Engineering and learning of adaptation knowledge in Case-Based Reasoning

Amélie Cordier, Béatrice Fuchs, and Alain Mille

LIRIS UMR 5205
CNRS/INSA de Lyon/Université Claude Bernard Lyon 1/
Université Lumière Lyon 2/Ecole Centrale de Lyon
Bâtiment Nautibus (710),
43, Boulevard du 11 Novembre 1918 - 69622 VILLEURBANNE CEDEX
{acordier,bfuchs,amille}@liris.cnrs.fr,
http://liris.cnrs.fr/

Abstract. Case-based reasoning (CBR) uses various knowledge containers for problem solving: cases, domain, similarity, and adaptation knowledge. These various knowledge containers are characterised from the engineering and learning points of view. We focus on adaptation and similarity knowledge containers that are of first importance, difficult to acquire and to model at the design stage. These difficulties motivate the use of a learning process for refining these knowledge containers. We argue that in an adaptation guided retrieval approach, similarity and adaptation knowledge containers must be mixed. We rely on a formalisation of adaptation for highlighting several knowledge units to be learnt, i.e. dependencies and influences between problem and solution descriptors. Finally, we propose a learning scenario called "active approach" where the user plays a central role for achieving the learning steps.

1 Introduction

Case-based reasoning (CBR) is a reasoning paradigm which consists in solving new problems by adapting the solutions of previously solved problems. The CBR cycle is constituted of five steps: elaborate, retrieve, reuse, revise and retain. Each step is of particular importance in the resolution of the problem and involves specific knowledge.

In CBR, problem-solving experiences constitute basic knowledge units: the cases. During a reasoning cycle, cases are stored in a case-base which may possibly be reorganised. The storage of a solved case is considered as the most traditional approach to CBR learning. Stored cases can be used in later reasoning cycles and gradually improve the system's abilities.

Case-based reasoning is particularly well suited to situations in which domain theory is weak or not easy to formalise. CBR systems have long been considered as interesting alternatives to knowledge-based systems, since, in theory, they require a smaller knowledge engineering effort to become usable in real world domains. It has even been argued that CBR was a solution to the bottleneck of

knowledge acquisition since it is easier to collect a number of cases than to build a knowledge base.

However, CBR does not avoid completely the need for a knowledge base and one has to face the knowledge acquisition problem. In fact, CBR systems also rely on other types of knowledge containers to reason on cases: domain ontologies, similarity measures and adaptation knowledge.

Similarity knowledge is used to remember the relevant cases and adaptation knowledge is used to adapt the solutions of stored cases. Experience shows, however, that similarity and adaptation knowledge available are difficult to turn into models, being vague or incomplete, and furthermore, they may evolve. It is therefore advisable to propose tools enabling to acquire and/or learn this knowledge. This would allow us to refine and improve knowledge as the system is being used.

This raises the issue of the management of the knowledge base of a CBR system from its design to its implementation and maintenance. In this paper we propose to view CBR from a knowledge management perspective. First, we consider the problem of knowledge management during the design and use of CBR systems, then we analyse the reasoning cycle, highlighting the various types of knowledge involved in each step. In particular, we will show that it is very difficult to formalise similarity and adaptation knowledge as they evolve with time. After discussing the close link between these two kinds of knowledge units, we show how the different learning approaches can make use of such a link. We present the model of adaptation by substitution on which we base ourselves and we put forward several scenarios for learning adaptation knowledge. We will emphasise the main role the user has to play in this process.

2 CBR knowledge: a typology

Case-based reasoning systems are knowledge-based systems (KBS) which, if we follow Richter's proposition [18], make use of four distinct knowledge sources: domain description vocabulary, cases, similarity knowledge and knowledge of solution transformation which we call adaptation knowledge.

2.1 Knowledge management in CBR systems

We can distinguish several phases in the life cycle of a CBR system: design, production and maintenance.

During the system's design and realisation phase, the designers define –in agreement with domain experts– problem solving methods to be used. An important engineering effort must be made to build the system's knowledge bases, define an initial base of solved cases, describe domain knowledge and formalise similarity and adaptation knowledge. The system can also be used with known cases to instantiate the case-base with examples and provide a starting point for reasoning. The issue of the knowledge representation formalism is also addressed at that time. The main actors of that phase are of course the experts, who are

true vectors of domain knowledge, as well as the designers who facilitate the passage from the knowledge level to the symbol level [17].

During the production phase, the system is used to solve –or help solving– new problems. The reasoning cycle carrying out this task is examined in the following section. Problems may be posed to the system by users or system experts. Interactions between the users and system take place at the beginning and the end of the cycle, but also during the production phase, as we shall see later on. As soon as the system is used, a maintenance procedure must ensure the evolution of the initial knowledge base. At the end of each problem solving step, the newly solved case is stored in the case-base to be re-used later on. As a result, there is a gradual increase in the size of the case-base and this highlights the need to organise and maintain it throughout the life of the system. To deal with this issue, several works propose indexing or classification techniques to facilitate the retrieval of stored cases. Other approaches are based on strategies of retention and forgetting [19] to retain only the more relevant cases and avoid overloading the case-base. Among all these various approaches, some occur during the retain phase whereas others are done outside the production phase. Finally, let us note that maintenance operations can be done by the system itself or by the expert user. The system can also ask the expert for assistance.

2.2 Reasoning cycle

As we mentioned earlier, CBR solves new problems by remembering and adapting already solved problems. The CBR cycle is composed of five steps:

- *Elaborate.* This step is not included in the classic CBR cycle introduced in [1]. Even if it was implicitly done in several systems, it was firstly explicitly mentioned in [8]. During this step, the information necessary to the resolution of a problem are collected and structured to form a new case: the target case. The system solicits the user or its outside environment (databases, information systems) to obtain the information needed to continue its reasoning.
- *Retrieve.* The retrieval step consists in searching the case-base for one or several solved cases deemed to be similar to the target case. The selection of a similar case is based on a similarity measure. Some systems use several stored cases and combine them to solve a problem, but most of the time, only one case is used to continue the process. It is called the source case. The selection of the source case can be done either by the system or by the user.
- *Reuse.* This step enables the system to solve the target case by adapting the selected source case solution which is first copied, then possibly adapted to satisfy the requirements of a given problem. The adaptation rests on adaptation knowledge which can be of different forms according to the various systems.
- *Revise.* The solution proposed by the system may not suit the user, or, once it has been applied, might be unable to solve the given problem. The user has therefore the opportunity to modify, amend or even refuse the proposed

solution. The revise step allows one to identify the possible causes of failures and to propose further adaptations to obtain a satisfactory solution: the revised case. This step is the basis of the learning process, leading to the improvement of existing adaptation knowledge and giving rise to new adaptation knowledge.
- *Retain.* Traditionally, the retain step is considered as the step during which the case-base is enriched by the revised target case. This retention implies an update of indexes used to retrieve the cases and sometimes a maintenance process is needed to reorganise the case-base. But the retention step is also a means to learn other types of knowledge. Indeed, it is during this step that additional knowledge can be acquired in various ways.

A concise but complete overview of the work in each of theses steps can be found in [16].

2.3 Knowledge acquisition and learning

The study of the reasoning cycle in CBR has highlighted the diversity of knowledge involved in this process. Table 1 proposes a synthesis. For each knowledge unit, the following are defined: the various forms of knowledge, the steps during which this specific knowledge can be acquired and the methods used for its acquisition and learning.

One notes that, except for domain knowledge, it is rare to find in the system knowledge that can be formalised *a priori*. In fact, even if it is possible to represent similarity or adaptation knowledge in the initial knowledge base, this knowledge remains vague or uncertain and must be improved during the system use.

2.4 About similarity and adaptation knowledge

The relation between similarity and adaptation knowledge needs to be studied. Adaptation is one of the most difficult step of CBR and therefore any effort to facilitate it is useful.

In [20], Smyth introduces the concept of adaptation-guided retrieval. He argues that the sources cases most similar to the target case are not always the easiest to adapt, in particular when the similarity rests on surface features. Retrieval must therefore search not only for similar cases, but especially easily adaptable cases.

In the same light, Leake [13] suggests that a good retrieval of a case reduces the adaptation effort. In fact, the traditional semantic similarity measures may lead to bad results since they occasionally retrieve source cases which are certainly very like the target case, but are difficult or even impossible to adapt. This remark shows the limitations of similarity measures with regard to the whole reasoning process. Leake therefore proposes to include in the similarity measure a notion of adaptation cost to make it more pertinent. Hence, in this approach, the evaluation of the similarity between the target case and the various source

Knowledge type	Form of Knowledge	Acquisition Step	Acquisition/Learning Approaches
Case	Problem part and solution part (descriptor sets), Reasoning traces (steps from problem to solution)	Design: use of known cases to train the system Retain: storage of cases solved during the reasoning cycle	Classification Indexing
Domain knowledge	Concepts: properties and relations with other concepts, Rules, Dependencies	Initial acquisition relatively easy if domain theory is weak	Description and modelling by the expert
Similarity knowledge	Predefined numeric measures, Empirical measures based on descriptors comparison, More complex measures taking into account adaptability, Weights, Similarity paths, Etc.	Initial acquisition not easy, no design methodology, Retrieval: acquisition of new knowledge and improvement of existing knowledge	Modelisation by the expert, Introspective learning, Automatic symbolic learning (data mining, neural networks . . .) Etc.
Adaptation knowledge	Adaptation rules, Adaptation operators, Adaptation cases		

Table 1. CBR knowledge typology

cases takes place in two steps: first, a classic similarity measure is evaluated by comparing the cases, then, the most similar retrieved cases are ranked according to their adaptability.

Lieber on the other hand, proposes an adaptation approach making use of similarity paths. Behind this notion lies the idea of a decomposition of adaptation into simpler adaptation sub-tasks. To expose similarities between two complex problems, it is often necessary to use domain knowledge. The approach proposed in [14] aims to decrease the difficulty of adaptation by increasing the similarity between the problems, which involves decomposing a complex problem into several simpler sub-problems. Intermediary problems are linked together by relations. Each relation corresponds to a specific adaptation enabling the passage from one problem to another. A similarity path is therefore composed of a linear sequence of intermediary problems linked together by relations. The first step of adaptation which involves the building of the similarity path can take place during the retrieval step. All that remains to do during the second adaptation phase is to calculate the elementary adaptations corresponding to each step of the similarity path. In [15], the authors demonstrate how, in a concrete case (the treatment of breast cancer), the notion of similarity paths may appear as a tool to assist in the acquisition and creation of models of adaptation knowledge.

These three examples highlight clearly the dual relation existing between similarity knowledge and adaptation knowledge. More generally, it is not advisable to consider the different stages of CBR separately and independently from one another, but rather as contributing to a common objective. The elaboration stage, for example, aims to improve retrieval by establishing suitable descriptors. In the same way, the retrieval step tends to facilitate adaptation by using an adaptability criteria to select a source case. A case's adaptability must therefore be taken into account in the retrieval step. This is why learning adaptation knowledge if of particular importance. In the following part, we consider the strategies for knowledge learning.

3 Learning adaptation knowledge

3.1 Learning strategies

Adaptation is studied according to three main directions: unifying approaches which propose general adaptation models; catalogues of adaptation strategies applicable to several domains; and methods for acquiring adaptation knowledge which, in a particular domain, try to highlight the general principles to explain the adaptation process. A distinction is made between different approaches of acquisition of adaptation knowledge: knowledge light approaches (according to [21]) consist in re-using knowledge available in the system to infer new knowledge while other approaches try to acquire new knowledge by using the interface between the system and its environment. The former approaches take place outside the problem solving phase, whereas the latter take place during the solving process and therefore present numerous possibilities of interactions with the user.

The approach presented in [7] can be classified in the first category: it consists in determining pairs of cases and using differences between their attributes to improve adaptation rules. The adaptation rules thus created are then refined and generalised. Each rule has associated measures of confidence calculated according to its degree of generalisation.

On the same line of thought, [15] propose an approach of knowledge learning based on a particular search technique called *frequent pattern extraction*. The main idea is to use the differences between cases taken in pairs. Indeed, these differences can be interpreted as the result of an adaptation effort. It is then possible to deduce some adaptation knowledge.

Among the approaches of the second category, we may note that of [12]. According to Leake, knowledge learning takes on several forms. At first, Fox and Leake proposed an approach using introspective reasoning to give systems the possibility of learning new knowledge enabling them to improve their overall efficiency. In [3], the authors apply introspective reasoning to improve indexing of cases. They extend this approach to the other stages of CBR and in particular, to the adaptation stage. In the DIAL system, the proposed reasoning focuses mainly on case adaptation and the learning of various types of knowledge is more or less linked to this stage. [11] considers case adaptation as a process combining a group of abstract transformations with memory search strategies. A trace of the actions taking place during an adaptation phase is stored and constitutes an adaptation case. Thus, when a new case is encounter, it can be adapted either from scratch or based on the use of adaptation cases and introspective reasoning [10]. Adaptation knowledge is acquired via a CBR cycle within the main CBR cycle. This approach of learning of adaptation knowledge enables an ongoing refining of adaptation strategies by adapting adaptation cases [9]. Leake also proposes to evolve similarity knowledge as adaptation knowledge is being learnt. The idea is to use knowledge contained in adaptation cases to predict adaptation costs. The proposed method is called RCR (Re-application Costs and Relevance). It enables us to assess the difficulty of adapting a problem and brings therefore further detail to the similarity measure [13].

One of the drawback of the approaches that aim to use knowledge already available in the system to infer new adaptation knowledge is their limitation to the vocabulary of the case-base. They do not allow one to infer knowledge that is not explainable using the existing knowledge of the application. Furthermore, they only give the user a minor role which consist in validating the inferred knowledge. On the contrary, approaches which allow the learning of knowledge during the reasoning process provide the possibility of adding new knowledge to the system and the opportunity for the user to play an actual role in the process. We stick to the second approach and our wish is to place the user at the centre of the learning process so that he can simultaneously play an active role in the solution of the problem and in the learning of adaptation knowledge.

3.2 Learning to improve adaptation

In this work, we base ourselves on a formalisation of adaptation by substitution. The framework of our study was set out in [4]. Adaptation knowledge is modelled as a set of dependencies. The dependencies we use are similar to those used in analogical reasoning [5], [6].

After presenting the notions and notations used, we identify the sources as well as the knowledge units targeted by the learning process (learning targets) and we propose some learning strategies. We illustrate the various strategies in the domain of the assessment of the price of a second-hand motor vehicle. In this problem, cases are vehicles characterised by some features as well as by their selling price on the used car marketplace. The aim is then to calculate, given a certain number of dependencies, the estimated selling price of a new vehicle according to the set of known descriptors.

In our approach, we make a difference between acquisition and learning. We speak of learning in reference to machine learning, that is to say when the system is able to learn on his own, using knowledge already available. We use acquisition when knowledge comes from outside the systems. Thus acquisition approaches often involve a user which interacts with the system.

An adaptation model The adaptation model proposed in [4] is briefly described below. Our hypothesis is that a case is composed of a problem part and a solution part. It is possible to represent a case using a set of descriptors. A descriptor consists in a name and a value. We note:

- d as descriptors of problem parts and D as descriptors of solution parts,
- $\{d_i^s\}_{i=1..n}$ as descriptors of a source problem and $\{D_j^s\}_{j=1..N}$ as descriptors of its solution,
- $\{d_i^t\}_{i=1..n}$ as descriptors of a target problem and $\{D_j^t\}_{j=1..N}$ as descriptors of its solution calculated by adaptation.

In two given cases, the retrieval step estimates the differences between the pairs of problem descriptors (Δd_i). The adaptation is based on a group of relationships between the problem and its solution called *dependencies* which indicates that some problem descriptors have an influence upon some solution descriptors. Thereby, adaptation knowledge is mainly constituted of dependencies.

A dependency is a triple $(d_i, D_j, \mathcal{I}(D_j/d_i))$ indicating the variation of the solution descriptor D_j in relation to the problem descriptor d_i. $\mathcal{I}(D_j/d_i)$ is called influence function and indicates how to calculate the variation of D_j knowing the variation of d_i. Adaptation combines these influence functions $\mathcal{I}(D_j/d_i)$ with the differences Δd_i between problem descriptors to estimate the variations ΔD_j. These variations are applied to source solutions descriptors D_j^s in order to obtain target solution descriptors D_j^t.

Dependencies are therefore essential as they contain, through influences, adaptation knowledge. Dependencies are domain knowledge which must be assessed at the beginning of the system's design to enable its reasoning. But this

knowledge remains empirical and uncertain, it must therefore be refined through the use of the system. This remark is justified by the very existence of a revision step in the CBR cycle. Indeed, if adaptation knowledge was complete, the system would be able to guarantee that the adaptation result is correct.

In the adaptation model presented here, dependencies also explicit the close relationships between similarity and adaptation knowledge. They link problem and solution descriptors thus highlighting the role they play in the evaluation of similarity.

Learning targets Using the formalisation of adaptation presented before, we have identified three main adaptation knowledge learning targets: influence functions, dependencies and classes of problems.

Influence functions Influence functions allow one to calculate the variation of a solution descriptor according to the variation of a problem descriptor. They can be of various types and of variable complexities but, most of the time, they can be assimilated to numeric functions. These functions, even if they are assessed during the system's design, can be refined throughout the problem resolution experiences. For example, it is possible to adjust function applicability thresholds or to modify some parameters to make functions more and more precise.

Dependencies During the resolution of a new problem, an adaptation failure can points out an unknown dependency. Indeed, it is likely that an experience shows that a problem descriptor ignored until now has an influence, under specific conditions, on a solution descriptor. In such a situation, a new dependency must be elaborated and associated with a suitable set of dependencies. It is also possible that several dependencies put in relation a unique problem descriptor with a unique solution descriptor but using different influence functions. In this case, another problem descriptor should be available. This descriptor will be used to select the dependency and, as a result, the influence to use. It is the responsibility of the elaboration step of identifying these descriptors.

Classes of problems A class of problems correspond to a group of problems that can be solved using similar adaptation knowledge. Concretely, a class of problems is composed of a set of dependencies necessary to solve a particular kind of problem. Thus, discovering a new class of problems is equivalent to identify a new category of problems unknown until now and consequently impossible to adapt. Identifying a new class of problems is also a way to acquire adaptation knowledge.

Knowledge acquisition and learning methods An adaptation failure in a CBR system reflects a lack of adaptation knowledge. It's during the revise step that this failure is observed: the modifications made by the user on the solution or the inability of the system to find a suitable solution to the problem are good indicators of this situation. The revise step is thus, most of the time, the starting

point of the acquisition and learning process. In the following, we describe some methods combining acquisition and learning techniques applicable in the CBR field.

Exploiting the revise step The adaptation process, using the influence functions, estimates differences between solution descriptors. We note these differences: $\Delta_{adapted}D_j$. Applied to source solution descriptors, these differences allow one to estimate the values of the target solution descriptors (D_j^t). These differences represent the modifications made by the system.

Other differences are produced by the user during case revision. They are noted as $\Delta_{revised}D_j$. They allow one to quantify the difference between one target solution descriptor D_j^t before and after the user's revision. In consequence, these differences represent the adjustment made by the user. We note D_j^{tr} as target solution descriptor j after the revise step.

In this model $\oplus$ (resp. $\ominus$) is an abstract operator which should be defined according to the types of the descriptors. For simplicity sake, we will assimilate this operator to the numeric operator $+$ (resp. $-$) in our example. Thus, we have:

- $D_j^t = D_j^s \oplus \Delta_{adapted}D_j$, and
- $\Delta_{revised}D_j = D_j^{tr} \ominus D_j^t$

These notations are used in the figure 1 which presents relationships between the various descriptors considering in particular the retrieve and the reuse steps. D_j^{tr} are produced by the user: as soon as the system knows them, it is able to evaluate $\Delta_{revised}D_j$.

The differences $\Delta_{revised}D_j$ bring to light problems on the influence functions used to infer the values of the descriptors. Observing such differences can lead to the trigger of a learning process. Indeed, if a solution has been revised by the user before its storage, it is possible to exploit the differences represented by $\Delta_{adapted}D_j$ and $\Delta_{revised}D_j$ during a learning process.

An influence function is characterised by its parameters as well as by thresholds indicating domains on which the function can be applied. Studying $\Delta_{revised}D_j$ and $\Delta_{adapted}D_j$ can allow one to refine both of these elements.

Retrieve process on the solutions Another possibility to acquire adaptation knowledge, inspired by [9], consists in doing a *retrieve step* on the revised source solutions stored in the case-base and to classify the retrieved cases according to their similarity with the revise target solution. If the better case, from the solution point of view, does not match with the source case used to solve the target problem, then we can suppose that one or more dependencies used during the retrieve step were incorrect or incomplete and have to be adjusted.

We believe that various methods can allow the acquisition and learning of adaptation knowledge in this specific situation. Several ideas can be explored: applying an introspective reasoning to do a comparison of descriptors in order to deduce modifications to be made on influence functions; setting a cooperative environment to allow the user to specify on his own how dependencies have to be corrected; etc.

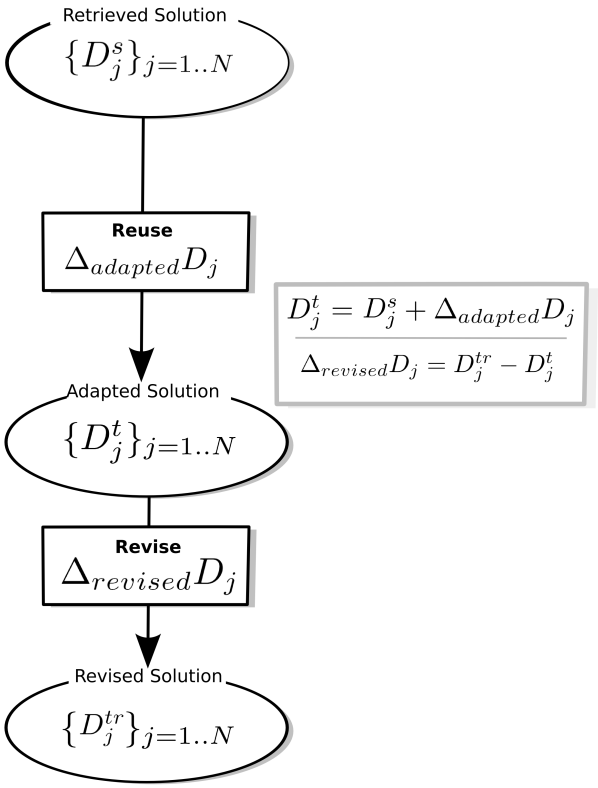

Fig. 1. Relationships between solution descriptors. This figure presents relationships and differences between solution descriptors during reuse and revise steps of the CBR cycle.

Replaying the reasoning cycle with the user If the revise step does not allow one to obtain a satisfactory solution to the current problem, it may then be useful to implicate the user in the reasoning cycle. The system and the user will then try to solve the problem together. In order to do this the system will provide an *assistance* to the user. This assistance can consist of a presentation of the knowledge used and of an explanation of the system's reasoning process. Allowing the user to specify or complete the knowledge used to solve the problem will certainly lead to a more satisfactory solution.

We also believe that it is possible to acquire and/or learn adaptation knowledge by exploiting a trace of the user's actions. This knowledge can certainly be represented as adaptation cases. Such interactive approaches enable one to

discover new classes of problems and even to guide the classification of a given problem into a suitable class of problems.

Acquisition and learning processes: a scenario As a synthesis, figure 2 presents various possible learning situations as well as applicable methods in each situation. We want to insist on the fact that is advisable to allow a cooperation between the system and the user at any time and not only after a reasoning failure.

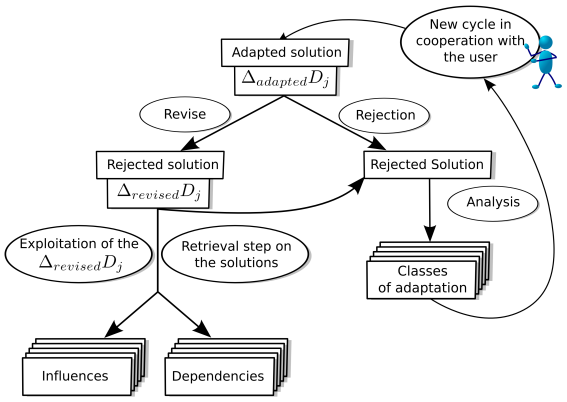

Fig. 2. Knowledge acquisition and learning process.

Finally, it is possible to draw a link with data mining approaches that can advantageously complete the approaches introduced before. For example, [15] use data mining techniques to help the discovery of new possible dependencies. In this work, the authors also use theses techniques to check the applicability of an influence function to some known cases.

Illustration through an example This example comes from a well known domain: the used cars selling marketplace. The problem is to estimate the price of a car knowing some of its characteristics and having experience in the form of cases stored in a case-base. A case is a car description composed of various descriptors. One of these descriptors is the price of the car: the price is known if the case is solved. In this section we briefly illustrate the concepts introduced before on this problem.

We assume that a car is described by several descriptors: mileage, age, power, colour, type of car (private car or collector car), price, etc.

We first consider the influence functions. A linear influence function allows one to compute a price variation of a car considering a variation of its mileage

in comparison with a reference car: this is a simple problem. A simple numeric function indicates that a price variation of one mile induce a variation of .01 euro. Thus a difference of 30 000 miles between two cars will implies a difference of 300 euros between their respective prices. It is possible to learn adaptation knowledge by refining some of the function parameters: for example, the value of the coefficient can be adjusted. The thresholds of the function can also be modified: for example, we can learn that if the car is less than 300 miles, the influence function is not applicable anymore.

The figure 3 presents an influence function in the domain of the simplified example we use: evaluating the price of a used car. In this domain, adjusting a dependency can be done by modifying an influence function or by discovering a new dependency: for example, the fact that the power has implications in the evaluation of a car price.

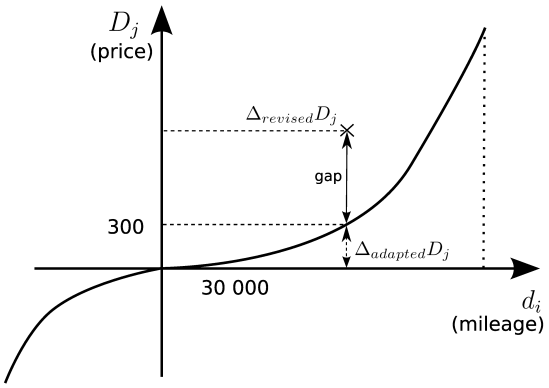

Fig. 3. Graphical representation of a part of an influence function. We can see that the function is only defined for a particular range of Δd_i values (mileage values). This means that the case is not adaptable in this domain. This figure also represents the difference $\Delta_{adapted}D_j$ and $\Delta_{revised}D_j$ to illustrate the possibility of an adaptation knowledge learning.

Let's suppose that we have learnt a new dependency: the price of a car depends not only on its mileage but also on its age. We now need to use two dependencies to solve the problem. This is an extremely simple example of the dependencies we can learn.

In this domain, we can consider that the methods used to estimate the price of a private car and those used to estimate the price of a collector car are not the same. These two problems correspond to two different classes of problems.

4 Conclusion

In this paper we have drawn up an overview of the different kind of knowledge involved in CBR allowing to characterise its reasoning process from the knowledge point of view. We have shown in what extent CBR in general and the learning of adaptation knowledge more precisely could take benefit of a unification of the similarity and adaptation knowledge. Then, based on an adaptation model using the dependency concept, we have identified knowledge units to be learnt and suggested several learning scenarios which have been illustrated through an simple example. Currently, an implementation of these ideas is being achieved using the JColibri tool [2], a framework for prototyping of CBR systems. There are several perspectives to this work. At this time, research is ongoing for setting an experimentation protocol. It aims at validating learnt knowledge and quantifying the global enhancement of the system's competence obtained by the learning scenario. This experimentation can serve as a basis for a comparative study with other approaches based on machine learning techniques applied to CBR. During our first experiment, we have limited the study to simple dependencies, i.e. where a single problem descriptor has an influence on a single solution descriptor. Next, we will have to take into account the most general case where a single solution descriptor is influenced by several problem descriptors. Furthermore, we have also limited the study to the case where dependencies are numerical functions. We have to study the generalisation of this approach to complex cases, i.e. when some descriptors are symbolic.

5 Acknowledgements

The authors would like to thank the referees whose remarks and comments were very helpful to improve this paper.

References

1. Aamodt, A. and Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AICOM **7**, pp. 39-59.
2. Bello-Tomas, J.J., Gonzalez Calero, P. and Diaz-Agudo, B.: JColibri: An Object-Oriented Framework for Building CBR Systems. European Conference on Case-Based Reasoning 2004, (2004).
3. Fox, S. and Leake, D.B.: Using Introspective Reasoning to Guide Index Refinement in Case-Based Reasoning. Sixteenth Annual Conference of the Cognitive Science Society, Atlanta, GA, (1994), pp. 324-329.
4. Fuchs, B., Lieber, J., Mille, A. and Napoli, A.: Towards a unified theory of adaptation in Case-Based Reasoning. Proceedings of the third International Conference on Case-based Reasoning, ICCBR-99, Lecture notes in Artificial Intelligence, Germany: Springer Verlag, (1999).
5. Gentner, D. and Forbus, K.: MAC/FAC: A model of similarity-based retrieval. Thirteenth Annual Conference of the Cognitive Science Society, Hillsdale, NJ: Lawrence Erlbaum, (1991), pp. 504-509.

6. Gick, M. L. and Holyoak, K.J.: Analogical problem solving. Cognitive Psychology, **12**, (1980), pp. 306-355.
7. Hanney, K. and Keane, M.T.: Learning Adaptation Rules from a Case-Base. Proceedings of the Third European Workshop on Advances in Case-Based Reasoning, Lecture Notes In Computer Science, (1996).
8. Herbeaux, O. and Mille, A.: ACCELERE: a case-based design assistant for closed cell rubber industry. Knowledge-Based Systems, **12**, (1999), pp. 231-238.
9. Leake, D.B.: Learning Adapatation Strategies by Introspective Reasoning about Memory Search. AAAI-93 Workshop on Case-Based Reasoning, AAAI Press, Menlo Park, CA, (1993), pp. 57-63.
10. Leake, D.B.: Becoming an Expert Case-Based Reasoner : Learning to Adapt Prior Cases. Eighth Annual Florida Artificial Intelligence Research Symposium, (1995), pp. 112-116.
11. Leake, D.B., Kinley, A. and Wilson, D.: Acquiring Case Adaptation Knowledge : A Hybrid Approach. Proceedings of the Thirteenth National Conference on Artificial Intelligence, AAAI Press, Menlo Park, CA, (1996).
12. Leake, D.B., Kinley, A. and Wilson, D.: Multistrategy Learning to Apply Cases for Case-Based Reasoning. Third International Workshop on Multistrategy Learning, AAAI Press, Menlo Park, CA, (1996), pp. 155-164.
13. Leake, D.B., Kinley, A. and Wilson, D.: Case-Based Similarity Assessment: Estimating Adaptability from Experience. Fourteenth National Conference on Artificial Intelligence, AAAI Press, Menlo Park, CA, (1997), pp. 674-679.
14. Lieber, J.: Reformulations and Adaptation Decomposition. International Conference on Case-Based Reasoning - ICCBR'99, LSA, University of Kaiserslautern, Munich, Germany, (1999).
15. Lieber, J., d'Aquin, M., Bey, P., Napoli, A., Rios, M. and Sauvagnac, C.: Acquisition of Adaptation Knowledge for Breast Cancer Treatment Decision Support.9th Conference on Artificial Intelligence in Medicine in Europe2003 - AIME 2003, Protaras, Chypre, (2003).
16. Lopez de Mantaras et al.: Retrieval, reuse, revision and retention in case-based reasoning. The Knowledge Engineering Review, (2005).
17. Newell, A.: The Knowledge Level. AI, **19**(2), (1982), pp. 87-127.
18. Richter, M.M.: Classification and Learning of Similarity Measures. Studies in Classification, Data Analysis and Knowledge Organisation, Springer, (1992).
19. Smyth, B. and Keane, M.T.: Remembering To Forget : A Competence-Preserving Case Deletion Policy for Case-Based Reasoning Systems. IJCAI, (1995), pp. 377-383.
20. Smyth, B. and Keane, M.T.: Adaptation-Guided Retrieval: Questioning the Similarity Assumption in Reasoning. Artificial Intelligence, **102**(2), (1998), pp. 249-293.
21. Wilke, W., Vollrath, I., Althoff, K. D. and Bergmann, R.: A Framework for Learning Adaptation Knowledge Based on Knowledge Light Approaches. Adaptation in Case-Based Reasoning: A Workshop at ECAI 1996, Budapest, (1996).