoning Workshop*, Clearwater, FL.

Kolodner J.L. and Simpson R.L., (1988) "The MEDIATOR: A Case Study of a Case-Based Problem Solver" Georgia Institute of Technology Technical Report: GIT-ICS-88/11.

Koton, P., (1988) "Reasoning about Evidence in Causal Explanations", *Proceedings of the 1988 Case-based Reasoning Workshop*, Clearwater, FL.

Mitchell T.M., et al, (1986) "Explanation-based Generalization: A Unifying View", *Machine Learning Journal*, Vol. 1, Number 1, Pg. 47.

Schank R.C., (1982) *Dynamic Memory: A Theory of Reminding and Learning in Computers and People*, Cambridge, England: Cambridge University Press.