Journal of ELECTRICAL ENGINEERING, VOL. 53, NO. 12/s, 2002, 20–23

CASE BASED REASONING — A POWERFULL ARTIFICIAL INTELLIGENCE APPROACH

Vladimir Kurbalija — Mirjana Ivanović *

Case-based reasoning is a relatively new and promising area of artificial intelligence. Generally speaking, case-based reasoning (CBR) is a reasoning method that facilitates knowledge management in which knowledge is a case base acquired by a learning process. Case-based reasoning can be used, for solving problems, in many practical domains such as: mechanical engineering, medicine, business administration, etc. Furthermore, for each domain, various task types can be implemented. Some of them are: classification, diagnosis, configuration, planning, decision support, etc.

The purpose of this paper is to present essential concepts of this promising area. At the beginning, some technical terms of artificial intelligence are introduced. Following this, the foundations of case-based reasoning are presented. At the end, Case Retrieval Net - an efficient memory structure for implementation case-based reasoning systems, is described.

K e y w o r d s: artificial intelligence, case-based reasoning

2000 *Mathematics Subject Classification*: 68T20, 68P20

1 INTRODUCTION

Case-Based Reasoning has become a very successful technique for knowledge based systems in different domains. This promising technique is based on the use of previous experience in the form of cases to better understand and solve new appearing problems in a particular domain. The main idea of CBR is the hypothesis that similar problems usually have similar solutions.

The purpose of this paper is to present the mathematical foundations of this promising area. In the second section some basic terms are introduced. Section 3 contains the definitions of all the functions needed for the retrieval process in CBR, while in Section 4 a possible implementation of CBR retrieval system is given. The conclusion is given in Section 5.

2 BASIC CONCEPTS OF CBR

Case-based reasoning is a problem solving technology. The basic scenario for case-based reasoning, from the simplified point of view, looks as follows: In order to find a solution of a particular problem one looks for a similar problem in an experience base, takes the solution from the past and uses it as a starting point to find a solution to the current problem.

A general desire in every knowledge-based system is to make use of the past experience. Experience may be concerned with what was true or false, correct or incorrect, more or less useful. It can be represented by a rule, constraint, some general law or advice or simply by saving a past event. From all of this the main idea of the case can be obtained. The case is some recorded situation where the problem was totally or partially solved. In its simplest form, the case is represented as an ordered pair

$$(problem, solution)$$

The existence of the case means that the corresponding episode happened in the past. This episode contains some decisions that a decision maker found useful. However, somebody else may not be happy with such a case and may neglect it. From this it follows that cases must be selected carefully, so different categories of cases can exist: good, typical, important, misleading or unnecessary.

A case base is a set of cases, which is usually equipped with some additional structure. A structured case base is usually called a case memory.

The next very important concept in the case-based reasoning is *similarity*. While in classical databases information can be retrieved by using only exact matches, in the case-based reasoning cases it can be retrieved by using even inexact matches. The notion of similarity is equivalent to a dual mathematical concept — distance.

In the functional way similarity can be defined as a function $sim : U \times CB \to [0, 1]$ where U refers to the universe of all objects, while CB refers to the case base (just those objects which were examined in the past and saved in the case memory). A higher value of the similarity function means that these objects are more similar.

Usually the concept *acceptance* is used instead of *similarity*, because acceptance includes similarity but

* Department of Mathematics and Informatics, Faculty of Science, University of Novi Sad Trg, D. Obradovića 4, 21000 Novi Sad, Yugoslavia, E-mail: kurba@im.ns.ac.yu, mira@im.ns.ac.yu.

Research is partially supported by the project no. 1844: "Development of (intelligent) techniques based on software agents for application in information retrieval and workflows", Ministry of Science, Technologies, and Development, Republic of Serbia.

ISSN 1335-3632

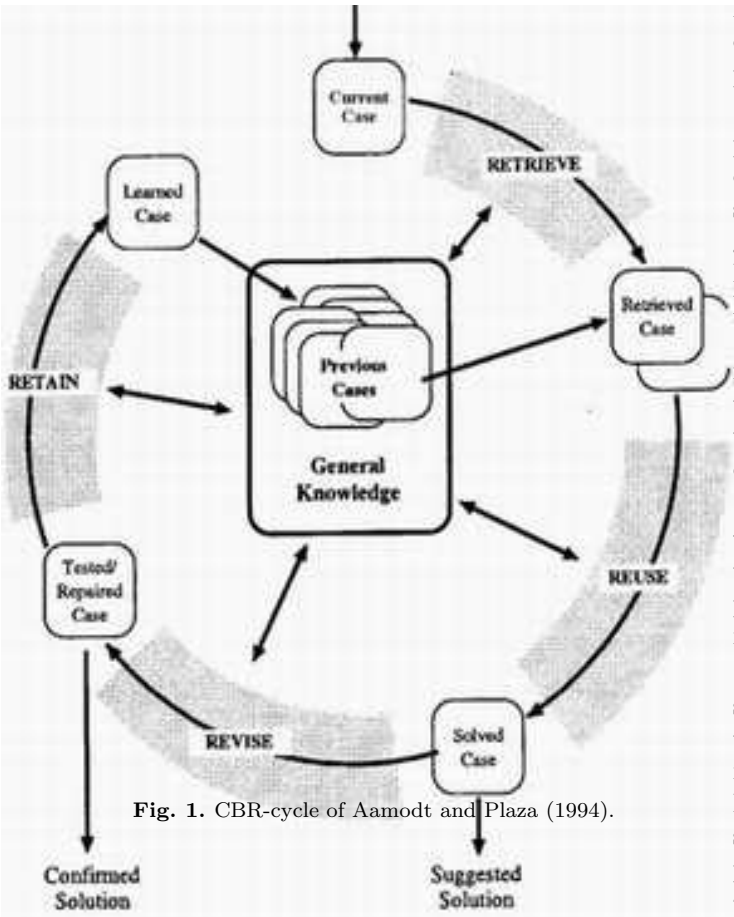

Fig. 1. CBR-cycle of Aamodt and Plaza (1994).

also other approaches related to "expected usefulness", "reminds on", etc.

Retrieval is a basic operation in databases and therefore in the case base too. A query to a database retrieves some information by an exact match by using a key, while a query to a case-based reasoning system presents a problem and returns a solution by using inexact matches with the problems from the cases in the case base. As in databases, trees play a major role in efficient retrieval. Examples of retrieval structures are: kd-trees (k-dimensional trees), case retrieval nets, discrimination nets, etc.

The *knowledge container* is a structural element which contains some quantity of knowledge. The idea of the knowledge container is totally different from the traditional module concept in programming. While the module is responsible for a certain subtask, the knowledge container does not complete the subtask but contains some knowledge relevant to many tasks. Even small tasks require the participation of each container. The concept of the knowledge container is similar to concepts of the nodes and propagation rules in neural networks.

In case-based reasoning the following knowledge containers can be modified: a) the vocabulary used; b) the similarity measure; c) the case base; and d) the solution transformation. In principle, each container can carry almost all knowledge available. From a software engineering point of view there is another advantage of case-based reasoning - the content of the containers can be changed locally. This means that manipulations on one container have little consequences on the other ones. As a consequence, maintenance operations [2], [4] are easier to be performed than on classical knowledge based systems.

The task of machine learning is to improve a certain performance using some experience or instructions. In inductive learning, problems and good solutions are presented to the system. The major desire is to improve a general solution method in every inductive step. Machine learning methods can be used in order to improve the knowledge containers of a case-based reasoning system (the case base, similarity measures and the solution transformation). However, one of the greatest advantages of the case-based reasoning system is that it can learn even through the work with users, modifying some knowledge containers.

The case based reasoning system has not only to provide solutions to problems but also to take care of other tasks occurring when it is used in practice. The main phases of the case-based reasoning activities are described in the CBR-cycle in [1] in Figure 1.

In the *retrieve* phase the most similar case (or k most similar cases), to the problem case, is retrieved, while in the *reuse* phase some modifications to the retrieved case is done in order to provide a better solution to the problem (case adaptation). As the case-based reasoning only suggests solutions, there may be a need for a correctness proof or external validation. That is the task of the phase *revise*. In the *retain* phase the knowledge learned from this problem is integrated in the system by modifying some knowledge containers.

3 RETRIEVAL FUNCTIONS

The case-based reasoning was developed in the context and in the neighborhood of problem solving methods, learning methods (Machine Learning, Statistics, Neural Networks) and retrieval methods (Data Bases, Information Retrieval). It has inherited the concepts of "problem" and "solution" and a notion of "similarity" based on the distance.

In this section, some formal or informal definitions of the essential concepts will be given. Definitions are taken from [3].

I Information Entities

The information entity is an atomic part of a case or query. E denotes the set of all information entities in a given domain.

A case is a set of information entities: $c \subseteq E$.

The set of cases (in the case memory) is denoted by C, $C \subseteq P(E)$.

A query is a set of information entities: $q \subseteq E$.

In many applications, the information entities are simply attribute-value pairs. Some examples of information entities are: (price, 1000), (price, 324), (color, blue),

(mass, 54 kg). We say that the first two information entities are comparable (because they have the same attribute) while the other information entities are not comparable. This causes a structuring of the set E into disjoint sets E_A, where E_A contains all attribute-value pairs from E for a certain attribute A.

II Acceptance

Usefulness of a case in the case completion process depends on real world circumstances that are not completely known at the retrieval time. This means that usefulness is only an a posteriori criterion. The retrieval from the case memory will be based on matching of certain information entities. Usefulness of former cases is not restricted to those cases that are similar to a given query for all information entities. Cases may contain information entities that have no counterpart in the query. It is also possible that some information entities of the query are not present in the useful case.

Some special desirable properties of acceptance are following:

P1: A case might be acceptable for a query even if there exist some information entities that are not comparable.

P2: A case might be unacceptable for a query if there exist an unacceptable information entity (a fixed budget may forbid expensive offers).

P3: The same information entity may have different importance for different cases (information entity (sex, male) has different importance in pregnancy testing and in testing for influenza).

P4: The same information entity may have different importance for different queries according to the user's intentions (material have different priorities in design queries).

P5: Information entities may not be independent of each other.

III Acceptance Functions

Queries have been defined as sets of information entities. A weighted query is the generalization of this concept.

Weighted query. The weighted query assigns an importance value to each information entity by a function:

$$\alpha_q : E \to R,$$

where $\alpha_q(e)$ denotes the importance of the information entity e for the query q.

High values indicate a high importance; negative values indicate the rejection of related cases. The value 0 is used as a neutral element ($\alpha_q(e) = 0$, means that information entity e is unimportant to the query q). Of course, values for $\alpha_q(e)$ can be taken from the set $\{0, 1\}$, meaning that "e is (un)important for the q".

Local Acceptance Functions for Attributes. A local acceptance function σ for the attribute A is defined over the domain $dom(A)$:

$$\sigma : dom(A) \times dom(A) \to R,$$

such that higher value $\sigma(e, e')$ denotes a higher acceptance of the value e (of a case c) for the value e' (of a query q).

By using function σ we can compute the acceptance of the information entity e' from the case for a single information entity e of a query. However, a query may contain several information entities e such that $\sigma(e, e')$ is defined for the single information entity e'. The question is: how these values can be combined to a single value for e' which expresses the resulting acceptance value of e' for that query.

Local Accumulation Function. Let $E_e = \{e_1, \cdots, e_n\}$ denote the set of all information entities to which the information entity e is comparable concerning acceptance ($E_e = \{e' | \sigma(e', e)$ is defined$\}$). The local accumulation function π_e for e is a function:

$$\pi_e : \underbrace{R \times \cdots \times R}_{n} \to R,$$

such that $\pi_e(a_1, \cdots, a_n)$ denotes the accumulated acceptance in e. The values a_i denote the contributions of the information entities $e_i \in E_e$ according to their occurrence in the query q and their local acceptance computed by $\sigma(e_i, e)$. The contributions are computed by a function:

$$f : R \times R \to R,$$

such that $a_i = f(\alpha_q(e_i), \sigma(e_i, e))$.

The retrieval of the cases can be considered as a process of reminding. Reminding may be of different strength; cases are in competition for retrieval according to the query. The cases receiving more reminders of more strength are the winners. The strength (importance, relevance) of reminding for an information entity $e \in c$ is given by a relevance function.

Relevance Function. The relevance between information entities and cases is described by a relevance function:

$$\rho : E \times C \to R.$$

The relevance $\rho(e, c)$ is considered as a measure for the relevance of information entity e for the retrieval of a case c. $\rho(e, c)$ is defined if and only if $e \in c$.

The acceptance of a case c for the query q is accumulated from the contributions of the information entities $e \in c$ according to their relevancies $\rho(e, c)$. The contributions of the information entities are computed by their local accumulation functions π_e. The accumulation in the cases is evaluated by *Global accumulation function.*

Global Accumulation Function. The global accumulation function π_c has the form:

$$\pi_c : \underbrace{R \times \cdots \times R}_{k} \to R,$$

for $c = \{e_1, \cdots, e_k\}$. The accumulated acceptance of the case c regarding its constituting information entities is then computed by $\pi_c(p_1, \cdots, p_k)$, where p_i is the contribution of the information entity $e_i \in c$. This contribution p_i depends on $\rho(e_i, c)$ and another real value x_i assigned to e_i (x_i is the accumulated local acceptance value computed by $\pi_{e_i}(a_1, \cdots, a_n)$). The contributions p_i are computed by a function:

$$g : R \times R \to R,$$

such that $p_i = g(x_i, \rho(e_i, c))$.

Global Acceptance function. Acceptance between weighted queries and cases is expressed by a global acceptance function:

$$acc : R^E \times P(E) \to R.$$

The acceptance $acc(\alpha_q, c)$ of a case c for a weighted query q can now be accumulated by using the introduced functions:

$$acc(\alpha_q, c) = \pi_c(p_1, \cdots, p_k),$$

where $c = \{e_1, \cdots, e_k\}$ and the values p_i are contributions of the information entities e_i from the case c.

If, for example, f and g are considered as products and π_c and π_e as sums then the global acceptance function looks like this:

$$acc(\alpha_q, c) = \sum_{e' \in c} \rho(e', c) \sum_{e \in E_{e'}} \sigma(e, e') \cdot \alpha_q(e)$$

Here, the properties P1,...,P4 are satisfied, but for satisfaction of the property P5, the appropriate selection of the functions f, g, π_c and π_e is needed.

4 A POSSIBLE IMPLEMENTATION — CASE RETRIEVAL NET

Case Retrieval Net (CRN) is a special memory structure that has been developed especially for being employed in large case bases. CRNs are able to deal with vague and ambiguous terms, they support the concept of information completion and can handle case bases of reasonable size efficiently.

CRN is a net structure with an information entity node for each information entity and case node for each case. There exists an "acceptance" arc from the information entity nodes e to the information entity node e' if $\sigma(e, e')$ is defined, and there exists an "relevance" arc from the information entity node e to the case node c if $\rho(e, c)$ is defined. The arcs in the net are weighted by the values $\sigma(e, e')$ and $\rho(e, c)$, respectively.

In practice, it may be impossible to include all information entities. Usually, it is sufficient to build a net from the information entities which occur in the cases from the case base only.

Acceptance values are computed by a spreading activation process in the net as follows: Information entity nodes are initially activated by $\alpha_q(e)$. The computation is performed by propagating along the acceptance arcs to further information entity nodes, and from these nodes over relevance arcs to case nodes. The functions f/π_e and g/π_c are responsible for the accumulation of activities in the information entity nodes and in the case nodes, respectively. The final activation at the case nodes denotes the acceptance value of the case for the weighted query.

5 CONCLUSION

Case-based reasoning is a problem solving paradigm that in many respects is fundamentally different from other major AI approaches. Instead of relying solely on general knowledge of a problem domain, or making associations along generalized relationships between problem descriptors and conclusions, CBR is able to utilize the specific knowledge of previously experienced, concrete problem situations (cases). A new problem is solved by finding a similar past case, and reusing it in the new problem situation. A second important difference is that CBR also is an approach to incremental, sustained learning, since new experience is retained each time a problem has been solved, making it immediately available for future problems. The CBR field has grown rapidly over the last few years, as seen by its increased share of papers at major conferences, available commercial tools, and successful applications in daily use.

References

[1] AAMODT, A.—PLAZA, E.: Case-Based Reasoning: Foundational Issues, Methodological Variations and System Approaches, AI Commutations, 1994, pp. 39–58.

[2] IGLEZAKIS, I.: The Conflict Graph for Maintaining Case-Based Reasoning Systems, 4th International Conference on Case-Based Reasoning (ICCBR 2001), 2001, pp. 263–276.

[3] LENZ, M.—BRTSCH-SPORL, B.—BURKHARD, H.—WESS, S.: Case-Based Reasoning Technology: From Foundation to Applications, Springer, 1998.

[4] REINARTZ, T.—IGLEZAKIS, I.—ROTH-BERGOFER, T.: On Quality Measures for Case Base Maintenance, 5th European Workshop (EWCBR 2000), 2000, pp. 247–260.

Received 28 May 2002

Vladimir Kurbalija is a research and teaching assistant at the Department of Mathematics and Informatics, Faculty of Science, University of Novi Sad, Yugoslavia. His Master-thesis supervisor (in Case-Based Reasoning) is Professor Mirjana Ivanović.

Mirjana Ivanović is a full professor at the Department of Mathematics and Informatics, Faculty of Science, University of Novi Sad, Yugoslavia. Her field of interest include Agent Technology, Artificial Intelligence Techniques, especially CBR, and programming languages.