A Case-Based Approach to Problem Formulation

L. Karl Branting
Department of Computer Science
University of Wyoming
Laramie, Wyoming 82071-3682
karl@eolus.uwyo.edu

Abstract

In domains requiring complex relational representations, simply expressing a new problem may be a complex, error-prone, and time-consuming task. This paper presents an approach to problem formulation, termed *case-based formulation* (CBF), that uses previous cases as a model and a guide for expressing new cases. By expressing new problems in terms of old, CBF can potentially increase the speed and accuracy of problem formulation, reduce the computational expense of retrieval, and determine the relevant similarities and differences between a new case and and the most similar old cases as a side-effect of expressing the new case. Three forms of CBF can be distinguished by the extent to which the retrieval and adaptation of previous cases are automated and the extent to which the facts of multiple cases can be combined. An initial implementation of one form of CBF is described and its ability to use previous cases to increase the efficiency and accuracy of new-case formalization is illustrated with a complex relational case.

The Task of Problem Formulation

Problem formulation, the expression of a problem in a representation amenable to manipulation by a computer, is an essential step in every form of automated problem solving. In systems that use featural representations of cases, problem formulation is typically quite straightforward. In MYCIN, for example, a new case is described by specifying values for parameters appearing in subgoals during a consultation. Similarly, a new case is represented in Protos as a vector of feature-value pairs. Problem formulation is easy in such systems because they use featural representations that have been engineered to represent only those aspects of cases known *a priori* to be relevant to the single specific task for which the systems were designed.

However, there is a growing recognition that featural representations are inadequate for a wide variety of applications. General-purpose knowledge bases intended to support a variety of different tasks clearly cannot use featural representation languages capable of expressing only case attributes relevant to a single task. Instead, such knowledge bases require relational representations that are capable of expressing a wide variety of relations among domain entities (Lenat and Feigenbaum, 1991; Porter et al., 1988). Moreover, featural representations

> cannot be used in areas requiring essentially relational knowledge representations. These areas include temporal reasoning, scheduling, planning, qualitative reasoning, natural language and spatial reasoning. Problems also occur in other areas involving arbitrarily complex structural relationships such as prediction of protein folding and DNA gene mapping (Muggleton, 1991).

However, relational representations are typically much more complex than featural representations because relational information implicit in the latter is made explicit in the former. This complexity can drastically complicate the process of formulating new cases. For example, in the context of a qualitative simulation program such as QSIM, a case consists of a set of qualitative differential equations specifying the structure of a physical system (Kuipers, 1989). Qualitative differential equations are represented in a relational language capable of expressing qualitative constraints among domain variables. The price of the expressive power of this representation is that creating and debugging qualitative differential equations is a complex, lengthy, and error-prone process (Farquhar et al., 1990).

The practical consequences of the difficulty of problem formulation in relational representation languages are illustrated by GREBE, a legal reasoning system that integrated case-based with rule-based reasoning (Branting and Porter, 1991; Branting, 1991a). GREBE used a representation language in which arbitrary causal, temporal, and intensional relations could be stated explicitly. This representation contributed significantly to GREBE's performance: in a preliminary evaluation, GREBE's analysis of 18 worker's compensation hypotheticals was found to compare well with analyses by law students (Branting and Porter, 1991). However, the expressiveness of this representation came at a cost. Representations of GREBE's test cases consisted on average of 89 propositions, each of which had to be entered by hand. Entering cases of this size was a lengthy process—typically several orders of magnitude longer than GREBE's run-time—and the resulting representation often required considerable debugging. Moreover, it was the experience of the legal reasoning group at the University of Texas that different knowledge enterers often chose to represent identical facts differently, creating the danger of inconsistent analyses of equivalent cases. Limitations in problem formulation, rather than problem solving, prevented GREBE from being usable in any practical setting (Branting, 1991a).

This paper proposes an approach to problem formulation in which previous cases are used as a model and a guide for expressing new cases. The use of existing information as a model for expressing new information makes it possible to "use what we know to help us process what we receive" (Schank, 1982). I refer to this approach to problem formulation as *case-based formulation* (CBF).

The Elements of Case-Based Formulation

The fundamental assumption underlying CBF is that new situations can be efficiently formalized using previous cases as models. CBF exploits the phenomenon that useful knowledge seldom consists of isolated facts, but instead tends to consist of collections of related facts. A simple example is a frame. The object/slot/value triples constituting a frame can be viewed as a collection of propositions that are related because they all concern the same object. In the context of CBF, a case can be any fact-collection/abstract-description pair[1]. The fact collection is referred to as the *facts of the case*, and the abstract description is referred to as the *consequent* of the case.

Although various approaches to case-based formulation are possible, all share the following basic steps:

1. **Retrieval.** Fetch an appropriate previous case from memory. Let F be the facts of the previous case.
2. **Substitution.** Substitute the names of the entities to which the new facts apply for the entities in F.
3. **Adaptation.** Add any necessary and delete any superfluous facts from the resulting set of new facts.
4. **Match Refinement.** If additions or deletions make the facts of some other case more similar to the new facts than the current case, refine the match by fetching the more similar case and go to step 2.
5. Return the description of the new case together with a record of all substitutions, additions, and deletions, since these constitute the relevant similarities and differences between the new facts and the previous case.

Three different forms of case-based formalization can be distinguished by the extent to which the steps of retrieval, substitution, and match refinement are automated and the extent to which the facts of multiple cases can be combined.

Copy and Edit

The simplest form of case-based formulation is the *copy and edit* approach to knowledge entry used extensively in the development of the Cyc knowledge base (Guha and Lenat, 1990). In the copy and edit methodology, a frame is added to a knowledge base by finding a similar frame, copying it, and modifying the copy.

The copy and edit methodology can speed knowledge entry and tends to enforce representational consistency. However, it doesn't address the potentially difficult task of retrieving the most appropriate frame, and it requires that the correspondence between entities in the new and old frames be determined manually. Moreover, it provides no mechanism for reuse of groupings of related knowledge larger than individual frames.

Single-Case CBF

The shortcomings of copy and edit can be addressed by automating the retrieval and substitution steps. I will refer to case-based formulation in which retrieval and substitution are automated and in which only a single case at a time can be used as a model for the new facts as *single-case CBF*. Single-case CBF begins when the user asserts some small number of facts. These facts are used by the system as a cue or memory probe to retrieve the facts of the most similar case. The system determines the correspondence between the new entities and the entities in the case that leads to the best match and makes the appropriate substitutions. If the system detects that the addition or deletion of facts makes some other case more closely match the new facts, the more similar case is automatically substituted for the current case.

Multiple-Case CBF

A new collection of facts is often best represented as a combination of several cases. In the domain of law, for example, a new case may match portions of several different precedents more closely

[1]This use of the term "case" is somewhat broader than in traditional CBR usage, where the term usually refers either to reusable plans (Hammond, 1986) or designs (Goel et al., 1991; Sycara and Navinchandra, 1991) or to *exemplars* (Porter et al., 1990), distinguished points in an instance space.

than it matches all the facts of any single precedent (Branting, 1991b). Similarly, designs are often best modeled as combinations of portions of multiple previous designs (Goel et al., 1991; Sycara and Navinchandra, 1991). Moreover, a single fact in a new case may itself be the consequent of some other case. I refer to the extension of single-case CBF to permit multiple cases to be combined as *multiple-case CBF*.

An Implementation of Single-Case CBF

CBF1 is an initial implementation of single-case CBF that provides a set of utilities for creating, viewing, and manipulating relational representations. CBF1 has been tested with a small knowledge-base of vehicles and with GREBE's knowledge base of worker's compensation precedents and hypotheticals.

The algorithm of CBF1 is as follows:

GIVEN:

- A partial description D consisting of a collection of propositions $(Pred_1\ A_{11} \ldots A_{1m}) \ldots (Pred_n\ A_{n1} \ldots A_{nk})$
- Optionally, a goal $(Pred_g C_1 \ldots C_p)$.

DO:

1. **Retrieval.** Fetch the case, C, whose facts, $F = (Pred_1\ B_{11} \ldots B_{1m}) \ldots (Pred_i\ B_{i1} \ldots B_{ik})$, most closely match D using structural congruence (Winston, 1980; Branting, 1991a; Holyoak and Thagard, 1989) as a similarity metric (limiting the search to cases whose consequents have the same predicate, $Pred_g$, as the goal, if a goal has been specified). F will be the model for the new description.
2. **Substitution.** Let M be the structurally most consistent mapping from entities in F to entities in D, that is, the mapping that maximizes the number of corresponding relations. Let D be the result of replacing each entity A_{ij} in F with $M(A_{ij})$, creating a new variable name if M is not defined for A_{ij}.
3. **Adaptation.** Add any necessary and delete any superfluous facts from D.
4. **Match Refinement.** If additions or deletions make the facts of some other case C' more similar to the D than F, let F equal the facts of C' and go to step 2.
5. Return D, along with all substitutions, additions, and deletions, since these constitute the relevant similarities and differences between D and F, the facts of the most similar case C.

The behavior of CBF1 is illustrated by the following example in which CBF1 was used to represent a case from GREBE's domain.

One of the cases used to compare the performance of GREBE to that of law students in GREBE's evaluation (Branting, 1991a) concerned Stanley, the head of a surveying crew at a large construction site. Stanley performed some of his duties—making architectural charts—at home during hours he set himself. One day, after doing some work at home, Stanley was injured in an accident while driving to the construction site.

The manually-constructed representation of Stanley's case used in the evaluation of GREBE consisted of 51 tuples. This representation took a number of hours to construct, and it is likely that a different knowledge-enterer (or even the same knowledge enterer on a different occasion) would have represented the case somewhat differently. With CBF1, however, entry of the case is relatively simple and the resulting representation is consistent with the conventions of the cases that have already been entered.

Representing Stanley's case using CBF1 begins with the assertion of the basic facts that Stanley was employed by Tower Construction Company to direct a surveying crew:

```
(employee Stanley-employment
          Stanley)
(employer Stanley-employment
          tower-construction-company)
(had-duties Stanley-employment
            directing-surveying-crew)
```

CBF1 uses a user-specifiable retrieval technique to determine the cases that most closely match these facts.[2] The system retrieves three candidate cases, each an instance of an employment-related activity: the Vaughn case, the Brown case, and the Prototypical Work Case. For each of these cases, CBF1 displays:

- The mapping from the entities in the case to entities in the new case that leads to the best match.
- The facts of the case that are unmatched in the new case. These facts constitute default conclusions about the new case under the assumption that the case is used as a model.
- The proportion of facts of the case that are matched in the new case.

For example, the best mapping from the entities in the Prototypical Work Case includes the following:

```
typical-employee ⇒ Stanley
typical-employer ⇒ Tower-Construction-Co.
```

[2]Three techniques for retrieving cases represented relationally were empirically compared in (Branting, 1991a). For convenience, the simplest of these algorithms, retrieval by best-first incremental matching (RBIM), is used in this example. Various alternative approaches to retrieval of relationally represented cases are discussed in (Branting, 1990).

```
typical-work-activity ⇒ directing-surveying-crew
...
```

Fourteen defaults are associated with the match to the Prototypical Work Case, including the assumptions that Stanley had some work hours and received a salary, that Stanley's being at the construction site was a prerequisite for Stanley's directing the surveying crew, and that Tower Construction Company had some goal that was achieved by Stanley's directing the surveying crew. Variable names are gensymed for entities in the case for which no corresponding entities exist in the new case. For example, the default that Stanley's being at the construction site was a prerequisite for Stanley's directing the surveying crew is represented by the tuple:

```
(prerequisite-for being-at-place.1635
                  directing-the surveying-crew)
```

All of the defaults of the Prototypical Work Case are true of Stanley's case, whereas each of the other cases has defaults that are not true of Stanley's case. As a result, the user selects the Prototypical Work Case as the initial model.

The adaptation step consists of the user entering the distinguishing facts of Stanley's case that are not true in the Prototypical Work Case. Such facts include that Stanley had the additional duty of making architectural charts, that he performed this duty at home, that he set his own hours for making the charts, and that he traveled from home to the construction site:

```
(had-duties Stanley-employment
          making-architectural-charts)
(activity-occurring-there Stanley-home
                          making-architectural-charts)
(prerequisite-for Stanley-being-at-home
                  making-architectural-charts)
(determined-by Stanley-work-hours
               Stanley)
(destination traveling-to-the-construction-site
             construction-site)
(source traveling-to-the-construction-site
        Stanley-home)
```

CBF1 permits the user to specify a goal to constrain the matching process. In Stanley's case, we can specify the goal of determining whether Stanley's traveling to the construction site is an employment-related activity (this determines whether Stanley is entitled to worker's compensation for his injuries). This goal will constrain matches only to cases of employment-related activities. Moreover, the mappings between each such case and Stanley's case will be constrained to pair the employment-related activity in the case with Stanley's traveling to the construction site, and the employment relation in the case with Stanley's employment.

After the user asserts the additional facts and the goal, the system performs a match refinement step in which the cases that most closely match the description under construction are retrieved. The closest match is with the Meyer case, which involved a real estate broker who was injured while traveling between his home, where he performed part of his job duties, and his office. This match provides an additional 11 defaults (*e.g.*, that Stanley was the driver in traveling to the construction site). Two of these defaults are not true in Stanley's case (Meyer had an additional job duty). When the user deletes these two tuples, the representation of Stanley's case is complete.

The representation of Stanley's case produced by CBF1 is more concise than the manual representation—32 propositions as opposed to 51 propositions—because facts irrelevant to the goal (*i.e.*, facts not contributing to the match with the controlling precedent) were omitted. Of these 32 propositions, only 9 tuples—28%—had to be explicitly asserted. The other tuples were obtained as defaults from matches with the Prototypical Work Case and the Meyer case. CFB1 reduces the time required to represent Stanley's case from hours to minutes and insures that the resulting representation is consistent with previous cases. Moreover, no additional retrieval or matching is necessary to analyze Stanley's case. This is because the most relevant precedent, Meyer, has been found and the relevant similarities and differences between Meyer and Stanley's case determined by the process of formulating the case.

Integrating Problem Formulation with Problem Solving

CBF is an application of case-based reasoning to the task of problem formulation. In domains for which problem formulation is complex enough to impede system use, CBF can be the first part of a two-step process: (1) case-based formulation of the problem, followed by (2) applying the appropriate problem-solving method to the problem thus formulated.

However, there are many tasks, such as such as precedent-based legal reasoning and case-based heuristic classification, for which problem solving consists at least in part of determining the relevant similarities and differences between a new case and the most similar past cases. CBF can perform these tasks, in part or entirely, as a side-effect of problem formulation. Problem solving in GREBE, for example, consists of (1) determining the mapping from the most similar precedents of the concept at issue to the facts of a new case, and (2) using this information to construct one or more explanation structures. As discussed in the previous section, CBF performs the first of these steps in the very process of formulating the facts of a new case. Thus, for tasks amenable to case-based reasoning, CBF can integrate problem formulation with problem solving.

This integration is desirable because a rigid separation between problem formulation, retrieval, and case comparison can exacerbate the difficulty and computational expense of each of these steps. The previous section illustrated how interleaving case retrieval and comparison with problem formulation can improve the accuracy and efficiency of the latter. The converse relation holds as well: interleaving these steps makes case retrieval and comparison more tractable. The combinatorics of matching relational cases makes the computational expense of finding the structurally most consistent case in memory steeply increase with the complexity of the probe[3]. By using a small set of initial facts as a probe and then incrementally refining the initial match as facts are added or deleted, CBF can avoid the computational expense of using a complete case description as a probe.

Range of Applicability of CBF

CBF as Knowledge Acquisition.

Problem formulation is a form of *knowledge acquisition*, the process of extracting knowledge from non-computer sources and encoding that knowledge in a form that is usable by a computer for problem solving. CBF is not restricted to problem formulation, but is applicable to acquisition of any type of knowledge organized around collections of related facts that can be manipulated as wholes. Viewed as a knowledge-acquisition technique, CBF has the virtue of being interactive, of automatically insuring consistency with existing collections of related facts, and of potentially improving, rather than degrading, as the knowledge base expands and a larger set of models for new cases becomes available.

Multiple-Case CBF

Extending the applicability of CBF to domains in which problems are best described as compositions of multiple previous cases will require implementing multiple-case CBF. Two mechanisms are required for multiple-case CBF. First, combining cases at the same level of abstraction requires the ability to partition the description under construction, apply single-case CBF to the partitions, and combine the results. Second, combining cases at different levels of abstraction requires the ability to view a single fact in a description under construction as the consequent of a collection of facts at a lower level of abstraction. For example, a swamp boat could be represented by using a motor boat as a model at a high level of abstraction (*e.g.*, a motor boat consists of a hull, a rudder, an engine, etc.) but, at a lower level of abstraction, using an airplane engine as a model of the swamp boat's engine. Current research is directed toward developing these two mechanisms.

Limitations

The effectiveness of any case-based reasoning system depends upon the existence of a library of cases relevant to the task addressed by the system. As an application of case-based reasoning to the task of problem formulation, CBF depends on the existence of a library of cases that can serve as suitable models for the problems that the system will encounter. If there is little similarity between new problems and past problems, CBF can provide little assistance.

Although CBF's use of past cases as models can reduce the danger of inconsistent representations of patterns of case facts, the current implementation of CBF nevertheless presupposes a consistent vocabulary of representational primitives. For example, if the initial description of Stanley's Case had been

(had-job Stanley Stanley-employment)

CBF1 would have failed to find any relevant past case because had-job is not part of the vocabulary in which the past cases were described. CBF1 would be improved by some mechanism for detecting possible inconsistent uses of primitives while at the same time permitting new primitives to be added when necessary.

A second limitation of CBF1 is its rudimentary *knowledge presentation* (Musen, 1988), *i.e.*, the conceptual model presented to the user. CBF1's knowledge presentation consists simply of the tuples that constitute the facts of a case in GREBE's representation idiom. An iconic presentation or a subset of English would greatly improve interaction with CBF1.

Conclusion

Relational knowledge representations are necessary for general-purpose knowledge bases intended for multiple tasks and for any of a wide variety of individual tasks. However, the price of the increased expressiveness of relational representations is that they make the task of expressing new cases correspondingly more complex. In domains involving sufficiently complex cases, simply expressing the facts of a problem in a relational representation language can itself be a complex, error-prone, and time-consuming task. This paper has presented an approach to problem formulation that uses previous cases as a model and a guide for expressing new cases.

[3]While various approaches to retrieval of relationally represented cases have been proposed, *e.g.*, MAC/FAC (Gentner and Forbus, 1991), ARCS (Thagard et al., 1990), and MRSDL (Branting, 1992), no approach has been shown both to guarantee a high level of accuracy and to cost significantly less than exhaustive matching.

CBF has a number of potential benefits. As illustrated by the example of Stanley's case, CBF can reduce the time necessary to pose new problems because modifying an existing representation is often much simpler than creating a new representation *ab initio*. CBF can reduce the danger of representational inconsistency by reusing conventions for representing particular patterns of facts rather than requiring them to be recreated in every new case. Moreover, when new cases are expressed in terms of old, the relevant similarities and differences between new and old cases are determined *a fortiori* by the very process of formulating each new case. CBF can also simplify case retrieval. By using a small set of facts as a probe and then incrementally refining the initial match as facts are added or deleted, CBF can avoid the computational expense of using a complete case description as a probe.

Finally, psychological plausibility argues for CBF over a rigid division between problem formulation and problem solving. Previous experience is not merely the yardstick by which new experiences are measured, but is the very medium in which they are expressed.

References

Branting, L. K. (1990). Techniques for retrieval of structured cases. *Working Notes of the AAAI Spring Symposium on Case-Based Reasoning*, Palo Alto, California.

Branting, L. K. (1991a). *Integrating Rules and Precedents for Classification and Explanation: Automating Legal Analysis*. PhD thesis, University of Texas at Austin.

Branting, L. K. (1991b). Representing and reusing explanations of legal precedents. In *Proceedings of the Third International Conference on Artificial Intelligence and Law*, pages 145–154, Oxford, England.

Branting, L. K. (1992). A model of the role of expertise in analog retrieval. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*, Bloomington, Indiana.

Branting, L. K. and Porter, B. W. (1991). Rules and precedents as complementary warrants. In *Proceedings of Ninth National Conference on Artificial Intelligence*, Anaheim. AAAI Press/MIT Press.

Farquhar, A., Kuipers, B., Rickel, J., and Throop, D. (1990). QSIM: The program and its use. Technical report, Artificial Intelligence Laboratory, Department of Computer Sciences, University of Texas at Austin.

Gentner, D. and Forbus, K. (1991). MAC/FAC: A model of similiarity-based retrieval. In *Thirteenth Annual Conference of the Cognitive Science Society*, pages 504–509.

Goel, A., Kolodner, J., Pearce, M., and Billington, R. (1991). Towards a case-based tool for aiding conceptual design problem solving. In *Proceedings of the Third DARPA Case-Based Reasoning Workshop*, pages 109–120. Morgan Kaufmann.

Guha, R. V. and Lenat, D. B. (1990). Cyc: a midterm report. *AI Magazine*, 11(3):32–59.

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

Holyoak, K. and Thagard, P. (1989). Analogical mapping by constraint satisfaction. *Cognitive Science*, 13(3).

Kuipers, B. (1989). Qualitative reasoning: modeling and simulation with incomplete knowledge. *Automatica*, 25:571–585.

Lenat, D. B. and Feigenbaum, E. A. (1991). On the thresholds of knowledge. *Artificial Intelligence Journal*, 47(1–3).

Muggleton, S. (1991). Inductive logic programming. *New Generation Computing*, 8.

Musen, M. A. (1988). *Generation of Model-Based Knowledge-Acquisition Tools for Clinical-Trial Advice Systems*. PhD thesis, Stanford University.

Porter, B., Lester, J., Murray, K., Pittman, K., Souther, A., and Acker, L. (1988). AI research in the context of a multifunctional knowledge base project. Technical Report AI-TR8-88, Artificial Intelligence Laboratory, Department of Computer Sciences, University of Texas at Austin.

Porter, B. W., Bareiss, E. R., and Holte, R. C. (1990). Concept learning and heuristic classification in weak-theory domains. *Artificial Intelligence Journal*, 45(1–2).

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

Sycara, K. and Navinchandra, D. (1991). Influences: A thematic abstraction for creative use of multiple cases. In *Proceedings of the Third DARPA Case-Based Reasoning Workshop*, pages 133–144. Morgan Kaufmann.

Thagard, P., Holyoak, K., Nelson, G., and Gochfeld, D. (1990). Analog retrieval by constraint satisfaction. Technical Report CSL-Report 41, Princeton University.

Winston, P. H. (1980). Learning and reasoning by analogy. *Communications of the ACM*, 23(12).