Towards An Evolutionary Approach to Case Retrieval

NABILA NOUAOURIA-AMRI, YAMINA MOHAMED BEN ALI, MED TAYEB LASKRI
Lab.LRI/GRIA Département d'Informatique
Université Badji Mokhtar de Annaba
B.P. 12 Annaba (23000)
ALGERIA

Abstract: Case-Based Reasoning (CBR) is a problem solving paradigm which is able to *retrieve* and *reuse solutions* that have worked for *similar* situations *in the past.* Past situations and their solutions are stored in a memory called case base. To find the good experiment in memory is the key of success in the reasoning. The good experiment is the one that can perform the best inference. To identify adequate experiment in memory constitutes the process of recall. In this paper, we present an evolutionary approach of the recall process applied to associative memory architecture. The main idea is to compute the neighbourhood of a new problem by an evolutionary algorithm. This will delimit our search space in the case base.

Keywords: Case Based Reasoning (CBR), Case Retrieval, Evolutionary Computing (EC), Associative Memory.

1. Introduction

Case-Based Reasoning (CBR) is a problem solving paradigm which is able to *retrieve* and *reuse solutions* that have worked for *similar* situations *in the past*. Past situations and their solutions are stored in a memory called case base.

To find the good experiment in memory is the key of success in the reasoning. The good experiment is the one that can perform the best inferences. To identify adequate experiment in memory is the recall process. Recall is highly influenced by memory organization and by retrieve strategies. The accuracy (in the sense of exhaustiveness) and speed of recall task constitute two important parameters in the performance evaluation of a CBR system.

Case based reasoning is an Artificial Intelligence paradigm that can be synergistically combined with other approaches to facilitate a broad array of tasks [1].

Among those possible combinations, we will present in the following, an approach to perform a quick and complete recall, in an associative memory, using evolutionary computing.

The main idea is to compute the neighbourhood of a new problem by an evolutionary algorithm. This will delimit our search space in the case base. And then, access directly to this neighbourhood via a network in an associative memory style.

For best understanding of the paper, we will start with a fast skimming of CBR paradigm (section 2) followed by memory models used in CBR (section 3). The proposed approach will be presented in two steps. Section 4.1, will present knowledge representation in terms of evolutionary algorithm. Section 4.2, will present the memory structure supporting the approach, followed by conclusion and related works in section 5.

2. Case Based Reasoning Paradigm

Problem solving with CBR proceeds as follows: *A* new problem is posed and is described as the problem part of a new case, sometimes also called the query. Then, old cases containing problems that are similar to the new problem are retrieved and the most suitable solution among retrieved solutions is suggested to become the solution of the new problem. This solution is then tested in reality and may lead to a revised solution worth to be stored as a new case. This last step is a form of incremental learning that enables CBR systems to adapt to changing environments rather smoothly.

In theory, the basic cycle of CBR is in three phases: «retrieve, reuse and store». The system looks for a similar case to the input case, reuse the recovered solution, and finally, store the current case for a future utilization.

This cycle can be extended to five stages [2], [3]:

1) Presentation or specification: a description of the problem is provided at the entrance of the system. This description must be suitable to the comparison between the case in entrance and cases stored in memory (uniformity of the representation). One of the key points of the CBR is the research of applicable cases. It is what justify the importance of the process that is going to label cases with indexes so that they could be recalled at the appropriate moment. This indexing leans mainly on the

extraction of the most characteristic descriptors of the case.

2) Retrieval: the system looks for cases that are best unified to this description (closest matching cases). These cases are stored in a case base or case memory (i.e.: data base of cases). If the case base is organized according to a particular structure, an algorithm of research describes then a path in this structure. A phase of filtering or selection is often done permitting to eliminate a subset of worst cases. A measure of similarity can be then used to measure the resemblance, more precisely, between the current case and selected cases. Then returns ordered cases.

3) Adaptation: the system uses the current problem and the matching case to generate a solution to this problem. The adaptation constitutes the second difficult point (after the indexing) when conceiving a CBR system. It is necessary to decide what type of knowledge it is interesting to transfer from the best case remembered. We can do a transformation analogy, consisting in transforming the solution of retrieved case to adapt it to the current case. Or to proceed by derivation when adapting the method of solution generation. Otherwise, the possibility to adapt several cases to solve a problem, in a simultaneous way or operating several remembering and simple adaptation to the different stages of the resolution, has been judged more creative [4].

4) Validation: this phase includes the possibility of an assessment of the solution proposed while testing it in an either simulated or real environment. The return of information can guide, in case of failure of the proposed solution, a process of repair.

5) Storage: the validated solution is added to the case base for a future utilization. We can have systems which store cases systematically in memory. A more selective memorization is however possible and would use some specific criteria to judge if the new case is useful to learn according to the current case memory. In general a case is useful to learn when using possibilities of adaptation and it can reach a point of the solution space that was inaccessible before the arrival of this new case.

3. Memory model

In order to function correctly, the case based reasoning uses cases stored in a case base. This one is supposed to be representative of all problems encountered in the field. The more it contains cases, the better selected case will be similar to the new case. The elaborate solution will be thus better. But with increasing base size, the calculating cost will be more prohibitive. This is why techniques of memory organization and search algorithms are particularly important in this reasoning mode.

There are several memory organizations according to which search algorithms exist [2], [3]:

The flat memory: cases are stored sequentially in a simple list, array or file. Cases will be retrieved by applying a matching function sequentially to each case in the file, keeping track of the degree of match of each case and returning those cases that match best. There is no particular organization put on the top of the cases in this scheme and the retrieval algorithm is very simple. The matching heuristics, in fact, do all the work. The major advantage is that the entire case library is searched. As a result, the accuracy of retrieval is a function only of how good the match functions are. Moreover the addition of a new case is not expensive. However, the organization is expensive when the base is too large. To remedy this disadvantage, we can use alternatives such as: Surface indexing to reduce the total of candidates, or partitioning of the base in sections, or also parallel implementations.

Shared-feature Network: It is based on gathering cases presenting similarities in the same cluster. The hierarchies are formed when the clusters are subdivided in under-clusters. The methods of regrouping used are those met in machine learning. This technique offers the advantage of better partitioning the case base making search more effective than a sequential search. However, the storage of new cases is complex. It is difficult to maintain the optimality of the network. An additional space is necessary for the organization. Several networks with different priorities would be necessary to increase the precision of search. In addition, there is no guarantee that a candidate is not forgotten.

Discrimination Networks: a discrimination of cases is made as a side effect of clustering in shared-features networks. In discrimination networks, the priority is put on discrimination. Each internal node is a question that subdivides the set of items stored underneath it. Each child node represents a different answer to the question posed to its parent. The most significant questions are put in first. We find the advantages of Shared-feature networks and search is more efficient for the latter, because the implementation of the traversal of the arcs is based on the answering of questions so easier than the first network. We can also meet the disadvantages of the Shared-feature networks. In addition, the missing information makes the search algorithm incompetent to answer a question so inefficient to pursuit the traversal of the network.

Redundant discrimination networks: They provide an answer to the problem of missing information. They organize items by using various discrimination networks, each one with a different ordering of questions. A search is done in parallel on the various networks. If in one of the networks a question does not have an answer, search in this network is discounted. At least, one of the networks will find the case which is matching if there exists. The major disadvantage of such organization is the complexity of its implementation.

The most frequently used models of memory relying on a Top-Down search, present some common features [5]:

- They support a structuring of data by regrouping together related objects.
- They support an efficient retrieval by utilizing traditional tree search algorithms.
- Traversing a Top-Down memory structure is performed by answering questions in the internal nodes in order to choose which path to follow. This requires a particular order in answers. In the case of incomplete information, it could mislead the utilization of an erroneous path.
- Once a certain cluster of cases has been reached in the leaf of a tree, it is hard to access neighbouring clusters containing similar cases.

For those reasons, we will expose another vision of retrieval problem based on the construction of problem neighbourhood.

4. Proposed Approach

The retrieval of applicable cases can be formulated in how to extract from the search space a sub-space of similar cases. This sub-space is what we call neighbourhood of the target problem. It is classically obtained by a search strategy.

The main idea is to compute the neighbourhood of a new problem by an evolutionary algorithm (figure1). And then, access directly to this neighbourhood via a network in an associative memory style.

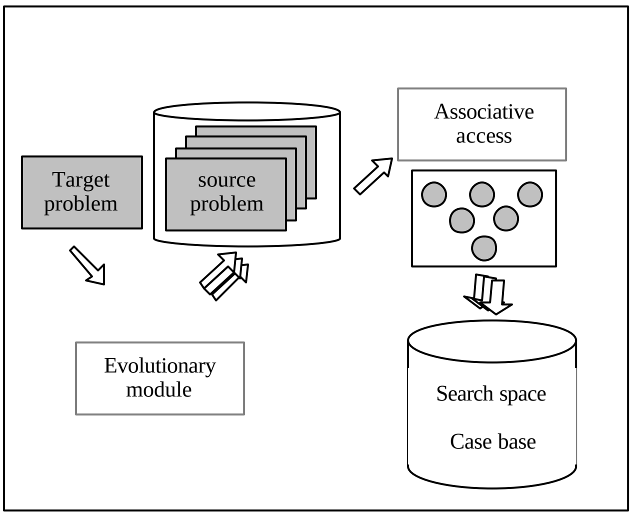

Figure.1. A global vision

In the following, we will focus on the evolutionary module.

4.1. The evolutionary vision

A case is an entity within which is gathered various information on a past situation. The term «situation» is very general. A case is also an entity about which an inference is possible by a process consisting in situating the new situation with regard to the definite circumstances in the case.

A case is constituted of descriptors, also called dimensions, distributed in three categories: the description of the problem, the solution and issues of the solution.

The description of the problem includes the context of the case. The solution is the solution of the problem or the reaction to this description (for example, the deliberation of a courthouse, the taken decision, etc.). It can also describe the used reasoning. The exit of the case is the description of the context after the implementation and execution of the solution. This part of the case is generally omitted and knowledge is reported on the other stages of the reasoning.

Coding

When a new problem is posed, the request to retrieve similar cases is generally, expressed with dimensions of problem description (figure2).

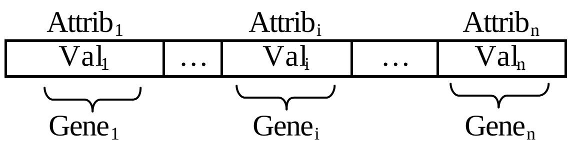

Figure.2. Problem description as a chromosome

The coding of problem will be:

Problem description : pbm	Chromosome
Descriptors : d_i	genes
Descriptor values val_j	Alleles

Tab.1. matching between CBR and EC entities

Pbm = { d_i } : an array of descriptors.
$d_i = (Attrib_i, val_{ij})$: a couple of attribute/value
val_{ij} € Dom_j : each value belongs to a specific domain which could be symbolic or numeric.

Population genesis

The initial population is randomly generated.

Selection

This step is based on a strategy guided by similarity.

Reproduction

Reproduction is essentially made by mutation of genes. The chromosome mutation corresponds to the troubling of the entry problem description in order to generate a neighbourhood.

Fitness function

The fitness function is based on similarity assessment in terms of distance between the input problem and the actual chromosome.

The whole algorithm will be:

1. Initialise a population of chromosomes.
2. Evaluate each chromosome in the population.
3. Create offspring problems population by mating the current generation.
4. Evaluate offspring population.
5. if <stop criterion> is satisfied then stop else goto 3.

The stop criterion = population stabilisation or max time

4.2. Memory structure

The case memory is indeed, a flat structure on which we construct a nested structure. There are two types of node: value node and case node.

Each value node represents a particular value of a problem attribute. It is linked to all case nodes where it occurs.

The case node point out to the case base where the whole case is stored.

The particularity of this structure is that we reach the case by its contents (the principle of associative memories). Every source problem computed by the evolutionary module will be directly pointed in the search space via the net.

Another particularity is that the structure could be easily and automatically build by simply scanning the case memory.

5. Conclusion

Many different approaches of case memory models have been proposed in literature (see [6]). However Evolutionary computing approach seems to be interesting for multiple reasons:

- Flexible knowledge representation.
- Good computation performances
- A large scale of applicability

Up to now their application in CBR was limited to the adaptation task. An evolutionary approach to case adaptation is presented in [7]. In [8], case adaptability is improved by a Genetic Algorithm.

The proposed approach leans on a memory structure reachable by the contents. Flexible, easy to construct and having a uniform knowledge representation according to the Evolutionary computing module.

It is very important to emphasize that the presented approach represents a general framework. When considering a specific application field we have to tune parameters of our system in order to improve the convergence. It is why no illustration is made.

Since our previous work were on adaptability guided retrieval memory (see [9] and [10]), it will be interesting to an extension of the approach taking into account the adaptability guided retrieval within the fitness function.

References:

[1] C.Marling et al., « Case-Based reasoning Integrations », in AI Magazine, vol. 23, N°1, Spring 2002.

[2] J. Kolodner, « Case Based Reasoning », ed. Morgan Kaufmann, 1993.

[3] A. Aamdot, E. Plaza, « Case Based Reasoning: Foundational Issues, Methodological Variations and System Approaches », published in IOS Press, Vol. 7:1, pp.39-59, 1994.

[4] J. Kolodner, « Judging which is the best case for a case based reasoner », in DARPA Workshop on CBR, 1992.

[5] M. Lenz et al, « Diagnosis and decision support » in LNAI 1400, éd. Springer 1998.

[6] N. Nouaouria, M.T. Laskri «Toward a formal model of case based reasoning from the roots », proceedings CESA'2003, Lille France, July 9-11 2003.

[7] A. Gomez de Silva Garza, M.L. Maher « An Evolutionary Approach to Case Adaptation », 3rd ICCBR'99, in LNAI 1650 , Germany, July 1999.

[8] L. Purvis, S. Athalye « Towards Improving Case Adaptability with a Genetic Algorithm », 2nd ICCBR'97, in LNAI 1266 , USA, July 1997.

[9] N. Nouaouria, M.T. Laskri « un modèle de mémoire pour la recherche de cas adaptables », in proceedings of ISPS'2003, Algiers, Algeria, May 5-7, 2003.

[10] N. Nouaouria, M.T. Laskri « Retrieving adaptable cases into an associative memory », submitted to ECAI 2004.