Reasoning with Generalized Cases

Alexander Tartakovski

Similarity-based retrieval of stored objects is one of the main building blocks of a case-based reasoning approach (CBR). Over the last three decades several sophisticated retrieval methods have been developed which are more suitable for many real-world scenarios than the traditional database search. However, the CBR approach and in particular the published retrieval methods do not support sufficiently special "complex" objects called generalized cases. My Ph.D. thesis [6] clarifies the concept of generalized cases and describes application scenarios. It discusses representation approaches and then focusses on novel methods for similarity assessment and efficient retrieval.

1 Introduction

Case-based reasoning (CBR) [1,2] is an approach to problem solving by reuse of previously made experience. Its basic idea is to solve new problems by applying solutions of similar past problems [1]. In order to realize this idea a CBR-solver is required to incorporate several steps, among them a *retrieval step* used for finding the similar problems and their solutions as well as a *reuse step* used for adaptation of found solutions to fit the new problem.

The retrieval approach proposed by CBR is proven useful also for scenarios which do not require full problem solving, but only a search technology. It has been successfully integrated in many applications, for example in electronic commerce and knowledge management [2]. Its advantage over traditional database search is the ability of finding useful objects, such as products or documents, which do not match the query exactly.

The similarity measure plays a central role in case-based reasoning since it allows a comparison of a new problem/query with previously solved problems/stored objects. It is usually modelled by a domain expert, who represents on this way her/his knowledge of similarity in the respective domain. The similarity measure is applied by the retrieval approach in order to determine k objects having the highest similarity to a query.

However, current CBR technology and in particular the retrieval methods do not sufficiently support special "complex" objects occurring in certain application scenarios [3,2]. Contrary to trivial objects which can be represented by use of attribute-value pairs, a representation of "complex" objects requires variables and constraints defining dependencies between them. Bergmann and Vollrath [3,2] propose to call these objects generalized cases.

In context of the CBR-based problem solving generalized cases extend the concept of traditional cases. A traditional case, according to the CBR terminology, is a single experience item which includes a description of a problem occurred in the past and a description of its solution. Hence a single case (point case) can be considered as a point in the space defined by the Cartesian product of the problem space P and solution space S: $pc \in P \times S$. A generalized case however is a possibly infinite set of closely related problems and solutions: $GC \subseteq P \times S$. In other words it provides solutions to a set of problems [3,2].

Generalized cases occurring in scenarios which require the application of the similarity-based search only do not necessarily contain problem and solution parts. However, they also naturally cover a set of closely related simple objects. An example for such a generalized case is a life insurance product, which covers a set of possible contracts differing among others in the period of insurance, the insurance premium, and the benefits [4]. Other examples are configurable travel products, configurable furniture, flexible chip designs, etc.

The study of related literature from the past 30 years has shown that the concept of generalized cases was used for different purposes:

- for aggregation of point cases from a case base to "complex" objects and reasoning with them
- for representation of "complex" objects (e.g., experiences in problem solving, parameterized or configurable products), which naturally occur in different scenarios, and reasoning with them
- for representation and handling of "complex" queries
- for representation and handling of incomplete information in descriptions of cases and queries

2 Representation

There are two main types of generalized cases, *attribute dependent generalized cases (ADGCs)* and *attribute independent generalized cases (AIGCs)*, which differ from each other in the representation formalism. Compared to traditional point cases both types cause an increased representation effort.

For both types of cases an attribute-value representation is chosen [2,6]. It is assumed that the attribute space A is subdivided into a problem and a solution space in the following manner: $A = (A_1, \ldots, A_k, A_{k+1}, \ldots, A_n)$ where $A_1, \ldots, A_k$ represents the k attributes of the problem space P and $A_{k+1}, \ldots, A_n$ represents the $n-k$ attributes of the solution space S. The range of the Attribute A_i is denoted by T_i.

ADGCs are cases which attributes are dependent on each other. The dependencies are usually represented by constraints and/or if-then conditions. For instance a generalized case with numeric attributes "a", "b" and a symbolic attribute "c" can look as follows:

$$\begin{array}{llll} ADGC & \subseteq T_a \times T_b \times T_c & s.t. & \\ a \leq & -2b + 42 & if & c = 'q' \\ a \leq & -5b + 8 & if & c = 'k' \\ & \vdots & & \end{array} \quad (1)$$

Auszug aus: Künstliche Intelligenz, Heft 1/2009, ISSN 0933-1875, BöttcherIT Verlag, Bremen, **www.kuenstliche-intelligenz.de/order**