Applied Artificial Intelligence, 18:735–760, 2004

ISSN: 0883-9514 print/1087-6545 online
DOI: 10.1080/08839510490496978

Taylor & Francis
Taylor & Francis Group

CASE-BASED REASONING WITH SUBJECTIVE INFLUENCE KNOWLEDGE

YORAM REICH and ADI KAPELIUK
Faculty of Engineering, Tel Aviv University,
Tel Aviv, Israel

Problems in domains that are highly dimensional, inhomogeneous, and context dependent are difficult to support by computational tools. If solutions to these problems must be devised based on little information that is highly subjective, the situation worsens. In this paper, we propose a new case-based reasoning (CBR) method for addressing such problems. The method is based on augmenting case descriptions with knowledge in the form of influence graphs. We use these influence graphs to cluster the space of problems. These clusters, in turn, are used to retrieve the most relevant cases given a new problem specified only by three to four attributes. We tested the system in a lab setting and found it very promising. The second contribution of the paper involves an analysis of the incorporation of knowledge in CBR. This analysis provides a basis for classifying future interactions between CBR and other knowledge sources.

Case-based reasoning (CBR) has become a method of choice for building many knowledge-based or decision-support systems. CBR is mostly suited to problems that are ill structured, whose solution involves subjective judgment. Instead of trying to formalize knowledge from various sources (including experts), decontextualize it, and apply it to all situations, CBR offers an alternative.

Starting from previous cases, CBR provides means to solve problems by constructing solutions from parts of previous solutions. In many cases, this saves significant effort in trying to "structure" a complex problem since all that is needed to solve problems is their input-output relationships. Given a new problem described by some input parameters, construct the output from the output parameters of some solutions. The solution process itself can remain a "black box." Early examples of such systems are (Kolodner 1993): Ask—user-directed exploration of stories and guidelines describing

Address correspondence to Yoram Reich, Faculty of Engineering, Tel Aviv University, Tel Aviv, 69978, Israel. E-mail: yoram@eng.tau.ac.il

a task and/or domain; Casey—explanation and diagnosis of heart failures; or Hypo—legal reasoning and trade secrets law.

The traditional application of CBR is concept learning in which a case described by a set of features is assigned to a particular class. For example, in medical diagnosis, the features are the symptoms and the class is the diagnosis. In concept learning, we can say metaphorically that the information supplied for solving the problem is considerably more than the information returned by the system (i.e., a single attribute). Nevertheless, there are problems for which solutions are formulated based on minimal data due to problem characteristics such as urgency, lack of information, or lack of knowledge, as in the case of decisions made in a trauma room. In situations that permit extended user interaction, conversational CBR could be used to extract more information about the problem through a guided dialogue (Aha et al. 2001).

In doing so, conversational CBR can address situations in which users lack further case information or domain knowledge. Even so, these applications are still considered concept learning.

In contrast, there are applications for which minimal data give rise to elaborate solution description. One example is design and an exemplar of this problem type is Bridger—a system for designing cable-stayed bridges (Reich 1993a; Reich and Fenves 1995). Bridger is capable of creating an elaborate set of design attributes from a minimal set of design specifications. A complete set of specifications consisting of nine attributes leads to determining a bridge configuration described by 30 attributes, its approximate analysis described by 15 attributes, and its the final evaluation by four attributes. This problem is considered *many-to-many* mapping compared to the *many-to-one* mapping in concept learning (or classification). Many-to-many mapping problems could be addressed by using clustering (Reich and Fenves 1991). Clustering is appealing also because it can cope with inhomogeneities in a domain and may improve the overall efficiency and even effectiveness of a system (e.g., in information retrieval) (van Rijsbergen 1979).

Bridger was an experiment to show how pure correlations between input and output case attributes could be used to create synthesis knowledge for design and how weak domain theory can augment CBR to generate redesign knowledge. The latter part was implemented as an extension of Protos (Porter et al. 1990), a system for heuristic classification. Heuristic classification differs from concept learning along three dimensions (Porter et al. 1990): (1) the classification has to be explained; (2) it needs to accommodate incomplete case descriptions; and (3) it should learn knowledge for identifying case features relevant to classification. These dimensions are very valuable for many real-world problems. In order to support them, Protos is used to acquire knowledge from an expert.

Additional knowledge about the problem and the cases, including intricate relations between them, provide better ground for problem solving (e.g., forming knowledge-oriented similarity measures, or filling the gaps between cases). There is an implicit assumption that follow reasonable training; the system will be competent enough to remove the expert from the loop. The problem of Protos is the difficulty of domain experts to use the system.

In contrast to many simple classification problems, the problem addressed in this paper has the following attributes:

1. It is based on a many-to-many mapping.
2. The domain is highly dimensional.
3. The problem domain is inhomogeneous and context dependent.
4. The solution is highly subjective.
5. Problem solving is based on few (e.g., three to four) characteristic case attributes.
6. No available decontextualized domain knowledge exists.
7. Future auditing or quality control may require explaining solutions.
8. Practitioners are the sole source of knowledge and they have little motivation to spend effort beyond their usual work. Any solution devised must fit naturally into their present work practice.
9. There is high cost to solution failure.

We are not familiar with studies that address all these problem attributes simultaneously; however, we do make use of ideas developed elsewhere for addressing them. The solution to the present problem involves two activities (see also Figure 5): clustering (supports the 1st–4th problem attributes) and elaborating minimal case description with additional knowledge (supports the 5th–7th problem attributes). The latter activity conflicts with the 8th attribute. This conflict creates a serious dilemma:

1. We could ignore any additional knowledge and risk failing the system due to poor performance.
2. We could incorporate complex knowledge elicitation and risk lack of cooperation from practitioners.

Our solution avoids both extremes by utilizing a new way of representing simple form of influence knowledge and using it to build a decision support system based on CBR. We show that our solution also satisfies the 8th attribute. Therefore, with two activities, we can, in principle, address all these problem attributes. Finally, the last problem attribute mandates that we obtain good quality solutions.

BACKGROUND

CBR Process

Most references on CBR present a typical CBR process composed of several steps (e.g., Aamodt and Plaza 1994; Kolodner 1993). Figure 1 depicts our version that includes seven steps:

1. **Building a database of past cases, i.e., case memory.** The database contains n cases $C = \{C_1, C_2, \ldots, C_n\}$ and access and retrieval procedures. Each case C_i is represented by the tuple (P_i, S_i, E_i), where,
   a. $P_i = (p_{i1}, \ldots, p_{ik_i})$ is the case problem definition attributes; k_i is not constant across cases and P_i can have empty attributes.
   b. $S_i = (s_{i1}, \ldots, s_{ij_i})$ is the case solution attributes; j_i is not constant and S_i can have empty attributes. Since, in general, $j_i \gg 1$, the problem is based on a many-to-many mapping (1st aforementioned problem attribute).
   c. The features in P_i and S_i are assigned values that could be numbers, text, fuzzy values, or symbols. Interestingly, the solution could be *indifferent* to the attribute values and based only on the attribute presence in case descriptions.
   d. $E_i = (e_{i1}, \ldots, e_{il_i})$ is knowledge or information about case C_i, l_i is not constant. Its form and content have to be specified for each problem. E_i is optional and does not appear in conventional "black-box" CBR.
   e. In general k_i, j_i, and l_i are both much greater than 1; therefore, the problem is highly dimensional (2nd attribute).

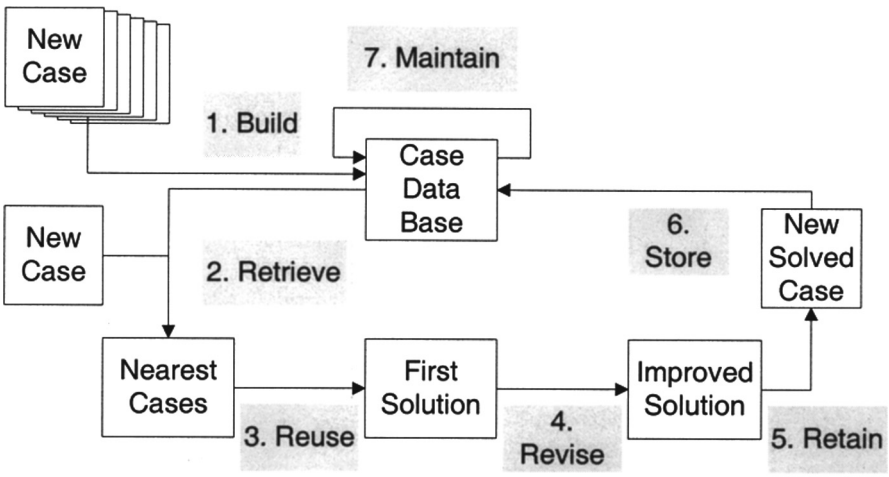

FIGURE 1. CBR process.

f. Often, only a small number a of the k_i problem definition attributes is used for solving the case (5th attribute).

Representing a case with the tuple (P_i, S_i, E_i) is different than common CBR practice. The common representation is a pair (p,s) where $p \in P$ is the language for describing the problem and $s \in S$ is the language for describing the solution (Bergmann 2002). While the latter can represent a connection between the problem description and the solution parts by making P and S overlap $(P \cap S \neq \phi)$, our formal representation is more precise and general.

When a new case C_{n+1} arrives at time $t+1$, practitioners need to find a qualified solution S_{n+1} for C_{n+1} using the P_i's. There is no domain knowledge available for executing this task (6th attribute) and, given the context dependency and inhomogeneity of the domain (3rd attribute), the solution is highly subjective (4th attribute). The consequences of exercising a solution are significant (9th attribute); therefore, in spite of the solution subjectivity, practitioners might be required to explain their rationale for solving a particular problem (7th attribute). At solution time, practitioners have little time or no motivation to seriously record their rationale (8th attribute).

2. **Case retrieval.** The system locates several cases that are similar to the given case.
3. **Case reuse and adoption.** The user or the system can replace parts of an old solution (requires substitution methods), transform/add some part of an old solution (requires transformation methods), use repair heuristic methods on old cases, or adopt the process of past solutions (derivational replay methods). The knowledge and information must be catalogued and indexed so that they will function in quick and effective way. This step adds S_{n+1} to case C_{n+1}.
4. **Case revision.** The system checks the new solution S_{n+1}, and makes changes to the solution if needed. During this step, the additional knowledge E_{n+1} is created and added to the case.
5. **Case retaining**. A decision is made whether the new case and its new solution should be added into case memory. If so, the new case and its knowledge are prepared for such inclusion.
6. **Case storage.** The new case is incorporated into the database in an appropriate place, including generating the necessary retrieval information.
7. **Maintenance.** Each additional case or a batch of cases is a trigger for checking the quality of the case-base. Maintenance could involve restructuring the database, modifying various parameters, as well as removing cases.

Each of these tasks must be performed *efficiently* and output *good quality results* (i.e., be effective). In *efficiency*, we can include the measurements of

retrieval, comparison, and database updating speed, and memory resources needed. We also include in efficiency, the effort spent by the user on working with the system. In *quality*, we include the provision of good assistance to practitioners as perceived by them, as well as improved case results following the implementation of system recommendations in reality.

Representing Knowledge in Cases

In general, the more knowledge we have about a problem domain, the better would be our solutions to problems; the exception include situations in which knowledge is considered harmful (Markovitch and Scott 1988), when unfiltered learned knowledge degrades problem solving performance (Miyashita and Sycara 1995). In CBR, there have been numerous attempts to incorporate knowledge in case representation, including the creation of a language that combines cases and knowledge (Manago et al. 1994). However, these studies differ in the role assigned to knowledge in the CBR process, i.e., whether and how they address questions such as: how knowledge is represented (inside, outside case), is it collected from experts or given by system developer, how is knowledge maintained, is it general or problem specific, etc.? Note that even defining a case structure (attributes and solutions), which is central to the success of CBR, requires exercising significant knowledge about the problem with the help of domain experts, but this is true of all problem solving.

Typology of Combining Knowledge in CBR

Knowledge can be combined with CBR in many ways. The space of possible combinations can be organized along several dimensions. First, knowledge in various problem-solving (PS) methods can be used concurrently or sequentially with CBR to add robustness to the overall system. This can be called horizontal integration (Aamodt 1994), see Figure 2. A vertical integration involves incorporating different knowledge sources (KS) that augment or modify the traditional CBR steps. Like Aamodt (1994), we focus on vertical integration.

There are four main approaches for vertically combining knowledge in CBR. The first and most common is augmenting a regular case-based reasoning system with an external knowledge-base system (KBS) for conducting various functions such as case organization and maintenance (e.g., Bergman et al. 1996). This approach can also describe the horizontal integration.

The second approach adds knowledge, which is context dependent, to each case. It can also utilize external KBS as well. In this approach, it could be subtle to ascertain whether a piece of information is knowledge or output. The classification depends on the *role this information plays in the CBR process*. If it modifies or augments the classical similarity-based functioning then we consider it as knowledge, otherwise, we call it output. An example of

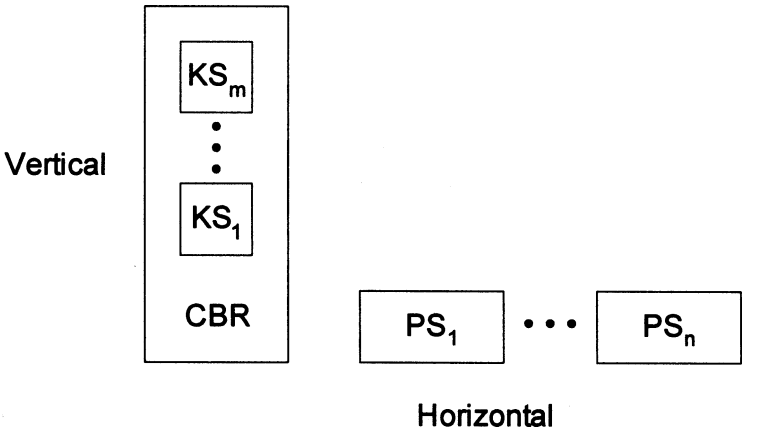

FIGURE 2. Integration of knowledge in CBR.

this elusive nature of information is found in Carbonell's (1986) work on transformational and derivational analogy. Transformational analogy treats solution paths as the output part of a case that transfers to a new similar situation and is replayed to get a result. Therefore, it might not be considered knowledge in the sense of this paper.

The third approach employs structured cases with or without knowledge. The structure is perceived as a model of domain knowledge. As in the previous approach, we have to distinguish between two types of structured representations. For example, case representation as a graph (e.g., as in FABEL) (Gebhardt et al. 1997) is a structured representation but is not necessarily considered knowledge (FABEL, nevertheless, has knowledge sources of the first approach that can be used to complete or adapt retrieved cases). In contrast, a structured connection between input and output used to augment classical CBR mechanisms is considered knowledge since it provides explicit connection between input and output that is otherwise made implicit by the CBR system. This approach is used in our study.

Graph representation of cases that does not include the kind of knowledge we discussed was used in the past in projects such as architectural topology (FABEL) or course timetabling (Burke et al. 2001). The importance of these studies to our discussion is in the retrieval mechanisms they developed for structured representations that could be used also when the knowledge is structured (e.g., graph edit operations) (Messmer and Bunke 1998).

In problems where structured case representation could be an arbitrary graph, the retrieval process could lead to computational difficulties (Gebhardt 1997). This situation could be ameliorated by trading off some of the computations with memory organization (Messmer and Bunke 1998). In our case, the graph has a fixed structure and its content is very simple, making it easy to handle, as discussed in the next section.

The fourth approach uses cases that are connected by a web of relations. The cases themselves could be structured or not, and this approach could be used with or without external knowledge. Protos is an example of this type where knowledge in the form of heuristic links (e.g., reminding) connects cases (Porter et al. 1990). The knowledge is used for structuring the database, for retrieval (spread of activation), and for maintenance (i.e., activations leading to a decision to store a new case due to lack of knowledge). Another approach that resembles Protos is case retrieval nets (Lenz and Burkhard 1996), which is a directed graph with nodes representing cases and information elements and links denoting the degree to which they influence each other. The graph representation is less elaborate than Protos, but the simplicity allows for greater applicability.

We now provide examples of using knowledge to modify or enhance the steps in the CBR process. This is an orthogonal dimensional to the four listed approached and an elaboration of the vertical integration. The list of examples is only illustrative and certainly not exhaustive.

Knowledge Used to Build Database—CBR Step 1

This and the next class differ from the rest in that the knowledge part takes an active role in the core CBR process: organization and retrieval of cases. Protos was perhaps the first CBR system to use domain knowledge to build the database. During knowledge acquisition, Protos tried to classify a new case and if it was wrong, the expert would provide an explanation that would finally lead to the correct classification, including the addition of various links that tell why the new case is an exemplar of one class and not the other. In our context, Protos knowledge representation is very appealing due to its weak nature; however, we are more interested in simpler forms of knowledge since complicated knowledge is difficult to extract, especially when knowledge is collected from field practitioners during their work. In fact, our experience suggests that an attempt to elicit more knowledge about cases that extend beyond the actual problem solution is doomed to failure in some domains.

Knowledge Used to Retrieve Cases—CBR Step 2

Contribution to this process requires a mechanism for calculating similarity between knowledge elements or using knowledge to enhance similarity calculation. Protos used rules to augment case descriptions (Porter et al. 1990) and Bergmann et al. (1996) used knowledge to complete case descriptions that in turn can be used to support similarity calculations. Smyth and Keane (1998) use knowledge in the retrieval stage to improve retrieval accuracy, flexibility, and greater overall problem solving efficacy. They address the problem that similarity does not always reflect utility of a case for solving another problem. Their goal is to retrieve the best or easily adaptable cases.

Therefore, a case is considered similar to a given problem if there is adaptation knowledge that suggests that it could be easily adapted.

Knowledge Used for Case Adaptation—CBR Step 3

Bergmann et al. (1996) also used external knowledge in the form of adaptation rules to reuse an old case to solve a new case. As already mentioned, Smyth and Keane (1996) provide a direct link between retrieval similarity and adaptation needs. The technique creates external adaptation knowledge that during retrieval facilitates the computation of a precise measure of a case's adaptation requirements. Bridger encoded adaptation knowledge in the form of proportions between bridge elements. This knowledge was also learned from design cases. When an old bridge was retrieved to address a new specification, its adaptation proportions were retrieved as well and used to adapt its dimensions (Reich 1993a; Reich 1993b; Reich and Fenves 1995).

Knowledge Used for Case Revision—CBR Step 4

In our model, at this stage, the additional internal knowledge E_{n+1} is created and added to the case. This knowledge can be considered as an explanation of why the solution S_i solves P_i. Nevertheless, it is different than systems that construct explanations (Aamodt 1994) or systems where the user provides explanations (Porter et al. 1990). In our model, the knowledge is very simple and structured in order to add little overhead to users and to enable efficient manipulation.

An example that differs with this study is Bridger. Following a bridge adaptation to suite a new specification, Bridger performed strength analysis according to American design codes. Bridges that failed the test could be modified using an interactive redesign system based on an enhanced version of Protos (Reich 1993a; Reich and Fenves 1995). This therefore is an interesting vertical integration where one CBR system serves to enhance one CBR step of another CBR system.

Knowledge Used for Case Retaining—CBR Step 5

A decision on whether the case should be kept depends on whether it adds to the overall system's quality, e.g., its ability to make predictions, increase coverage, or prevent the system's failure. As already noted, knowledge is not always useful (Markovitch and Scott 1988; Miyashita and Sycara 1995). In Protos, case retaining is done simply if the present case structure cannot classify the new case correctly. In this situation, the case is added to the case structure in the appropriate place. In addition and as already mentioned, an explanation is also added that justifies the user classification and the case retaining.

A different study discussed not simple case retaining but a recommendation of a case that could best improve the present system performance

(McSherry 2000). The case-authoring tool, CaseMaker-2 recommends new cases for addition to a case library based on their coverage or competence contribution. The system evaluates the space of uncovered cases using a procedure that could employ domain knowledge.

Knowledge Used for Case Storage—CBR Step 6

Usually, case storage uses various algorithms for indexing cases for efficient retrieval. An example of using knowledge for this step is Protos. While retaining a case, explanations are entered and linked to reminders. This modifies the future retrieval behavior of the system.

Knowledge Used for Case Maintenance—CBR Step 7

Most knowledge-related work in this step falls under the first type: use of external KBS. Reinartz et al. (2001) adds two maintenance steps to the CBR cycle. The review step covers assessment and monitoring of the case database, whereas the restore step actually modifies the case database according to recommendations resulting from the review step. By defining several problem-dependent quality measures, the user monitors the case database in order to improve its quality during maintenance. In our model, by clustering the case-base according to influence graphs, we can maintain the cases better since similar cases are gathered together.

Problem Definition

We can summarize the foregoing discussion as a problem definition: Create a CBR system based on the outlined framework that addresses the nine problem attributes discussed in the introduction. As seen in the previous subsection representing knowledge in cases has a significant potential to improve the execution of the different CBR steps. The difficulty remains in deciding where to employ knowledge and how to make it easily usable to field practitioners.

SOLUTION

Our solution has two main stages. Stage I organizes the memory cases by clustering the cases according to an influence graph representation and characterizing these clusters. Stage II solves a new case by retrieving the best matching clusters and nearest neighbor cases from these clusters. Subsequently, the solutions of these cases are used to solve the new case.

Case Elaboration with Knowledge

We propose a solution that represents a case with an *influence graph*. As shown in Figure 3, an influence graph defines the inputs (or case problem definition part, P_i) and the output (or case solution part, S_i) of a case with lists

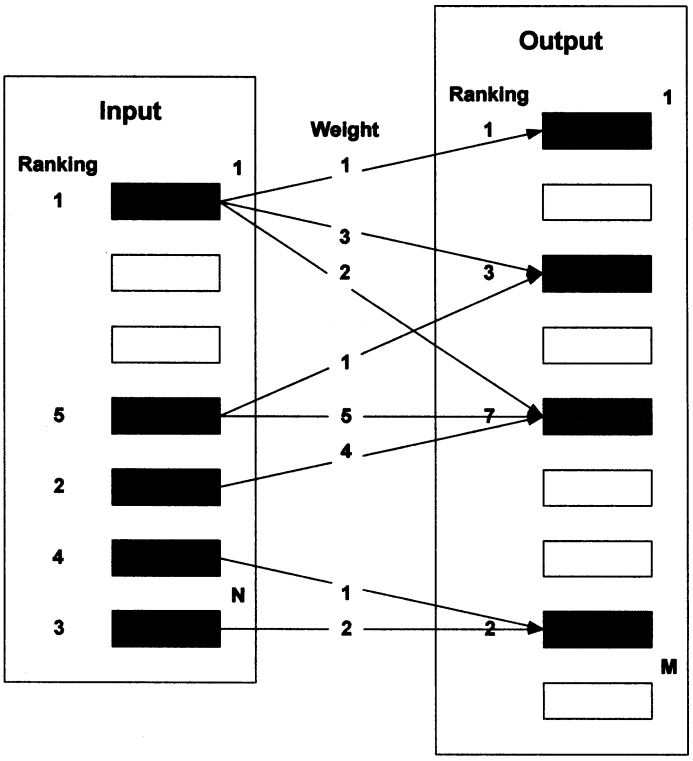

FIGURE 3. Influence graph representation of a case.

of attributes, similar to the classic case representation. In addition, the influence graph depicts, for each selected output, the input attributes that led to its selection. For example, output 1 was selected due to input 1. Each participating input, output, and influence can have an associated numeric value that defines its strength. The influence graph can be easily represented with a matrix as shown in Figure 4. The edges of the influence graph constitute a very simple form of knowledge $E_i \subseteq P_i \times S_i$. This knowledge is *indifferent to the values of the features or solution parts.* As such, it is simple to elicit.

The use of CBR with a database of cases represented with influence graphs is described in Table 1. The description is rather a framework that could be implemented in many ways. We first discuss the framework and subsequently discuss our implementation in detail. The framework consists of two stages: memory organization and solution. In the memory organization stage, cases are clustered and characterized. These activities could be realized in many ways. For example, through the use of ECOBWEB, which performs hierarchical clustering with cluster characterization (Fisher et al. 1993; Reich 1993a). Clustering with characterization can also be done with traditional clustering techniques such as agglomerative hierarchical clustering

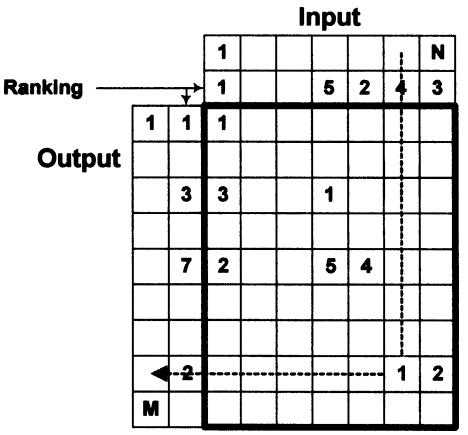

FIGURE 4. Matrix representation of a case.

(Everitt 1993; Jain et al. 1999) or fuzzy clustering (Höppner et al. 1999). Since most clustering methods do not provide cluster characterization, this part could be done in a separate step either by supervised cluster characterization (Reich and Fenves 1991) or manually.

In the solution stage, a new case is described only by its input. The decision maker identifies a few inputs as central to the case. These inputs are used to retrieve a cluster by matching them to the clusters' descriptions. Subsequently, few cases are retrieved from the selected cluster. The decision

TABLE 1 Solution Algorithm

Stage I: Memory organization

Given: n cases represented by an influence graph
1. Cluster cases (using E) into g clusters, $G = \{G_1, \ldots, G_g\}$ (g is not specified *a priori*).
2. Characterize clusters.

Stage II: Solution

Given: a new case C_{n+1} described by a set of inputs P_{n+1} and a set of g clusters, $G = \{G_1, \ldots, G_g\}$:
1. Identify $a \ll k_{n+1}$ key inputs.
2. Retrieve b best matching clusters $G_{best} \subseteq G = \{G_1, \ldots, G_g\}$, $b \leq g$..
3. Retrieve c nearest neighbors from members of G_{best}.
4. Solve C_{n+1} using the outputs S_i of the c cases.
5. Complete E_{n+1}.

Remarks:

a may be as low as three.
b would be one or two in most cases.
c should be manageable, e.g., about three from each cluster. If b is larger than two, c should be adjusted accordingly.

maker synthesizes a solution to the new problem from the solutions of the cases retrieved.

There is one feature of our solution that is worth emphasizing. As we show later, the solution works without considering attribute values in the memory organization or the retrieval steps. The only influencing factors are the relations between attributes and their inclusion in, or absence from, case descriptions. This makes the solution different from the three types of CBR representations listed by Bergmann (2002), which are textual, conversational, and structural. In our application, cases have a structured representation of attributes and values that are a mix of different types, including numbers, symbols, and text. In addition, this application can accept a conversational solution as well. Thus, the solution can be best characterized as a hybrid approach. We now elaborate on the details of each stage of our solution.

Memory Organization

Clustering Cases

This activity subdivides the database into a set of mutually exclusive classes. It could equally create fuzzy clustering or other forms of overlapping clusters (Dash and Liu 2001; Höppner et al. 1999; Tao 2002). The basis of clustering is unique to our system. Whenever cases are clustered (e.g. Cheng et al. 1997; Reich 1993a; Yang and Wu 2001), the classical case descriptions (i.e., P_i and S_i) are used by the clustering algorithm. In the application we describe later, we found that such clustering was insufficient: The clusters were not homogeneous thus ineffective. In an attempt to enrich the representation, we devised the influences graph concept E_i and used it to cluster the cases. We assumed that the relations between the problem input and output, made explicit in this representation, provide better ground for considering cases as similar and that they would let clustering capture more of the similarity between the cases. This assumption was confirmed in the results of the clustering and in subsequent testing.

Central to any clustering algorithm is a measure of distance between the objects being clustered. In our case, the objects are the influences graphs represented by matrices. The distance between matrix M_i and matrix M_j was defined in two ways. One was measuring the distance directly using graph edit operations (see Eq. (1)) and the second was measuring the distance through similarity based on maximal common part (see Eq. (2)). Measuring the distance directly is done by calculating the number of steps needed in order to get from one matrix to the other. A step can be writing 1 instead of 0 or vice versa. This method is based on simplified graph edit operations as a means for measuring the distance between graphs (Messmer and Bunke 1998). Since the method input is binary (0,1), in order to find the distance between two matrices, we can subtract them and sum

all absolute values. The result is normalized by the sum of "1" factors in both matrices:

$$\text{Distance} = \frac{\sum_{\text{non-zero matrix elements}} |E_i - E_j|}{\sum_{\text{non-zero matrix elements}} |E_i| + \sum_{\text{non-zero matrix elements}} |E_i|}; \quad (1)$$

where

E_i, E_j are two knowledge matrices,

$|E|$ is the matrix whose values are absolute values of E.

In this calculation, identical matrices will result in 0 distance and matrices that share no factor (that is, differing in all the places having 1), will result in the maximal distance 1. A single overlap of 1 placed in both matrices will result in a distance smaller than 1.

Calculation according to similarity will detect the similar factors common to both matrices and divide by the maximal number of factors in either matrix (Bunke and Shearer 1998):

$$\text{Similarity} = 1 - \frac{\max common(E_i, E_j)}{\max matrix(E_i, E_j)}, \quad (2)$$

where

"max *common*" is the count of non-zero elements equal to matrix E_i and E_j, and

"max *matrix*" is the maximal non-zero elements of matrix E_i or E_j.

The distance is 1 minus the similarity. Using this calculation, we will also find the matrices with identical factors to have 0 distance (maximal proximity) and in matrices differing altogether to have distance 1 (maximal). Single correspondence will suffice to yield a distance smaller than 1. Bunke (1997) defined a particular cost function for graph edit distance and showed that, under this cost function, graph edit distance computation is equivalent to the maximum common subgraph computation. In out case, the graph edit calculation is different, therefore, the two methods yield different results.

We examined several possible clusterings by the use of different clustering methods. These methods yielded very similar results. Consequently, we decided to use a classical agglomerative method that builds up a hierarchical cluster tree called dendrogram representing the proximity between the cases (Jain et al. 1999). The similar cases are placed closer and connected at the lower branches and the more distant cases are connected at the upper levels. Crossing the tree at a certain level will create branches, each of which represents a cluster.

Characterize Clusters

Following clustering, each cluster can be given a concise characterization. Cluster characterization could be performed manually by field experts or

automatically using supervised learning programs such as WEKA (Witten and Eibe 1999). The choice depends on the size and type of case database and clusters. Combining methods can lead to better understanding and characterization of clusters by providing different perspectives of the data (Ali et al. 1994; Reich et al. 1996). In our implementation, due to the size of the case database and the large number of attributes defining a case, we employed manual characterization by a field expert and examined the ability to use supervised learning with WEKA.

Solution Process

Retrieve Best Matching Clusters* G *Based on Decision-maker* a *Key Inputs

The first step performed by field experts when they start handling a new case is identifying $a \ll k_{n+1}$ inputs as the most important attributes defining the problem. Subsequently, few clusters (typically one) that match these inputs are selected. This can be done manually by inspecting the cluster characterization, or automatically using decision tree/rules created by some supervised learning programs. In our tests, we tested both methods for retrieving the best matching clusters, G, based on three key inputs. While the main method was manual, the automated method turned to be very accurate. Lack of cases prevented using the automated method as the primary cluster retrieval mechanism.

Retrieve* c *Nearest Neighbors from Clusters* G *Considering the* a *Key Inputs

In order to retrieve the most compelling cases from clusters G, using the knowledge matrix, $E = |e_{ji}|$, we calculate for each case in the clusters the score in Eq. (3) and pick the c cases with the higher score. The score simply points to those cases in which the key inputs were most influential on the solution

$$score = \sum_{i \epsilon a} \sum_{j=1}^{M} e_{ji} \tag{3}$$

Solve* p *Using the Outputs of the* c *Cases

This stage can be done manually by field experts that examine the solutions retrieved and reuse them. Alternatively, this reuse could be done automatically by using various CBR reuse strategies. In our implementation, we let the experts reuse the solutions manually and were interested in assessing their effort in this activity. This effort was measured by the assessment of the effort they spent in solving the case beyond the available solution, and the amount of changes (adding or reducing actions) they had to perform to arrive at a satisfactory solution.

In summary, the application of the improved CBR method to our problem consists of the following stages:

1. The decision maker receives a new case and identifies the prominent characteristics of the case.
2. The decision maker chooses the most appropriate clusters for this case (automated execution is also functioning).
3. Similar cases are retrieved out of the selected clusters.
4. The decision maker determines the best solution, after consulting the solution part of the cases retrieved.
5. While solving the case, new information is added to the case, including the knowledge part.
6. After examining and approving the solution, the case in added as a new case to the database.

EXAMPLE APPLICATION: ATTENDANCE OFFICERS

The Problem

Close to 9% of the students in Israel (Israel's Central Bureau of Statistics 1999) drop out of school. This figure is close to the 10.9% U.S. dropout rate (U.S. Department of Education 2001). School dropout can be seen as a systemic problem with both human and social aspects. Reduction in education budgets lead to an unsatisfactory treatment of the problem, thus there has been no significant change in these figures throughout the years. Moreover, the present economic situation may even cause the situation to worsen.

Attendance officers try to investigate the reasons students drop out and then try to prevent it. Attendance officers monitor the parents and the education system in order to enforce the education law. Consequently, attendance officers are linked to many functions in the community and they try to work together to enforce the law and realize students' learning potential. Attendance officers find themselves dealing with many different cases in a situation with limited resources. Since there is no system that accumulates knowledge when attendance officers work alone, they have little assistance, and moreover, creative solutions adopted by one attendance officer are not shared with others.

The Work of the Attendance Officer Today

The attendance officer operates as part of the municipal educational department. Large cities have a number of attendance officers working together. Small cities may have only one officer. The attendance officer is

familiar with the welfare and education services in the area and, based on her experience and personal skills, tries to confront each problem with the best solution. She acts to execute the plan and follows it as it is being executed. She may also report the problem to other authorities, if necessary.

Regional meetings take place every month. These four-hours-long meetings give officers at a certain region a chance to exchange experiences; they are usually arranged as a short workshop aimed at providing the attendance officers with a more holistic view of the issues they are dealing with.

Monthly reports are sent by every attendance officer to the state's Bureau of Education. These reports include descriptions of the cases being handled and the progress of each case. The reports have an administrative purpose and are seldom reviewed for generating feedback.

Building an Interface to Fit the Present Operation of Attendance Officers

Figure 5 shows how we tried to deal with the aforementioned work context. The figure captures how the initial limited understanding led to the problem attributes (the nine mentioned in the introduction are organized slightly differently). These led to intermediate conclusions such as the need to address heterogeneous space, multiple sources of information, and the use of simple accessible tools. These and other concerns led to choosing CBR as the main technology with clustering the space and augmenting case descriptions with case influence knowledge. We now elaborate only on the generation of case representation.

Following an analysis stage in which attendance officers participated in various ways, including taking part in the research, we divided the details of each case into three parts as we detailed in the previous section:

1. Case characteristics.
2. Solutions/modes of action.
3. Professional knowledge about the case.

In the definition of the case characteristics, we used the existing monthly attendance officer report sent to the Bureau of Education, plus additional characteristics, which are not discussed here. After several meetings and consultations and using majority voting, we arrived at 24 characteristics. We expect that while developing similar systems, the process of defining the characteristics will be very time consuming. This definition would be subjective, and in general, 100% agreement cannot be reached.

The characteristics contain empty fields that should be filled with objective data, e.g., age, birthplace, and present educational institution. Other

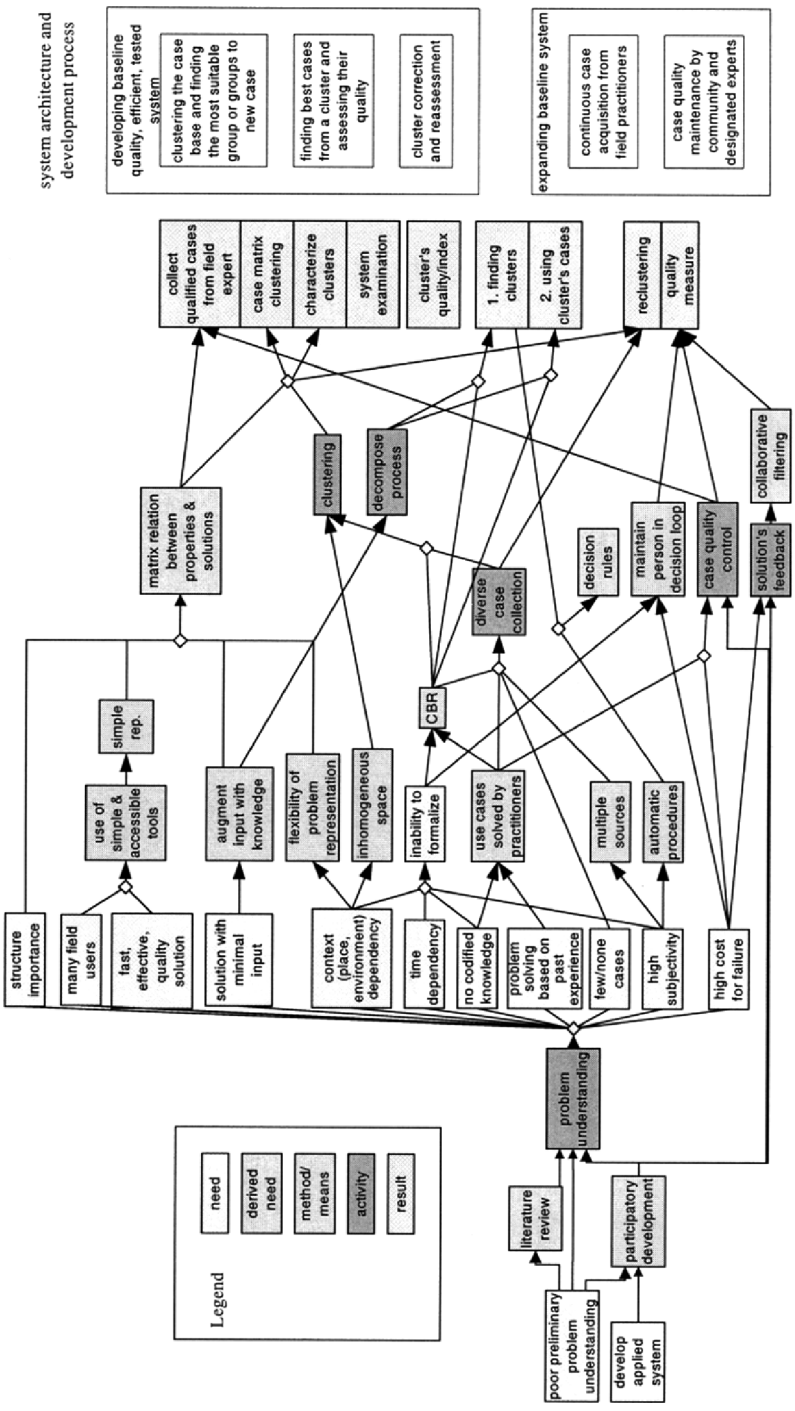

FIGURE 5. Rationale of DSS for school dropout prevention.

fields include "yes/no" fields such as: probation officer, school dropout, emotional problems, prior reassignment committee, and domestic violence. A third type of characteristic requires filling by fix values. These fields include: residential area, family background, educational performance, school attendance, psychological diagnosis, behavior, abnormalities, and decisions made by previous reassignment committees. Other fields may be filled freely and contain unique data describing the child such as distinguished talent, e.g., sport, computers, music, pets, reason for submission, and degree of parent/child collaboration. It seems that these characteristics give a comprehensive description of the child's condition in the education system.

Following discussions in several meetings, and again, applying majority vote, we arrived at 18 possible modes of actions. As in the definition of problem characteristics, defining the solutions is a very time consuming process. Since the definition is subjective, in general, it is impossible to arrive at 100% consent.

Here, too, some of the data are fixed (submission to regular/special education institution), some of the fields are "yes/no" fields (legal suit), some will be filled with fixed values, and some will be creative solutions applied to the specific child. For example, consider the following actions: detection, forming an interdisciplinary team, reassignment committee, treatment given by welfare authorities, treatment plan, tutoring, job assignment, parental guidance, reinforcing, boarding school, psychological therapy, school discontinuation, and psychiatric care.

The knowledge component is the most essential and major aspect of the research. When solving a case by selecting a set of actions, the attendance officer explains, based on her professional experience, which characteristics led her to choose a certain solution. Knowledge as a concept to be recorded was a novel concept for the attendance officers. Consequently, we introduced them to this subject through a short presentation that included basic terms used in knowledge management as a life-long process (Reich 1999). Knowledge management imposes conflicting demands on our project. On the one hand, the attendance officer should not be overloaded with additional work that is not performed in addressing the primary goal of solving the problem. Therefore, knowledge coding should be simple and matching as much as possible the regular line of work. On the other hand, a simple structured knowledge may be insufficient for the appropriate description of the case, making the whole endeavor futile.

The coding method presented earlier matches these conflicting demands, enabling the creation of valuable knowledge for the attendance officers themselves, as will be explained. After describing the modes of action, the attendance officer depicts the characteristics that influenced choosing the specific mode of action. The attendance officers were asked to mark, in a table of 18 lines by 24 columns, similar to Figure 4. Each line represents a solution (1...18) and each column represents a characteristic (1...24).

Summing the X's marked in a column indicated the characteristics influencing the solutions most dramatically.

It is evident that such a knowledge structure postulates several presuppositions:

1. Any pair of characteristics and any pair of solutions are independent. That is, if two characteristics together are needed for one solution, we are unaware of that.
2. There is no limit to the number of characteristics that may influence a certain solution or solutions. That is, hypothetically one may mark all the characteristics as influencing the choice of the solution.

The characteristics marked as dominant when the cases were first obtained (three characteristics chosen in the beginning of the process) did not always correlate with the characteristics marked as influential in the knowledge matrix. There is no doubt that attendance officers, while reexamining their modes of action as influenced by the case characteristics and marking the knowledge in a structured manner, improved their understanding of the case and their solutions. All attendance officers who filled out the forms asserted that the actual knowledge definition was helpful in arriving at a better solution and understanding the processes underlying these cases.

In this project, we collected 190 forms filled out by four attendance officers. An initial examination of the matrices *indicate quite distinctly a treatment pattern and a certain perspective*. By combining all the matrices filled out by a certain attendance officer, one could establish the solutions she usually applies and characteristics she focuses on when trying to decide on the desirable mode of action.

Knowledge Utilization

The data was processed in three stages:

1. In the first stage, the cases were clustered according to the knowledge matrices. Using matrices in CBR is not common, not even in the fields of knowledge discovery or artificial intelligence; therefore, we had to develop the aforementioned technique for handling them.
2. In the second stage, we detected typical clusters and created a system for accessing them.
3. In the third stage, we built a computer system that would support the retrieval stage of the CBR. By receiving a certain cluster and retrieving the cases from that cluster that are similar to the new case, a retrieval is made as a function of the dominant characteristics of the case (identified by the attendance officer, who is the decision maker).

After clustering, we arrived at 20 groups of varying sizes. Upon a closer inspection, we found 12 overlapping groups (consisting of 159 cases) that emerged by using the two techniques for distance calculation mentioned earlier. The clusters were given to an expert attendance officer who was asked to characterize them using the problem characteristics. These groups seemed to be significant to the work of attendance officers. An additional investigation consisted of identifying proximate clusters having a common denominator. We found six pairs of proximate clusters having closely related problem descriptions.

The final step in completing the process is the retrieval of cases from the selected cluster and presenting them to the user. The program we developed receives from its user an input of three major characteristics of a new case and retrieves similar cases from the selected group.

TESTING THE MODEL AND THE SOLUTION PROCESS

In this section, we describe the testing of the system. Testing the system with many attendance officers is troublesome because it has been difficult to gather several volunteers for such testing. Nevertheless, the results so far are very promising.

The examination of the systems is composed of qualitative and quantitative elements. The qualitative element was mentioned earlier. Qualitatively, the feedback indicated that using the system helps focus the process of finding a solution (even if one does not use the tool of case retrieval), and that after being exposed to such a system, the quality of decisions of the attendance officer improved.

This section focuses on the quantitative evaluation whose purposes are:

1. Checking the applicability of the model, including the various stages with respect to its quality and efficiency.
2. Checking the performance of the model in different cases.
3. Checking the clustering into group types.
4. Checking the applicability of the model for field work.

Performing the Test

We tested the model twice. Initially (test I), the model was tested by a group of seven attendance officers. Among them:

- Two participated actively in the project by providing case data.
- Three participated in the project but did not fill out forms.
- Two other examiners who first heard of the research and the model on the day the test was conducted.

In the second test (test II), the model was tested by a group of five attendance officers who did not take part in the project. The second test was conducted in a similar way with a few changes made after the first test, which included improving the style of some questions. It is clear that the two examining groups constitute a small sample that is insufficient for statistical inference; however, the rather decisive nature of the results hints that they might be significant and certainly noteworthy.

In the tests, each attendance officer received six cases to examine. In test I, the division was as follows:

- Three typical cases (their classification to the type group is obvious and frequent).
- Two average cases (their classification is clear and frequency is average).
- One border case (belongs to the group but is marginal).

This division was made in order to test the model efficacy in different situations. In the second test, the division was two, two, and two, respectively, in order to obtain more information on the border cases.

The cases were chosen randomly, in proportion to the size of each group, and there were no two cases from the same group (each case belonged to another group). For example: from group 10, which is the largest, we obtained six significant cases, four average cases, and two border cases. From group five, which is the smallest, we got two significant cases, one average case, and one border case. The sampling was also based on the proximity between the groups. If we identified groups one and three to be proximate, we made sure that the attendance officer would receive a case only from one of these groups (one or three).

The attendance officers were asked to perform the following actions for each case:

- Examine the characteristics of the case.
- Choose the group that best fits this case from the cluster groups.
- Identify three major characteristics of the case.
- Based on the three characteristics and the group, obtain three similar cases.
- Determine the appropriate modes of action, taking into consideration similar cases and using personal judgment.
- Complete the feedback form (Table 2 or Table 3).
- After solving all cases, complete a general feedback form (Table 4).

In the second test, the support system also determined automatically the group based on the three characteristics. If this system group was different from the group chosen by the attendance officers, they were asked to

TABLE 2 Case Feedback Form for Test I

Question	Average
1. To what extent does the case fit the chosen group?	3.8
2. To what extent are the retrieved cases similar to the present case?	3.7
3. To what extent do the retrieved cases help to solve the present case?	3.5
4. What is the degree of change (adding or reducing) of the retrieved actions taken to solve the present case in comparison to the actions taken to solve the retrieved cases?	2.4
5. How much did the system contribute to solving the case?	3.4

Answers were given on a scale of 1–5 (in right column): 1 – very minor, 2 – minor, 3 – fair, 4 – greatly, 5 – very greatly.

examine three additional cases belonging to the system selected group and compare their quality to the cases retrieved earlier.

Due to the small number of attendance officers available for testing, we did not administer a control group that solved problems without using the system. We intend to do so in future testing.

Tables 2, 3, and 4 include the questions attendance officers were asked to address. In the right column the average response of all 12 officers on a scale of 1–5 is given. We discuss these figures in the next section.

Analyzing the Testing Results

In general, the feedback received by the other examiners (a total of seven officers) and those who filled out forms (two) was more favorable than the feedback from those who participated in the project from the beginning, but did not fill out forms (three). Also, the case feedback forms (Table 3) were more favorable than the general feedback forms (Table 4). In the second test, the feedback forms, both case related and general, were more favorable than those received in test I.

TABLE 3 Case Feedback Form for Test II

Question	Average
1. Is the case easy to solve?	2.5
2. Is the case frequent at your work?	3.6
3. To what extent is the case really close to the group you chose?	4.1
4. What is the degree of change (adding or reducing actions) required to get from the solutions taken from the retrieved cases to the solution of the present case?	1.7
5. In case you changed the modes of action, how easy was it to arrive at the change?	3.9
6. What is the degree of contribution of the method in solving the present case?	4.3

Answers were given on a scale of 1–5 (in right column): 1 – very minor, 2 – minor, 3 – fair, 4 – greatly, 5 – very greatly.

TABLE 4 General Feedback Form (Average of Group I & II)

Question	Average
1. How well can a case characterization be described in a form?	4.0
2. How well can the actions of the attendance officer be described in a form?	3.8
3. How well can a total attendance case be fully represented by a form?	3.6
4. How frequent is the clustering process in your work?	3.5
5. Is the support system easy to use and is it efficient?	4.1
6. How much can such a system help to improve your decision making?	4.1
7. Is the system friendly and easy to understand?	3.8
8. Can such a system give a quick solution to problems?	3.8
9. Can such a system give quality solutions to problems?	3.7
10. Would you use such a system in your work?	4.0
11. How difficult was it to match solutions to the cases you examined?	2.5
12. Will the system damage the quality or efficiency of the solution?	1.6
13. Would your work be harmed as a result of implementing a decision support system?	1.6
14. Did your exposure to the project improve your understanding of your work?	3.8

Answers were given on a scale of 1–5 (in right column): 1 – very minor, 2 – minor, 3 – fair, 4 – greatly, 5 – very greatly.

Encouraging results were obtained especially in all case specific questions (questions 3, 4, 5, 6, Table 3). They indicate that the case classification was good (question 3) and case retrieval effective (question 6). They further confirm that the changes required were minor (question 4) and that making changes was rather easy (question 5). The differences between the tests can be explained by some improvements made between them and by motivational factors.

The general feedback forms (Table 4) yielded good results as well. We were especially pleased with the result 3.8 for the last question, regarding the contribution of the project to the improvement of the attendance officers' understanding of their work. This was achieved following a mere two-hours session including method presentation and testing.

CONCLUSIONS

Starting from a set of problem attributes, we developed a CBR approach that employs subjective influence knowledge as key element in the process. This knowledge served to organize the cases into clusters and also was used to retrieve the most relevant cases from the best clusters. The CBR approach solely uses the structure of the case for organization and retrieval. The values of the attributes are not used until the practitioners manipulate the cases when they solve problems. Given the encouraging results, this seems to be a powerful approach.

The next step in developing the approach is observing how it behaves in a real-world setup. In order to do this, we develop mechanisms for maintaining

the case database as it grows. Clearly, in this situation, the manual characterization we employed no longer works effectively and the automated methods we explored would take precedence.

Other developments include testing the approach in similar domains in order to check its generality. We think that the same representation is appropriate in diverse problems. In many help-desk applications (in engineering and social domains) the need arises to provide quick solutions based on a simple situation description. We are presently trying to use the approach in two additional domains: addressing academic problems of students at a university and improving customer service of a CAD software provider.

REFERENCES

Aamodt, A. 1994. Explanation-driven case-based reasoning. In *Topics in Case-Based Reasoning*, eds. S. Wess, K. Althoff, M. Richter, 274–288. Berlin: Springer Verlag.

Aamodt, A. and E. Plaza. 1994. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications* 7(1):39–59.

Aha, D. W., L. A. Breslow, and H. Munõz-Avila. 2001. Conversational case-based reasoning. *Applied Intelligence* 14(1):9–32.

Ali, K., C. Brunk, and M. Pazzani. 1994. On learning multiple descriptions of a concept. In *Proceedings of Tools with Artificial Intelligence*, New York, NY, pages 476–483.

Bergmann, R. 2002. *Experience Management*. Berlin: Springer Verlag.

Bergmann, R., W. Wike, I. Vollrath, and S. Wess. 1996. Integrating general knowledge with object-oriented case representation and reasoning. In *Fourth German Workshop on CBR—System Development and Evaluation*, eds. H.-D. Burkhard, and M. Lenz, 120–127. Berlin: Informatik-Berichte.

Bunke, H. 1997. On the relation between graph edit distance and maximum common subgraph. *Pattern Recognition Letters* 18:689–694.

Bunke, H. and K. Shearer. 1998. A graph distance metric based on the maximal common subgraph. *Pattern Recognition Letters* 19(3–4):255–259.

Carbonell, J. G. 1986. Derivational analogy: A theory of reconstructive problem solving and expertise acquisition. In *Machine Learning: An Artificial Intelligence Approach*, Volume 2, eds. R. Michalski, J. Carbonnel, and T. Mitchell, 371–392. Los Altos, CA: Morgan Kaufmann.

Cheng, C. H., J. Motwani, A. Kumar, and J. Jiang. 1997. Clustering cases for case-based reasoning systems. *Journal of Computer Information Systems* 38(1):30–37.

Dash, M. and H. Liu. 2001. Efficient hierarchical clustering algorithms using partially overlapping partitions. In *Proceedings of the (PAKDD 2001)*, pages 495–506. Berlin: Springer-Verlag.

Everitt, B. 1993. *Cluster Analysis*, 3rd ed. London: Arnold.

Fisher, D., X. Ling, R. Carnes, Y. Reich, S. J. Fenves, J. Chen, R. Shiavi, G. Biswas, and J. Weinberg. 1993. Selected applications of an AI clustering technique to engineering tasks. *IEEE Expert* 8(6):51–60.

Gebhardt, F., A. Voß, W. Gräther, and B. Schmidt-Belz. 1997. *Reasoning wit Complex Cases*. Boston: Kluwer.

Höppner, F., F. Klawonn, R. Kruse, and T. Runkler. 1999. *Fuzzy Cluster Analysis*. Chichester: Wiley.

Israel's Central Bureau of Statistics. 1999. *Survey of Constancy and Dropout of High Schools During 1987–1996*, Jerusalem.

Jain, A. K., M. N. Murty, and P. J. Flynn. 1999. Data clustering: A review. *ACM Computing Surveys* 31:264–323.

Kolodner, J. 1993. *Case-Based Reasoning*. San Matco, CA: Morgan Kaufmann Publishers.

Lenz, M. and H.-D. Burkhard. 1996. Case retrieval nets: Basic ideas and extensions. In *KI-96: Advances in Artificial Intelligence*, Lecture Notes in Artificial Intelligence, 1137, eds. G. and S. Hölldobler, 227–239. Berlin: Springer-Verlag.

Manago, M., R. Bergmann, S. Wess, and R. Traphöner. 1994. *A Common Case Representation Language*, ESPRIT Project 6322 Deliverable D1, University of Kaiserslautern, Kaiserslautern.

Markovitch, S. and P. D. Scott. 1988. The role of forgetting in learning. In *Proceedings of The Fifth International Conference on Machine Learning*, pages 459–465, San Francisco, CA, USA. Morgan Kaufmann.

McSherry, D. 2000. Intelligent case-authoring support in CaseMaker-2. In *Proceedings of the European Workshop on CBR (EWCBR)*, pages 198–209. Berlin: Springer–Verlag.

Messmer, B. T. and H. Bunke. 1998. A new algorithm for error-tolerant subgraph isomorphism detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20(5):493–504.

Miyashita, K. and K. Sycara. 1995. Improving system performance in case-based iterative optimization through knowledge filtering. In *Proceedings of the International Joint Conference on Artificial Intelligence*, AAAI Press, Menlo Park, CA, pages 371–376.

Porter, B. W., R. Bareiss, and R. C. Holte. 1990. Concept learning and heuristic classification in weak-theory domains. *Artificial Intelligence* 45(1–2):229–263.

Reich, Y. 1993a. The development of Bridger: A methodological study of research on the use of machine learning in design. *Artificial Intelligence in Engineering* 8(3):217–231.

Reich, Y. 1993b. A model of aesthetic judgment in design. *Artificial Intelligence in Engineering* 8(2):141–153.

Reich, Y. and S. J. Fenves. 1991. The formation and use of abstract concepts in design. In *Concept Formation: Knowledge and Experience in Unsupervised Learning*, eds. D. H. J. Fisher, M. J. Pazzani, and P. Langley, 323–353. Los Altos, CA: Morgan Kaufmann.

Reich, Y. and S. J. Fenves. 1995. A system that learns to design cable-stayed bridges. *Journal of Structural Engineering*, *ASCE* 121(7):1090–1100.

Reich, Y., M. Medina, T.-Y. Shieh, and T. Jacobs. 1996. Modeling and debugging engineering decision procedures with machine learning. *Journal of Computing in Civil Engineering*, 10(2):157–166.

Reinartz, T., I. Iglezakis, and T. Roth-Berghofer. 2001. Review an restore for case-base maintenance. *Computational Intelligence* 17(2):214–234.

Smyth, B. and M. T. Keane. 1996. Using adaptation knowledge to retrieve and adapt design cases. *Knowledge Based Systems* 9:127–135.

Smyth, B. and M.T. Keane. 1998. Adaptation-guided retrieval: questioning the similarity assumption in reasoning. *Artificial Intelligence* 102:249–293.

Tao, C. W., 2002. Unsupervised fuzzy clustering with multi-center clusters. *Fuzzy Sets and Systems* 128:305–322.

U.S. Department of Education. 2001. National Center for Education Statistics, *Dropout Rates in the United States 2000*, NCES 2002–114, U.S. Government Printing Office, Washington, D.C.

van Rijsbergen, C. J. 1979. *Information Retrieval*, 2nd ed. London: Butterworths.

Witten, I. H. and F. Eibe. 1999. *Data Mining: Practical Machine Learning Tools with Java Implementations*. San Francisco, CA: Morgan Kaufmann.

Yang, Q. and J. Wu. 2001. Enhancing the effectiveness of interactive case-based reasoning with clustering and decision forests. *Applied Intelligence* 14(1):49–64.