A Framework and an Analysis of Current Proposals for the Case-Based Organization and Representation of Procedural Knowledge

Roland Zito-Wolf and **Richard Alterman**
Computer Science Department/Center for Complex Systems
Brandeis University, Waltham, MA 02254
rjz@cs.brandeis.edu, alterman@cs.brandeis.edu

Abstract

Case-based reasoning refers to the class of memory-based problem solving methods which emphasize the adaptation of recalled solutions (explanations, diagnoses, plans) over the generation of solutions from first principles. CBR has become a popular methodology, resulting in a proliferation of case organization and representation proposals. The goal of this paper is to sort through some of these proposals. Using the formal models of "procedure" and "case-based reasoning" introduced in Zito-Wolf and Alterman (1992), we compare three current proposals for the organization of procedural case-bases: individual cases, microcases, and multicases. We give a worst-case analysis that shows the advantages of the multicase in terms of case storage and retrieval costs. The model predicts that multicases reduce case storage and retrieval costs as compared to the other two models. We then provide some empirical evidence from an implemented system that suggests that the trends observed in the formal model are also observable in case bases of practical size.

1 Introduction

In recent years, Artificial Intelligence researchers have become increasingly interested in techniques for reasoning directly from examples rather than from the abstract knowledge one might distill from them. *Case-based reasoning* (CBR) refers to the class of memory-based problem-solving methods which emphasize the adaptation of recalled solutions (explanations, diagnoses, plans) over the generation of solutions from first principles (such as a domain theory). People rely heavily on such techniques both in expert domains, such as medical diagnosis, legal reasoning, and in coping with the more mundane problems that arise in everyday life (Kolodner & Simpson, 1989).

The variety of applications of CBR has resulted in a proliferation of case organization and representation proposals. Until now the evaluation of proposals has been informal making comparisons difficult. The goal of this paper is to put some of these issues on a firmer foundation.

Because it is difficult to discuss these issues in any detail independent of specific tasks, this paper will focus on the organization of procedural knowledge for planning tasks. Because procedures are typically executed many times, providing large numbers of related yet distinct cases, and they are complex, with many interrelated components (i.e., steps), procedural domains are a good test domain for examining these issues. It is also an area where perhaps the largest number of different proposals have been made, which we interpret as reflecting the difficulty of the representational problem.

We will present a formal model which will allow for the comparison of case-base organization proposals. We discuss three existing proposals, the third of which, the *multicase*, combines the benefits of the other two. We give a worst-case analysis that shows the advantages of the multicase in terms of case storage and retrieval costs. We also provide some empirical evidence that suggests that the trends observed in the formal model are also observable in case bases of practical size. For an extended analysis that also discusses additional factors (e.g., adaptation costs) see Zito-Wolf (1993).

2 Representing Procedures

Let a *problem* as presented to the system consist of a *situation* (a world state) plus a *goal* to be achieved. The solution to a problem will be a *procedure*, that is, a sequence of steps that achieves the goal in that situation. Assume we are given a set of examples of some procedure. What shall we call a case? In this paper we will use the terms *example* or *episode* for an instance of a problem and a solution procedure. The term *case* will be reserved for the unit of storage and retrieval from memory. Many CBR systems, especially early ones (e.g., CHEF, CYRUS) have equated the two; however, they are logically distinct, and it is useful to distinguish them.

Storage Requirements Although memory is becoming increasingly plentiful, at any given time mem-

ory is a finite resource which needs to be traded off against possible uses. Consequently, one significant issue is the amount of memory required to store all the cases. As the quality of solution retrieved by a case-based reasoner is expected to be monotonically increasing with the number of distinct, relevant cases available to it, one wants to accommodate as many cases as possible. On the other hand, the number of potential cases can be very large – it is exponential in the complexity (number of features) of the cases.

Retrieval Cost Retrieval cost is a function of the number of cases examined and the effort required to determine their relevance to the current situation. Both the number of cases to examine and the cost of deciding among them can be large in practical case-bases. When the number of cases becomes large, retrieval normally relies on *indexes*. An index is an auxiliary data structure that provides a direct mapping from each specific feature of interest to cases having that feature. Complete indexing may not be practical for all features, however; for example, the type of data involved may not admit of simple indexes, or the feature of interest may be computed only at execution time. Hence, we will look at both indexed and unindexed retrieval.

3 Modelling Procedure Organizations

In this section we will compare the behavior of three representations for procedural case-bases, based on the formal models of "procedure" and "case-based reasoning" introduced in Zito-Wolf and Alterman (1992). The results of this section are summarized in Table 1.

We assume the procedural knowledge to be captured has the form of a complete binary decision tree T of uniform depth n. Each node $i \in T$ contains a *step* to be performed plus a *decision* selecting the next node to be executed. T therefore contains $|T| = 2^n - 1$ steps and $2^{n-1} - 1$ decisions (those in the leaves are ignored). Each procedure execution *episode* will consist of $n - 1$ decisions selecting n steps along some path in T from the root to a leaf node.

Let the input to the decision at a node i be the set of binary features F_i, so that $F = \bigcup_{i \in T} F_i$ is the set of all features referenced by the procedure. We assume there exists some upper bound $f = \max_{i \in T} |F_i|$ on the number of features tested by any specific decision, and that f is small compared to F.[1] To estimate F we choose $(n-1)f$. This corresponds (for example) to a procedure composed of $n - 1$ distinct decisions occurring in a fixed order.

Storage Case-based reasoning for procedure generation is the example-based selection of a sequence of steps to achieve a given goal. Each occasion for selection is a *problem* P_i, the process of searching through the case-base to solve a problem is a *retrieval*, and the number of steps determined by each problem is the *problem size* S_P. The solution to each problem will be encoded in memory as some set of *cases* C_P. Each case pairs a problem solution with a conjunction of features for which it applies. Since it has been stipulated that a given decision references at most f features, at most 2^f cases will be required to represent a decision, one for each possible conjunction of the features and their negations. The union of all the C_P is the case-base C. The *case-base size* $S(C)$ will be measured as the number of step instances in the case base C, that is, the product of the number of cases $|C|$ and the problem size. The set of examples from which a given case base is derived will be denoted E.

Retrieval and Indexing Consider a linear search model of case retrieval, in which the *retrieval effort per problem* R_P is proportional to the number of feature tests made. R_P is the product of the number of cases to be searched through and the number of features to be tested per case. If P is the number of problems per episode, then the *total retrieval effort* per episode $R = R_P P$.

Because case-retrieval via linear search involves effort exponential in the procedure size (n), most CBR systems use some form of *indexing* for faster retrieval.[2] We will model an index as a boolean discrimination network which tests just enough features to discriminate all the cases. Assuming that the index is well-constructed, the decision cost per problem R_{XP} is proportional to the depth d of the index, that is, log base 2 of the number of cases entering into a given decision, and the pre-episode cost $R_X = R_{XP}P$ The size of an index (in nodes) is $2^d - 1$, of the same order as the number of cases indexed.

3.1 Individual Cases

The first representation we will consider is the storage of individual cases (CHEF, Hammond 1990; COOKIE, McCartney 1990). In this method the unit of retrieval from memory, the case, is taken to be the same as the unit of knowledge presentation, the episode. Procedure execution over such a case base consists of a single up-front decision among alternative cases (Figure 1a). That is, case retrieval returns a single complete example episode for the target task, which (usually after some tweaking) is interpreted as a procedure for the desired task. Also in this class are MOP-based systems (Kolodner, 1983; Lebowitz, 1983; Turner, 1989).

[1] For simplicity, in the remainder of the paper we will write X for $|X|$ where there is no likelihood of confusion.

[2] The term "indexing" is used in the CBR literature in at least three distinct senses: to refer to *performance* methods that accelerate access to subsets of the case base; to refer to *organizing* methods that group cases observed to have similar features, typically in the service of generalization (cf. CYRUS and IPP); and to refer to the process of *encoding knowledge* by adding features to a case-base to define sets of cases with related content. Our analysis focuses on the first of these meanings.

MOPs indexes, though more complex then our model index, serve the same function. The key similarity is that cases are stored and accessed as wholes; for the purpose of this paper the indexing differences can be ignored.

Storage Each episode of (i.e., path through) T is a case, so the problem size $S_P = n$. The entire mapping from features to procedure is performed in one retrieval ($P = 1$) with 2^{n-1} potential outcomes. The number of potential cases can be estimated from the total number of features referenced, yielding $C = 2^{(n-1)f}$, $S(C) = nC$, and $R_P = FC$).

Retrieval Cost Unindexed retrieval cost is determined by the number of feature comparisons made, which is the size of the case base times the number of features per case. Only one retrieval is required, so that $R = R_P P = F2^F$. Using an index, the decision cost is the log base 2 of the number of cases entering into a given decision, or $O(F)$, hence the popularity of indexing. A simple index requires on the order of as many nodes as there are cases indexed, so its storage can be ignored.[3]

Most notable here is the rapid growth in possible cases as either the number of features or procedure steps increases. This is due to the fact that the cases are significantly redundant – each case instantiates a complete path through the tree. To represent procedural knowledge in an individual-case-based system, for, say, one's knowledge of procedures for telephoning, one would have to store a case for every possible event sequence and every situation type that could be encountered in executing one's phone-call procedure, or at least a significant number of them. Note that as the case-base fills up with variant episodes, retrieval also becomes more expensive.

3.2 Microcases

The second class of procedure representations is the *microcase*. In this method, each example presented to the system is converted into *many* cases, typically one for each step of the episode. All the cases so created go into a common case-base. At execution time, procedures are not so much retrieved as incrementally reconstructed, steps being selected sequentially by separate retrievals over the case-base. (Figure 1b). Microcases avoid the redundant storage and difficulty with transfer that we encountered with individual cases. Micro-case-based systems have been applied to planning (Langley and Allen, 1990), parsing (Goodman, 1991), and word pronunciation (NetTalk, Stanfill and Waltz 1986).

[3]Note that whereas in a simple index the path to a case is unique (enumerating each case's features in some specific order), methods involving multiple indexing – such MOPs-based systems – will require significantly more storage.

Storage To encode procedure T using microcases we make each selection of a step a separate problem. Then $S_P = 1$, and $C_P = 2^f$ cases per problem. There are $P = n$ problems per episode, but $2^n - 1$ problems to be encoded to represent the entire procedure, giving $C = 2^f(2^n - 1)$.

Retrieval Cost The price of the microcase's additional flexibility and reduced storage is increased retrieval costs: a case retrieval occurs at every procedure step. Each such decision has to select among a large number of options, namely, all possible steps. Unindexed retrieval effort per problem is $R_P = (F+n)2^{n+f}$. Note that n additional features are added to distinguish the 2^{n-1} potential "current positions" within the represented procedure; that is, the structure of the procedure T needs to be encoded implicitly as extra features referenced by the cases. Total retrieval effort R is n times the per-problem figure, or $O(n^2 2^{n+f})$. Indexed decision cost is the log base 2 of the number of cases entering into a given decision, or $O(n+f)$. There is a single index of size $|C|$, or $O(2^{n+f})$.

3.3 The Multicase

We have proposed a third organization, the *multicase* (Zito-Wolf and Alterman 1992). By a multicase we mean a structure which merges many individual episodes but retains a representation of the overall structure of those episodes. Episodes are merged through the introduction of conditionals, so that the details of the individual episodes can be retained (Figure 1c). Each example is represented by some specific path through the procedure graph. Episodic memory is organized around the underlying procedure, which serves to partition the procedure into many individual decisions. This organization efficiently accommodates variation among episodes, and is morcover a convenient vehicle for organizing related knowledge, such as unexpected events or episodes where the plan failed.

The key difference between individual cases and multicases is that individual cases store episodes in memory as separate structures linked by indexes or abstraction hierarchies, whereas multicases index them at a finer-grained level by segmenting them and distributing them across the partonomic (i.e., step) structure of the procedure.

The key difference between the multicase and microcase is that for multicases the decision overhead is reduced by partitioning the pool of cases according to the structure of the procedure, whereas for microcases all the cases go into a single pool, so that each decision must decide among a much larger range of options.

The historical antecedents of all of the case models discussed here are the ideas of scripts (Schank & Abelson, 1977), MOPs, and their elements, scenes and tracks (Schank, 1982). The issue is how to organize an agent's episodic memory using ideas like this in the most effective manner. This is the problem the multicase addresses.

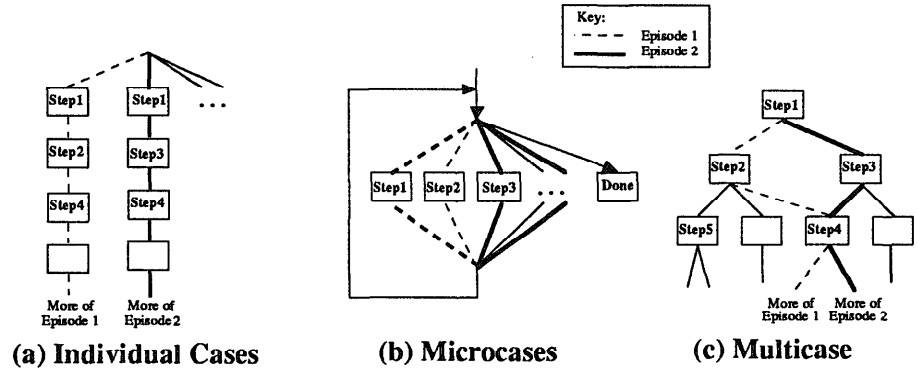

Figure 1: Three Methods of Representing Episodes

Storage The case base is partitioned both by the type of decision (e.g., next-step, role-choice); the structure of the procedure is expressed explicitly in the structure of the multicase. We have $S_P = 1$, $|P| = n-1$ problems per episode, with $|C_P| = 2^f$ and $2^{n-1} - 1$ problems overall, for a total of $|C| = O(2^{n+f-1})$.

Retrieval Cost Because the multicase focuses on only one decision at a time, the number of cases that must be searched through and the number of features needing to be consulted at any given decision point are greatly reduced. For the multicase, only f features need be consulted per decision, so only 2^f cases need be examined; the rest of the cases are only relevant to *other* decisions. Thus $R_P = f \cdot 2^f$ and $R = O(nf \cdot 2^f)$. Indexed decision cost is the log base 2 of the number of cases entering into a given decision, or $O(f)$. For multicases there may be as many 2^{n-1} indexes, one for each decision, the total index size is 2^{n+f-1}.

4 Empirical Demonstration

Our source data derives from a sample of runs of the FLOABN system on problems involving the operation of household and office devices such as telephones, copiers, and vending machines (Alterman, Zito-Wolf, and Carpenter 1990; Alterman, Carpenter, and Zito-Wolf 1990). FLOABN was provided initially with a skeleton multicase for a simple procedure for the usage of each class of device. It was then presented with a sequence of 50 problem situations involving these procedures; for example, phone calls varied in destinations, locations, call types, and phone features. There were on average 25 steps per episode, yielding in excess of 1200 cases. For each episode we collected over 60 items of data about the evolution of memory and procedure performance. Each run of the example sequence required approximately 8 hours on an 8 Mbyte Macintosh IIx under Allegro Common Lisp.

4.1 Comparing Representation Methods

Our methodology for comparing the three case-organization models will be to use our formal model of procedures to define mappings between the multicase model and the individual-case and microcase models. We will gather data from runs of FLOABN to estimate practical values for the relevant parameters of the model: the total number of different features that were observed (F), the average number of features referenced in making a decision (f), the average number of decisions needing to be made per episode, and the number of cases C. We then run this data through the model to derive costs for the three methods.

It would have been preferable in some sense to compare implementations of the three methods directly rather than comparing projections based on a single implementation. Problems emerged from such an attempt. Several operations that were facilitated by the multicase – for example, instruction-processing and plan-modification operations – were hard to do, and in some cases even to define, on other representations. This is because these functions required an evolving plan schema representation, which the multicase provides and the other organizations do not.

4.2 Storage Costs

Figure 2 compares memory requirements for case storage. The greatest storage requirements are for the individual case method. Individual cases save much redundant information with each nominally different case. In contrast, case memory growth for the microcase and multicase methods tail off as as the memory becomes familiar with the range of variation of its procedures. The multicase method uses less storage than the microcase method. This is because the multicase partitions its space of decisions much more finely, and consequently, many more decisions have only one option and hence do not require that any cases be stored for them. Figure 3 compares the number of decisions with > 1 option for each method. The graph for microcases represents something of an upper bound, since

Item	Individual cases		Microcases		Multicases	
	formula	example	formula	ex.	formula	ex.
1. Total cases $\|C\|$	2^F	256	$O(2^{n+f})$	124	$O(2^{n+f-1})$	75
2. CB size	$n2^F$	1280	$O(2^{n+f})$	124	$O(2^{n+f-1})$	75
3. Effort/problem R_P	$F2^F$	2048	$O(F2^{n+f})$	1612	$f2^f$	8
4. Effort/problem R_{XP}	F	8	$O(n+f)$	7	f	2
5. Effort/episode R	$F2^F$	2048	$O(nF2^{n+f})$	8060	$(n-1)f2^f$	40
6. Effort/episode R_X	F	8	$O(n^2+F)$	35	F	8

Table 1: Storage and Retrieval Cost Summary

microcases strive to have as many options as possible at each decision. Comparison with the graph for multicases shows that in our example situations, anywhere from 50-75% of these decisions are unnecessary. This suggests that microcases overemphasize transfer at the cost of greatly increasing the amount of knowledge required to "learn" the procedure.

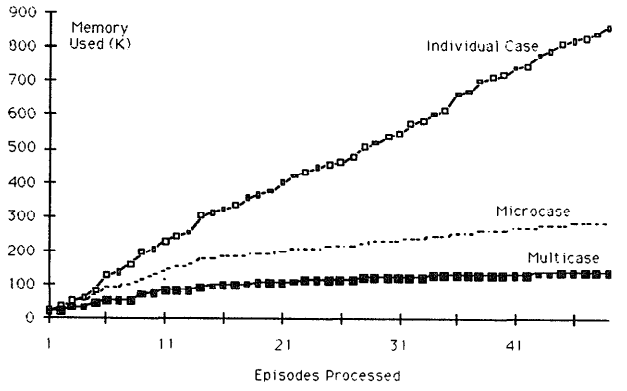

Figure 2: Memory Usage Comparisons

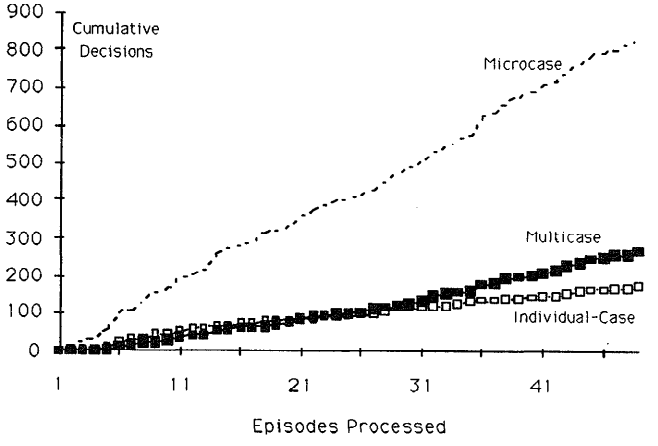

Figure 3: Cumulative Decisions per Episode

4.3 Decision Cost

Now we wish to use our empirical data to compare decision costs. Recall that our model of unindexed decision cost R_P was the product of features tested and cases examined per problem, whereas for indexed retrieval, we took R_{XP} to be the depth of the smallest index needed to discriminate the cases under consideration. The required quantities were determined from the example sequence. Figure 4 shows the number of cases available per decision for the three methods. The number of features referenced per case was measured at the end of the problem sequence to be $f \approx 7$ per case and total $F \approx 16$. Lastly, the number of retrievals per episode was presented in the previous section (Figure 3).

Individual cases introduce a difficulty here. It is unreasonable to assume that only one up-front case retrieval is needed per episode, because it ignores the cost of the runtime decision-making which the other methods are being "charged" for. Thus, since each such unanticipated circumstance incurs at least one additional retrieval, we have added to the count of retrievals per episode in Figure 3 one retrieval per unanticipated decision event or relevant situation feature.

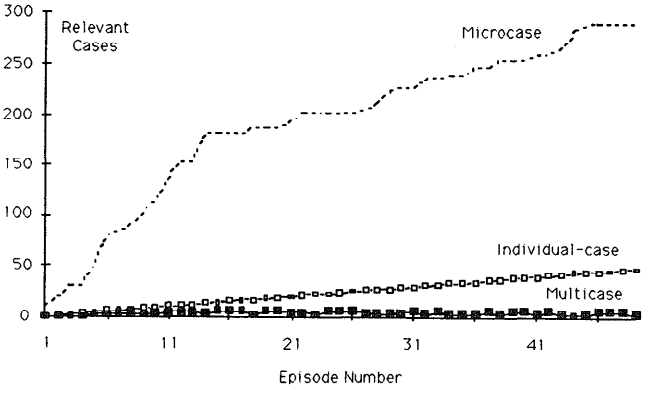

Figure 4: Choices Per Decision

Figure 5 graphs indexed retrieval effort for the three methods. The first result we observe is that microcases have much larger retrieval effort – an order of magnitude larger than multicases even in the fully indexed case. This difference is the product of the two

differences shown above: the number of cases requiring a decision, and the number of cases examined per decision. Secondly, we observe that multicases require effort less than (but roughly comparable to) that for individual cases. The individual-case method's advantage of making fewer decisions per episode is more than offset by the extra costs of handling more contingencies and sorting through more cases per decision. For unindexed retrieval (not shown), the multicase has about 1% of the cost of individual cases, while the microcase remains the most costly of the three.

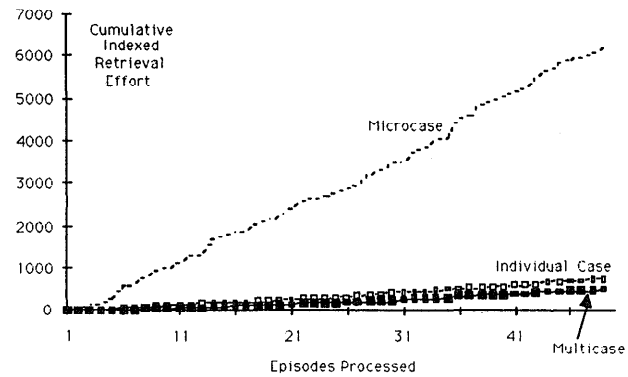

Figure 5: Indexed Retrieval Effort Comparisons

5 Concluding Remarks

The paper has presented a framework for comparing case-based procedure models over issues concerning case representation and organization. The framework is a formal model of procedures and their representation that enables us to characterize analytically several important properties of each method. In this paper we focused on storage cost and retrieval effort.

Three current proposals were analyzed within this framework: microcases, individual cases, and multicase. The formal analysis addressed the large scale behavior of the different models, a kind of worst-case analysis. Results of this analysis can be summarized: the multicase has the least storage requirements and individual cases are worse by an exponential factor. For retrieval cost, multicase and individual cases have comparable costs, with microcases a factor of n worse (where n is the depth of the procedure).

To evaluate the behavior in a more average case, empirical data was collected from a run of FLOABN in which over a thousand cases were collected. The results of this data can be summarized: The empirical data qualitatively confirms the the formal analysis with regard to storage and retrieval costs. We note, however, that the difference in indexed decision cost between individual cases and multicases, though of the expected polarity, was not as significant as the formal model leads us to expect. This is because the number of cases was not yet large enough for the proliferation of cases to dominate retrieval cost.

Both our formal analysis and our empirical data emphasize that representational choices in CBR systems have significant effects on system performance. We feel that this paper is the first step of a larger project in the exploration and characterization of efficient and useful case-base organizations, and an essential step in the development of truly large-scale CBR systems.

References

[1] Richard Alterman, Tamitha Carpenter, and Roland Zito-Wolf. An architecture for understanding in planning, action, and learning. *SIGART Bulletin*, 2(4):14–19, 1991. Special Issue on *Integrated Cognitive Architectures.*

[2] Richard Alterman, Roland Zito-Wolf, and Tamitha Carpenter. Interaction, comprehension, and instruction usage. *Journal of the Learning Sciences*, 1(4), 1991.

[3] Marc Goodman. A case-based, inductive architecture for natural language processing. In *AAAI Spring Symposium on Machine Learning of Natural Language and Ontology*, 1991.

[4] Kristian J. Hammond. Case-based planning: A framework for planning from experience. *Cognitive Science*, 14:385–443, 1990.

[5] Janet L. Kolodner. Reconstructive memory: A computer model. *Cognitive Science*, 7:281–328, 1983.

[6] Janet L. Kolodner and Robert L. Simpson. The MEDIATOR: Analysis of an early case-based problem solver. *Cognitive Science*, 13:507–549, 1989.

[7] Michael Lebowitz. Generalization from natural language text. *Cognitive Science*, 7:1–40, 1983.

[8] Robert McCartney. Reasoning directly from cases in a case-based planner. In *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*, pages 101–108, Hillsdale, NJ, 1990. Lawrence Erlbaum Associates.

[9] R. Schank and R. Abelson. *Scripts, Plans, Goals, and Understanding.* Lawrence Erlbaum Associates, Hillsdale, NJ, 1977.

[10] Roger Schank. *Dynamic Memory: A Theory of Reminding and Learning In Computers and People.* Cambridge University Press, Cambridge, 1982.

[11] Craig Stanfill and David Waltz. Toward memory-based reasoning. *Communications of the ACM*, 29(12):1213–1239, 1986.

[12] Roy M. Turner. A schema-based model of adaptive problem solving. Technical Report GIT-ICS-89/42, Georgia Institute of Technology, 1989.

[13] Roland Zito-Wolf. Case-based representations for procedural knowledge. Unpublished doctoral dissertation, 1993.

[14] Roland Zito-Wolf and Richard Alterman. Multicases: A case-based representation for procedural knowledge. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*, pages 331–336. Lawrence Erlbaum Associates, 1992.