MORGAN & CLAYPOOL PUBLISHERS

Case-Based Reasoning

A Concise Introduction

Beatriz López

Synthesis Lectures on Artificial Intelligence and Machine Learning

Ronald J. Brachman, William W. Cohen, and Peter Stone, *Series Editors*

Case-Based Reasoning

A Concise Introduction

Synthesis Lectures on Artificial Intelligence and Machine Learning

Case-Based Reasoning: A Concise Introduction
Beatriz López
2013

Answer Set Solving in Practice
Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub
2012

Planning with Markov Decision Processes: An AI Perspective
Mausam and Andrey Kolobov
2012

Active Learning
Burr Settles
2012

Computational Aspects of Cooperative Game Theory
Georgios Chalkiadakis, Edith Elkind, and Michael Wooldridge
2011

Representations and Techniques for 3D Object Recognition and Scene Interpretation
Derek Hoiem and Silvio Savarese
2011

A Short Introduction to Preferences: Between Artificial Intelligence and Social Choice
Francesca Rossi, Kristen Brent Venable, and Toby Walsh
2011

Human Computation
Edith Law and Luis von Ahn
2011

Trading Agents
Michael P. Wellman
2011

Visual Object Recognition
Kristen Grauman and Bastian Leibe
2011

Learning with Support Vector Machines
Colin Campbell and Yiming Ying
2011

Algorithms for Reinforcement Learning
Csaba Szepesvári
2010

Data Integration: The Relational Logic Approach
Michael Genesereth
2010

Markov Logic: An Interface Layer for Artificial Intelligence
Pedro Domingos and Daniel Lowd
2009

Introduction to Semi-Supervised Learning
XiaojinZhu and Andrew B.Goldberg
2009

Action Programming Languages
Michael Thielscher
2008

Representation Discovery using Harmonic Analysis
Sridhar Mahadevan
2008

Essentials of Game Theory: A Concise Multidisciplinary Introduction
Kevin Leyton-Brown and Yoav Shoham
2008

A Concise Introduction to Multiagent Systems and Distributed Artificial Intelligence
Nikos Vlassis
2007

Intelligent Autonomous Robotics: A Robot Soccer Case Study
Peter Stone
2007

Case-Based Reasoning: A Concise Introduction

Beatriz López

www.morganclaypool.com

ISBN: 9781627050074 paperback
ISBN: 9781627050081 ebook

DOI 10.2200/S00490ED1V01Y201303AIM020

A Publication in the Morgan & Claypool Publishers series
SYNTHESIS LECTURES ON ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Lecture #20
Series Editors: Ronald J. Brachman, *Yahoo Research*
William W. Cohen, *Carnegie Mellon University*
Peter Stone, *University of Texas at Austin*
Series ISSN
Synthesis Lectures on Artificial Intelligence and Machine Learning
Print 1939-4608 Electronic 1939-4616

Case-Based Reasoning

A Concise Introduction

Beatriz López
University of Girona

SYNTHESIS LECTURES ON ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING #20

MORGAN & CLAYPOOL PUBLISHERS

ABSTRACT

Case-based reasoning is a methodology with a long tradition in artificial intelligence that brings together reasoning and machine learning techniques to solve problems based on past experiences or cases. Given a problem to be solved, reasoning involves the use of methods to retrieve similar past cases in order to reuse their solution for the problem at hand. Once the problem has been solved, learning methods can be applied to improve the knowledge based on past experiences. In spite of being a broad methodology applied in industry and services, case-based reasoning has often been forgotten in both artificial intelligence and machine learning books. The aim of this book is to present a concise introduction to case-based reasoning providing the essential building blocks for the designing of case-based reasoning systems, as well as to bring together the main research lines in this field to encourage students to solve current CBR challenges.

KEYWORDS

knowledge-based systems, problem-solving, reasoning, machine learning, learning from experiences, knowledge reuse

To Pep and Bernat.

Contents

Preface

Would you board a plane with a pilot who has no flying experience? Would you start a day's sailing under the command of a captain who has never sailed before? How many years of practical training in a hospital does a physician need before graduating? When solving complex problems, experience is mandatory. Case-based reasoning concerns the study of intelligent decision systems based on past experiences.

Case-based reasoning is a methodology for developing knowledge-based systems. Case-based reasoning means solving problems based on past experiences, remembering previous cases to guide the solution to current problems, and adapting past solutions to new problems. For example, an experienced sailor sets a course according to his experience of how the wind blows and the waves behave in the geographical area where he is used to sailing, in addition his knowledge of the origin and destination points. How to compute the course is something learned at nautical school; however, the sailor's experience is definitive in reaching the best outcome. Thus, using knowledge acquired from studying books, one could decide on course N, but the experienced sailor could decide NE because he remembers that in a similar situation the boat escaped from the influence of a cliff, the wind became stronger, and the speed faster.

In turn, the solution to new problems becomes a new experience from which to learn, in a never-ending cycle. To continue with the nautical example, a sailor who gets involved in a very strong wind learns that, under the particular circumstances of the present case, he would follow a N course instead of a NE one, or maybe a completely different one altogether.

Thus, case-based reasoning has two closely connected mainstays: problem solving and learning. Problem solving requires acquired knowledge, while solving a new problem provides insights for improving the problem-solving process. However, observe that the knowledge required for building a knowledge-based system following this methodology does not consist exclusively of the set of past experiences, but incorporates the knowledge to determine the mappings between the data presented in the current problem and the past experiences, compute similarities, and adapt solutions.

The key question is when the first experience acquired? The process of how a novice becomes an expert can be useful in answering that. A novice professional, such as a physician who has never practiced before, tends to follow the protocol learned at a university, step by step. Protocols are knowledge stored, usually in the form of books. Moreover, novices do not usually work alone, but under expert supervision. Such supervisors provide useful cases to guide the novice when he is not able to apply learned knowledge. The cases provided by supervisors are the novices' first cases. As the novice professional solves problems, he accumulates case experience. These cases help the novice in the solution of new problems, enabling him to progressively manage without relying on the books. In the end, he becomes an expert who rarely needs to consult a book to solve a problem.

When copying the way humans solve problems and learn, we can skip the first stage. We do not need to develop novice systems which solve problems from books; we can provide the system directly with experiences. This is the idea behind case-based reasoning.

Moreover, it is important to highlight that different problem solvers, placed in different contexts, acquire different expertise, as they are learning according to the context in which the problems are being posed. For example, an oncologist in Spain, where people follow a Mediterranean diet, learns about predicting breast cancer in a different way than an oncologist in the U.S., where food habits are completely different. Both oncologists become experts and are efficient in predicting breast cancer in their context; their experiences are different in the sense that the importance of health factors differs depending on the habits of the citizens and the environment. The expertise of both is continuously evolving and adapting to the problems they solve.

Case-based reasoning has been demonstrated to be useful in practice in many application fields. The reason for this could be that it is a lazy method: you can build a case-based reasoning system with data coming from a database, without the necessity of learning from the data-specific models or patterns, as other eager machine methods do. For example, records of patients treated can be the input of a case-based reasoning system to understand other new patients, without the need to learn an explicit illness model. Case-based reasoning postpones the main inductive work until problem solving. At the same time, since the solution of a problem depends on the particular context in which the decision has been made, such a lazy approach preserves all problem details. Generalized patterns are useful for the majority of the situations, while case-based reasoning is valid for all individuals, whenever a past, similar experience exists.

This introductory lecture reviews the elements required to develop a case-based reasoning system, with the aim of providing essential knowledge on this topic as well as bringing together the current challenges of the research community. Readers are encouraged to follow the bibliographical notes for further information about the introductory concepts provided in this book.

Beatriz López
March 2013

Acknowledgments

I have taken efforts in creating this book. However, it would not have been possible without the kind support and help of many individuals. I would like to gratefully acknowledge the enthusiastic discussion of Dr. Vicenç Torra about distances and aggregation measures. I thank Prof. Enric Plaza for the technical discussions on case-based reasoning fundamentals and assessing some key issues, without making him responsible for the final result. Postgraduates and master students of the University of Girona are thanked for numerous stimulating discussions. I would like to acknowledge the help of Albert, Ferran, and Pablo for their assistance with all types of technical problems regarding the editing of this document. I am grateful to the editor of this series for his understanding in the scheduling. I am in debt to the reviewers who provided useful comments to improve the manuscript.

Beatriz López
March 2013

CHAPTER 1

Introduction

Case-based reasoning (CBR) is a methodology for developing knowledge-based systems where the central issue is past cases or experiences. Case-based reasoning means solving problems based on past experiences, remembering previous cases to guide the solution to current problems, and adapting past solutions to new problems.

The case-based reasoning methodology incorporates four main stages, according to Agnar Aamodt and Enric Plaza, who provide the foundations of this form of knowledge-based system development. They define the process of solving a new problem or query case through the following four stages (Figure 1.1):

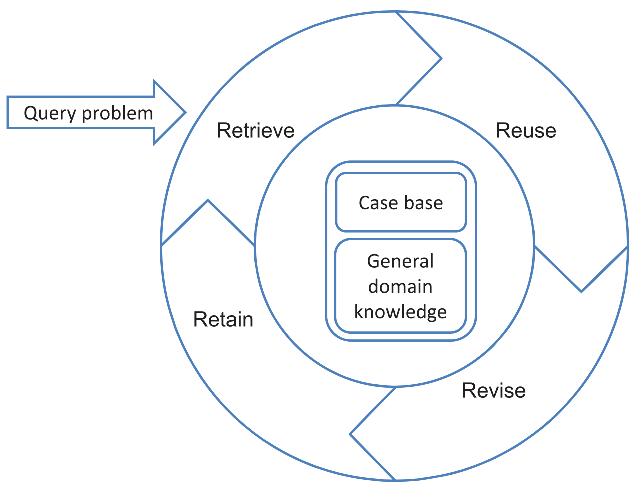

Figure 1.1: Case-based reasoning methodologies (based on Aamodt and Plaza [1994]).

1. Retrieve: past cases similar to the query case are retrieved; in this stage, inductive methods can be applied to find cases in memory in accordance with common features of the query case, meaning that learning from cases is postponed until using them, following a lazy approach.

2. Reuse: the solutions of the best (or set of the best) retrieved cases are used to construct the solution for the query case; again, new generalizations and specializations may occur as a consequence of the solution transformation.

Table 1.1: Example of sailing course prediction

Query case		Case C1		Case C2	
Origin	Base port	Origin	Base port	Origin	Base port
Destination	Maho	Destination	Maho	Destination	Maho
Wind speed	22	Wind speed	20	Wind speed	30
Wind direction	N	Wind direction	N	Wind direction	N
Wave height	0.2	Wave height	1.5	Wave height	2.5
Wave direction	S	Wave direction	S	Wave direction	S
Cloudscape	unknown	Cloudscape	cloudy	Cloudscape	sunny
Course	?	Course	W	Course	SW

3. Revise: the solution provided by the query case is evaluated and information about whether the solution has or has not provided a desired outcome is gathered.

4. Retain: when the new problem-solving experience can be stored or not stored in memory, depending on the revise outcomes and the CBR policy regarding case retention.

Observe that this process model is defined at the knowledge level; it identifies what is being processed and determines how the process is done. This approach is also known as the four-R approach, as the spelling of all of its phases starts with R.

Example 1.1 Sailing course prediction. Consider a CBR system for recommending a sailing course depending on origin, destination, and weather forecast. The current problem-solving situation is the one described in the query case in Table 1.1. Suppose that there are two cases in the memory, C1 and C2 as shown in the table, which are retrieved from the memory. In the retrieval stage, a similarity assessment is performed. Regarding the different components of the problem-solving descriptions, one can observe that the query case is closer to C1 (the wind speed differs by two units, wave heights by 1.3 units, but the cloudscape is dissimilar) than to C2 (the wind speed differs by eight units, wave heights by 2.3 units, and the cloudscape is again dissimilar). Thus, the system reuses the solution from C1 to solve the query case, by copying the solution (W). In the revise phase, the outcome of the solution is evaluated as satisfactory, since the target position (Maho) has been reached without incidences. In the retain phase, the new case, namely C3, is stored.

Observe that in the example, once the problem solving is finished, there are two cases with the same outcome (C3 and C1). An eager learning method could probably build a pattern for W sailing courses like the following:

Origin	Base port
Destination	Maho
Wind speed	[20, 22]
Wind direction	N
Wave height	[0.2, 1.5]
Wave direction	S
Cloudscape	{unknown, cloudy}

However, CBR does not pursue the generation of such patterns, but stores past experiences, infers similarities among them when problem solving. This is the lazy learning principle of the methodology.

Example 1.2 Continuing with the previous sailing course prediction example, suppose that the situation shown in the first column of Table 1.2 is posed as a new problem to be solved. The case-base contains the experiences of Table 1.1, but now with the outcome (W) of the first case (renamed as C3) known. The differences of the current case to the cases stored in memory are shown in Table 1.2. Assuming that we can sum up the differences to decide about the most similar case, then we have that this is C1. Thus, the solution of C1 (W) is provided to the query case. In the revise phase, however, the outcome of the solution provided is evaluated as undesirable, because passengers could become seasick. Possible reasons for that include the fact that sailing perpendicular to the waves could be worse than sailing backwards into the waves. A better solution is SW, provided by C2. In the retain phase, since the case has not been solved by the system successfully, the new case is stored in memory to improve problem-solving efficiency in the future. Moreover, some other learning mechanism could take place to avoid future failure, after analyzing the differences between C1 and C2 regarding the query case, as to consider that wave direction and height is more relevant for problem solving than other case information.

Problem solving and learning are then loosely coupled in CBR. Viewing problem solving from a constructivism point of view, cases are problem-solving states or instances. Thus, when a new

Table 1.2: A new problem in the example of sailing course prediction. C3 is the query case of Table 1.1

Query problem		C3	C1	C2
Origin	Base port	0.0	0.0	0.0
Destination	Maho	0.0	0.0	0.0
Wind speed	20	2.0	0.0	10.0
Wind direction	N	0.0	0.0	0.0
Wave height	1.7	1.5	0.2	0.8
Wave direction	E	1.0	1.0	1.0
Cloudscape	unknown	0.0	1.0	1.0
Course	?	W	W	SW

problem is posed, it is described as a state in the problem-solving space with an unknown solution, and problem solving consists of moving toward a state that contains the solution. Once the solution is known, learning from the experience of solving it is posed as a learning problem to the system.

All of the stages rely on a knowledge corpus which includes knowledge that could be general domain knowledge, tasks, and methods required to reason from cases—or specific, as past experiences or cases. Nowadays, two additional stages have also been considered in order to highlight the maintenance stage regarding such knowledge.

- Review: in this stage the knowledge in the system is analyzed according to different quality measures.

- Restore: modifies the knowledge according to certain operators.

1.1 CBR SYSTEMS TAXONOMY

The process described above at the knowledge level can be instantiated in different CBR systems. In doing so, different CBR system typologies can be considered according four criteria: knowledge source, function, organization, and distributiveness (see Table 1.3).

Table 1.3: CBR kind of systems. The kind of system that are mainly addressed in this lecture appear in bold

Knowledge source	Function	Organization	Distributiveness
Textual	**Classification**	**Sole**	**Single memory**
Structural	Recommendation	Multiple level	Multiple memories
Conversational	Tutoring	Hybrid CBR	**Single agent**
Temporal	Planning	Meta CBR	Multiple agents
Images	Monitoring		
	Knowledge management		

The first dimension is the *knowledge source*, which refers to the form of the past experiences or cases. Following that criterium, five main kinds of case-based reasoning are identified:

- Textual: when cases are text documents. There are large collections of documents that are easy to acquire. A typical example is a FAQ system.

- Structural: when cases are well defined according to a predefined vocabulary. For example, in a medical decision system in which the patients records are stored according to certain predefined variables (age, IMC, drug history).

- Conversational: when a case is iteratively defined through a user-system conversation. This is the case of a help desk center, where customers ask for support.

- Temporal: in which the information represented in the cases has, in an either implicit or explicit way, a temporal relationship (such as in traces or episodes). For example, in a game-playing system, when cases represent user histories in a game.

- Images: when CBR is used to support image interpretation, taking into account the different factors that influence such a process.

The second dimension is the *function* for which case-based reasoning is developed. The most popular applications of case-based reasoning are the following:

- Classification: in which the CBR system is used to predict classes or labels. This task is quite generic and two popular types of classification are the following:

  - Prognosis: when the prediction consists of two classes, positive or negative. For example, the prognosis about suffering or not suffering from an illness.

  - Diagnosis: when there is a discrete number of classes upon which to perform the prediction. For example: working normally, abnormally, or failure.

- Recommendation: when a product is recommended to a user on the basis of previous products acquired or her similarity to and interactions with other users.

- Tutoring: when dealing with the collection and assessment of exercises in a given discipline.

- Planning: when helping first-principle planners to be more efficient. In that case, the CBR system is called the second-principle planner, as it reasons at the planning level instead of the action level.

- Monitoring: when the CBR system predicts a deviation in the system behavior it is supervising.

- Knowledge management: where CBR matches perfectly, since knowledge management involves the use of information resources and knowledge assessment by remembering and applying experience.

Of course this list is not exhaustive.

The third dimension deals with *organizational* issues. Organizational criteria involve the combination of different case-based reasoning systems for solving a problem, the same as with other knowledge-based systems. According to that dimension, four classes of case-based reasoning are distinguished as follows:

- Sole approaches: when a single CBR is considered for problem solving.

- Multiple-level approaches: when several CBR systems are used, in a multilevel organization, to solve a problem. This is often the case with image interpretation tasks.

- Hybrid approaches: when CBR is hybridized with another problem-solving methodology. For example, CBR is used to diagnose an illness and a planning system to determine the illness treatment.

- Meta CBR: when another knowledge-based system, including a second CBR system, is used to reason about the best methods to be applied in each CBR stage according to the information in the query problem.

Regarding the hybrid dimension, it is important to highlight that CBR is a methodology, not a technology. So it is clear that several technologies may be required to implement the different stages for solving a problem. For example, k-nearest neighbor can be used to retrieve cases, and constraint satisfaction to adapt the solution. This means that several AI technologies are integrated in a CBR system to deploy reasoning like CBR does; but it should not be confused with hybrid CBR.

Finally, the last dimension is about *distributiveness*, which can be characterized by two criteria: 1) number of memories or case-bases a system has, and 2) how they are distributed for processing (one agent or system, or multiple agents in a multi-agent system). That characterization results in four types of case-based reasoning systems:

- Single memory, single agent

- Single memory, multiple agent

- Multiple memory, single agent

- Multiple memory, multiple agent

They are self-explanatory by their definition and are reviewed in Chapter 6.

1.2 FOUNDATIONAL ISSUES

CBR has its roots in four different disciplines:

- Cognitive Science (Analogical reasoning): Analogical reasoning is an area of study of artificial intelligence (AI) and cognitive science, with different models, and case-based reasoning is one of them. The cognitive psychologist aims to clarify the mechanisms underlying analogy, such as perception, memory organization, reminding, creativity, and self-awareness, in such a way that they become understandable. AI wishes to emulate analogical reasoning on computers, to produce more flexible systems. Moreover, CBR has been characterized as a particular case of analogical reasoning in which analogies are intra-domain, instead of inter-domain. CBR presents a challenge to other formal approaches in AI since it promotes a different kind of reasoning than deductive inference and emphasizes the role of memory in human cognition. Most of the methods that will be introduced in this lecture are influenced by this origin of CBR, as well as the cognitive sciences. Tversky's studies of similarity, which will be discussed in Chapter 3, are good examples of this.

- Knowledge representation and reasoning: Knowledge plays an important role in CBR as there are different types of explicitly represented structures needed for a system to perform reasoning, namely, domain, task, and method knowledge. The role of domain knowledge is discussed in Chapter 2 and the other two in Chapter 3.

- Machine learning: CBR is known as a lazy learning method as it postpones learning when problem solving. However, since there is knowledge explicitly represented in the system (including cases), CBR could integrate eager learning mechanisms for learning such knowledge and alleviate the knowledge acquisition problem, as well as maintain the knowledge. Chapter 4 focuses on the learning aspects involved in CBR.

- Mathematical foundations: Mathematical methods have influenced the retrieval stage of CBR in that they provide similarity measures which enable the retrieval of similar cases for solving query cases. Moreover, some mathematical theories have been used to formalize parts of CBR, to provide mechanisms for analyzing the inference properties of CBR systems. Chapter 3 explores similarity measures in depth, while Chapter 5 introduces some formal models for CBR. Mathematical foundations have some cross-cut topics with other disciplines such as utility theory and decision theory (operational research and economics) from which CBR borrows some ideas.

1.3 RELATED FIELDS

Case-based reasoning is a methodology but it does not prescribe any specific technique. The 4R stages method describes a process for solving problems and several techniques can be used to apply it. There are a variety of techniques which are usually employed to apply CBR, and sometimes they are confused with methodology. On the other hand, some AI fields, such as recommender systems, have applied CBR to achieve their goals; but they are not CBR. To clarify such cutting edges, several of the techniques and AI areas are reviewed below.

K-nearest neighbor (KNN) is a lazy learning technique in which induction is performed at run-time. It is a technique mainly used for pattern classification, so that a class is recognized by identifying the nearest neighbors to a query case and using the information about the class of those neighbors to provide the label or solution. This technique is broadly used in the retrieval and reuse CBR stages.

Instance-based learning, also a lazy learning technique, focuses on selecting good cases for classification, reducing storage requirements, tolerating noise, and learning attribute relevance. CBR also modifies cases and uses parts of cases during problem solving.

Information retrieval is a process that starts with a user's need for information and ends in a set of references to documents that should be relevant to the user's request. CBR can be applied to help the user define her requests. On the other hand, information retrieval can support CBR when a query is interpreted as describing a user problem and the documents obtained enable the user to solve it.

Pattern recognition is a multidisciplinary area that studies the operation and design of systems that recognize patterns in data, mainly based on either a priori knowledge or on statistical information extracted from the patterns. When patterns are complex, CBR could be used as a pattern recognition technique without the need for defining explicit patterns.

In an attempt to look for a unified model of induction, CBR has also been connected to Bayesian reasoning under the umbrella of statistical learning methods. In this scenario, cases are viewed as instantiations of random variables describing the domain and problem solving deals with the generation of hypotheses concerning how the domain works; see Chapter 5 for further details.

1.4 BIBLIOGRAPHIC NOTES

The most relevant paper is Aamodt and Plaza [1994]. Although there exists several previous works, this paper gathers for first time the methodological issues underlying CBR. Later on, the methodological aspects are reviewed and updated in Lopez de Mantaras et al. [2005], including the maintenance steps (Reinartz et al. [2001]). Another interesting work is Slade [1991], a short review in which the classical CBR system applications are compiled.

Reference books about CBR are Riesbeck and Schank [1989] (introducing the concept of case-based reasoning as dynamic memory), Kolodner [1993] (a shorter version was published as a journal article in Kolodner [1992]), and Leake [1996] (which is a collection of "recent" developments in 1996, when CBR was considered a mature field and also includes a tutorial). Regarding technology required to deploy CBR, Watson [1999] analyzes the synergies between these two topics (methodologies and technologies), and identifies hybrid and integrated approaches with other AI areas.

CBR strengths and weaknesses are discussed in Cunningham [1998].

For further information regarding recommender systems see the introductory book Jannach et al. [2010]. For a close analysis of the use of CBR in recommender systems, read Lorenzi and Ricci [2005] and Burke [2002]. Conversational CBR is described in Aha et al. [2001] and Weber et al. [2006], and see Shimazu [1998] for a textual CBR system example. An example of multi-layer CBR system for medical image interpretation is given in Grimnes and Aamodt [1996]. The classification of CBR systems according to distributiveness is given in Plaza and McGinty [2005].

There are a myriad of hybrid CBR systems, some examples are: CBR with rule-based systems (Bellazzi et al. [2002], López and Plaza [1997]), CBR and data mining (Aamodt et al. [1998]), and CBR with model-based reasoning (López-Arévalo et al. [2007]). A general architecture for problem solving and learning including CBR is PRODIGY, which is described in Veloso et al. [1995]. Marling et al. [2002] provides a survey on CBR integrations and hybrid CBR.

A closer look at foundational aspects is provided in Richter and Aamodt [2005]. Cognitive approaches to case-based reasoning are discussed in Ekbia [2008], and at the problem solving level in Newell [1972] and Langley and Rogers [2005]. Analogical problem solving and learning were introduced in Carbonell [1981] and Carbonell [1982] correspondingly; its relation with case-based reasoning is discussed in Falkenhainer et al. [1989]. The limitations of analogy are discussed in Holyoak and Thagard [1995], where the authors point out that "analogies should enhance thinking, not substitute for it." An updated review of analogy can be found in Holyoak and Morrison [2005], together with a comprehensive review of thinking and reasoning. These works should be read together with Stanfill and Waltz [1986] about memory-based reasoning. Gentner and Forbus [1991] presents similarity-based reasoning from a psychological point of view.

K-nearest neighbour classifiers are reviewed in Cunningham and Delany [2007]. For an introductory reading about instance-based learning, refer to Aha et al. [1991]. For deepening in lazy learning, read the special issue Aha [1997]. A good example of the application of CBR for information retrieval is given in Gronau and Laskowski [2003] and for pattern recognition in Chantaraskul and Cuthbert [2004]. See Gilboa et al. [2009] for a discussion of the Bayessian framework and a unified view with CBR.

A good collection of CBR application domains is in Watson [1997] and Watson [2003], the later including a knowledge management perspective (see also Watson [2001]).

Bergmann et al. [2003] provides a knowledge engineering perspective of CBR, with the goal of developing industrial applications. CBR is also being used in composite services (Limthanmaphon and Zhang [2003]) and workflow management in business (Weber et al. [2004]). One of the major areas of CBR application is the health sciences; Bichindaritz and Montani [2011] and Begum et al. [2011] are good reviews of CBR application in this domain. Image interpretation through CBR is analyzed in Perner [2001]. Another promising field is game industry where, for example, CBR offers a way of personalizing players' interactions and thus improving the overall player experience, as shown in Sharma et al. [2010].

CHAPTER 2

The Case-Base

The case-base contains the experiences and conforms to one of the four sources of knowledge required in a CBR as shown in Figure 2.1. They are the vocabulary, the case-base, the similarity measure and adaptation containers. The first, the vocabulary, contains the terms which support the others. The case-base comprehends what is in a case and how cases are organized. The similarity measure container contains knowledge to determine the similarity between two cases in the retrieval phase. The solution adaptation container contains knowledge to adapt past solutions to new problems in the reuse stage.

In addition to these containers, other knowledge can be considered, such as that required for case argumentation in the revision stage (feedback), or maintenance knowledge required in the review and retain stages. Developing and maintaining the contents of the CBR containers can be analyzed from the knowledge-level point of view, which offers the possibility of defining knowledge types according to how the system performs reasoning, that is, distinguishing between task knowledge, method knowledge, and domain knowledge (see Figure 2.2). Task knowledge models what to do (e.g., retrieve a case from memory), method knowledge shows how to do it (e.g., search the case base), and domain knowledge is the knowledge about the application domain that a method needs to be applied (cases, heuristics, rules, etc.). In this chapter we deal with domain knowledge, particularly on cases, and the next chapter particularizes about methods and tasks.

2.1 VOCABULARY

The vocabulary is the set of terms, words, or symbols in which cases are formulated. Case elements are defined based upon the vocabulary. Each term in the vocabulary has a meaning in the application domain. For example, the term "milk" is "a white fluid, rich in fat and protein, secreted by female mammals for the nourishment of their young." When changing the term from one language to the other, i.e., "leche" in Spanish, the meaning is preserved.

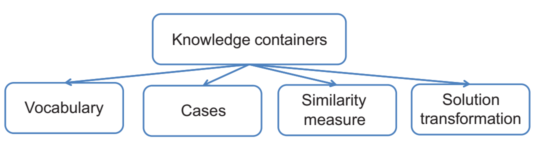

Figure 2.1: Knowledge containers (based on Richter [1995]).

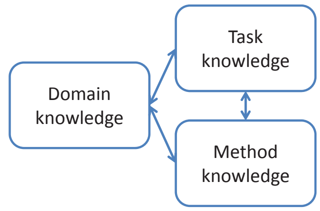

Figure 2.2: Knowledge types according to the knowledge-level point of view (Aamodt [2001]).

Nowadays, a vocabulary cannot be understood without the use of an ontology. An ontology contains the terms of the vocabulary and their relationships, so that the meanings of the terms are explicit. The most common relations used in ontologies are type-of, subtype, part-of, class and instance-of. Ontology development has been promoted by researchers who need to share information between systems, and case-based reasoning has taken advantage of it for enhancing relationships between case elements at the semantic level.

An ontology should be implemented in a language that enables its management. Description logic is currently the standard for representing ontologies based on logic.

Ontology provides an expressive framework for building structured representations of cases, enabling the semantic retrieval of cases. Moreover, when a logic representation formalism is used, it enables the detection of incoherences of the case-base organization, as illustrated in the following examples.

Example 2.1 Dairy products. Figure 2.3 shows the ontology of the different dairy products, milk and cheese among them. It is possible to distinguish a tighter relationship between mascarpone and quark than between mascarpone and mozzarella, because mascarpone and quark are kinds of fresh unripened cheese, while mozzarella is a stretched curd cheese. Suppose that there is a case C1 in the case-base that is related to quark, and another C2 related to mozzarella, and that there is a query problem about mascarpone. In the absence of an identic case about mascarpone, case C1, closest to mascarpone, is retrieved.

Example 2.2 Suppose now that there are two cases, C3 and C4, both with mascarpone data, but with different outcomes, e.g., positive and negative classes correspondingly. From the logic point of view, any retrieval based on the cheese data would be incoherent, since it leads to contradictory outcomes. So the cheese information should be avoided when organizing the cases in the case-base.

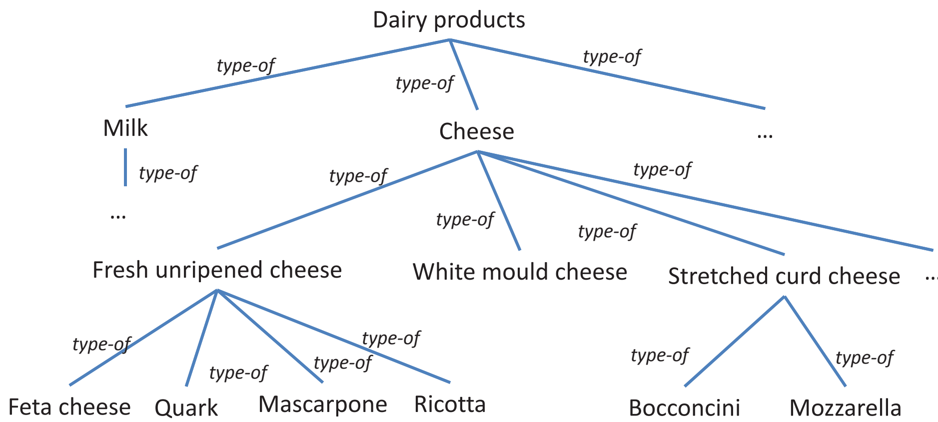

Figure 2.3: Dairy products.

2.2 CASE MODELING

A case is an instance of a problem-solving process. Two components of the case need to be distinguished: the problem description and the solution. The former comprehends the goals, task description, constraints, and initial data, among others. The last item can include the solution as it is, the steps to achieve the solution (or trace), solution justification and annotation, alternative solutions, and expectations (what could happen when obtaining the solution). Additionally, the outcome of the solution can be captured in the case representation, that is to say, if the solution has achieved the desired outcome or not. Therefore, a general approach to represent a case can be a tuple $\langle p, s, o \rangle$, p being the problem, s the solution, and o the outcome. This is not an exhaustive list of possible case components. Other components (e.g., explanations) can be considered, as well as some variations of them (for example for textual and image data representation).

To represent each case component at the semantic level, it is important to distinguish between features, objects, and relational objects. Features are attribute-value assignments, $f_i = (a_i, v_i)$, so that they describe a property. Attributes a_i are defined in the vocabulary. Values v_i are related to the attributes, so they can be either numerical (discrete or continuous), or defined in the vocabulary at a different refinement than the attributes. So, we can have the attribute *food habits* with the value $milk$, with $milk$ being a member of the ontology depicted in Figure 2.3.

Objects are attributes grouped into more complex data to facilitate reasoning at a higher level. When we enrich knowledge representations with relations between objects, then we have relational representations. All the components of a case, p, s, o, can be represented by features, objects, or relational objects, to obtain the knowledge model required for the application domain.

2.2.1 PROBLEM DESCRIPTION

There are two main types of knowledge models used for representing the problem description, p: attribute-value pairs and relational objects. Attribute-value pairs is the simplest and most frequently used representation. A problem description is described by a sequence of features, $(f_1, \dots f_n)$.

Example 2.3 Breast cancer prognosis. Consider a CBR system in charge of assessing the risk of suffering breast cancer. For that purpose, some information about the patient should be gathered as cases. A possible attribute representation for such a scenario is given in Table 2.1. The solution is the *illness sufferer* attribute.

Table 2.1: Attribute-value pair representation for the breast cancer example

Attribute	**Value**
Family risk	low
BCRA mutation	confirmed
Family checker	no
Illness sufferer	yes
Sex	woman
Place of birth	Girona
Height	1.65 m
Weight	70 Kg
Alcohol	yes
Smoking	yes
...	...

With this kind of problem it is possible to define prototypes or generalizations of cases. In the example above, if two cases are identical with the exception of height, with one patient being 1.65 m and the other 1.70 m, it is possible to think of a prototype that includes both cases by setting the height value to an interval [1.65 - 1.70]. Dealing with prototypes could improve case retrieval.

As handling hundreds of features could be annoying, features can be grouped so that objects can be used to represent cases. The example object representation for example 2.3 is provided in Table 2.2. This representation is rarely used, as, from the practical point of view, it can be easily reduced to the attribute-pair representation.

The third main kind of case representation is relational objects which are usually visualized as trees or graphs. Since there is not a homogeneous representation for all the cases, attributes cannot be localized by their position, and then the use of attribute names is required together with the path that helps in the identification of the attribute from the root of the graph. This can be seen with the breast cancer example under this knowledge representation schema as shown in Figure 2.4. To distinguish between food habits in adolescence and current food habits, the path person.foodhabits and person.adolescenthabits.foodhabits added to the corresponding attributes

Table 2.2: Object representation for the breast cancer example

Personal data	Family risk	low
	BCRA mutation	confirmed
	Family checker	no
	Illness sufferer	yes
Epidemiological data	Sex	woman
	Place of birth	Girona
	Height	1.65 m
	Weight	70 Kg
Toxic habits	Alcohol	yes
	Smoking	yes
...	...	...

solves any disambiguation. This kind of representation has been also formalized as feature terms (see Chapter 5).

More complex knowledge models for cases are also possible, including plans and workflows. Moreover, series and sequences can also be represented by a temporal component.

2.2.2 SOLUTION DESCRIPTION

The information about the solution of a case, s, depends on the problem-solving tasks. In the simplest and most frequent case, CBR is used to predict a label or class (classification). So given a set of labels L, the solution $s \in L$. There is a huge usage of binary classification, in which two labels indicate a positive or negative outcome. For example, if a device will fail or not in the near future; if a person will suffer a given illness or not (prognosis). In a credit approval domain, there may be more than two labels, for example, low, medium, and high risk for the credit approving of a person. Multi-class labeling involves the assignment of a subset of labels as the solution of the problem, $s \subseteq L$. An example of multi-class labeling is diagnosis, in which a set of possible bacteria has been identified as causing a patient's illness.

The solution can also include the way the solution was obtained, the quality of the solution, constraints restricting the solutions' application, and alternative solutions.

2.2.3 OUTCOME

One design decision when defining a case structure is to represent and store information about how the solution solves the problem. This can involve accepting that there will be cases with wrong solutions in the case-base that can be used to prevent repetition. There are very few CBR systems which incorporate such information, because a failure is handled at the retain stage, where learning methods are applied to avoid future failure. However, if we are dealing with an outlier (atypical or extreme) case, it can be helpful to keep the outcome (i.e. failure) for future use.

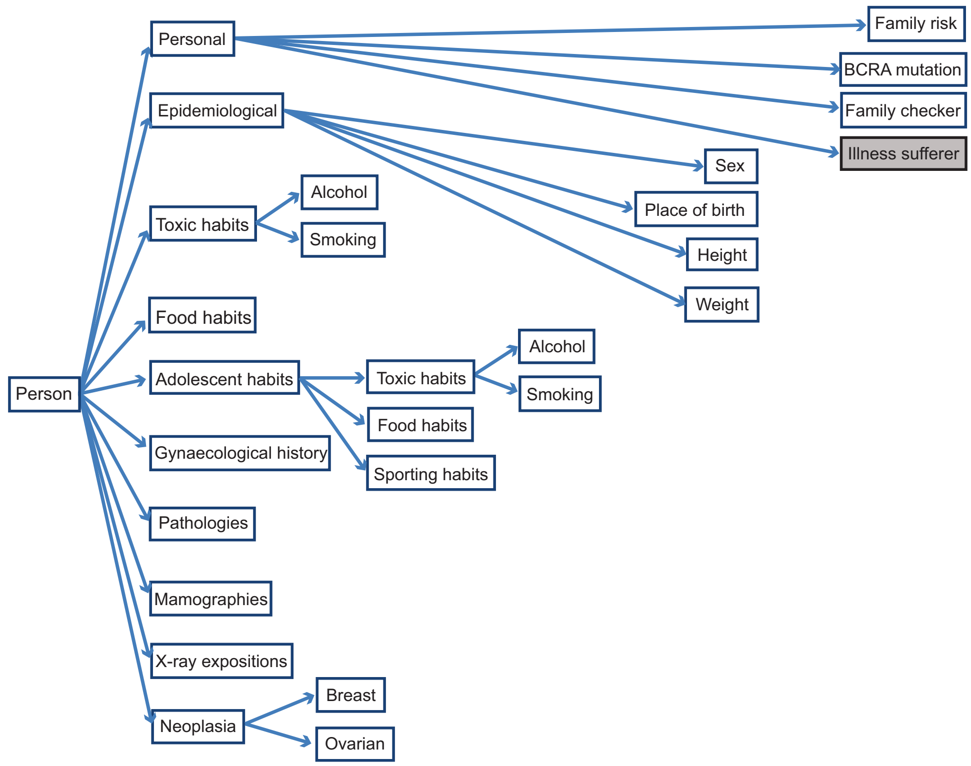

Figure 2.4: Relational representation of cases for the breast cancer example.

2.3 CASE-BASE ORGANIZATION

Case-base organization relates to how cases are indexed and retrieved from memory. Regarding the vocabulary terms used to represent cases, it is not mandatory to consider all of them as indexes for cases. Indexes are features which discriminate among cases. Indexes should have some properties, such as being predictive, discriminative, or explicative. First, predictive means that it recovers a cluster of cases that are appropriate for a query case. Therefore, it is related to their generality. Second, discriminative means that it should be specific enough to avoid retrieving erroneous cases. Finally, explicative means that it should be capable of contextualizing retrieval with the other features of the case or among all of the retrieved cases.

Pure CBR lazy approaches do not have any kind of indexing mechanisms. Other approaches determine the best indexes by solving a learning problem, as discussed in Chapter 4.

Once the indexes have been determined, different case-base organizations can be considered. Figure 2.5 depicts a possible classification of the most frequently used organizations. Some of the approaches store and access case instances, while others organize cases by defining some kind of generalizations about instances. The instance approach is important because all of the information concerning the individual problem-solving instance can be used to access a case.

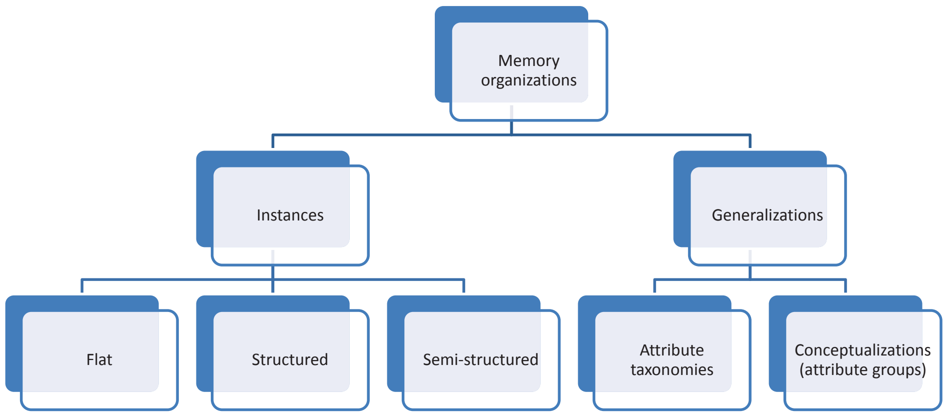

Figure 2.5: Case-base organisations.

The instance approach includes three main kinds of organizations: flat, structured and semi-structured. First, in a flat organization, cases are organized in a single table, with rows being cases, and columns, attributes. Thus, each cell of the table contains the value of the attribute for a given case. This kind of organization is linked to the use of sequential search methods to access cases. Flat organizations have been widely used until now because they can be implemented and managed by non-expert users with spreadsheets tools.

Second, structured organizations are related to the relational case representation. Cases are stored according to the relations of attributes and can be retrieved in accordance with subsumption methods, showing a robust behavior regarding missing-values management.

And third, in a semi-structured organization there is no given schema to represent cases in a uniform way throughout the case-base; this is why they are called loosely structured or semistructured. Attributes are represented by tags whose name and place inside the case may vary from case to case. Tags enable the identification of the semantic interpretation of the attributes between the different cases, thanks to the use of the ontologies, which should be annotated somewhere in the case. So the challenge for the CBR system is to handle different vocabularies for representing cases to identify and match the same semantic entities. The semi-structured organization is typical of the internet approaches and applications that handle XML documents (textual CBR).

The generalization approach includes two main kinds of organizations: attribute taxonomies and conceptual taxonomies. In attribute taxonomies, cases are organized according to a tree in which nodes represent attributes and there are as many branches from the node as there are attribute values.

Cases are placed as leaves on the tree. Figure 2.6 illustrates a possible attribute organization for the breast cancer case example.

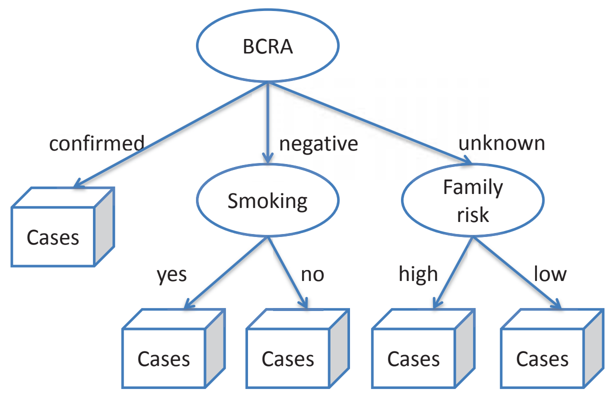

Figure 2.6: Attribute taxonomy for the breast cancer example.

Some conversational CBR considers the organization at the feature level instead of the attribute. This kind of organization is valid for help-desk applications since the answer to one question (attribute with value or feature) determines the next question.

Which attribute should be placed at the root node is a critical decision. Thus, it is possible to consider the integration of machine learning algorithms to decide which order of attributes will determine the path from the root to the cases. When doing so, CBR is being integrated with other techniques. This is a feasible system design option, since CBR is a methodology, not a technology, so it is possible to choose any technique for each of the CBR stages.

Any of the approaches that considers a single attribute in a node has the risk of failure when there are missing or unknown values in the query case. Thus, given a problem to be solved, if the root attribute is unknown, the retrieval mechanism will have no options to decide which branch of the tree to follow in order to recover some matching cases from memory. One way to mitigate such a situation is to enable multiple root entries, one per attribute, like a network, or to enable the inclusion of more than one attribute in a node. This is what conceptual organizations do.

Conceptual organizations organize cases as a tree or graph in which nodes contain generalizations of features of the cases contained within according to certain optimization criteria. Another approach is to organize cases as Galois lattices over the feature set. While a lattice on the set of all case features is a partial order set, so any two elements have a supreme (or least upper bound) and a infimum (or greater lower bound), a Galois lattice is a compacted representation of features according to some data (cases in CBR). A formal concept algorithm (FCA) performs the reduction from a lattice to a Galois lattice without information loss, so that all the combinations of features presented in the cases are kept in the Galois lattice.

Example 2.4 Suppose there are four boolean attributes available to represent data cases: A,B,C,D. Figure 2.7 left shows the lattice corresponding to this set of features. Moreover, suppose the following cases: C1=(1,1,1,0), C2=(1,1,0,0), C3=(1,0,1,1), C4=(0,1,0,0). Cases are annotated on the lattice according to the features they have. After the application of the FCA, the Galois lattice of Figure 2.7 right is obtained.

Galois lattices have good structural properties because they are closure sets: each node in the lattice is a pair of featured set and cases. Closure makes data retrieval much more efficient since cases are organized around maximal groups of shared features.

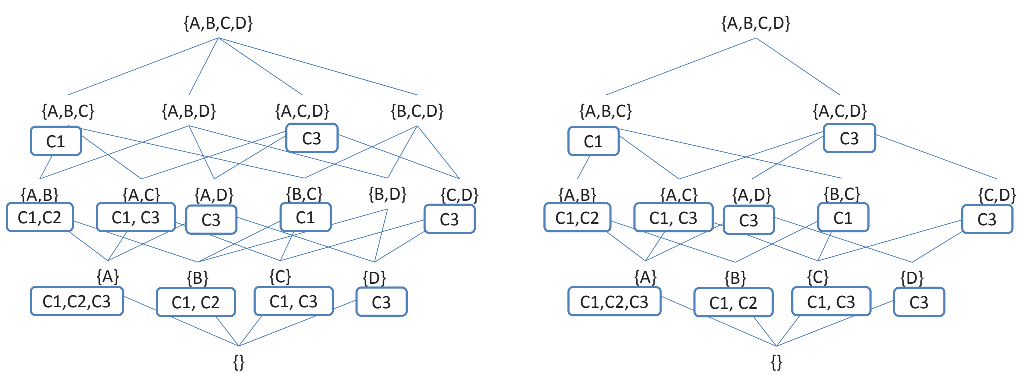

Figure 2.7: Lattice on the features A,B,C,D and Galois lattice for a given set of cases.

2.4 BIBLIOGRAPHIC NOTES

Michael Richter proposes the idea of knowledge containers in Richter [2005] and Richter [1995]. Aamodt [2001] provides a knowledge-level perspective of what a CBR system is and the knowledge it should include (domain models, tasks, and methods), based the approaches of reusable libraries.

The knowledge representation approach about features, objects, and relations is based on Chapter 12 of Russell and Norvig [2010], where they distinguish among features, composite objects, and relational objects.

For an introduction to ontologies, read Guarino [1998]. Bergmann and Schaaf [2003] discusses the relationship between structural representation and ontologies. This proposal is also followed in Bichindaritz [2004], as enabling the use of CBR for distributing and sharing cases on the Web. In Plaza [1995], feature terms are proposed as cases for CBR. Temporally related cases are handled in Sánchez-Marré et al. [2005] by means of episodes. Time series management is dealt with in Montani et al. [2009], for case retrieval by means of temporal abstractions at different levels. Regarding sequences of actions Hammond [1986], Veloso et al. [1995] and López and Plaza [1997] were the first works to deal with plans in cases in very different fields (cuisine, rockets, medicine). The former, the CHEF system, is very well known by its capability of keeping and handling cases with wrong solutions. A set of sequences is required to represent the solution of cases in a soccer team scenario in Ros et al. [2009]. Graphs as workflows were studied in Minor and Görg [2011]. An example of XML case representation is in Shimazu [1998].

PROTOS (Bareiss et al. [1988], Bareiss [1989]) is a classical example of a CBR that uses exemplars to organize cases (that the authors differentiate from prototypes, as exemplars do not attempt to obtain a pattern, but a way of improving efficiency in retrieval). PROTOS and its successor ORCA (Bareiss and Slator [1993]) provide a variety of indexing mechanisms. A review on the role of prototypes in Medicine is given by Schmidt et al. [2008] and particularly in a diabetes system in Bellazzi et al. [1998]. At a different generalization level, Yager [1997] proposes the use of fuzzy sets as a way of generalizing values. In Bergmann and Vollrath [1999], the authors discuss the convenience of dealing with generalized cases represented with the use of constraints, that enable-the use of off-the-self constraint-based techniques for their management. Maximini et al. [2003] provides different kinds of generalizations and discusses about the membership relationships required at the retrieval case.

A review of memory organization is given in Bichindaritz [2008]. Kolodner [1993] provides a deep understanding of the different features required for indices. Díaz-Agudo and González-Calero [2001] introduced the lattice approach to case-based reasoning through formal concept analysis. The use of genetic algorithms to explore the best organization of cases is explored in Garcia-Piquer et al. [2011]. Ricci and Senter [1998] analyzes the retrieval efficiency of organizational structures. As further reading on particular kind of memory organizations, have a look at feature taxonomies organizations in Gupta et al. [2004], where cases are hierarchically organized in a tree where nodes keep information of attribute-values, instead of attributes alone.

From a practical point of view, it is important to clarify that the CBR organization should respond to the appropriate storage capability. A small case-base can be stored in memory, while others require the support of data management systems. However, they are not the same. See Schumacher and Bergmann [2000] and West and McDonald [2003] for a discussion about the implementation issues of CBR using DMBS. In Sengupta et al. [1999], the practical approaches discussed include XML as knowledge representation.

CHAPTER 3

Reasoning and Decision Making

Given a query case with an unknown solution, CBR consists of finding the solution to the case by retrieving and reusing the experiences (problem-solving episodes) stored in the knowledge base. It involves the retrieve and reuse stages of CBR, as shown in Figure 3.1. Moreover, as it is possible to observe in the diagram, some feedback can be obtained in the revise stage and a remade solution can be generated and annotated with the evaluation outcome.

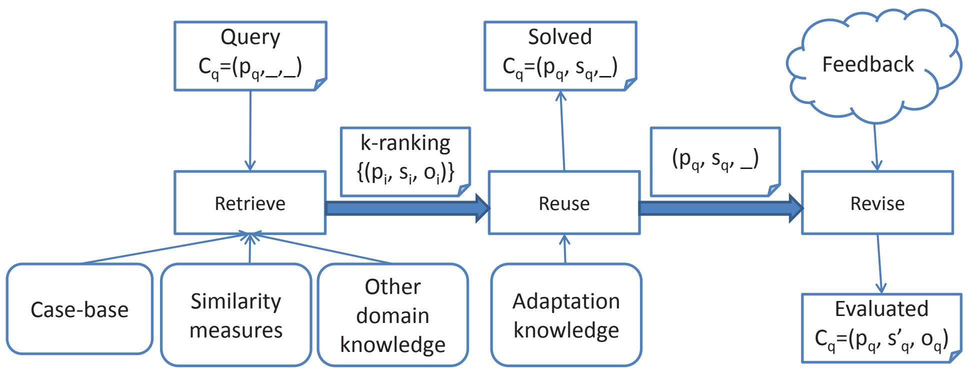

Figure 3.1: Problem-solving process in CBR.

3.1 RETRIEVE

Given the query case, C_q, the retrieval stage consists of obtaining a list of similar cases from the case-base that could be useful in providing the solution to C_q. For that purpose, the following tasks should be performed:

- Feature identification: interpret the problem and infer the problem description.
- Search the case-base to retrieve cases.
- Match retrieved cases following a certain similarity measure.
- Select the best cases as providers for the solution to the query case.

Feature identification could involve the use of domain knowledge to derive from a problem the case description suitable for the case-base system.

Example 3.1 Expressive music performance. Suppose that a CBR system should generate an expressive performance for a musical score. For doing that, the case-base contains cases about expressive performances annotated with the role of each note. Thus, musical domain knowledge can be used to enrich the input information so as to obtain the role of the notes, deriving a suitable case to be compared with past experiences in the case-base (see Figure 3.2).

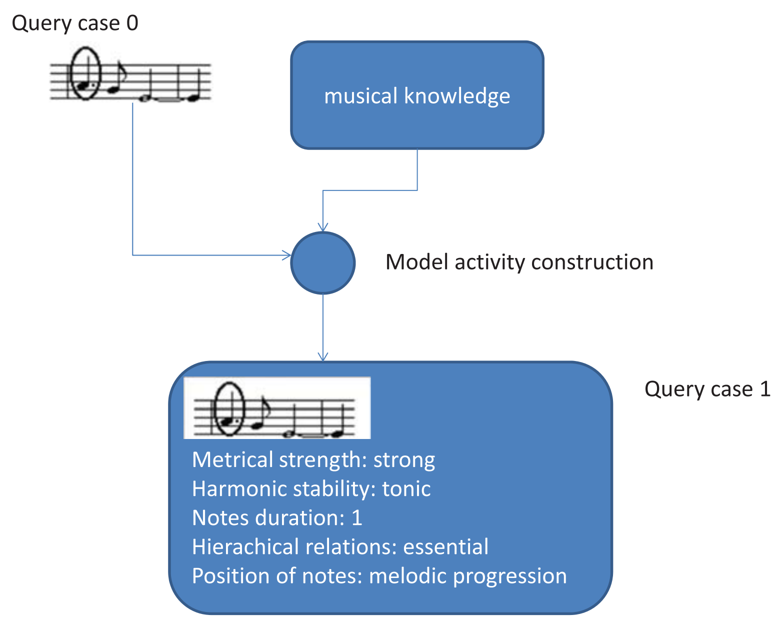

Figure 3.2: Deriving the case description for the music example (based on López de Mantaras and Arcos [2002]).

Regarding the search task, the effort dedicated to this task depends on the case-base organization. For example, if there is a good indexing structure, and it is predictive enough, similarity assessment can be simpler.

Example 3.2 Suppose that case descriptions can be represented by two boolean features, A, B. Consider three case organizations: plain (Figure 3.3 left), hierarchical (Figure 3.3 center), and Galois lattice (Figure 3.3 right). Moreover, suppose that the query case is (0,1). The plain organization would involve a great search effort, since all of the cases in memory there is almost no search. The hierarchical organization will recover half of them. Finally, with the Galois lattice, there is no search, since the indexing mechanism returns all the relevant cases.

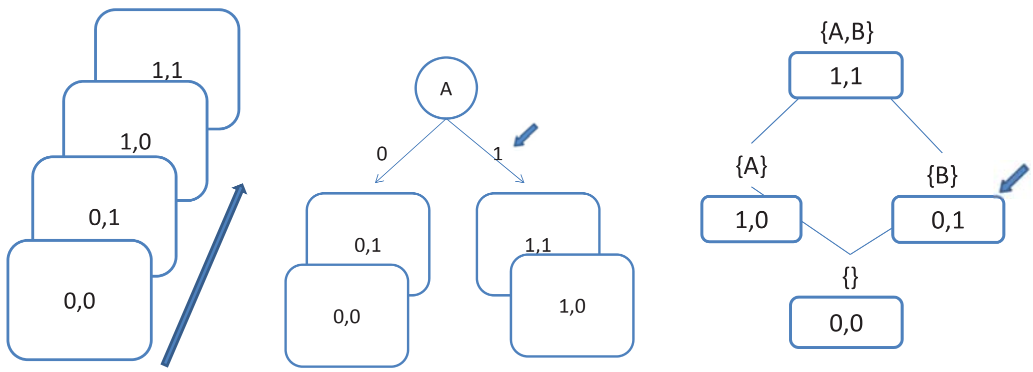

Figure 3.3: Search effort depending on the case-base organization.

Indexing mechanisms that condition the search for similar cases in the case-base have been described in the previous chapter. In this section, the focus is on the methods with which to deploy similarity assessment and ranking and selection of the k-best cases.

3.1.1 SIMILARITY ASSESSMENT

CBR bases the similarity between the query case and cases in memory in terms of similarity measures. Similarity measures are defined to highlight commonalities over pairs of problem descriptions of cases, $Sim(p_m, p_q)$, so that the solutions of past cases C_m can be reused to find the solution of the query case C_q. Similarity measures are assumed to fulfill the following properties:

- Identity: $Sim(X, X) = Sim(Y, Y)$.
- Monotonic: Similarity increases with the addition of common features (attribute values) and/or the deletion of distinctive features.
- Independence: The order of the joint effect of any two components is independent of a third factor.

For simplicity reasons, we can assume that similarity is defined as a value in [0,1], with 1 being the maximum similarity ($Sim(X, X) = 1$).

The *local-global principle* states that similarity can be computed on two levels: first, at the feature level (feature matching), taking into account the values of common attributes, and second, at the case level, combining the results according to the case representation.

Given two problem descriptions p_m and p_q, and an attribute which characterizes the problems, a_j, a local similarity measure sim_j is defined between the attribute values $p_m(a_j)$, $p_q(a_j)$, namely $sim_j(p_m(a_j), p_q(a_j))$ and $sim(x, y)$ for short. One can say that this is the result of feature matching.

Local similarity functions are often approached with distances. In this case, similarity measures are defined as the inverse of distances: $sim(x, y) = 1 - d(x, y)$, $sim(x, y) = \frac{1}{1+d(x,y)}$, $sim(x, y) = e^{-d(x,y)}$ each of which provides a different gradient behavior; in general, sim(x,y)=f(d(x,y)). To combine local measures at the global level, distance measures are often scaled up from the single-dimensional (i.e., dealing with one attribute, $d(p_i(a_j), p_q(a_j)) = d(x, y)$) approach to the multi-dimensional approach (i.e., dealing with all the case description, $d(p_i, p_q) = d(X, Y)$). In that case, careful attention should be paid regarding the inverse function employed to define similarities from distances, particularly if an iterative aggregation process is used (see below).

Distances are characterized by the following properties:

- Non-negativity: $d(x, y) \geq 0$
- Identity: $d(x, x) = 0$
- Symmetry: $d(x, y) = d(y, x)$
- Triangle inequality: $d(x, z) \leq d(x, y) + d(y, z)$

However, similarity measures should not necessarily be symmetrical, because, when making a comparison, subjects focus more on the features of the subject than on the referent. Moreover, the triangle inequality has a confusing meaning in non-geometrical spaces.

Example 3.3 Consider the attribute "uses of the computer," with three different values {domestic, industrial, space}. It is clear that we can use an industrial computer for domestic purposes with a high degree of success, so it is the similarity between "industrial" and "domestic," $sim(industrial, domestic) = 0.8$; while the opposite is not true, since using a domestic computer for industrial purposes is not suitable, $sim(domestic, industrial) = 0.4$ (see Table 3.1 for the specific similarity values which can be considered in that domain).

Table 3.1: Local asymmetric similarity measure

	Domestic	Industrial	Space
Domestic	1.0	0.4	0.0
Industrial	0.8	1.0	0.2
Space	0.6	0.8	1.0

When dealing with distances at the local level, so that a single attribute is taken into account, they depend on the type of value being handled. The following distances are the most popular (according to the data type they handle):

Numeric: when values are continuous. Possible methods include absolute difference of the values ($d(x, y) = |x - y|$) or other non-lineal functions such as $d(x, y) = \frac{1}{x-y}$ $(x \neq y)$, when the differences among closed values is not significant, while it is for remote values.

Nominal: also known as categorical or discrete values, so that the attribute values are defined in a set of labels.

- *Overlap distance*:
$$d(x, y) = \begin{cases} 0 & if x = y \\ 1 & otherwise \end{cases}$$
For binary attributes the overlap distance reduces the Hamming distance.

- *Value difference metric* (VDM), is a broadly used asymmetric distance, requiring more computational cost than that in the simple overlap measure. It is used in CBR classifiers,
$$d(x, y) = \omega(x) \sum_{c \in \mathcal{C}} (\mathcal{P}(c|x) - \mathcal{P}(c|y))^2 ,$$
where $\mathcal{C}$ is the set of all class labels, $\mathcal{P}(c|x)$ is the conditional probability of class c given value x, and $\omega(x)$ is a weighting factor related to the feature relevance x. The weighing factor $w(x)$ makes the distance not symmetric.

Ordered: when categorical values are sorted, for example, low, $medium$, $high$. In that case, values are usually mapped as a numeric value and numeric methods are applied.

Structured value: also known as taxonomic or ontological value refers to the values pertaining to an ontology. For example, the value *feta cheese* is a structured value regarding the ontology described in Figure 2.3. One popular distance is the *Palmer distance*:

$$d(x, y) = \frac{2 * nodes(lcs(x, y), root)}{nodes(x, lcs(x, y)) + nodes(y, lcs(x, y)) + 2 * nodes(lcs(x, y), root)} ,$$

where $lcs(x, y)$ is the least common superconcept of x and y (most specific generalization of x and y); and $nodes(x, y)$ is the number of nodes on the path from x to y. For example, in the taxonomy provided in Figure 2.3, the lcs between $mozzarella$ and $mascarpone$ is $cheese$, $lcs(mozzarella, mascarpone) = cheese$. The path from $cheese$ up to the root, and down to $mascarpone$ and $mozzarella$ have the following number of nodes:

$$n1 = nodes(cheese, dairyProducts) = 1 ,$$

$$n2 = nodes(mascarpone, cheese) = 2 ,$$

$$n3 = nodes(mozzarella, cheese) = 2 .$$

Finally, the distance is computed combining all of the above numbers as follows:

$$d(mozzarella, mascarpone) = \frac{2 * n1}{n2 + n3 + 2 * n1} = \frac{2 * 1}{2 + 2 + 2 * 1} = 0.33 .$$

Fuzzy values: when the attribute values are expressed by fuzzy sets, that is, the range of an attribute a_i is defined as fuzzy values $\mu_x(Z) \in [0, 1]$ for some $Z \subset \Re$. For example, Figure 3.4 shows a fuzzy attribute defined, acidity, with three possible fuzzy values: acidic ($\mu_{acidic}(pH)$), neutral ($\mu_{neutral}(pH)$), and basic ($\mu_{basic}(pH)$). The universe of acidity is $pH = [0, 14]$. One simple measure is the *disconsistency distance*:

$$d(x, y) = 1 - sup_{z \in Z} \mu_{x \cap y}(z) ,$$

where $x \cap y$ is the min t-norm, $\mu_{x \cap y}(z) = min\{\mu_x(z), \mu_y(z)\}, \forall z \in Z$.

Heterogeneous: where the distance measure handles different kinds of attributes, nominal and numeric. For example, the heterogeneous VDM:

$$d(x, y) = \begin{cases} 1 & \text{if x or y is unknown} \\ |x - y| & \text{if x and y are numeric} \\ \sum_{c \in \mathcal{C}} (\mathcal{P}(c|x) - \mathcal{P}(c|y))^2 & \text{if x and y are nominal} \end{cases}$$

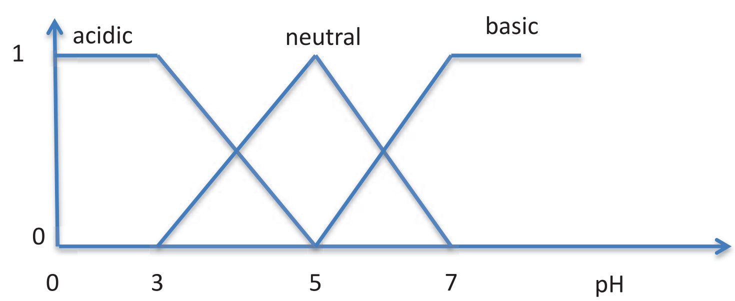

Figure 3.4: Fuzzy sets as attribute values.

Observe that some of the measures defined above are defined in $\Re$. The use of normalized distances is convenient so that their combination of values at the global level is comparable. For that purpose, numerical attribute values can be normalized before use in a local similarity measure. There are some measures, however, which include an embedded normalization method. For example, the value difference metric of numerical attributes can be redefined in a normalized version as follows:

$$d(x, y) = \frac{|x - y|}{4\sigma} ,$$

with σ being the standard deviation of the numeric values of the attribute. There is also a previous issue related to local similarity, which is attribute missingness, as outlined in Section 3.1.3.

Different local similarity measures can be considered in the same CBR system, $sim_i(x, y)$, one per attribute a_i. Once the local similarity measures are known, $sim_i(p_m(a_j), p_q(a_j))\forall i$, the global similarity measure $Sim(p_m, p_q)$ can be computed. Whenever an attribute-value representation is used, we can say that:

$$Sim(p_i, p_q) = f(sim_1(p_m(a_1), p_q(a_1)), \ldots, sim_n(p_m(a_n), p_q(a_n))) .$$

Observe that combining local results in such a composite way, local similarity measures are normalized in [0, 1] and comparable. A plain addition to all of the measures returns a positive value in $\Re$. If we wish to have a global result value in [0, 1], as required for the final outcome of $Sim(p_m, p_q)$, we need to average the results of all of the features. If that is the case, then we are applying average functions, for example, the arithmetic mean:

$$Sim(p_m, p_q) = \frac{1}{n}\sum_{i=1}^{n} sim_i(p_m(a_i), p_q(a_i)) .$$

Moreover, some aggregation schemas introduce weights to express the feature relevance. As for example, for the weighted mean:

$$Sim(p_m, p_q) = \omega_i \sum_{i=1}^{n} sim_i(p_m(a_i), p_q(a_i)) ,$$

where ω_i expresses the relevance of attribute i regarding the CBR task, $\sum_i \omega_i = 1$. Some other aggregation schemas are also possible.

Some of the most used distances and similarity functions at the global level are reviewed below, according to their applicability, from simple case structures (vector of numeric values, set of features), to more complex ones (graph and sequences).

Bottom-up distance-based measures: these come from the aggregation of single-dimensional distances, $d(x, y)$, as defined previously. The most popular distances are Minkowsky and their particular cases known as the Euclidean and Manhattan or city-block distances.

$$d(X, Y) = \underset{\text{Minkowsky}}{\left(\sum_{i=1}^{n} |x_i - y_i|^r\right)^{\frac{1}{r}}} \quad \underset{\text{Euclidean (r=2)}}{\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}} \quad \underset{\text{Manhattan (r=1)}}{\sum_{i=1}^{n} |x_i - y_i|} .$$

Also, the Chebychev distance belongs to this group, and it consists of using the highest feature difference as the representative value to determine similarities. These kinds of distances require all the data are normalized in [0,1].

Direct distance-based measures: direct distance measures are not formed directly from $d(x, y)$, but include other feature comparisons. The Canberra distance is an example of this:

$$d(X, Y) = \sum_{i=1}^{n} \frac{|x_i - y_i|}{|x_i| + |y_i|} .$$

Canberra distance is sensitive to values around 0.

Information-theoretic-based measures: measures which use information about the distribution of values throughout the case memory. Although they can provide accurate results, they can require some maintenance cost in a CBR system when cases are added and deleted according to the system experience. Typical measures are Mahalanobis, quadratic, correlation, Chi-square, and the Kendall rank correlation:

$$
\begin{aligned}
d(X,Y) = & \\
& \sqrt{(X-Y)^T V^{-1}(X-Y)} && \text{Mahalanobis,} \\
& \sqrt{(X-Y)^T W(X-Y)} && \text{Quadratic,} \\
& \frac{\sum_{i=1}^{n}(x_i-\mu_i)(y_i-\mu_i)}{\sqrt{\sum_{i=1}^{n}(x_i-\mu_i)^2 \sum_{i=1}^{n}(y_i-\mu_i)^2}} && \text{Correlation,} \\
& \sum_{i=1}^{n} \frac{1}{sum_i}\left(\frac{x_i}{sum_X} - \frac{y_i}{sum_Y}\right)^2 && \text{Chi-square,} \\
& 1 - \frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{j=1}^{j-1} sign(x_i - x_j) sign(y_i - y_j) && \text{Kendall,}
\end{aligned}
$$

where V^{-1} is the covariance of attributes $(a_1, \ldots, a_n)$; W is a weight matrix (observe that V^{-1} can equal W); μ_i is the average value for attribute a_i; sum_i is the sum of all the values occurring for attribute a_i; sum_X and sum_Y is the sum of all values occurring for all of the attributes of p_i and p_q respectively; and

$$
sign(x) = \begin{cases} -1 & if x < 0 \\ 0 & if x = 0 \\ 1 & otherwise. \end{cases}
$$

The main drawback of such measures is their computational cost.

Feature-based similarity measures: when considering cases as sets of features, $p_i = \{p_i(a_j)\} = \{f_{ij}\}$, we can apply this kind of measure. They are usually characterized as being asymmetric.

The most representative measure is the contrast model of Tversky. It is based on the relationship between the feature sets of p_i and p_q, as shown in Figure 3.5. Similarity increases with the measure of common features while it decreases with distinctive features, as shown below:

$$Sim(p_i, p_q) = \theta f(p_i \cap p_q) - \alpha f(p_i \cap p_q) - \beta f(p_i \cap p_q) ,$$

with $\theta, \alpha, \beta \geq 0$, and f reflects the salience of the various features and depends on the context. One example provided by Tversky is the following: when comparing $Spain$ within a context set up by $France$ and $Sweden$, it seems reasonable to think that $Spain$ is closer to $France$ than $Sweden$; however, if the context is $China$ and $Sweden$, it is clear that the closeness degree between $Spain$ and $Sweden$ should be significantly larger than between $Spain$ and $China$.

Graph-based similarity measures: whenever cases are represented in relational structures, the complexity regarding similarity increases: the identification of the features that should match is not always straightforward, nor is it how to combine the local similarities. Subsumption is a very well-known procedure for recovery, but it could involve a complex computational cost (graph isomorphism problem). So, several restrictions are user-defined to provide tractable algorithms. This happens analogously with the edit distance. It is defined as the minimum number of character insertion and deletion operations needed to transform one string to the other.

Sequence-based similarity measures: these involve the alignment of features before matching cases. They are mainly used in temporal CBR systems.

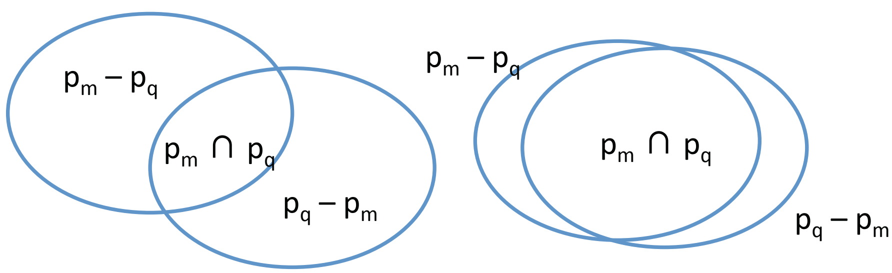

Figure 3.5: Relations between features of a memory case p_m and a query case p_q.

3.1.2 RANKING AND SELECTION

The matching task of the retrieve stage returns a set of cases with a different similarity degree to the query case C_q. From them, the best cases should be selected for use in the reuse stage. For that purpose, cases are ranked according to a preference relation induced by their utility for solving a case. When there is a set of preferences, as cases to be chosen for reuse in CBR, utilities are defined as a mapping of the preferences to real numbers that express the preferred outcome. Utility functions are difficult to provide and they are approximated by weighting schemas embedded in the similarity measure.

Weights allow the importance of one feature to be expressed over another in a decision process. Observe that similarity measures provide a solution Pareto set. For example, consider a domain with two binary attributes a_1 and a_2, and a query case C_q, with $(a_1 = 1, a_2 = 1)$. Two cases in memory, C_i and C_j, can be retrieved with the same similarity degree, with their attributes having opposite values $(C_i \rightarrow (0, 1), C_j \rightarrow (1, 0))$, as shown in Figure 3.6. Thus, weights enable the decision for one case over another.

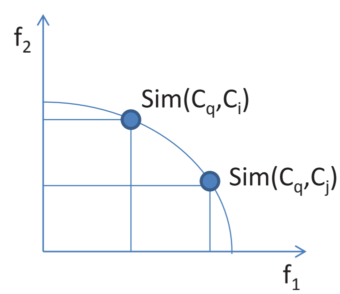

Figure 3.6: Retrieved Pareto solutions set.

Features can be understood as the criteria upon which the decision making will be based, becoming a multi-criteria decision-making (MCDM) research problem. The goal of MCDM discipline is to provide tools when several alternatives are possible, if they are limited. Therefore, functions such as the arithmetic mean, or weighted mean, are studied, among other functions. When the number of alternatives is infinite, then we are dealing with a multi-objective problem and optimization techniques should be applied. In a CBR system we can assume that the number of attributes is limited and can be handled by weighted means and similar functions.

Weights constitute what it is known as the similarity container because they represent the knowledge of an expert in a domain or heuristics to guide the retrieval process. Weights can be defined at several levels:

- Case-base level: each attribute has the same weight for all of the cases.
- Context level: attributes' weights depend on their role to discriminate among cases. For example, the W matrix in the Quadratic distance.

Weights are provided to the CBR system following a knowledge acquisition strategy or by means of machine learning techniques as detailed in Chapter 4.

The use of multi-criteria decision-making techniques such as the weighted mean helps to establish a single value for all of the values for a given case and enable their sorting. However, such an approach focuses on additive features, since all of them contribute in some degree to the similarity measure. The case may be that the presence of a particular feature is a key issue when discarding a case as a possible reference for solving the query case, threatening the monotonic assumption about similarities. When that happens, CBR systems often use filtering methods. Another way of handling such features is by adding constraints in the reuse stage to prevent the use of cases when building the final solution.

As a result of the ranking tasks, an ordered list of cases $RC = [C_1^r, \ldots, C_k^r]$ is provided. Sometimes the cut is given by a k number, other times by a threshold τ, so that ranked cases have a similarity degree over this threshold. $k = n$ (the list containing all of the cases in memory or a small number k of them) depends on the CBR implementation. Often, if k cases are returned, then the system is following a k-nearest neighbor approach. Be aware, however, of the differences and commonalities of k-NN and CBR discussed in Chapter 1.

3.1.3 NORMALIZATION, DISCRETIZATION, AND MISSING DATA

There are some related issues to the retrieve stage, such as attribute-value normalization, discretization, and missing attribute management.

Regarding normalization of numerical attributes, a key issue is having all the local similarity measures in the same scale $[0, 1]$ to combine them at the global similarity measure. There are different normalization methods, the simplest one is the following:

$$\frac{p_i(a_j) - min(a_j)}{max(a_j) - min(a_j)} .$$

Sometimes, numeric attributes are discretized so they can be manipulated as nominal attributes. Discretization may respond to several issues, such as, to determine discriminate interval values of a numeric attribute or specific intervals of interest regarding the application domain. A simple, non-informed discretization method consists of splitting the attribute range in k equal-length intervals and assigning a label to each of the intervals.

Regarding missing values, four missingness categories have been identified:

1. Missingness completely at random (MCAR): when the missing value depends on the probability of the values of the variable. For example, for a variable with a probability 0.5 of being A and 0.5 of being B, when a missing value can be substituted by either of the values with the same probability.

2. Missingness at random (MAR): when the probability of a value depends on the value of the other variables. For example, elderly people do not usually drink Coca-cola. Thus if the Coca-cola value is missing, a false value can be set instead.

3. Missingness that depends on unobserved predictors (also known as non-MAR or NMAR): For example, if the missing value is for the milk variable, and it is known that the person is lactose intolerant, then the milk value can be turned to null (no intake at all of milk). However, if the lactose intolerant variable is not in the vocabulary, and this information has not been collected in the case, the missing treatment cannot be performed.

4. Missingness that depends on the missing value itself: For example, when the alcoholic variable is missing because people who drink huge amounts of wine or beer do not want to fill that variable. If there are some other variables in the vocabulary such as a glass of wine or mug of beer, the alcoholic value can be derived from them by providing the appropriate rules (e.g., one glass of wine is approximately 3 gr. ethanol).

Solutions for every kind of missingness can be found in the statistical literature. They are sorted into five groups:

1. Discard incomplete data, either from cases or variables with missing values: Discarding incomplete data, however, can cause a bias in the resulting system.

2. Single imputation methods: such as mean or mode.

3. Random imputations based on approximating a single variable with missing data according to other informed variables: regression and prediction methods belong to this group.

4. Model-based imputations, when more than one variable has missing data: multivariate methods should be considered here.

5. Multiple imputations: to combine the different methods above.

Observe, however, that for some problem domains there could be cases with a high percentage of missing data, without meaning that the cases or variables should be discarded. Particularly, in the medical domain, where most of the variables are the outcome of laboratory tests, there could be a lot of missing values for healthy people. Thus, keeping them inside the reasoning process, without any missing value treatment that could impact the accuracy of the case-based reasoning system, is an alternative that should be considered. In this regard, it is more important to highlight structure case representations based on graph-like structures, than to handle missing values in retrieval by subsumption mechanisms.

3.2 REUSE

Given a query case C_q , and list of retrieved cases, $RC = [C_1^r, \ldots, C_k^r]$ $(k \leq n)$, eventually ordered based on certain preferences, the reuse stage consists of determining the solution for the query case s_q by re-using the solutions from the cases in RC, $s_1^r, \ldots, s_k^r$.

Example 3.4 Remembering the nautical example of Chapter 1, suppose that there are two cases retrieved, C_1^r and C_2^r, that recommend two different sailing courses, N and E correspondingly.

Which would be the final solution of the query case? One possibility is to copy the solution of the best case. Since the cases are sorted according to their similarity degree (ranked), then C_1^r is the best, and therefore the solution provided for the query case is N. Moreover, some reasoning can take place regarding differences between the information from the query case and the best case, to assess the applicability of the solution. Another alternative is to merge both solutions, obtaining NE.

Of course, this example is quite simple. When having plans as solutions, the problem of merging them could be more complex. Moreover, in medicine, the presence or absence of a feature could require a deep analysis of the implications of the difference and apply some changes in the solution provided by the case retrieved from memory before transferring the solution to the query case (e.g., changing a drug dose, the length of a therapeutic treatment, etc.). Thus, there are different approaches to the reuse stage mainly, case-based decisions (copy) and case-based adaptations. Moreover, when none of the other methods apply, special purpose adaptation and repairing heuristics can be used.

3.2.1 SOLUTION COPY

Case-based decisions are usually applicable in classification systems, in which a class should be assigned to the query case based on the most similar cases. Thus, each case $C_i^r = \langle p, s\rangle$, with each solution being a label class, $s \in \{c_j\}$. There are several methods to be considered here:

Majority rule: $s_q = arg_i maxcount(c_i, RC)$, that is, solution is the class c_i with the larger number of votes, i.e., the class that has solved the larger number of cases in RC,

Probabilistic method: which assigns probabilities to each possible outcome or class c_i:

$$p(s_q = c_i) = \frac{\sum_{j \in RC} Sim(p_q, p_j)\delta^j}{\sum_{j \in RC} Sim(p_q, p_j)} ,$$

where $\delta^j = 1$ if $s_j = c_i$, 0 otherwise.

Class-based method: is based on obtaining, for each class c_i, the mean distance of the query case with the cases belonging to c_i, and taking the class with the minimum average.

$$s_q = arg_i minmean_{c_i} Sim(p_q, p_j), \forall j ,$$

where $\mathcal{C}$ is the set of all possible classes.

Example 3.5 Consider the four scenarios given in Table 3.2. In each scenario the same cases are retrieved $C_1^r, \ldots, C_5^r$, but with different degrees of similarity with the query case, as represented in each cell of the table. The CBR is being used for a classification, having two possible labels: "+" and "-." The solution of cases C_1^r, C_2^r, C_3^r is "+" while the solution of C_4^r, C_5^r is "-." Table 3.3 shows the different values obtained by the majority, probabilistic, and class-based method for all of the labels, and the final solution assigned to the case. It is worth noting that the majority rule requires an odd number of cases; thus if C_3^r is removed from the scenarios, then it is unable to find any solution. Regarding the probabilistic method, it fails to find a solution for the first scenario, since the probabilistic outcome of each class is the same; in this case, it is convenient to combine it with the majority rule. Finally, it is interesting to observe the dependency of the applied method regarding the last scenario.

Table 3.2: Different possible ranked cases and their similitude with the query case

Scenario	C_1^r (+)	C_2^r (+)	C_3^r (+)	C_4^r (-)	C_5^r (-)
1	0.5	0.5	0.5	0.5	0.5
2	0.8	0.8	0.5	0.2	0.5
3	0.4	0.4	0.5	0.5	0.5
4	0.4	0.5	0.8	0.4	0.5

Table 3.3: Results of the different reuse methods per class and the final solution to the query case

	Majority			Probabilistic			Class-based		
Scenario	+	-	solution	+	-	solution	+	-	solution
1	3	2	+	0.40	0.40	?	0.33	0.50	-
2	3	2	+	0.46	0.25	+	0.43	0.35	+
3	3	2	+	0.39	0.43	-	0.30	0.50	-
4	3	2	+	0.46	0.35	+	0.40	0.45	-

Classification systems cover a wide range. Consistently, and in addition to the fact that these methods are simply to deploy, they are the most popular. The remaining methods are not so frequent and incur a computational cost that reduces the benefits of using CBR in real environments. Reuse is one of the weaknesses of CBR and needs further research.

3.2.2 SOLUTION ADAPTATION

Case-based adaptations are applicable to CBRs other than classification and the solution of the query case should be built up from the solutions of the recovered cases. Traditionally, adaptation methods have been put into three main groups: substitution, transformation, and derivational reply. These methods usually work with a single case, the best case recovered from memory C_1^r.

Substitution methods are applicable when the solution of the best case C_1^r is not applicable to the query case C_q unless a value is changed. There are three main methods:

Parameter adaptation: when the solution of s_1^r is copied to s_q but changing some parameters. For example, in a diabetes domain, where the CBR system is used to recommend insulin doses (bolus), the solution of a case is composed of two parameters, the insulin-to-carbohydrate ratio

(ICR) and the bolus B. When a solution is transferred, only the ICR parameter is adapted to other parameters of the query case and the B is derived from the ICR.

Local search: when a feature of the solution is changed according to a certain taxonomy. For example, if the CBR system is providing menus for people and the suggested solution contains $mascarpone$ when in the query case there is no $mascarpone$ available, then, local search methods could change that feature to $quark$ whenever $quark$ is present in C_q and the available knowledge is that provided in Figure 2.3. Observe, that if $mozzarella$ is also available, according to the closest principle described for structured value matching, $quark$ is preferred to $mozzarella$. Of course, other heuristics could be defined, and needed, depending on the permitted substitutions.

Memory-based adaptation: when different parts of the solution can be distinguished and each part can be solved by a different case in the memory. For example, a user consults a CBR routing system for going from location A to B. Two cases are recovered from memory that go from A to B: case C_1^r matches better than C_2^r, because, the time the user wants to travel fits at the end of the time interval gathered in C_1^r regarding when the route should be traversed, which is during a rush hour. However, case C_2^r provides a shorter solution for non-rush hours, which also intersects with the time interval requested in C_q. In the end, the adaptation method can generate a mixture of the two solutions, starting the route on C_1^r until a time t is reached (or a cross), and then followed on by the solution of the second case s_2^r. This substitution method may require more computational effort than the previous ones, but with a limited number of alternatives to consider (RC).

Transformation methods change the structure of the retrieved solution s_1^r to obtain the solution for the current query case s_q. There are two main kind of approaches:

Heuristic methods: used when the solution is a sequence of steps or plan-like structures. The retrieved solution s_1^r is complemented with a sequence of operations such as adding, removing, and changing the parameters of the actions of the plans or steps, until the requirements of C_q are satisfied. This is known as plan adaptation and is proven to be less computationally costly than planning from scratch.

Model-based methods: assume that the known solution has failed to deliver the requirements of the query case (model) and apply modifications to achieve the desired behavior. Constraints-based techniques are useful in this approach.

Finally, derivational reply methods are based on solutions that model the sequences of problem-solving steps required to obtain the solution instead of the solution itself. This approach is well understood in the field of planning problems, where several actions are chosen toward achieving a goal. After the execution of each action, several conditions should be fulfilled which condition the application of the next action. Thus, the solution representation requires, as explained in Chapter

2, the action, the justifications of the action selection, and the annotated alternatives in case some of the conditions are not satisfied. Derivational reply requires the use of a planner generator since it starts from a null-plan and tries to reuse the decisions recovered from s_1^r, with the aim of replicating the same steps of the plan. Derivational reply has been successful in several domains. The difficulty which conditions the use of this technique is the information required to hold onto the solution traces.

3.2.3 SPECIFIC PURPOSE METHODS

When none of the above methods apply, special purpose methods can be used. In point, rules can be used to express how to include or delete particular parts of the solution of the retrieved cases to generate the solution to the query case. For example, a special purpose adaptation heuristic rule (from Kolodner [1993]) is the following:
If the function of a component part is redundant and there is no other reason for that component then delete it.

Example 3.6 Blocks world. Consider the query problem given in Figure 3.7 left, and the selected case for reuse of the one shown at the right part of the figure. The difference in the description of both cases relies on the presence of the predicate $On(c, d)$ in the memory case. This predicate does not condition any of the preconditions of the solution ($Free(a)$, $Free(b)$, and $Free(c)$), so it can be ignored. As there is no other difference between the cases, the solution is copied.

All of the methods require some knowledge to be applied, be that heuristics or not, which, according to the knowledge representation point of view of CBR, are stored in what is known as the adaptation container (see Figure 2.1).

3.3 REVISE

The revise stage involves solution evaluation and solution repair, if needed. Solution evaluation may be provided directly by an evaluator or indirectly by measuring certain conditions of the application domain. In any case, the outcome of the solution should be to enact the appropriate solution reparation, if needed.

To deal with solution reparation, a loop between reuse and revise is often performed, until the correct solution is achieved, s'_q. In this process, some additional information, such as explanations of the failure, could be gathered and kept for future problem-solving improvements.

3.4 BIBLIOGRAPHIC NOTES

The local-global principle is introduced in Bergmann [2002]. Burkhard and Richter [2001] discusses the local-global principle and the problems that could arise to use inverse distance functions when aggregating values at the global level.

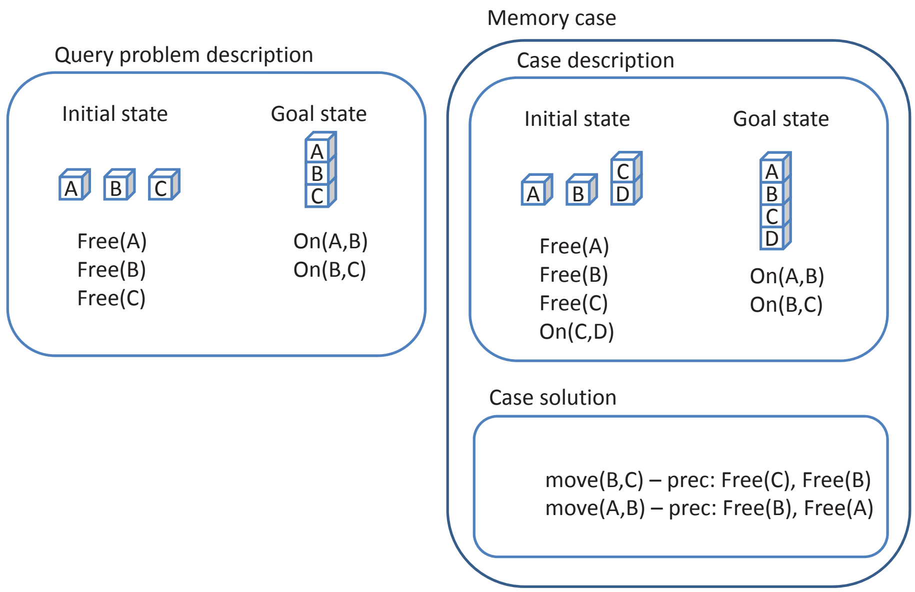

Figure 3.7: Adaptation example in the block world domain. Left: query problem. Right: selected case for reuse.

In Jakulin [2005], the author discusses the attribute interactions when generalizing, since most of the machine learning approaches assume attribute independence and are always dealing with additive measures. If CBR was a purely K-nearest neighbor method, with a similarity measure that sums up the similitude among attributes, it could suffer from the same problem. However, attribute interactions can be handled in CBR with the use of features that censor the retrieval of a case, as done in Bareiss and Slator [1993]. This is possible because CBR is learning when problem solving, and all the details of cases are retained.

Regarding similarity measures and distances, there is a vast amount of literature, because they are used in a broad scope of AI disciplines. One of the key works is the one of Tversky [1977], as mentioned in the main matter of this chapter. Cunningham [2009] organizes different similarity measures in a taxonomy. Other recommended papers on the different kind of similarity measures when applied to CBR include the following: Wu and Palmer [1994] and Resnik [1995] introduce measures among taxonomic features; Stanfill and Waltz [1986] for overlapping and value difference metrics, the latter extended in Wilson and Martinez [1996] and in Cheng et al. [2004] (and learning dissimilarities); Boriah et al. [2008] for measures for categorical attributes; Gabel [2005] for ordered values distances and taxonomic distance; Gamero et al. [2011] and Montani et al. [2009] provide similarity measures to deal with time series; Armengol and Plaza [2001] and Ontañón and Plaza [2012] deal with similarities for feature terms; Minor et al. [2007] for edit distances on workflows with some constraints due to the computation of the edit distance is a NP-complete problem; also, in Bunke and Messmer [1994], several structured value distances are presented, including an algorithm for computing the edit distance; Bergmann and Stahl [1998] analyzes similarity measures for object representations, too; Bonissone and de Mantaras [1998], Santini and Jain [1999], and Johanyák and Kovács [2005] deal with the particularities of fuzzy value attributes. For asymmetric distances, see Stahl [2002]. Wilson and Martinez [1997] provides a good review on heterogeneous distance functions. Lin [1998] deals with information-theoretic definitions of similarity.

For an extended introduction to aggregation operators read Detyniecki [2001]. Atkeson et al. [1997] provides a good survey of different weighted functions. From the point of view of multi-criteria decision making, read Torra and Narukawa [2007]. Again, the work of Tversky and Simonson [1993] provides some insights into the dependence of context when dealing with alternatives, and in Tversky [2003], preferences are analyzed in different choice settings. Emde and Wettschereck [1996] deals with feature weights in relational structures, so that feature weights depend on the depth of their use.

Weber-Lee et al. [1996] was one of the first works on CBR and proposes the use of typicality as a way of selecting cases. In the middle of ranking and reuse, Hüllermeier and Schlegel [2011] analyzed the role of preferences in CBR.

CBR as case reuse is the core of Kolodner [1993]. Later on, Hanney et al. [1995] organized CBR adaptation methods according to several criteria, as if the adaptation is carried out with one or multiple cases, if adaptation is carried out in an iterative process. Some principles of case reuse were discussed in Voß [1996]. Schmidt et al. [2003] analyzes the difficulties of reuse cases, particularizing the case in the medical domain. In Schmidt and Waligora [2006], the same authors provide a solution approach based on adaptation rules. Regarding copying solutions, Billot et al. [2005] proposes dealing with class prediction as probabilities. Bichindaritz [2011] discusses several voting schemas as well as Horváth et al. [2001]. The bolus calculation example has been taken from Herrero [2011].

Maybe one of the most prolific fields on reuse is planning. Nebel and Koehler [1995] compares the complexity and difficulties between plan reuse and plan generation. Derivational analogy was described in Carbonell [1985], from which later works are based, such as derivational reply (Veloso and Carbonell [1993], Veloso [1994], and Muñoz Avila [2001]), and transformational analogy (Kuchibatla and Muñoz Avila [2008]), among other. In this field, Rousu and Aarts [1996] proposes the use of the reuse effort as an evaluation measure.

Another fecund field is design, were model base reasoning (Goel [1992]) and constraint-satisfaction techniques (Sqalli et al. [1999], Purvis and Pu [1998]) have been used. In Manzoni et al. [2005] substitutional methods are used for truck design.

CHEF, Hammond [1989], is a very well known system regarding the use of a simulation in the revise stage.

References of missing value management can be found in statistical textbooks as Gelman and Hill [2006]. Other modern approaches such as Ennett et al. [2008] discuss the use of neural networks for value imputation. Zhang et al. [2005] reflects about the usefulness of missing values in eager learning environments. Regarding normalization, discretization algorithms, and missing value methods, see Witten et al. [2011].

CHAPTER 4
Learning

Machine learning is one of the mainstays of CBR together with problem-solving. Although pure CBR has been defined as a lazy learning approach, that is, no learning effort is dedicated when storing cases in memory, but is at problem-solving time, practical CBR requires the use of more eager methods to improve efficiency in problem solving. Learning can be applied to obtain case models, as prototypes, to recognize the relevant features of the domain or the problem at hand to learn the corresponding similarity measure, or to improve retrieval through a better case organization. That is, learning can be applied to any knowledge involved in the CBR, i.e., the knowledge containers.

Learning can be performed at different CBR lifecycle activities (see Figure 4.1) in the set up, application, and maintenance stages. In the former situation, learning from scratch methods can be applied to any container. Vocabulary can be acquired thanks to the use of inductive methods. Note that the vocabulary, as the pillar of all the knowledge involved in CBR, is mainly defined when setting up a CBR system, and later modifications could inhibit its use in the remaining containers. The case-base can be automatically generated, thanks to inductive methods, and updated later on. In this process, case generalizations or prototypes can be obtained. Moreover, genetic algorithms can be used to select the best features to index cases or to learn the weights of features for similarity retrieval. Finally, an example of adaptation knowledge learning could involve the use of inductive learning mechanisms to discover rules that relate one case solution with the solutions of other cases with the same solution, taking into account the similarities in their problem description.

During CBR system application, adaptive methods can be used in the retain stage so that the knowledge is adapted thanks to the feedback provided in the revise stage. Finally, during CBR maintenance, the case-base quality is monitored and machine learning methods enable the application of a maintenance strategy.

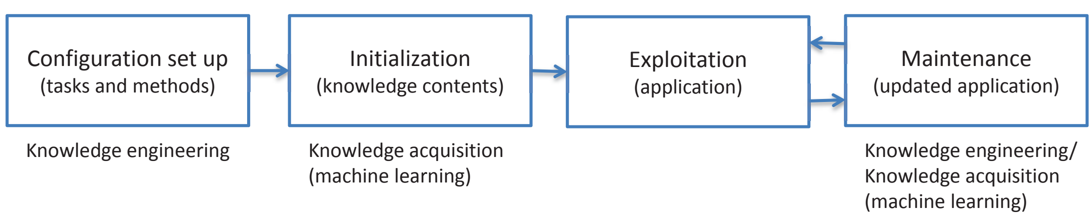

Figure 4.1: Processes for CBR system development and maintenance, their related products, and role of machine learning (knowledge acquisition).

From the point of view of machine learning, CBR systems apply most of the methods developed in this discipline. For example, the use of inductive learning methods, such as decision tree learners, for dealing with case organization (case-based knowledge learning). However, CBR has contributed significantly in the development of machine learning methods regarding similarity measures and knowledge maintenance.

4.1 SIMILARITY LEARNING

The determination of the similarity measures is a difficult and cumbersome task for knowledge engineers, and machine learning methods are being used to alleviate this task. Depending on the characteristics of the problem space, similarity learning may involve learning the similarity measure or a part of it, and the weights used in a similarity measure which indicate the feature relevance.

4.1.1 MEASURE LEARNING

Measure learning has been applied for determining an asymmetric local similarity function based on the availability of a feedback utility. Given a query case C_q, with an unknown similarity function for attribute a_j ($sim(p_i(a_j), p_q(a_j))$ is unknown), it can be estimated according to the utility feedback, assuming that the desired utility of the value $u(p_q(a_j), p_i(a_j))$ is the similarity $sim(p_q(a_j), p_i(a_j))$. For that purpose, the utility of a sequence of problem-solving episodes is collected, $C_{k_1}, \ldots, C_{k_l}$, together with its utility. If the attribute value is $p_q(a_j) = f_u$, a utility outcome $u(f_u, p_i(a_j))$ for $j = k_1, \ldots, k_l$ can be gathered. Then, $sim(f_u, x)$ can be obtained by means of interpolating the utilities. Whenever the utility value is expressed by means of a relative sorting, such as $u(f_u, p_i(a_j)) > u(f_u, p_k(a_j))$, constraint solvers can be used to derive the similarity profile.

4.1.2 FEATURE RELEVANCE LEARNING

Mainly, similarity learning is understood as the process of determining the feature relevance embedded in similarity measures. Therefore, most of the feature relevance learning methods are dedicated to weight learning. Since there are different similarity measures, there could be several models to be learned, i.e., a vector of weights $w_i \in [0, 1]$ with $\sum_i w_i = 1$, or a weight matrix with $w_i \in [-1, 1]$, among others.

Weight learning could be used in combination with a threshold (parameter of the CBR system) to drop features out when $w_i < \tau$. Moreover, some systems follow a selection strategy. Feature weights are defined in $\{0, 1\}$ and by using a threshold τ, some features are removed while the other weights are set to 1. This is known as feature selection. Following a selection strategy or leaving the overall range of weight values in the unit interval is a design choice, but several works have demonstrated that the latter approach leads to better outcomes.

From the learning process point of view, cases in memory are then considered the training data for the learner, either as a whole or a portion of it. When learning is carried out for the whole training set, then the learning method is global; otherwise it is local. Local methods enhance the lazy facet of CBR, since weights can be learned at query time.

The majority of these learning methods have been dedicated to classification tasks that use weighted similarities to decide the solution of a query case. Given the set of classes $\mathcal{C}$, the following methods can be applied:

- Cross-category feature importance method: learns weights for binary attributes a_j. Then, the weight of each feature is determined as follows:

$$w(a_j) = \sum_{c_i \in \mathcal{C}} (\mathcal{P}(c_i|a_j))^2 .$$

  Observe that with the appropriate discretization methods continuous attributes can be reduced to a set of binary ones.

- Value difference method: The value difference measure introduced in Chapter 3 provides a method to determine the relevance of features for each attribute according to its distribution along the classes, as follows:

$$w(f_{ij}) = \sqrt{\sum_{c_k \in \mathcal{C}} (\mathcal{P}(a_j|f_{ij}))^2} ,$$

  where f_{ij} is the j value of the a_i attribute.

- Mutual information method: which is based on the information theory, is as follows:

$$w(a_i) = \sum_{j} \sum_{c_k \in \mathcal{C}} \mathcal{P}(c_k, f_{ij}) log \frac{\mathcal{P}(c_k|f_{ij})}{\mathcal{P}(c_k)\mathcal{P}(f_{ij})} .$$

- Between-group and within-group sum-of-squares method (BSS/WSS): discriminates features which have a higher discriminatory power between classes. It is defined as follows:

$$\frac{BSS(f_{ij})}{WSS(f_{ij})} = \frac{\sum_{k=1}^{n} \sum_{c_l \in \mathcal{C}} \delta(C_k, c_l)(\mu_k - \mu_l)^2}{\sum_{k=1}^{n} \sum_{c_l \in \mathcal{C}} \delta(C_k, c_l)(p_k(a_i) - \mu_k)^2} ,$$

  where μ_k is the average value of attribute a_i in all the cases; μ_l is the average value of the attribute a_i regarding the class c_l; and δ is a binary function related to the class (or solution) of the cases:

$$\delta(C_k, c_l) = \begin{cases} 1 & if s_k = c_l \\ 0 & otherwise \end{cases} .$$

  It is a feature selection algorithm, so the features with the highest BSS/WSS ratio are the ones used for reasoning.

Other computationally costly methods for feature relevance learning include genetic algorithms. The methods mentioned above exploit the information of the class to learn weights, that is, they are supervised methods. There are some other methods, however, than can be used under unsupervised schemas.

Moreover, the use of some distances, such as Euclidean and Manhattan, involves the assumption of predefined information about feature data distribution (Gaussian and exponential respectively). When the distribution is unknown or the data is heterogeneous, recent approaches such as boosting have been proposed so that the best measure is selected in accordance with the current data.

4.2 MAINTENANCE

Traditional process models for software engineering and knowledge engineering define maintenance as an activity which is executed after the system has been finished and the application has already been deployed to exploitation (see Figure 4.1). Then, maintenance consists of modifying the system to correct faults, improve performance or other system features, or to adapt the system to changes in the environment in what is known as corrective, perfective, and adaptive maintenance respectively. The particularity of CBR is that maintenance is integrated in the methodology. Thus, a running CBR is continuously collecting maintenance data from the case-base system and triggering maintenance operations according to given policies.

Maintenance involves three different stages: retain, review and restore (Figure 4.2). Retain involves corrective maintenance, such as dealing with problem-solving faults. Moreover, the modification of the CBR knowledge is also considered as the decision to add or not to add newly solved cases in memory, and their consequences (case generalizations, etc.) are processed. However, retain performs a kind of myopic maintenance that can lead to a problem-solving efficiency reduction due to the case-base size, or to the accumulation of difficulties due to the distributed case collection and other issues coming from real-world data. Thus, review and restore provide mechanisms for handling maintenance over a broader scope, taking care of accuracy and quality measures of the system regarding the knowledge containers.

The first CBR systems only considered a retain stage. Nowadays, review and restore are mandatory tasks when designing CBR systems, but their actual implementation strongly depends on the application domain. Changing the system behavior is something that could be considered harmful in some domains, for example, a system controlling a chemical process. In other domains, such as recommender systems or spam filtering applications, all of these maintenance stages facilitate handling with concept drift, a hot problem in dynamic environments that other techniques can hardly manage.

Concept drift is the problem of machine learning methods when the concept of interest (target concept) may change depending on the context. For example, the buyer behavior (i.e. target concept) may vary depending on the day of the week. Other changes can be caused by the aging of the person. It is clear that the rate of change (weekly or yearly) is different, and so there are different degrees of drift. The context elements responsible for the changes are often unknown or hidden. CBR with maintenance mechanisms is able to track those changes and adapt to them.

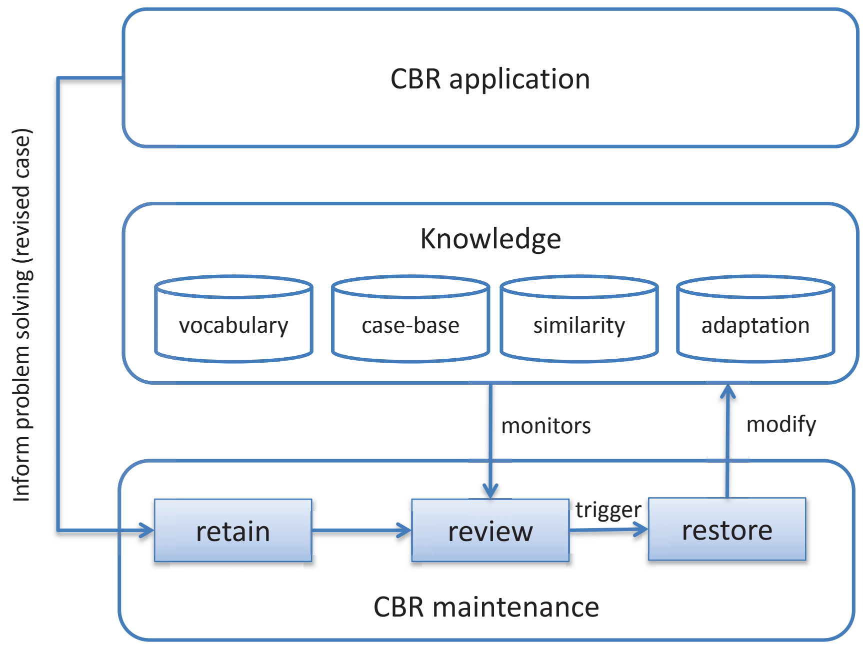

Figure 4.2: Maintenance activities in CBR.

Each application domain involves a particular maintenance policy. Maintenance policies are described in terms of how they gather data relevant to maintenance, how they decide when to trigger maintenance, the types of maintenance operations available, and how selected maintenance operations are executed.

4.2.1 RETAIN

Retain starts after the solution of a case is evaluated and repaired, if necessary. Maintenance in the retain stage is then triggered in each problem-solving experience, as the revise outcomes then offer several opportunities for learning:

- Learning when problem solving has succeeded, by storing of the new case so that:
  - Case-base structure can be modified
  - New case generalizations can be obtained
- Learning when problem solving has failed, by:

- Learning the error
- Learning about the error

The decision to retain a case or not in memory depends on whether the CBR has some introspective maintenance policy, such as analyzing the state of the case-base and future problem solving, or the utility problem. The utility problem deals with the case-base size and the retrieval time, since, if a case is added in the case-base, a saturation point could be reached and the system efficiency could start to degrade. For example, a CBR planning system which follows a derivational reply method at the reuse stage, could define a utility metric for controlling the addition of cases based on the computational effort saved by the retrieved case and the effort expended by a first-principle planner. Thus, if $Saved$ is the decision replayed from the retrieved case and $First$ the decisions performed by a first-principles planner during the problem-solving experience, this new experience or case will be stored in memory whenever

$$\frac{Saved}{Saved + First} \leq \tau \ ,$$

where τ is a given threshold. The utility problem has been widely used in CBR, but there are other factors which determine the addition of a case in memory, as shown in Section 4.2.2 (review process of CBR).

When the new case can be stored in memory for future uses, the case-base should be modified to fit the new case accordingly. For example, if the case-base is organized according to a particular case hierarchy, the case should be hung from the appropriate node and the information regarding that node updated accordingly. This modification could become global, depending on the organization and the machine learning method employed to obtain it.

Analogous to case modification, when case memory includes prototypes, the incorporation of a new case could imply the use of some inductive learning method to obtain a new prototype. The amalgam of possibilities then is wide, and several machine learning approaches could be chosen, the same as when initializing the case-base.

Example 4.1 Consider a new case C_q which has been solved with a retrieved case C_m, and that the differences between both cases rely on the value of a single attribute a_i, so that $p_m(a_i) = v_m, p_q(a_i) = v_q, v_m \neq v_q$. Then, a prototype p which represents both cases can be formed by both the attribute values, that is, $p_p(a_i) = \{v_m, v_q\}$. Moreover, if a_i is a taxonomic attribute, $p_p(a_i) = mostSpecificGeneralization(v_m, v_p)$.

Learning the error means storing error cases to avoid providing the incorrect solution for the same problem description. Therefore, the case-base should contemplate the definition of cases with incorrect answers, so that, when recovering, the mechanism should take this kind of case into account.

Learning about the error means revising the knowledge containers to identify and analyze the knowledge elements which caused the failure and conduct the appropriate modifications to avoid that in future. Therefore, the information provided at the revise stage is crucial. It should include the repaired case C_r, with the correct solution. Then, the first task to perform in this stage is to extract the relevant knowledge to distinguish the erroneous retrieval case C_e from the desired case C_d. This desired case C_d is the closest case to C_r that should be retrieved in accordance with the final solution. Knowledge about case divergences could involve different feature descriptions and/or different adaptation rules, among others. Then, the indexes related to both C_d and C_e should be updated after a trade-off analysis: C_d should have more chances than C_e of being recovered in a similar situation.

Example 4.2 Consider a CBR system that uses an attribute-value pair representation and a weighted mean to retrieve cases. Each feature (f_{ij} corresponding to the assignment (a_i, v_c)) has a weight $w(f_{ij})$. Features are updated depending on whether or not they have contributed to recovering the erroneous case from memory. Thus, three situations can be distinguished (remind $C_e = (p_e, s_e, o_e)$, $C_d = (p_e, s_e, o_e)$, being p_e and p_d the problem descriptions):

- If $f_{ij} \in p_d$ and $f_{ij} \notin p_e$: then $w(f_{ij}) = w(f_{ij}) + \alpha w(f_{ij})$, with $\alpha \in [0, 1]$, meaning that the features present in the desired case but not in the erroneous case should have matched with higher relevance, but they did not.
- If $f_{ij} \in p_e$ and $f_{ij} \notin p_d$: then $w(f_{ij}) = w(f_{ij}) - \alpha w(f_{ij})$, meaning that the features present in the erroneous case but not in the desired case have mismatched.
- If $f_{ij} \in p_e \cap p_d$: then $w(f_{ij}) = w(f_{ij})$, meaning that features common to both cases are not modified.

One needs to be sure that these operations are applied in a non-empty set, to assure some changes in the knowledge base and avoid repeating the failure. Of course several variations to this example can be considered. Simpler approaches could always update feature relevance when cases are successfully retrieved, while decreasing their relevance otherwise. More refinement modifications can also be handled, to take into account the probability of each class. The α value plays the role of a learning rate; so great values incur in big changes during different problem-solving episodes, with probably some oscillations and a hard convergence value for $w(f_{ij})$, if needed, while a small α value implies a smooth change.

In addition to weight adaptation, another strategy for the corrective maintenance of the case-base is to deal with prototypes. In the same way that a generalization can be performed when the case is solved successfully, a specialization of prototypes can be done when a failure occurs. Observe, then, that the retain mechanism is behaving as a search strategy in a hypothesis version space for prototype learning.

All the methods described above emphasize the retention of cases, while more knowledge in the system can actually degrade the system performance. So, forgetting cases can be a good retain strategy to keep performance under control. However, forgetting cases can cause a loss of knowledge.

Thus, any forgetting method should be based on the current knowledge available in the system, and how to measure that is explained in the following section.

4.2.2 REVIEW

The review stage consists of measuring and monitoring the case-base as a consequence of the activities performed in the retain stage as well as in problem solving. After an original initialization of the case-base, the inclusion of new cases can degrade the system in several ways, among them, efficiency. Moreover, the CBR system should respond to changes in the environment. Review measures are monitored, so when two consecutive measurements have a difference higher (or lower) than a certain threshold, the associated maintenance policy is triggered (and executed in the restore stage, see Figure 4.2).

There are several measures which can be grouped depending on if they are a measuring case or case-base properties:

Case properties:

- Coverage of a case, $coverage(C, CB)$: is the set of problems of the case-base CB that the case C can actually face and can solve successfully.
- Reachability of a case, $reachability(C, CB)$: is the set of cases in CB that can be used to provide solutions for C.
- Competence of a case: is the combination of the above measures, $competence(C, CB) = coverage(C, CB) \wedge reachability(C, CB)$.
- Complexity: is related to the neighborhood of a case and its solution (see Figure 4.3 left). Let $Disagree(C_i, k)$ be the proportion of cases in a neighborhood of k (k-nearest neighbors) that disagrees with the outcome of C_i. Thus, complexity is defined as the average disagreement of the progressive neighborhood, $Disagree(C_i, 1), \ldots, Disagree(C_i, k)$, as follows:
$$Complexity(C_i) = \frac{1}{k} \sum_{i=1}^{k} Disagree(C_i, k) \; .$$
When complexity is 0 (low), it means that there is no overlapping of the case description with any other case providing a different solution. This can be the cause of some redundancy in the case-base. On the other hand, when complexity is 1, the case could be a noisy case, thus causing error (high complexity). This measure is applied to CBR classifiers, so the solution is a class, but it can be extended to other problem-solving tasks as well.
- Quality: quality measures have been defined as an alternative for the previous ones, as being simpler calculation measures.
  - Correctness: whenever the case correctly solves a problem. The following measures provide additional quality measures assuming that the cases are correct.

– Consistency: a case is consistent when there does not exist any other case in the case-base which solves the same (or a more general) problem differently.

– Uniqueness: a case is unique if there does not exist any other in the case-base which solves exactly the same problem.

– Minimality: a case is minimal if there does not exist any other case with a more general description which solves the same problem. Minimal problems could be too specific and could be a matter for modification,

– Incoherency: a case is incoherent if there does not exist any other case in memory which provides the same solution with a minimum overlap of their problem descriptions. Incoherency is a good property since it favors diversity regarding the feature space.

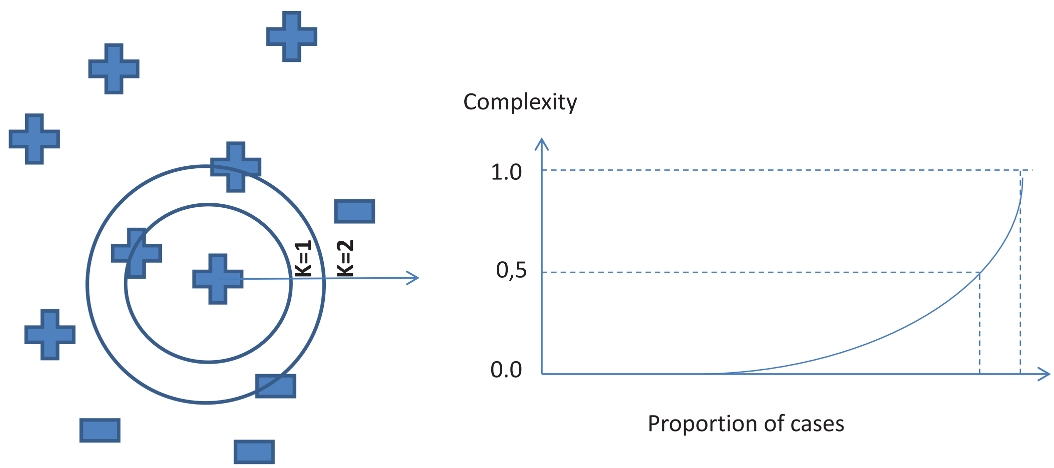

Figure 4.3: Left: Case neighborhood. Right: Complexity plot of a case-base (Massie [2006]).

Case-base properties:

- The size of the case-base: obviously, since the higher the size the higher the probability of having redundant cases, for example.
- The density of the case-base: A case in a dense region with same solution neighbors, has a low contribution to the competence of the case-base. Given a similarity measure Sim, the density of a case C in a case base CB can be computed as:

$$Density(CB, C) = \frac{\sum_{C_i \in CB \setminus \{C\}} Sim(C, C_i)}{|CB| - 1} .$$

Thus, the case-base density is obtained as the average of all of the cases densities:

$$Density(CB) = \frac{\sum_{C \in CB} Density(CB, C)}{|CB|} .$$

- Distribution: a case-base with an uneven distribution of cases is at risk of failing when solving problems in sparse regions.
- Competence: is defined as the ratio of possible problems a system can face to what it can solve successfully. Competence has been defined under the assumption that the cases in the case-base are a representative sample of the target problems. It depends on several factors: statistical and problem solving. Statistical factors include: the size, density, and distribution of cases. Problem-solving factors are the coverage and reachability of the case-base. They are measured for each case in the case-base thanks to a leave-on-out process. Next, competence groups are set up according to coverage and reachability maximal overlapping. $G \subset CB$ is a competence group if

  $$\forall C_i \in G, \exists C_j \in G \setminus C_i : competence(C_i, CB) \wedge competence(C_j, CB) \neq \varnothing$$

  and

  $$\forall C_k \in CB \setminus G, \forall C_j \in G : competence(C_i, CB) \wedge competence(C_j, CB) = \varnothing .$$

  The group coverage is defined proportional to its size and inverse proportional to its density, as follows:

  $$competence(G) = 1 + [|G| * (1 - Density(G))] ,$$

  where $Density(G)$ is defined as above (case-base). Finally, the coverage of the case-base is the summation of the coverages of all of its groups:

  $$competence(CB) = \sum_{G_i \subset CB} competence(G_i) .$$

  Figure 4.4 illustrates this measure. Observe that C_3 can be removed without losing competence, meanwhile removing C_2 could lead to be a similar unreachable situation.
- Efficiency: this property is related to the utility problem, so the benefit of adding knowledge as a consequence of a maintenance activity should not outweigh the cost of applying it.
- Complexity: is related to the boundaries of clusters formed by cases solving the same problem. It can be computed by plotting the individual complexity of cases increasingly sorted by their complexity (see Figure 4.3 right) and analyzing the properties of the generated profile. The proportion of cases from the origin to the point where the curve

breaks from the x-axis corresponds to the proportion of cases in a neighborhood providing the same solution, that is, it shows that there are redundant cases in the case-base. The area under the curve corresponds to the average case complexity, and then, to the expected error. Thus, when the complexity is over half of its value, there are potentially noisy cases.

- Quality: includes the same concepts as for an individual case: correctness, consistency, uniqueness, minimality, and incoherency. Any of the measures can be an aggregation at the case level of the individual case measurements.

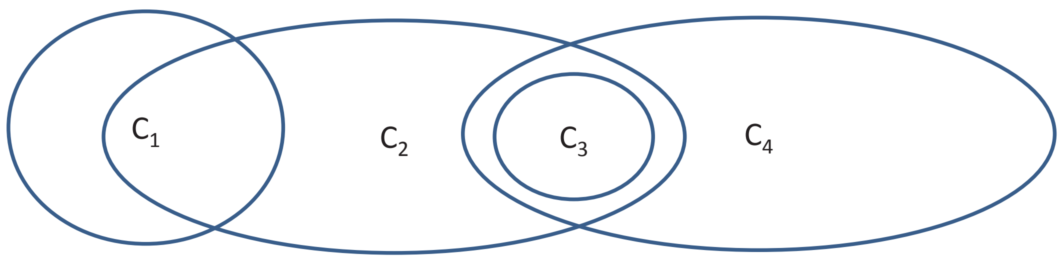

Figure 4.4: Example of coverage (based on Smyth and McKenna [1998]).

Measuring the system knowledge involves a data collection task that could be deployed in different ways according to the granularity, timing, and integration of the measurement with the CBR application. First, granularity (type of data) refers to the details of the data collected. Some non-introspective methods perform maintenance operations without analyzing the state of the case-base. Synchronic approaches measures the case-base as a snapshot each time a problem-solving episode is finished, while diachronic methods consider sequences of snapshots of the case-base. The latter approach enables the monitoring of the case-base according to changes in the environment.

Second, timing specifies when the data could be analyzed. Periodic methods collect data according to a given frequency, as for example, each problem-solving cycle. Conditional methods respond to certain well-defined conditions which prevent, for example, the case-base size going over a given threshold. Ad hoc approaches are used when the user or expert activates data gathering.

And third, integration could be online or offline, whenever data is collected while the CBR application is running, or is not.

The monitoring task receives as input the measures performed on the system and triggers maintenance operations. Analogously to the measuring task, monitoring can follow different maintenance policies according to timing and integration. Space-based policies activate maintenance operations according to the space needed to store the case-base. Time-based policies actuate when efficiency decays, that is, the retrieval time exceeds a given value. There are also some result-based policies that trigger maintenance operations when there is evidence of low-quality results. Regarding integration, monitoring can be online or offline, the same as measuring.

Maintenance operations may require the support of users, and measures obtained in this stage play an important role in providing case authoring support.

4.2.3 RESTORE

There are several maintenance operations that can be carried out in the restore stage. Some of them have a local effect, such as adding a new case, removing a case, specializing or generalizing a case, adjusting a case, or altering a case. Global modifications involve cross cases, join cases, combine cases, and abstract cases.

Different maintenance operations on different knowledge containers can have the same effect. Thus, a CBR system should incorporate a maintenance policy to guide this CBR stage. There are as many policies as combinations of the aforementioned strategies: strategies for data gathering, timing, and integration regarding measuring and monitoring; strategies for adding or modifying cases.

For example, the iterative case filtering algorithm (ICF) uses case coverage and reachability to identify cases to be deleted. Particularly, this algorithm is based on the following rule: remove cases which have a reachability set greater than the coverage one, meaning that a case C is removed when more cases can solve C than C can solve itself. Other deleting policies could be guided by the complexity measure, in order to deal with redundancy and error pruning. Sometimes, it is also important to add a lower bound to the case-base competence by adding artificial cases.

Other methods rely on forgetting strategies, as the time-weighted window (TWW) one, which consists in keeping the most recent experiences. In TWW, cases are stored together with a weight w_i, which is initialized to 1 when the case C_i is added to the memory. The updating rule for the weight is the following: $w_i = \varphi w_i$ with $0 \leq \varphi \leq 1$ being the forgetting rate. When the weight drops under a deletion threshold τ, the case is deleted. Different variations of this kind of forgetting mechanism exist, such as also considering to increase the weight when the case is used.

4.3 BIBLIOGRAPHIC NOTES

A good starting point to understand lazy learning in CBR is Armengol and Plaza [2003]. In that work, the authors describe how inductive learning can be used in CBR by following a lazy approach, that is later used for explanation purposes. This approach is reviewed and discussed with other generalization methods in Armengol [2007]. The inductive approach is related to the versions space method in Mitchell [1982], as discussed in Wess and Globig [1993]. Bichindaritz [2007] analyzes case prototypes in the biomedical field. Hanney and Keane [1996] gives an example of how to approach learning adaptation rules for CBR. Gupta et al. [2004] provides an approach to learn feature taxonomies for case indexing for conversational CBR. Young and Wu [2004] discuss this kind of CBR systems, the use of clustering techniques to deal with large case-bases.

Regarding similarity measures, most of the approaches are, in fact, feature relevance techniques, since they focus on weight learning. An exception is Stahl and Gabel [2003] where some of the distances are asymmetric and the authors propose to learned the functions that approximate them by using an evolutionary program. The work is extended in Gabel [2005] so that the use of vocabulary knowledge enhances learning. In Yu et al. [2008], distance learning is presented as a relation with the data distribution thanks to maximum likelihood theory. However, since most of the measures depend on weights, some of the results also rely on weight estimation.

Some of the feature relevance methods provided in this chapter come from Wettschereck et al. [1997], which provides four categories for feature weighted method classification: bias, weight space, representation, generality, and knowledge. The BSS/WSS method has been taken from Bichindaritz and Montani [2011]. Xing et al. [2002] discusses methods to deal with features in continuous spaces. Tishby et al. [1999] approaches learning from information theory, analyzing the strengths and limitations. Atkeson et al. [1997] provides a local view of feature learning, so that feature weights are related to the depth they are used in a relational case representation. To some extent, López and Plaza [1997] follows this local approach. When feature spaces are high regarding the number of cases available, support vector machines are proposed, as in Domeniconi and Gunopulos [2001]. Cheng et al. [2004] approaches the learning problem, taking into account attribute interactions.

First approaches to case-based maintenance can be found in Leake and Wilson [1998] and Smyth [1998], and the utility problem in Smyth and Cunningham [1996]. The need for case forgetting as well as case adding is discussed in Smyth and Keane [1995] and Zhu and Yang [1999], respectively. The former related to the drift concept as described in Delany et al. [2005]. Reinartz et al. [2001] describes the review and restore stages, and Iglezakis et al. [2004] reviews the CBR maintenance methods under a unified methodology approach, including appropriate maintenance memories. See also Craw [2003] for an introspective approach to CBR maintenance, and Mehta et al. [2009] and Murdock and Goel [2001] for a meta-reasoning perspective. Baumeister [2004] analyzes CBR development from a knowledge engineering perspective, including the monitoring and controlling of the knowledge bases.

Competence measures are detailed in Smyth and McKenna [1998]. CBR quality measures are defined in Reinartz et al. [2000] and Bierer and Hofmann [2009]. Craw et al. [2007] and Massie [2006] describe the complexity measure. The iterative case filtering algorithm is described in Brighton and Mellish [2002]. The forgetting mechanism by means of a time window, together with other mechanisms, are analyzed in Salganicoff [1997]. An application of a forgetting mechanism to recommender systems is provided in Montaner et al. [2002].

CHAPTER 5

Formal Aspects

Formal approaches to CBR specification facilitate the determination of properties, such as correctness, completeness, and complexity of the developed systems. The behavior of a system should be predictable, thus the formal specification and then verification of CBR systems should be possible. CBR, as a methodology, encompasses several stages, while formal approaches used to cover part of them. For example, description logic focuses on knowledge representation; fuzzy sets and probabilistic approaches and case-based decision theory center their attention on problem solving. Below, a summary of very well-known formalization approaches to CBR are provided, sorted by an approximate chronological order. It is beyond the scope of this introductory lecture to provide long debates and demonstrate CBR properties derived from the models, but the summaries which follow should be taken as the tools offered by the mathematical foundations of CBR.

5.1 DESCRIPTION LOGICS

Description logics are knowledge representation formalisms specifically oriented to domains that are describable by means of taxonomic organizations of complex objects. The application of these formalisms to CBR consists of representing case-base indexes as concepts in a description logic. Thus, the specifications of problem descriptions are mapped into indexes, so that retrieval is performed by reasoning approximation and similarity.

The reason for such a mapping is the following. A new case to be solved is represented by a problem specification, $C_q = \langle p, s \rangle$, from which a solution s_q has to be found. Searching for the solution of a case is done at the problem description space by looking for similar problem specifications. That means that a solution of a previous case s_i solves a case C_i in the case-base, so that:

$$s_i \models C_i \ .$$

Therefore, looking for cases in the problem description space that entail the description of the new case, is the same as stating that solutions of previous cases are solutions for the new problem:

$$C_i \models C_q \ .$$

As a consequence, the solutions s_i stored in C_i will solve C_q. Observe, however, that according to this notation, solutions are not adapted by merely copying from previous cases. Case-based reasoning requires a more flexible approach.

Therefore, the retrieval process is based on indexes instead of direct case specifications. An index of a case is computed by an encoding schema w which formalizes an abstract process:

$$w(C_q) = \langle w(p_q), w(s_q) \rangle$$

The encoding schema would depend on the particularities of the CBR system (case representation, index representation, domain), but it has to posses the following formal property:

$$\text{If} \quad C_i \rightarrow C_q \quad \text{then} \quad w(C_i) \rightarrow w(C_q) \ .$$

As a consequence, the models of cases M_{C_i} and M_{C_q} are preserved in the models of the indexes, $M_{w(C_i)}$ and $M_{w(C_q)}$, in what is known as the *monotonicity property*:

$$\text{If} \quad M_{C_i} \rightarrow M_{C_q} \quad \text{then} \quad M_{w(C_i)} \rightarrow M_{w(C_q)} \ .$$

According to this property, a solution can be found by searching the case-base along $\models$ the dimension between indexes. A retrieved case, the index of which entails the new index, will not necessarily provide a solution to the problem with certainty, but by approximation. The retrieval algorithm approximates the $\models$ relationship between cases when it compares the indexes of the cases.

Regarding reasoning by similarity, it consists of testing $w(C_i) \models w(C_q)$, which can be addressed by taking into account that each index comprises two components, $\langle w(p), w(s) \rangle$. Thus, two retrieval algorithms are proposed:

- Strong: $w(p_q) \models w(p_i)$ and $w(s_i) \models w(s_q)$
- Weak: $w(p_q) \models w(p_i)$ or $w(s_i) \models w(s_q)$

Thus, the retrieval of cases can be based on well-defined relations between indexes that possess formal semantics. Moreover, partial matches between cases are naturally handled in the indexes relationships.

Using description logics then, indexes are represented as concepts in the description logic and their $\models$ relationship can be determined by *subsumption* between concepts, $\sqsubseteq$. Subsumption is computationally intractable, unless some restrictions are imposed, which is being done in most of the practical CBR approaches, where polynomial time is guaranteed.

Taking advantage of the logical formalism underlying the indexes representation, inductive learning methods can be applied for learning and transforming indexes, as done in feature terms: obtaining the least general generalization (or *anti-unification*) or the most general specialization (*unification*). Description logics could be useful for dealing with domain contents and also with methods such as task domains.

5.2 BAYESIAN MODEL

In the Bayesian point of view of CBR, the attributes are interpreted as random variables, and the case-base is used to approximate the underlying joint probability of the attributes. Thus, this approach requires dealing with discrete attribute values. Moreover, since the elaboration of such a joint distribution could require a lot of cases, an approximation is taken by clustering attributes, i.e., grouping cases sharing similar properties.

The Bayesian model of CBR is then defined upon n random variables, $X_1, \ldots, X_n$ (attributes a_i as said in Chapter 2). A case $\overrightarrow{c_i}$ is a data instantiation of such variables (x_i are the features f_i according to previous chapters),

$$\overrightarrow{c_i} = (X_1 = x_1, \ldots, X_n = x_n) .$$

A case-base CB is a set of m independent and identical distributed data instances.

Cases can be clustered in K groups and obtain a probability distribution that represents the clusters. That is, for all k, $P(\overrightarrow{c_i}|Y = y_k)$ is the probability that a case belongs to a cluster y_k, where Y is the random variable related to clusters.

Given a case $\overrightarrow{c_i}$, it can be approximated by the weighted sum of the distributions, as follows:

$$P(\overrightarrow{c_i}) = \sum_{k=1}^{K} P(Y = y_k) P(\overrightarrow{c_i}|Y = y_k) .$$

Assuming that variables X_i inside each cluster are independent,

$$P(\overrightarrow{c_i}) = P(X_1 = x_1, \ldots, X_n = x_n) = \sum_{k=1}^{K} P(Y = y_k) \prod_{i=1}^{n} P(X_i = x_i|Y = y_k) .$$

With that model, it is possible to solve various probabilistic reasoning tasks. Thus, when CB data has been observed, and $\overrightarrow{c_q}$ is an unobserved case (query case), its predictive distribution can be defined as

$$P(\overrightarrow{c_q}|CB) = P(\overrightarrow{c_q}|CB, \Theta) ,$$

where Θ are the model parameters, $\Theta = (\alpha, \Phi)$:

- α are the parameters regarding the cluster distributions, $\alpha = (\alpha_1, \ldots, \alpha_k)$, $\alpha_i = P(Y = y_i)$, and

- Φ are the parameters related to the conditional probabilities of clusters regarding variable values, $\Phi = (\phi_{11}, \ldots, \phi_{1n}, \ldots, \phi_{K1}, \ldots \phi_{Kn})$. Each ϕ_{ij} is a set of parameters according to the cardinality D_i of variable X_i (variables are discrete). Thus, $\phi_{ij} = (\rho_{ij_1}, \ldots, \rho_{ij_{|D_i|}})$, $\rho_{ij_l} = P(X_i = x_l|Y = y_j)$.

As CB is conditionally independent given Θ, then

$$P(\overrightarrow{c_q}|CB) = P(\overrightarrow{c_q}|\Theta) .$$

In a practical situation, it could be required to know the value of a uninstantiated variable X_i (solution of a query problem) given some instantiated values (problem description). To simplify the notation, assume that the $n-1$ first variables $X_1, \ldots, X_{n-1}$ are instantiated, with values $x_1, \ldots, x_{n-1}$ correspondingly. The distribution to be determined is

$$P(X_n | CB, X_1, \ldots, X_{n-1}) \ .$$

According to the previous explanations, that means:

$$P(X_n = x_{ni} | \Theta, X_1, \ldots, X_{n-1}) =$$

$$= \frac{P(X_n = x_{ni}, X_1, \ldots, X_{n-1} | \Theta)}{P(X_1, \ldots, X_{n-1} | \Theta)} =$$

$$= \frac{\sum_{k=1}^{K} P(Y = y_k | \Theta) P(X_n = x_{ni} | Y = y_k, \Theta) \prod_{j=1}^{n-1} P(X_j = x_j | Y = y_k, \Theta)}{\sum_{k=1}^{K} P(Y = yk | \Theta) \prod_{j=1}^{n-1} P(X_j = x_j | Y = y_k, \Theta)} = \ .$$

This model can be used for classification (variable X_n is the class). Among all of the possible values, $x_{n1}, \ldots, x_{n|Dn|}$, the most probable value of the distribution is taken.

5.3 FUZZY SET FORMALIZATION

Fuzzy sets are sets of objects with a continuum of grades of memberships. Thus, while CBR is based on the principle: "the more similar the problem description attributes are, the more similar the outcome attributes," the fuzzy set model involves the membership of a previous case into the set of *similar enough* cases to the current problem.

According to this principle, reasoning holds at the similarity level (versus the instance level) and is subject to the definitions of the similarity measures, S and T, in the problem description space S and the solution space T, respectively, (see Figure 5.1). S and T are fuzzy relations, defined in [0, 1], and applied to attribute-value pairs case representations. Thus, the principle can be modeled as follows:

$$\forall (p_i, s_i), (p_j, s_j) \in CB, S(p_i, p_j) \leq T(s_i, s_j) \ .$$

This means that the similarity of the problem space constrains the similarity in the solution space: if two problems are similar, their solution should be as similar as the problem description. When solving a new problem $C_q = \langle p_q, s_q \rangle$, where the solution is unknown, the constraint defines the set of possible values for s_q: $E = \bigcap_{(p,s) \in CB} \{s_q \in T | S(p_i, p_q) \leq T(s_i, s_q)\}$.

Example 5.1 Consider a CBR system that predicts wave heights according to wind speed. Table 5.1 shows the three cases available in the CB. The similarity function $S(x, y)$ is defined as the membership of $|x - y|$ to μ_{wind} as shown in Figure 5.2; analogously for T. The prediction to be forecast is about a wind speed of $24 km/h$, that is, $p_q = (24)$. First of all, the similarity between the query case and C_1 is computed: $S(26, 24) = \mu_{wind}(2) = 0.5$. The 0.5 closeness degree of the

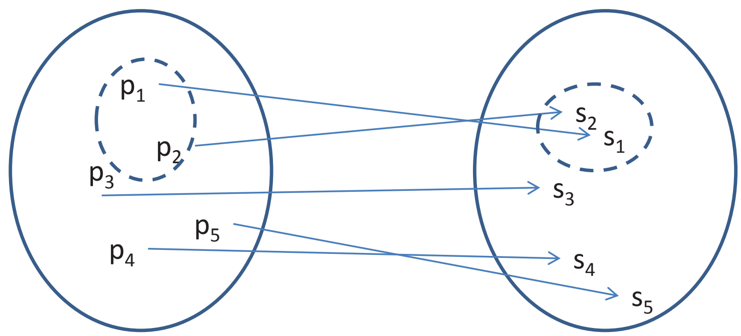

Figure 5.1: Left: problem description space where retrieval takes place. Right: solution space where inference takes place.

problem description constrains the closeness of the solutions, as shown in the figure. Thus, the solutions should be as close as 0.5 according to μ_{wave}, which corresponds to a deviation of $0.25m$ of the solution of C_1 ($1.5m$), that is, [1.25, 1.75].

Proceeding analogously with the next case C_2, we get $S(C_2, C_q) = 0$. Of course, we cannot use this similarity result to constrain the solution space, because there would be no solution at all. So, an α parameter is required that determines a similarity cut. The next case has a similarity of $S(p_3, p_q) = 0.5$, which constrains the solution space to $T(s_3, s_q) = 0.375$, which means that $s_q \in [1.125, 1.875]$. As it is possible to observe, since C_1 was closer to C_q than C_3, the solution interval provided by C_1 was narrower. Moreover, the intersection of both intervals is non-empty. Continuing with case C_4, we get $S(p_4, p_q) = 0, 5$, and then the solution should be in [0.05, 0.55]. It is possible to observe that the intersection of all intervals leads to an empty set.

Table 5.1: Wave prediction example

	Wind Speed (km/h)	Wave height (m)
C_1	26	1.5
C_2	16	1
C_3	21	1
C_4	22	0.3

For deterministic situations, non-empty sets, an interpolation method can be applied to obtain the final solution. To handle such non-deterministic situations in which empty sets could occur, a weaker form of the CBR principle is considered: "the more similar the problem description attributes are, the more *possibly* similar the outcome attributes." This rule corresponds to a particular case of possibility rules called *possibility rules*. See the bibliographical notes for further reading on this topic.

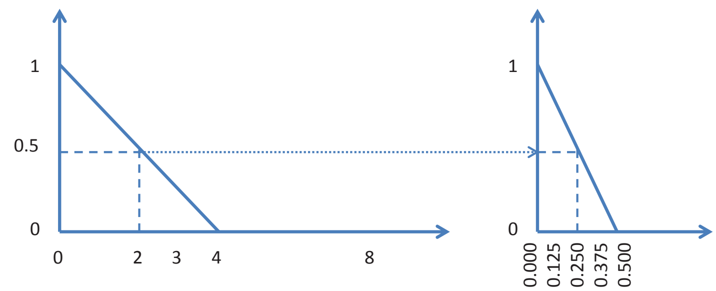

Figure 5.2: Fuzzy sets for the similarities of the wave prediction example. Left: μ_{wind}. Right: μ_{wave}.

5.4 PROBABILISTIC FORMALIZATION

Based on a similar approach to fuzzy sets, in which the solution space is constrained by the similarities at the description space, probabilistic models reason at the similarity level too. Both fuzzy and probabilistic models enable incomplete and uncertain information management (approximate reasoning), since they rely on the hypothesis that "similar inputs are *likely* to have similar outcomes;" so both kinds of formalizations focus on *case-based inference*. In the probabilistic model, the meaning of likely is analyzed in terms of probability distributions that depend on the similarity of the inputs.

As case-based inference takes place at the similarity level, some mapping is required from the original CBR set up (or instance level) to the similarity level. In general, the main steps of case-based inference regarding the probabilistic approach, are the following:

1. Characterize the problem at the similarity level by means of a similarity structure.
2. Utilize the similarity schema to derive a probabilistic characterization of the unknown outcome.
3. Translate the similarity degree on the outcome at the instance level.

To characterize the CBR at the similarity level, first a description of the CBR at the instance level is required. A CBR system is described at the instance level by (S, R, φ), where S and R are the sets of different attribute-value pairs representing problem descriptions and solutions respectively. Function φ assigns solutions to problems $\varphi : S \rightarrow R$, such that a case $C = \langle p_i, \varphi(p_i)\rangle \in SxR$.

Thus, a case-based inference set up Σ is characterized by $\langle(S, \mu_s), R, \varphi, \sigma_S, \sigma_R, CB\rangle$, where S is endowed with a probability measure μ_S, so that the sequence of problem-solving description pairs in S can be seen as the realization of a random sequence of inputs characterized by the following probability measure:

$$(\mu_S)^{n+1} = \mu_S \times \ldots \mu_S \in \mathcal{P}(S^{n+1}) .$$

This measure determines the probability space in which inference holds.

CB is the case base, as in the previous chapters.

σ_S and σ_R are the similarity functions on the problem description space S and solution space R correspondingly. They are used to map the system at the similarity level. Given that S is finite, it is possible to define the set of similarity degrees of inputs and outputs (D_S and D_R) that can be attained:

$$D_S := \{\sigma_S(p_i, p_j) | p_i, p_j \in S\} ,$$

$$D_R := \{\sigma_R(\varphi(p_i), \varphi(p_j)) | p_i, p_j \in S\} .$$

Thus, a fingerprint of the system regarding the φ structure at the similarity level is represented by a *similarity profile* $h_\Sigma : D_S \rightarrow [0, 1]$, as follows:

$$h_\Sigma(x) := inf_{p_i, p_j \in S, \sigma_S(p_i, p_j) = x} \sigma_R(\varphi(p_i), \varphi(p_j)) .$$

The similarity profile has a simple structure which enables the formulation of the *similarity hypothesis*, as a function $h : [0, 1] \rightarrow [0, 1]$, so that:

$$(\sigma_S(p_i, p_j) = x) \rightarrow (\sigma_R(\varphi(p_i), \varphi(p_j)) \geq h(x)) ,$$

which can be read as "the more similar two situations, the more similar the results."

Given a case-base inference set up Σ, a memory of cases CB, and a query problem p_0, the similarity structure of the problem (Σ, p_0) is defined by the profile $(h_\Sigma, \sigma_S, \sigma_R)$, as follows:

$$SST(CB, p_0) = \{z_{ij} = (x_{ij}, y_{ij}) | 1 \leq i < j \leq n\} \cup \{x_{0j} | 1 \leq j \leq n\} ,$$

where $x_{ij} = \sigma_S(p_i, p_j)$ and $y_{ij} = \sigma_R(s_i, s_j)$. Analogously, the outcome structure is defined as the set of values:

$$OST(CB, p_0) = SST(CB, p_0) \cup \{s_j | 1 \leq j \leq n\} .$$

However, in the probabilistic approach the focus is not predicting values but probabilities, that means,

$$P(S_0 = s | OST(CB, p_0))$$

is to be read as the probability that the outcome $\varphi(p_0)$ is s given the information provided by $OST(CB, p_0)$.

The last step of case-based inference for the probabilistic model, consisting of mapping the probability distribution of the outcome space to the instance level, is not trivial and requires further reading.

It is important to observe that the probability formalization of case-base inference has been considered an extension to statistical reasoning, since it assumes observations to be generated under conditions that are similar, instead of identically distributed random variables.

5.5 CASE-BASED DECISIONS

Case-based decisions (CBD) center the modeling on choosing acts by finding similarities to past experiences. Meaning,

"which is the best case among all the retrieved ones to reuse?"

Choosing from a set of options accompanied by utility value is a matter of studying utility theory. But this theory, as well as probabilities, tends to fail when the state of the world and the probabilities are neither given in the problem nor can be easily constructed. Thus, CBD has been proposed as an alternative.

Given a query problem C_q, CBD ranks retrieved cases, M, according to the similarities and utilities of cases. Thus, each case $C_i = (p_i, s_i, o_i)$ should be annotated with the outcome o_i of the solution s_i regarding the CBR tasks. This outcome can be understood, for example, as the assessment obtained in the revise case. Therefore, a utility function is provided to evaluate outcomes, with $u(o_i) > 0$ for desirable outcomes, while $u(o_i) < 0$ otherwise.

The CBD decision ranks actions, where actions can be understood as possible solutions to problems. So it is assumed that the CBR system has a limited number of alternatives or solutions available for choosing A ($\mathcal{A} = \bigcup_{C_i \in CB} \{s_i\}$; also it is possible to equate $\mathcal{A}$ to R of the previous formal approach).

The simplest model of CBD computes the utility of an action $a_i \in \mathcal{A}$ regarding a query problem p_q as a similarity-weighted sum of the utilities of the retrieved cases M:

$$U(a_i) = U_{q,M}(a_i) = \sum_{(p_j, a_i, o_j) \in M} Sim(p_j, p_q) u(o_j) .$$

The actions can be ranked according to their utility, and the CBD selects the action that maximizes it.

5.6 BIBLIOGRAPHIC NOTES

Koehler [1994] describes the terminological logic approach to CBR, that is reviewed in Kamp [1996] under the representation framework of description logics. Salotti and Ventos [1998] provides a formalization of CBR using description logics. They use two criteria, similarity and dissimilarity, that are used to induce a partial order on the subsumption relation. Tractability issues about subsumption are analyzed in Sebastiani and Straccia [1991]. Feature terms in CBR are described in Plaza [1995]; regarding the axiomatic approach to generalization, read Aït-Kaci and Sasaki [2001].

The Bayesian framework of CBR is described in Tirri et al. [1996]. An inductive model has been proposed as a unified framework for Bayesian reasoning, case-based reasoning, and rule-based reasoning (Gilboa et al. [2009]).

Dubois et al. [1998] describes the fuzzy formalization of case-based reasoning, which is further reviewed in Chapter 3 of Pal et al. [2001]. A logical approach to the fuzzy model is provided in Plaza et al. [1998]. Read Esteva et al. [1997] also, for a close concept on similarity-based reasoning, and Sandri et al. [2012] for a weight learning analysis.

For a short introduction to the probabilistic formalization, see Hullermeier [1999] and for further readings Hüllermeier [2007], which also includes the constraint and fuzzy modelling.

The theory of case-based decision is from Gilboa and Schmeidler [2001] (a short version in Gilboa and Schmeidler [1995]). Further discussion about similarity and utility foundations are in Richter [2007]. Hüllermeier and Schlegel [2011] provides a first step toward a preference-based methodological framework of CBR.

Globig et al. [1997] discusses case-based reasoning and learnability trade-off, how the case representation and the similarity measure limits learnability.

CHAPTER 6

Summary and Beyond

CBR is a methodology for developing knowledge-based systems based on storing previous problem-solving experiences for their use in solving future, similar problems. Past experiences or cases are stored by keeping all the particular information of a problem-solving past episode, without dropping any particularities out, the removal of which could lead to the generation of general patterns. The experiences are, by themselves, part of the domain model. When solving a problem, reasoning is carried out thanks to a neighborhood generalization procedure guided by a similarity measure. Thus, lazy learning is postponed until problem solving, and only part of the cases, those that are relevant to the problem at hand, participate in that process.

CBR presents several advantages regarding the knowledge acquisition bottleneck that other knowledge-based systems suffer, as the key knowledge to feed a CBR system can be based on the observation of problem-solving episodes from a teacher. Nevertheless, the development of a CBR system is not without effort. Throughout the process, the amalgam of options a CBR system designer is faced with, in order to build a CBR system, has been demonstrated to be wide, due to the many different knowledge models, tasks, and techniques that can be used in each stage. Several tools have been developed with the aim of aiding the CBR system designer in the development process, with most of them being object-oriented, since it is a software methodology that provides the flexibility and modularity to include new methods as required. See the bibliographical notes for references to some of them. It is worthy to comment that most of the off-the-shelf tools focus on the retrieval and reuse processes, emphasizing the lazy learning approach of CBR.

Moreover, the experience factory concept should be applicable in the development of CBR systems, so as to reuse previous experiences in CBR system development in the decisions of new ones. So, thinking of defining a CBR factory for CBR systems seems a recursively nice way of approaching CBR system development support.

Beyond this lecture, and before addressing the development of a CBR system, issues such as the explanation of the CBR, case provenance management, and CBR distribution should be clarified so that they can be considered as a matter for further study.

6.1 EXPLANATIONS

Explanations of the reasoning process is a mandatory issue for improving system usability. Explanations provide contextual information from which the solutions to problems have been provided. CBR systems usually rely on providing the most similar case, the case from which the solution has been derived, as an explanation of the system answer. Moreover, this information can be enlarged in a visual map of all the solutions and similarities between cases.

However, the reference to a past case as an explanation, when some complex similarity measures, as well as reuse techniques, can be involved, could, instead of providing user support, be completely uninformative. Then explanation mechanisms should be designed according to certain explanation goals, like the following:

- Transparency: how the system reaches the answer,
- Justification: why the system answer is good,
- Relevance: why the answer is important in relation to the most frequent answers,
- Conceptualization: which is the meaning of the concepts in the answers, and
- Learning: which is the insight provided by the case, when the goal is to teach the user about the problem solving application domain.

In defining the explanation mechanism, it should be taken into account that past experiences may not necessarily be good lessons. On the other hand, the knowledge inside the CBR system could plays an important role in providing good explanations. Moreover, the effort required to produce such explanations should be balanced with the effort required for reasoning. In the end, we want a CBR system that solves complex problems while providing simple, clear explanations that empower the user.

6.2 PROVENANCE

CBR focuses on case retaining (and maintenance), but not on the provenance of cases, that is, how the cases have been derived. A case that has been provided by a user is not the same as one derived from the system. Moreover, it is not the same to have a derived solution without any further feedback from the environment than one with some feedback about the outcome of the proposed solution.

When a case receives negative feedback, what is the causal root? The current retrieved case or previous cases from which it was derived? Thus, tracking the case source or provenance of cases could help in the revise stage. Moreover, in maintenance operations, when the user could be asked about a myriad of cases to be dropped or retained, case provenance could support the case authoring decision process.

6.3 DISTRIBUTED APPROACHES

CBR has been introduced in multi-agent environments for coordination learning and reasoning in what is known as distributed CBR.

Multi-agent means that more than one agent is involved in solving a problem, such as when different coverage of a given domain must be provided. Thus, in addition to the local-global principle, a social policy should be added to CBR. While the local-global principle relates to the synergies between similarity functions at the feature level (e.g., similarities between two age values) and how these similarities are aggregated at the case level, the social policy combines the outcome of several agents. Moreover, several schemas for case exchange related to the machine learning dimension of case-based reasoning should be considered.

Other approaches of distributiveness relate to the use of one or multiple case-bases (by a single agent or multiple agents). Multi-case-based reasoning (MCBR) works by enriching a local case-base with cases from other bases with different task or execution environments. MCBR then focuses on strategies to decide when to access case-bases and how to apply their cases.

6.4 BIBLIOGRAPHIC NOTES

Abdrabou and Salem [2008] provides a survey on CBR tools. Updated information can be found in `http://cbrwiki.fdi.ucm.es/`. Some of the tools are teaching oriented, as Díaz-Agudo et al. [2007], while other are research tools oriented in a particular field, as López et al. [2011] for medicine. The experience factory is described in Klaus-Dieter et al. [1997]. See Bergmann et al. [2003] for further understanding of industrial applications of CBR.

The foundations of CBR explanations are in Schank et al. [1994]. Ram [1993] focuses on the issue of explanations to understand the lessons provided from problem-solving experiences. Roth-Berghofer [2004] and Roth-Berghofer and Cassens [2005] provides a knowledge description of this CBR task, and Armengol and Plaza [2006] provides a similarity-based perspective. Goel and Murdock [1996] uses a meta-level to deal with explanations. Sørmo et al. [2005] addresses CBR explanations according to the systems and users goals. A recommender system approach is given in Tintarev and Masthoff [2007], Mcsherry [2005], and Bilgic [2004].

Provenance is discussed in Leake and Whitehead [2007].

The taxonomy of CBR systems according to distributiveness, has been taken from Plaza and McGinty [2005]. The use of XML technology to deal with distributiveness is explained in Hayes et al. [1998]. Ontañón and Plaza [2008] presents close connections between multi-agent system research and CBR, by including argumentation components in the CBR methodology, as well as other social level decisions, as to define committees. Reasoning with multiple cases is tackled in Leake and Sooriamurthi [2002].

Bibliography

Aamodt, A. (2001). Modeling the knowledge contents of CBR systems. In *Workshop Program at the Fourth International Conference on Case-Based Reasoning , Vancouver. Naval Research Laboratory Technical Note AIC-01-003*, pages 32–37. 12, 19

Aamodt, A. and Plaza, E. (1994). Case-based reasoning; foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59. 1, 8

Aamodt, A., Sandtorv, H. A., and Winnem, O. M. (1998). Combining case based reasoning and data mining - a way of revealing and reusing rams experience. In *In Lydersen, Hansen, Sandtorv (eds.), Safety and Reliability; Proceedings of ESREL 98*, pages 16–19. 8

Abdrabou, E. A. M. L. and Salem, A.-B. M. (2008). Case-based reasoning tools from shells to object-oriented frameworks. In *ICCOMP*, pages 781–786. 67

Aha, D. W. (1997). Editorial on lazy learning. *Artificial Intelligence Review*, 11:7–10. DOI: 10.1023/A:1006538427943 9

Aha, D. W., Breslow, L. A., and Muñoz Avila, H. (2001). Conversational case-based reasoning. *Applied Intelligence*, 14(1):9–32. DOI: 10.1023/A:1008305406188 8

Aha, D. W., Kibler, D., and Albert, M. (1991). Instance-based learning algorithms. *Machine Learning*, 6:37–66. DOI: 10.1023/A:1022689900470 9

Aït-Kaci, H. and Sasaki, Y. (2001). An axiomatic approach to feature term generalization. In Raedt, L. and Flach, P., editors, *Machine Learning: ECML 2001*, volume 2167 of *Lecture Notes in Computer Science*, pages 1–12. Springer Berlin Heidelberg. 62

Armengol, E. (2007). Usages of generalization in case-based reasoning. In *Proceedings of the 7th international conference on Case-Based Reasoning: Case-Based Reasoning Research and Development*, ICCBR, pages 31–45. Springer-Verlag. DOI: 10.1007/978-3-540-74141-1_3 52

Armengol, E. and Plaza, E. (2001). Similarity assessment for relational CBR. In *Proceedings of the 4th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development*, ICCBR, pages 44–58. Springer-Verlag. DOI: 10.1007/3-540-44593-5_4 37

Armengol, E. and Plaza, E. (2003). Discovery of toxicological patterns with lazy learning. In Palade, V., Howlett, R., and Jain, L., editors, *Knowledge-Based Intelligent Information and Engineering Systems*, volume 2774 of *Lecture Notes in Computer Science*, pages 919–926. Springer Berlin Heidelberg. 52

Armengol, E. and Plaza, E. (2006). Symbolic explanation of similarities in case-based reasoning. *Computing and Informatics*, 25(2-3):153–171. 67

Atkeson, C. G., Y, A. M., and Z, S. S. (1997). Locally weighted learning. *Artificial Intelligence Review*, pages 11–73. DOI: 10.1023/A:1006511328852 38, 53

Bareiss, E. R., Porter, B. W., and Wier, C. C. (1988). PROTOS: an exemplar-based learning apprentice. *International Journal of Man-Machine Studies*, 29(5):549–561. DOI: 10.1016/S0020-7373(88)80012-9 20

Bareiss, R. (1989). *Exemplar based knowledge acquisition: a unified approach to concept representati on, classification, and learning*. Academic Press Professional, Inc. 20

Bareiss, R. and Slator, B. M. (1993). The evolution of a case-based computational approach to knowledge representation, classification, and learning. In Nakumura G., Medin D., Taraban R. (Ed.) *Categorisation by Humans and Machines*, pages 157–186. Academic Press. 20, 37

Baumeister, J. (2004). *Agile Development of Diagnostic Knowledge Systems*. Ph.D. thesis, Bayerischen Julius Maximilians Universität Würzburg. 53

Begum, S., Ahmed, M., Funk, P., Xiong, N., and Folke, M. (2011). Case-based reasoning systems in the health sciences: A survey of recent trends and developments. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 41(4):421–434. DOI: 10.1109/TSMCC.2010.2071862 9

Bellazzi, R., Larizza, C., Montani, S., Riva, A., Stefanelli, M., d'Annunzio, G., Lorini, R., Gomez, E., Hernando, E., Brugues, E., Cermeno, J., Corcoy, R., de Leiva, A., Cobelli, C., Nucci, G., Del Prato, S., Maran, A., Kilkki, E., and Tuominen, J. (2002). A telemedicine support for diabetes management: the T-IDDM project. *Computer Methods and Programs in Biomedicine*, 69(2):147–161. DOI: 10.1016/S0169-2607(02)00038-X 8

Bellazzi, R., Montani, S., and Portinale, L. (1998). Retrieval in a prototype-based case library: A case study in diabetes therapy revision. In Smyth, B. and Cunningham, P., editors, *Advances in Case-Based Reasoning*, volume 1488 of *Lecture Notes in Computer Science*, pages 64–75. Springer Berlin Heidelberg. 20

Bergmann, R. (2002). *Experience Management Foundations, Development Methodology, and Internet-Based Applications*, chapter 4. Assessing Experience Utility. LNCS 2432. Springer. 36

Bergmann, R., Althoff, K., , Breen, S., Goker, M., Manago, M., Traphoner, R., and Wess, S. (2003). *Developing Industrial Case-Based Reasoning Applications: The Inreca Methodology. Second edition*. Springer-Verlag New York, Inc. DOI: 10.1007/b94998 9, 67

Bergmann, R. and Schaaf, M. (2003). Structural case-based reasoning and ontology-based knowledge management: A perfect match? *Journal of Universal Computer Science*, 9(7):608–626. 19

Bergmann, R. and Stahl, A. (1998). Similarity measures for object-oriented case representations. In Smyth, B. and Cunningham, P., editors, *Advances in Case-Based Reasoning*, volume 1488 of *Lecture Notes in Computer Science*, pages 25–36. Springer Berlin Heidelberg. 38

Bergmann, R. and Vollrath, I. (1999). Generalized cases: Representation and steps towards efficient similarity assessment. In *Proceedings of the 23rd Annual German Conference on Artificial Intelligence: Advances in Artificial Intelligence*, KI '99, pages 195–206. Springer-Verlag. DOI: 10.1007/3-540-48238-5_16 20

Bichindaritz, I. (2004). Memoire: Case based reasoning meets the semantic web in biology and medicine. In Funk, P. and Gonzáez Calero, P. A., editors, *Advances in Case-Based Reasoning*, volume 3155 of *Lecture Notes in Computer Science*, pages 47–61. Springer Berlin Heidelberg. 19

Bichindaritz, I. (2007). Prototypical cases for knowledge maintenance in biomedical CBR. In Weber, R. O. and Richter, M. M., editors, *Case-Based Reasoning Research and Development*, volume 4626 of *Lecture Notes in Computer Science*, pages 492–506. Springer Berlin Heidelberg. DOI: 10.1007/978-3-540-74141-1 52

Bichindaritz, I. (2008). Memory structures and organization in case-based reasoning. In Perner, P., editor, *Case-Based Reasoning on Images and Signals*, volume 73 of *Studies in Computational Intelligence*, pages 175–194. Springer Berlin Heidelberg. DOI: 10.1007/978-3-540-73180-1 20

Bichindaritz, I. (2011). Comparison of reuse strategies for case-based classification in bioinformatics. In Ram, A. and Wiratunga, N., editors, *Case-Based Reasoning Research and Development*, volume 6880 of *Lecture Notes in Computer Science*, pages 393–407. Springer Berlin Heidelberg. DOI: 10.1007/978-3-642-23291-6 38

Bichindaritz, I. and Montani, S. (2011). Advances in case-based reasoning in the health sciences. *Artificial Intelligence in Medicine*, 51(2):75–79. DOI: 10.1016/j.artmed.2011.01.001 9, 53

Bierer, A. and Hofmann, M. (2009). Dimensions of case-based reasoner quality management. In McGinty, L. and Wilson, D., editors, *Case-Based Reasoning Research and Development*, volume 5650 of *Lecture Notes in Computer Science*, pages 90–104. Springer Berlin Heidelberg. DOI: 10.1007/978-3-642-02998-1 53

Bilgic, M. (2004). *Explanation for Recommender Systems: Satisfaction vs. Promotion*. Ph.D. thesis, Department of Computer Sciences, University of Texas at Austin. Undergraduate Honor Thesis. 67

Billot, A., Gilboa, I., Samet, D., and Schmeidler, D. (2005). Probabilities as similarity-weighted frequencies. *Econometrica*, 73(4):1125–1136. DOI: 10.1111/j.1468-0262.2005.00611.x 38

Bonissone, P. and de Mantaras, R. L. (1998). Case-based reasoning. In Ruspini, Bonissone, P. E., editor, *Handbook of Fuzzy Computing*, chapter Section F4.3. Institute of Physics Publishers. DOI: 10.1887/0750304278 38

Boriah, S., Chandola, V., and Kumar, V. (2008). Similarity measures for categorical data: A comparative evaluation. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*, pages 243–254. 37

Brighton, H. and Mellish, C. (2002). Advances in instance selection for instance-based learning algorithms. *Data Min. Knowl. Discov.*, 6(2):153–172. DOI: 10.1023/A:1014043630878 53

Bunke, H. and Messmer, B. (1994). Similarity measures for structured representations. In Wess, S., Althoff, K.-D., and Richter, M. M., editors, *Topics in Case-Based Reasoning*, volume 837 of *Lecture Notes in Computer Science*, pages 106–118. Springer Berlin Heidelberg. DOI: 10.1007/3-540-58330-0 38

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12:331–370. DOI: 10.1023/A:1021240730564 8

Burkhard, H.-D. and Richter, M. M. (2001). On the notion of similarity in case based reasoning and fuzzy theory. In Pal, S. K., Dillon, T. S., and Yeung, D. S., editors, *Soft Computing in Case based Reasoning*, pages 29–45. Springer-Verlag. 36

Carbonell, J. G. (1981). A computational model of analogical problem solving. In *Proceedings of the 7th international joint conference on Artificial intelligence - Volume 1*, IJCAI'81, pages 147–152. Morgan Kaufmann Publishers Inc. 8

Carbonell, J. G. (1982). Learning by analogy : formulating and generalizing plans from past experience. Technical report, Carnegie Mellon University, Computer Science Department, Paper 2442. 8

Carbonell, J. G. (1985). Derivational analogy: a theory of reconstructive problem solving and expertise acquisition. Technical report, Carnegie Mellon University, Computer Science Department, Paper 1534. 38

Chantaraskul, S. and Cuthbert, L. (2004). Using case-based reasoning in traffic pattern recognition for best resource management in 3g networks. In *Proceedings of the 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, MSWiM '04, pages 252–259. DOI: 10.1145/1023663.1023708 9

Cheng, V., Li, C.-H., Kwok, J. T., and Li, C.-K. (2004). Dissimilarity learning for nominal data. *Pattern Recognition*, 37(7):1471–1477. DOI: 10.1016/j.patcog.2003.12.015 37, 53

Craw, S. (2003). Introspective learning to build case-based reasoning (CBR) knowledge containers. In *Proceedings of the 3rd International Conference on Machine Learning and Data Mining in Pattern Recognition*, MLDM'03, pages 1–6. DOI: 10.1007/3-540-45065-3_1 53

Craw, S., Massie, S., and Wiratunga, N. (2007). Informed case base maintenance: a complexity profiling approach. In *Proceedings of the 22nd national conference on Artificial intelligence - Volume 2*, AAAI, pages 1618–1621. AAAI Press. 53

Cunningham, P. (1998). CBR: Strengths and weaknesses. In Pasqual del Pobil, A., Mira, J., and Ali, M., editors, *Tasks and Methods in Applied Artificial Intelligence*, volume 1416 of *Lecture Notes in Computer Science*, pages 517–524. Springer Berlin Heidelberg. 8

Cunningham, P. (2009). A taxonomy of similarity mechanisms for case-based reasoning. *IEEE Transactions on Knowledge and Data Engineering*, 21(11):1532–1543. DOI: 10.1109/TKDE.2008.227 37

Cunningham, P. and Delany, S. J. (2007). k-nearest neighbour classifiers. Technical report, UCD School of Computer Science and Informatics. 9

Delany, S. J., Cunningham, P., Tsymbal, A., and Coyle, L. (2005). A case-based technique for tracking concept drift in spam filtering. *Knowledge-Based Systems*, 18(4-5):187–195. DOI: 10.1016/j.knosys.2004.10.002 53

Detyniecki, M. (2001). Fundamentals on aggregation operators. It is based on Detyniecki's doctoral thesis "Mathematical aggregation operators and their application to video querying" `http://www.lip6.fr/reports/index-eng.html` [Accessed November 2012]. 38

Díaz-Agudo, B. and González-Calero, P. A. (2001). Formal concept analysis as a support technique for CBR. *Knowledge-Based Systems*, 14(3–4):163–171. DOI: 10.1016/S0950-7051(01)00093-4 20

Díaz-Agudo, B., González-Calero, P. A., Recio-García, J. A., and Sánchez-Ruiz, A. A. (2007). Building CBR systems with jcolibri. *Special Issue on Experimental Software and Toolkits of the Journal Science of Computer Programming*, 69(1-3):68–75. DOI: 10.1016/j.scico.2007.02.004 67

Domeniconi, C. and Gunopulos, D. (2001). Adaptive nearest neighbor classification using support vector machines. In *Neural Information Processing Systems: Natural and Synthetic, NIPS*, pages 665–672. 53

Dubois, D., Prade, H., Esteva, F., Garcia, P., Godo, L., and López de Mantaras, R. (1998). Fuzzy set modelling in case-based reasoning. *International Journal of Intelligent Systems*, 13(4):345–373. DOI: 10.1002/(SICI)1098-111X(199804)13:4%3C345::AID-INT3%3E3.3.CO;2-I 62

Ekbia, H. R. (2008). *Artificial Dreams: The Quest for Non-Biological Intelligence*. Cambridge University Press, 1st edition. DOI: 10.1017/CBO9780511802126 8

Emde, W. and Wettschereck, D. (1996). Relational instance-based learning. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 122–130. Morgan Kaufmann. 38

Ennett, C. M., Frize, M., and Walker, C. R. (2008). Imputation of missing values by integrating neural networks and case-based reasoning. In *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pages 4337–4341. DOI: 10.1109/IEMBS.2008.4650170 38

Esteva, F., Garcia, P., Godo, L., and Rodriguez, R. (1997). A modal account of similarity-based reasoning. *International Journal of Approximate Reasoning*, 16(3–4):235–260. DOI: 10.1016/S0888-613X(96)00126-0 63

Falkenhainer, B., Forbus, K. D., and Gentner, D. (1989). The structure-mapping engine: Algorithm and examples. *Artificial Intelligence*, 41:1–63. DOI: 10.1016/0004-3702(89)90077-5 8

Gabel, T. (2005). On the use of vocabulary knowledge for learning similarity measures. In *Proceedings of the Third Biennial conference on Professional Knowledge Management*, WM, pages 272–283. Springer-Verlag. DOI: 10.1007/11590019_32 37, 52

Gamero, F. I., Meléndez, J., and Colomer, J. (2011). QSSI: A new similaritu index for qualitative time series. Application to classify voltage sags. *Applied Artificial Intelligence*, 25(2):141–162. DOI: 10.1080/08839514.2011.545213 37

Garcia-Piquer, A., Fornells, A., Orriols-Puig, A., Corral, G., and Golobardes, E. (2011). Data classification through an evolutionary approach based on multiple criteria. *Knowledge and Information Systems*, 33:35–56. DOI: 10.1007/s10115-011-0462-9 20

Gelman, A. and Hill, J. (2006). *Data Analysis Using Regression and Multilevel/Hierarchical Models*, chapter Chapter 5. Missing-data imputation. Cambridge University Press. DOI: 10.1017/CBO9780511790942 38

Gentner, D. and Forbus, K. D. (1991). Mac/fac: A model of similarity-based retrieval. *Cognitive Science*, 19:141–205. DOI: 10.1207/s15516709cog1902_1 8

Gilboa, I., Samuelson, L., and Schmeidler, D. (2009). A unified model of induction. `http://economics.mit.edu/files/4558` [Accessed November 2012]. 9, 62

Gilboa, I. and Schmeidler, D. (1995). Case-based decision theory. *The Quarterly Journal of Economics*, 110(3):605–639. DOI: 10.2307/2946694 63

Gilboa, I. and Schmeidler, D. (2001). *A Theory of Case-Based Decisions*. Cambridge University Press. DOI: 10.1017/CBO9780511493539 63

Globig, C., Jantke, K. P., Lange, S., and Sakakibara, Y. (1997). On case-based learnability of languages. *New Generation Computing*, 15(1):59–83. DOI: 10.1007/BF03037560 63

Goel, A. and Murdock, J. (1996). Meta-cases: Explaining case-based reasoning. In Smith, I. and Faltings, B., editors, *Advances in Case-Based Reasoning*, volume 1168 of *Lecture Notes in Computer Science*, pages 150–163. Springer Berlin Heidelberg. 67

Goel, A. K. (1992). Integrating case-based and model-based reasoning: A computational model of design problem solving. *AI Magazine*, 13(2):50–54. 38

Grimnes, M. and Aamodt, A. (1996). A two layer case-based reasoning architecture for medical image understanding. In Smith, I. and Faltings, B., editors, *Advances in Case-Based Reasoning*, volume 1168 of *Lecture Notes in Computer Science*, pages 164–178. Springer Berlin Heidelberg. 8

Gronau, N. and Laskowski, F. (2003). Using case-based reasoning to improve information retrieval in knowledge management systems. In Menasalvas, E., Segovia, J., and Szczepaniak, P., editors, *Advances in Web Intelligence*, volume 2663 of *Lecture Notes in Computer Science*, pages 94–102. Springer Berlin Heidelberg. DOI: 10.1007/3-540-44831-4 9

Guarino, N. (1998). Formal ontology and information systems. In *Proceedings of FOIS'98, Trento, Italy, 6-8 June 1998. Amsterdam*, pages 3–15. IOS Press. 19

Gupta, K., Aha, D., and Moore, P. (2004). Learning feature taxonomies for case indexing. In Funk, P. and González Calero, P. A., editors, *Advances in Case-Based Reasoning*, volume 3155 of *Lecture Notes in Computer Science*, pages 211–226. Springer Berlin Heidelberg. 20, 52

Hammond, K. J. (1986). CHEF: A model of case-based planning. In *AAAI*, pages 267–271. 20

Hammond, K. J. (1989). *Case-based Planning: Viewing Planning as a Memory Task*. Academic Press Professional, Inc. 38

Hanney, K. and Keane, M. (1996). In Smith, I. and Faltings, B., editors, *Advances in Case-Based Reasoning*, volume 1168 of *Lecture Notes in Computer Science*, pages 179–192. Springer Berlin Heidelberg. 52

Hanney, K., Keane, M. T., Cunningham, P., and Smyth, B. (1995). What kind of adaptation do CBR systems need?: A review of current practice. In *In Adaptation of Knowledge for Reuse, Proc. 1995 AAAI Fall Symposium*. AAAI. 38

Hayes, C., Cunningham, P., and Doyle, M. (1998). Distributed CBR using XML. Technical Report Computer Science Technical Report TCD-CS-1998-06, The University of Dublin, Trinity College. 67

Herrero, P. (2011). An insulin boulus calculator based on case-based reasoning. In *Diabetes Technology meeting*. 38

Holyoak, K. J. and Morrison, R. G., editors (2005). *The Cambridge Handbook of Thinking and Reasoning*. Cambridge University Press. 8

Holyoak, K. J. and Thagard, P. (1995). *Mental Leaps: Analogy in Creative Thought*. MIT Press. 8

Horváth, T., Wrobel, S., and Bohnebeck, U. (2001). Relational instance-based learning with lists and terms. *Machine Learning*, 43(1-2):53–80. DOI: 10.1023/A:1007668716498 38

Hullermeier, E. (1999). Toward a probabilistic formalization of case-based inference. In *Proceedings of the 16th International Joint Conference on Artifical Intelligence - Volume 1*, IJCAI, pages 248–253. Morgan Kaufmann Publishers Inc. 63

Hüllermeier, E. (2007). *Case-Based Approximate Reasoning*. Springer. 63

Hüllermeier, E. and Schlegel, P. (2011). Preference-based CBR: first steps toward a methodological framework. In *Proceedings of the 19th International Conference on Case-Based Reasoning Research and Development*, ICCBR'11, pages 77–91. Springer-Verlag. DOI: 10.1007/978-3-642-23291-6_8 38, 63

Iglezakis, I., Reinartz, T., and Roth-Berghofer, T. (2004). Maintenance memories: Beyond concepts and techniques for case base maintenance. In Funk, P. and Calero, P. A. G., editors, *Advances in Case-Based Reasoning*, pages 227–241. Springer-Verlag. 53

Jakulin, A. (2005). *Machine Learning Based on Attribute Interactions*. Ph.D. thesis, University of Ljubljana, Faculty of Computer and Information Science. ECCAI Ph.D. Dissertation Award 2005. 37

Jannach, D., Zanker, M., Felfernig, A., and Friedrich, G. (2010). *Recommender Systems: An Introduction*. Cambridge University Press. DOI: 10.1017/CBO9780511763113 8

Johanyák, Z. C. and Kovács, S. (2005). Distance based similarity measures of fuzzy sets. In *3rd Slovakian-Hungarian Joint Symposium on Applied Machine Intelligence (SAMI)*, pages 265–276. 38

Kamp, G. (1996). Using description logics for knowledge intensive case-based reasoning. In *3rd Europan Workshop on Case-Based Reasoning*, pages 204–218. Springer Verlag. DOI: 10.1007/BFb0020612 62

Klaus-Dieter, A., Birk, A., and Tautz, C. (1997). The experience factory approach: Realizing learning from experience in software development organizations. In *University of Karlsruhe*, pages 6–8. 67

Koehler, J. (1994). An application of terminological logics to case-based reasoning. In *Proceedings of the Fourth International Conference on the Principles of Knowledge Representation and Reasoning (KR-94)*, pages 351–362. Morgan Kaufmann. 62

Kolodner, J. (1993). *Case-Based Reasoning*. Morgan Kaufmann. 8, 20, 36, 38

Kolodner, J. L. (1992). An introduction to case-based reasoning. *Artificial Intelligence Review*, 6(1):3–34. DOI: 10.1007/BF00155578 8

Kuchibatla, V. and Muñoz Avila, H. (2008). An analysis of transformational analogy: general framework and complexity. In *Proceedings of the 23rd national conference on Artificial intelligence - Volume 3*, AAAI, pages 1555–1558. AAAI Press. DOI: 10.1007/11805816_34 38

Langley, P. and Rogers, S. (2005). An extended theory of human problem solving. In *Proceedings of the Twenty-seventh Annual Meeting of the Cognitive Science Society*. 8

Leake, D. and Whitehead, M. (2007). Case provenance: The value of remembering case sources. In *Proceedings of the 7th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development*, ICCBR, pages 194–208. Springer-Verlag. DOI: 10.1007/978-3-540-74141-1_14 67

Leake, D. B., editor (1996). *Case-Based Reasoning: Experiences, Lessons, and Future Directions*. AAAI Press/MIT Press. 8

Leake, D. B. and Sooriamurthi, R. (2002). Automatically selecting strategies for multi-case-base reasoning. In *Proceedings of the 6th European Conference on Advances in Case-Based Reasoning*, ECCBR, pages 204–233. Springer-Verlag. DOI: 10.1007/3-540-46119-1_16 67

Leake, D. B. and Wilson, D. C. (1998). Categorizing case-base maintenance: Dimensions and directions. In *Proceedings of the 4th European Workshop on Advances in Case-Based Reasoning*, EWCBR, pages 196–207. Springer-Verlag. DOI: 10.1007/BFb0056333 53

Limthanmaphon, B. and Zhang, Y. (2003). Web service composition with case-based reasoning. In *Proceedings of the 14th Australasian Database Conference - Volume 17*, ADC '03, pages 201–208. Australian Computer Society, Inc. 9

Lin, D. (1998). An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 296–304. Morgan Kaufmann Publishers Inc. 38

López, B. and Plaza, E. (1997). Case-based learning of plans and goal states in medical diagnosis. *Artificial Intelligence in Medicine*, 9(1):29–60. DOI: 10.1016/S0933-3657(96)00360-0 8, 20, 53

López, B., Pous, C., Gay, P., Pla, A., Sanz, J., and Brunet, J. (2011). eXiT*CBR: A framework for case-based medical diagnosis development and experimentation. *Artificial Intelligence in Medicine*, 51(2):81–91. DOI: 10.1016/j.artmed.2010.09.002 67

López-Arévalo, I., Bañares-Alcántara, R., Aldea, A., Rodríguez-Martínez, A., and Jiménez, L. (2007). Generation of process alternatives using abstract models and case-based reasoning. *Computers & Chemical Engineering*, 31(8):902–918. DOI: 10.1016/j.compchemeng.2006.08.011 8

López de Mantaras, R. and Arcos, J. L. (2002). AI and music from composition to expressive performance. *AI Magazine*, 23(3):43–57. 22

Lopez de Mantaras, R., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M., Cox, M. T., Forbus, K., Keane, M., Aamodt, A., and Watson, I. (2005). Retrieval, reuse, revision and retention in case-based reasoning. *The Knowledge Engineering Review*, 20(03):215–240. DOI: 10.1017/S0269888906000646 8

Lorenzi, F. and Ricci, F. (2005). Case-based recommender systems: a unifying view. In *Proceedings of the 2003 international conference on Intelligent Techniques for Web Personalization*, ITWP'03, pages 89–113. Springer-Verlag. DOI: 10.1007/11577935_5 8

Manzoni, S., Sartori, F., and Vizzari, G. (2005). Towards a general framework for substitutional adaptation in case-based reasoning. In *Proceedings of the 9th conference on Advances in Artificial Intelligence*, AI*IA, pages 331–342. Springer-Verlag. DOI: 10.1007/11558590_34 38

Marling, C., Sqalli, M., Rissland, E., Muñoz Avila, H., and Aha, D. (2002). Case-based reasoning integrations. *AI Magazine*, 23(1):69–86. 8

Massie, S. (2006). *Complexity Modelling for Case Knowledge Maintenance in Case-Based Reasoning*. Ph.D. thesis, The Robert Gordon University, Aberdeen, UK. 49, 53

Maximini, K., Maximini, R., and Bergmann, R. (2003). An investigation of generalized cases. In *ICCBR 2003, Case-Based Reasoning Research and Development, LNAI 2689*, pages 261–276. Springer-Verlag. DOI: 10.1007/3-540-45006-8_22 20

Mcsherry, D. (2005). Explanation in recommender systems. *Artificial Intelligence Review*, 24:179–197. DOI: 10.1007/s10462-005-4612-x 67

Mehta, M., Ontañón, S., and Ram, A. (2009). Using meta-reasoning to improve the performance of case-based planning. In *Proceedings of the 8th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development*, ICCBR, pages 210–224. Springer-Verlag. DOI: 10.1007/978-3-642-02998-1_16 53

Minor, M. and Görg, S. (2011). Acquiring adaptation cases for scientific workflows. In *Proceedings of the 19th international conference on Case-Based Reasoning Research and Development*, ICCBR, pages 166–180. Springer-Verlag. DOI: 10.1007/978-3-642-23291-6_14 20

Minor, M., Tartakovski, A., and Bergmann, R. (2007). Representation and structure-based similarity assessment for agile workflows. In Weber, R. O. and Richter, M. M., editors, *Case-Based Reasoning Research and Development*, volume 4626 of *Lecture Notes in Computer Science*, pages 224–238. Springer Berlin Heidelberg. DOI: 10.1007/978-3-540-74141-1 37

Mitchell, T. M. (1982). Generalization as search. *Artificial Intelligence*, 18(2):203–226. DOI: 10.1016/0004-3702(82)90040-6 52

Montaner, M., López, B., and Rosa, J. L. (2002). Improving case representation and case base maintenance in recommender agents. In Craw, S. and Preece, A., editors, *Advances in Case-Based Reasoning*, volume 2416 of *Lecture Notes in Computer Science*, pages 234–248. Springer Berlin Heidelberg. 53

Montani, S., Bottrighi, A., Leonardi, G., Portinale, L., and Terenziani, P. (2009). Multi-level abstractions and multi-dimensional retrieval of cases with time series features. In McGinty, L. and Wilson, D., editors, *Case-Based Reasoning Research and Development*, volume 5650 of *Lecture Notes in Computer Science*, pages 225–239. Springer Berlin Heidelberg. DOI: 10.1007/978-3-642-02998-1 20, 37

Muñoz Avila, H. (2001). Case-base maintenance by integrating case-index revision and case-retention policies in a derivational replay framework. *Computational Intelligence*, 17(2):280–294. DOI: 10.1111/0824-7935.00145 38

Murdock, J. W. and Goel, A. K. (2001). Meta-case-based reasoning: Using functional models to adapt case-based agents. In *Proceedings of the 4th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development*, ICCBR, pages 407–421. Springer-Verlag. DOI: 10.1007/3-540-44593-5_29 53

Nebel, B. and Koehler, J. (1995). Plan reuse versus plan generation: a theoretical and empirical analysis. *Artificial Intelligence*, 76(1-2):427–454. DOI: 10.1016/0004-3702(94)00082-C 38

Newell, A. (1972). *Human Problem Solving*. Prentice-Hall, Inc. 8

Ontañón, S. and Plaza, E. (2008). An argumentation-based framework for deliberation in multi-agent systems. In *Proceedings of the 4th international conference on Argumentation in multi-agent systems*, ArgMAS, pages 178–196. Springer-Verlag. DOI: 10.1007/978-3-540-78915-4_12 67

Ontañón, S. and Plaza, E. (2012). Similarity measures over refinement graphs. *Machine Learning*, 87(1):57–92. DOI: 10.1007/s10994-011-5274-3 37

Pal, S. K., Dillon, T. S., and Yeung, D. S., editors (2001). *Soft Computing in Case based Reasoning*. Springer-Verlag. 62

Perner, P. (2001). Why case-based reasoning is attractive for image interpretation. In *Proceedings of the 4th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development*, ICCBR '01, pages 27–43. Springer-Verlag. DOI: 10.1007/3-540-44593-5_3 9

Plaza, E. (1995). Cases as terms: A feature term approach to the structured representation of cases. volume 1010, pages 265–276. Springer, Springer. DOI: 10.1007/3-540-60598-3_24 19, 62

Plaza, E., Esteva, F., Garcia, P., Godo, L., and de Màntaras, R. L. (1998). A logical approach to case-based reasoning using fuzzy similarity relations. *Information Science*, 106(1-2):105–122. DOI: 10.1016/S0020-0255(97)10008-1 63

Plaza, E. and McGinty, L. (2005). Distributed case-based reasoning. *The Knowlowdge Engineering Review*, 20:261–265. DOI: 10.1017/S0269888906000683 8, 67

Purvis, L. and Pu, P. (1998). Composer: A case-based reasoning system for engineering design. *Robotica*, 16(3):285–295. DOI: 10.1017/S0263574798000368 38

Ram, A. (1993). Indexing, elaboration and refinement: Incremental learning of explanatory cases. *Machine Learning*, 10:201–248. DOI: 10.1023/A:1022634926452 67

Reinartz, T., Iglezakis, I., and Roth-Berghofer, T. (2000). On quality measures for case base maintenance. In *Proceedings of the 5th European Workshop on Advances in Case-Based Reasoning*, EWCBR, pages 247–259. Springer-Verlag. DOI: 10.1007/3-540-44527-7_22 53

Reinartz, T., Iglezakis, I., and Roth-Berghofer, T. (2001). Review and restore for case-base maintenance. *Computational Intelligence*, 17(2):214–234. DOI: 10.1111/0824-7935.00141 8, 53

Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 1*, IJCAI, pages 448–453. Morgan Kaufmann Publishers Inc. 37

Ricci, F. and Senter, L. (1998). Structured cases, trees and efficient retrieval. In Smyth, B. and Cunningham, P., editors, *Advances in Case-Based Reasoning*, volume 1488 of *Lecture Notes in Computer Science*, pages 88–99. Springer Berlin Heidelberg. 20

Richter, M. M. (1995). The knowledge contained in similarity measures. Invited talk at ICCBR-95, Sesimbra, Portugal. 11, 19

Richter, M. M. (2005). Knowledge containers. 19

Richter, M. M. (2007). Foundations of similarity and utility. In *FLAIRS Conference*, pages 30–. 63

Richter, M. M. and Aamodt, A. (2005). Case-based reasoning foundations. *Knowledge Engineering Review*, 20(3):203–207. DOI: 10.1017/S0269888906000695 8

Riesbeck, C. K. and Schank, R. C. (1989). *Inside Case-Based Reasoning*. Artificial Intelligence Series. Taylor & Francis Group. 8

Ros, R., Arcos, J. L., Lopez de Mantaras, R., and Veloso, M. (2009). A case-based approach for coordinated action selection in robot soccer. *Artificial Intelligence*, 173(9-10):1014–1039. DOI: 10.1016/j.artint.2009.02.004 20

Roth-Berghofer, T. R. (2004). Explanations and case-based reasoning: Foundational issues. In Funk, P. and Calero, P. A. G., editors, *Advances in Case-Based Reasoning*, pages 389–403. Springer-Verlag. 67

Roth-Berghofer, T. R. and Cassens, J. (2005). Mapping goals and kinds of explanations to the knowledge containers of case-based reasoning systems. In *Proceedings of the 6th international conference on Case-Based Reasoning Research and Development*, ICCBR, pages 451–464. Springer-Verlag. DOI: 10.1007/11536406_35 67

Rousu, J. and Aarts, R. J. (1996). Adaptation cost as a criterion for solution evaluation. In *Proceedings of the Third European Workshop on Advances in Case-Based Reasoning*, EWCBR '96, pages 354–361. Springer-Verlag. DOI: 10.1007/BFb0020622 38

Russell, S. J. and Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Pearson Education, 3 edition. 19

Salganicoff, M. (1997). Tolerating concept and sampling shift in lazy learning using prediction error context switching. *Artificial Intelligence Review*, 11:133–155. DOI: 10.1023/A:1006515405170 53

Salotti, S. and Ventos, V. (1998). Study and formalization of a case-based reasoning system using a description logic. In *Proceedings of the 4th European Workshop on Advances in Case-Based Reasoning*, EWCBR, pages 286–297. Springer-Verlag. DOI: 10.1007/BFb0056341 62

Sánchez-Marré, M., Cortés, U., Martínez, M., Comas, J., and Rodríguez-Roda, I. (2005). An approach for temporal case-based reasoning: Episode-based reasoning. In Muñoz Ávila, H. and Ricci, F., editors, *Case-Based Reasoning Research and Development*, volume 3620 of *Lecture Notes in Computer Science*, pages 465–476. Springer Berlin Heidelberg. 19

Sandri, S., Mendonca, J., Martins-Bedé, F. T., Guimarães, R. J. P. S., and Carvalho, O. S. (2012). Weighted fuzzy similarity relations in case-based reasoning: A case study in classification. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE, 2012)*, pages 1–7. DOI: 10.1109/FUZZ-IEEE.2012.6251149 63

Santini, S. and Jain, R. (1999). Similarity measures. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(9):871–883. DOI: 10.1109/34.790428 38

Schank, R. C., Kass, A., (eds.), C. K. R., Brown, M., and Paliouras, G. (1994). *Inside Case-Based Explanation*. 67

Schmidt, R., Vorobieva, O., and Gierl, L. (2003). Adaptation problems in therapeutic case-based reasoning systems. In Palade, V., Howlett, R., and Jain, L., editors, *Knowledge-Based Intelligent Information and Engineering Systems*, volume 2773 of *Lecture Notes in Computer Science*, pages 992–999. Springer Berlin Heidelberg. 38

Schmidt, R. and Waligora, T. (2006). Using prototypes and adaptation rules for diagnosis of dysmorphic syndromes. In *Proceedings of the 6th Industrial Conference on Data Mining conference on Advances in Data Mining: applications in Medicine, Web Mining, Marketing, Image and Signal Mining*, ICDM, pages 1–9. Springer-Verlag. DOI: 10.1007/11790853_1 38

Schmidt, R., Waligora, T., and Vorobieva, O. (2008). Prototypes for medical case-based applications. In *Proceedings of the 8th industrial conference on Advances in Data Mining: Medical Applications, E-Commerce, Marketing, and Theoretical Aspects*, ICDM '08, pages 1–15. Springer-Verlag. DOI: 10.1007/978-3-540-70720-2_1 20

Schumacher, J. and Bergmann, R. (2000). An efficient aproach to similarity-based retrieval on top of relational databases. In *Proceedings of the 5th European Workshop on Advances in Case-Based Reasoning*, EWCBR, pages 273–284. Springer-Verlag. DOI: 10.1007/3-540-44527-7_24 20

Sebastiani, F. and Straccia, U. (1991). A computationally tractable terminological logic. In *In Proceedings of SCAI-91, 3rd Scandinavian Conference on Artificial Intelligence*, pages 307–315. 62

Sengupta, A., Wilson, D. C., and Leake, D. B. (1999). On constructing the right sort of CBR implementation. In *In Proceedings of the IJCAI-99 Workshop on Automating the Construction of Case Based Reasoners*. 20

Sharma, M., Ontañon, S., Mehta, M., and Ram, A. (2010). Drama management and player modeling for interactive fiction games. *Computational Intelligence*, 26(2):183–211.
DOI: 10.1111/j.1467-8640.2010.00355.x 9

Shimazu, H. (1998). A textual case-based reasoning system using xml on the world-wide web. In *Proceedings of the 4th European Workshop on Advances in Case-Based Reasoning*, EWCBR '98, pages 274–285. Springer-Verlag. DOI: 10.1007/BFb0056340 8, 20

Slade, S. (1991). Case-based reasoning: a research paradigm. *AI Magazine*, 12(1):42–55. 8

Smyth, B. (1998). Case-base maintenance. In Pasqual del Pobil, A., Mira, J., and Ali, M., editors, *Tasks and Methods in Applied Artificial Intelligence*, volume 1416 of *Lecture Notes in Computer Science*, pages 507–516. Springer Berlin Heidelberg. 53

Smyth, B. and Cunningham, P. (1996). The utility problem analysed - a case-based reasoning perspective. In *In Proceedings of the Third European Workshop on Case-Based Reasoning*, pages 392–399. Springer Verlag. DOI: 10.1007/BFb0020625 53

Smyth, B. and Keane, M. T. (1995). Remembering to forget: a competence-preserving case deletion policy for case-based reasoning systems. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI, pages 377–382. Morgan Kaufmann Publishers Inc. 53

Smyth, B. and McKenna, E. (1998). Modelling the competence of case-bases. In *Proceedings of the 4th European Workshop on Advances in Case-Based Reasoning*, EWCBR, pages 208–220. Springer-Verlag. DOI: 10.1007/BFb0056334 51, 53

Sørmo, F., Cassens, J., and Aamodt, A. (2005). Explanation in case-based reasoning—perspectives and goals. *Artificial Intelligence Review*, 24(2):109–143. DOI: 10.1007/s10462-005-4607-7 67

Sqalli, M., Purvis, L., and Freuder, E. C. (1999). Survey of applications integrating constraint satisfaction and case-based reasoning. In *in Procs. of the First International Conference and Exhibition on The Practical Application of Constraint Technologies and Logic Programming*. 38

Stahl, A. (2002). Defining similarity measures: Top-down vs. bottom-up. In Craw, S. and Preece, A., editors, *Advances in Case-Based Reasoning*, volume 2416 of *Lecture Notes in Computer Science*, pages 406–420. Springer Berlin Heidelberg. 38

Stahl, A. and Gabel, T. (2003). Using evolution programs to learn local similarity measures. In *Proceedings of the 5th International Conference on Case-based Reasoning: Research and Development*, ICCBR, pages 537–551. Springer-Verlag. DOI: 10.1007/3-540-45006-8_41 52

Stanfill, C. and Waltz, D. (1986). Toward memory-based reasoning. *Communications of ACM*, 29(12):1213–1228. DOI: 10.1145/7902.7906 8, 37

Tintarev, N. and Masthoff, J. (2007). A survey of explanations in recommender systems. In *Proceedings of the 2007 IEEE 23rd International Conference on Data Engineering Workshop*, ICDEW, pages 801–810. IEEE Computer Society. DOI: 10.1109/ICDEW.2007.4401070 67

Tirri, H., Kontkanen, P., and Myllymäki, P. (1996). A bayesian framework for case-based reasoning. In *Proceedings of the Third European Workshop on Advances in Case-Based Reasoning*, EWCBR, pages 413–427. Springer-Verlag. DOI: 10.1007/BFb0020627 62

Tishby, N., Pereira, F. C., and Bialek, W. (1999). The information bottleneck method. In *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377. 53

Torra, V. and Narukawa, Y. (2007). *Modeling Decisions. Information Fusion and Aggregation Operators*. Springer. 38

Tversky, A. (1977). Features of similarity. *Psychological Review*, 84(4):327–352. DOI: 10.1037/0033-295X.84.4.327 37

Tversky, A. (2003). *Preference, Belief, and Similarity: Selected Writings*. The MIT Press. 38

Tversky, A. and Simonson, I. (1993). Context-dependent preferences. *Management Science*, 39(10):1179–1189. DOI: 10.1287/mnsc.39.10.1179 38

Veloso, M., Carbonell, J., Pérez, A., Borrajo, D., Fink, E., and Blythe, J. (1995). Integrating planning and learning: the prodigy architecture. *Journal of Experimental & Theoretical Artificial Intelligence*, 7(1):81–120. DOI: 10.1080/09528139508953801 8, 20

Veloso, M. M. (1994). Flexible strategy learning: analogical replay of problem solving episodes. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (vol. 1)*, AAAI, pages 595–600. American Association for Artificial Intelligence. 38

Veloso, M. M. and Carbonell, J. G. (1993). Derivational analogy in Prodigy: Automating case acquisition, storage, and utilization. *Machine Learning*, 10:249–278. DOI: 10.1023/A:1022686910523 38

Voß, A. (1996). Principles of case reusing systems. In Smith, I. and Faltings, B., editors, *Advances in Case-Based Reasoning*, volume 1168 of *Lecture Notes in Computer Science*, pages 428–444. Springer Berlin Heidelberg. 38

Watson, I. (1997). *Applying Case-based Reasoning: Techniques for Enterprise Systems*. The Morgan Kaufmann Series in Artificial Intelligence. Morgan Kaufmann. 9

Watson, I. (1999). Case-based reasoning is a methodology not a technology. *Knowledge-Based Systems*, 12(5-6):303–308. DOI: 10.1016/S0950-7051(99)00020-9 8

Watson, I. (2001). Knowledge management and case-based reasoning: a perfect match. In Russel I, Kolen J (eds) *Proc. of the Fourteenth Annual Conference of the International Florida Artificial Intelligence Research Society*. AAAI Press, Menlo Park, CA, pages 118–122. AAAI Press. 9

Watson, I. (2003). *Applying Knowledge Management: Techniques for Building Corporate Memories*. The Morgan Kaufmann Series in Artificial Intelligence. Morgan Kaufmann. 9

Weber, B., Reichert, M., and Wild, W. (2006). Case-base maintenance for CCBR-based process evolution. In Roth-Berghofer, T., Göker, M., and Güvenir, H., editors, *Advances in Case-Based Reasoning*, volume 4106 of *Lecture Notes in Computer Science*, pages 106–120. Springer Berlin Heidelberg. DOI: 10.1007/11805816 8

Weber, B., Wild, W., and Breu, R. (2004). CBRFlow: Enabling adaptive workflow management through conversational case-based reasoning. In Funk, P. and González Calero, P. A., editors, *Advances in Case-Based Reasoning*, volume 3155 of *Lecture Notes in Computer Science*, pages 434–448. Springer Berlin Heidelberg. 9

Weber-Lee, R., Barcia, R.-M., Martins, A., and Pacheco, R. C. (1996). Using typicality theory to select the best match. In Smith, I. and Faltings, B., editors, *Advances in Case-Based Reasoning*, volume 1168 of *Lecture Notes in Computer Science*, pages 445–459. Springer Berlin Heidelberg. 38

Wess, S. and Globig, C. (1993). Case-based and symbolic classification - a case study using version space. In *Selected papers from the First European Workshop on Topics in Case-Based Reasoning*, EWCBR, pages 77–91. Springer-Verlag. DOI: 10.1007/3-540-58330-0_78 52

West, G. M. and McDonald, J. R. (2003). An sql-based approach to similarity assessment within a relational database. In *Proceedings of the 5th International Conference on Case-based Reasoning: Research and Development*, ICCBR, pages 610–621. Springer-Verlag. DOI: 10.1007/3-540-45006-8_46 20

Wettschereck, D., Aha, D. W., and Mohri, T. (1997). A review and empirical evaluation of feature weighting methods for aclass of lazy learning algorithms. *Artificial Intelligence Review*, 11(1-5):273–314. DOI: 10.1023/A:1006593614256 53

Wilson, D. R. and Martinez, T. R. (1996). Value difference metrics for continuously valued attributes. In *Proceedings of the International Conference on Artificial Intelligence, Expert Systems and Neural Networks*, pages 11–14. 37

Wilson, D. R. and Martinez, T. R. (1997). Improved heterogeneous distance functions. *Journal of Artificial Intelligence Research*, 6:1–34. DOI: 10.1613/jair.346 38

Witten, I. H., Frank, E., and Hall, M. A. (2011). *Data Mining: Practical Machine Learning Tools and Techniques (Third Edition)*. Morgan Kaufmann. 39

Wu, Z. and Palmer, M. (1994). Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, ACL '94, pages 133–138. Association for Computational Linguistics. DOI: 10.3115/981732.981751 37

Xing, E. P., Ng, A. Y., Jordan, M. I., and Russell, S. (2002). Distance metric learning, with application to clustering with side-information. In *Advances in Neural Information Processing Systems 15*, pages 505–512. MIT Press. 53

Yager, R. (1997). Case based reasoning, fuzzy systems modeling and solution composition. In Leake, D. and Plaza, E., editors, *Case-Based Reasoning Research and Development*, volume 1266 of *Lecture Notes in Computer Science*, pages 633–642. Springer Berlin Heidelberg. 20

Yang, Q. and Wu, J. (2001). Enhancing the effectiveness of interactive case-based reasoning with clustering and decision forests. *Applied Intelligence*, 14(1):49–64. DOI: 10.1023/A:1008303008006

Yu, J., Amores, J., Sebe, N., Radeva, P., and Tian, Q. (2008). Distance learning for similarity estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):451–462. DOI: 10.1109/TPAMI.2007.70714 53

Zhang, S., Qin, Z., Ling, C. X., and Sheng, S. (2005). Missing is useful: Missing values in cost-sensitive decision trees. *IEEE Transactions on Knowledge and Data Engineering*, 17(12):1689–1693. DOI: 10.1109/TKDE.2005.188 39

Zhu, J. and Yang, Q. (1999). Remembering to add: competence-preserving case-addition policies for case-base maintenance. In *Proceedings of the 16th International Joint Conference on Artifical Intelligence - Volume 1*, IJCAI, pages 234–239. Morgan Kaufmann Publishers Inc. 53

Author's Biography

BEATRIZ LÓPEZ

Beatriz López graduated in Computer Science from the Autonomous University of Barcelona in 1986. Two years later she joined the Artificial Intelligence Research Group of the Spanish Scientific Research Council where she received her Ph.D. in Computer Science from the Technical University of Catalonia in 1993 for the work "Case-base reasoning of strategic plans." From that point on, her research interest has always been around case-based reasoning and planning and scheduling, now including optimization and learning in distributed environments. She was associate professor from 1992-1995 and 1998-2000 at the Rovira Virgili University and has served as a Computer Science Engineer in several private companies. In February 2011, she co-founded Newronia, a spin-off company from the University of Girona. Since 2000, she has been a senior lecturer in the Department of Electronics, Electricity, and Automation Engineering at the University of Girona. Taught courses include Artificial Intelligence and Machine Learning, in which case-based reasoning is embedded. She is member of the Catalan Association for Artificial Intelligence (member of ECCAI) and several scientific committees.