eXiT*CBR.v2: Distributed case-based reasoning tool for medical prognosis

Albert Pla, Beatriz López, Pablo Gay, Carles Pous

University of Girona, Girona, Spain
{*albert.pla,beatriz.lopez,pablo.gay,carles.pous*}*@udg.edu*

Abstract

In this work we propose a user-friendly medically oriented tool for prognosis development systems and experimentation under a case-based reasoning methodology. The tool enables health care collaboration practice to be mapped in cases where different doctors share their expertise, for example, or where medical committee composed of specialists from different fields work together to achieve a final prognosis. Each agent with a different piece of knowledge classifies the given cases through metrics designed for this purpose. Since multiple solutions for the same case is useless, agents collaborate among themselves in order to achieve a final decision through a coordinated schema. For this purpose, the tool provides a weighted voting schema and an evolutionary algorithm (genetic algorithm) to learn robust weights. Moreover, to test the experiments, the tool includes stratified cross-validation methods which take the collaborative environment into account. In this paper the different collaborative facilities offered by the tool are described. A sample usage of the tool is also provided.

Keywords: medical prognosis, case-based reasoning, multi-agent system, distributed reasoning, genetic algorithms,

1. Introduction

Medical research and practice is no longer conceived without the support of decision support tools that facilitate the tasks of the physicians. There are currently a lot of tools available off-the-shelf, but most of them are deliberately generic, (that is, designed to be used in any domain), while the focus of the tool is the kind of technique it employs (e.g. DROOLS for rule-based systems [1], or jCOLIBRI for case-based reasoning (CBR) [2]). Technique oriented tools support decision making according to the underlying theory, but the development of domain oriented tools brings the advantage of providing an adequate interface to their targeted users, offering added value as well as enhancing technology acceptance [3]. In the medical field, several particularities have been identified in [4], and we have addressed some of them with the development of a medically specific case-based reasoning tool called eXiT*CBR [5]. In particular, we provide support and interpretation of results in medical metrics, experiment reproducibility and heterogeneous data management.

eXiT*CBR was originally designed to provide support to isolated users, but nowadays physicians work in teams, sharing information among several units (endocrinology, paediatrics, etc) of a given hospital. As such, we have extended our tool to support the collaboration of medical collectives.We have renamed the tool as eXiT*CBR.v2 to distinguish it from the previous version. In the first version we considered a plain data input - a table - in which rows are cases and columns are attributes. This kind of plain input was used by physicians mainly for research purposes, as it is the format in which statistical tools such as SPSS [6] work. However, physicians are starting to adopt electronic health record (EHR) standards for sharing information with a view to improving the quality of health care [7]. Such standards capture health care data comprehensively, while enabling interoperability among hospitals and research projects. The manner in which individual clinical statements are recorded determines the context for their interpretation [7].Thus keeping the EHR structure in decision support tools, so that EHR fragments correspond to particular clinical settings, seems to be the best way of maintaining the appropriate context for their interpretation. In the second version of the tool, presented in this paper, we enhance it with a multi-agent approach that allows such context to be preserved.

Although the tool has undergone many changes due to the shift from isolated to distributed environments, we have retained the v1 interface in order to minimise possible disruptions in the work of current v1 users who wish to adopt the newest version of the tool. Thus, we have added additional working modes that allow the collaboration of different medical teams, offering privacy and scalability factors for the development of medical applications. First, privacy refers to the willingness to keep data apart from a centralised case repository [8]. This is an important issue in medicine, where physicians exchange opinions or recommendations, for example, but rarely patient data.

The question of patient data ownership is unresolved and bound by legal considerations [4, 7]. And second, it is very important to consider scalability, which "concerns the impracticability of processing a centralised case base when dealing with very large amounts of data" [8]. Contemporary data are not just medical notes and clinical information, but also high-quality images (from magnetic resonance imaging devices or computer tomography scans), process signals (from spirometers) and other formats that require thousands of bytes to be stored. Additionally, problem coverage can be improved when problems are solved by using different experts, and thus, wider knowledge [8]. The purpose of the distributed approach offered by the eXiT*CBR.v2 tool is to enable the collaborative work techniques in the previous eXiT*CBR.v1 platform.

eXiT*CBR.v2 can help in the development of medical applications in which several CBR systems cooperate in a distributed scenario. Physicians and knowledge engineers can use the tool work together to determine the parameters of a case-based reasoning application, including collaboration among different clinical units. The tool includes a coordination mechanism to enable cooperation among different CBR systems, a learning facility to enhance cooperation, and a validation method which takes into account the distributed environment. Like its predecessors, the current scope of eXiT*CBR.v2 includes prognosis, thus supporting the development of decision support systems for predicting the likely output of a disease, and similar classification tasks.

This paper is organised as follows: first, in Section 2, we briefly introduce the fundamentals needed to understand our work. In Section 3, we then present the architecture of our tool and its functionalities. In Section 4, we illustrate the performance of our approach employed for breast cancer prognosis. In Section 5, we compare the developed tool with other similar softwares. Finally, in Section 6, we present our conclusions and proposals for future work.

2. Background

In this section we introduce the main topics on which our software is based, which are namely: case-based reasoning (CBR), cooperative multi-agent systems and learning, and distributed case-based reasoning.

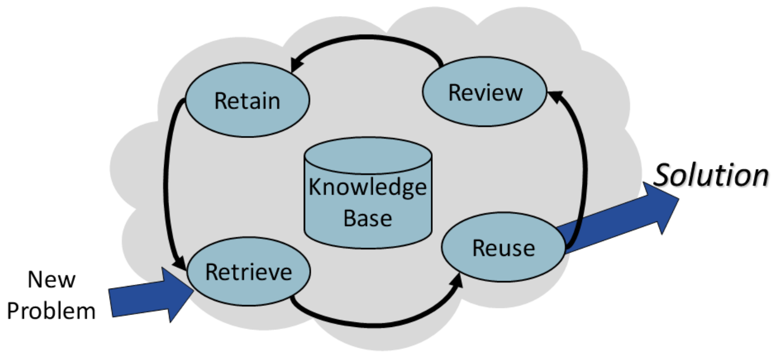


Figure 1: CBR stages from [9]

2.1. Case-based Reasoning

Case-based reasoning is a knowledge-based methodology which allows a problem to be solved based on past experiences [9]. Reasoning by reusing past experiences is a powerful and common way for humans to solve problems. When faced with a problem, humans tend to search for similar past situations and adapt the solution used to solve the current issue they are dealing with. Case-based reasoning tries to transfer this behavior to Artificial Intelligence. CBR seeks cases similar to the current one and analyses which decision or classification was taken in order to reuse it in the present solution.

All CBR shares a set of common tasks: identify the problem to be dealt with; find a similar past case; use the old case to propose a solution for the new one; evaluate the suggested solution and update the knowledge base with the new experience. CBR is consistently divided into four different stages, which are repeated for every new case [9]: retrieve, reuse, revise and retain (see Figure 1). The first stage, retrieve, searches for past situations similar to the new one; the second stage, reuse, employs the retrieved situations to propose a suitable classification or solution to the problem; the third stage, revise, consists of supervising and validating the proposed classification (this task is often carried out by a human expert); in the last stage, retain, it must be decided wether or not the treated case should be included in the knowledge database in order to help in future situations. See [10] for a discussion about CBR as a special case of inductive inference, and their relationship with rule-based and Bayesian reasoning.

The amalgam of options a CBR system designer is faced with in order to build a CBR system is wide, due to the many different parameters and techniques that can be used in each step. 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 flexibility and modularity to include new methods as required [11]. However, although generic tools can help, specific tools for a given domain can constrain the search as well as provide user interfaces adequate to the domain, as for example the first version of eXiT*CBR.

2.2. Cooperative Multi-agent Systems

Agents have been defined as autonomous, flexible computer systems able to interact with other agents in order to achieve a goal [12]. How agents interact in a multi-agent system (MAS) is the focus of study of distributed artificial intelligence [13, 14]. In a MAS, agents carry out a social activity thanks to a coordination mechanism. This mechanism consists of two main components: agent communication and agent interaction protocols [15].

On the one hand agent communication protocols define the structure of messages the agents should follow in order to understand each other. In this respect, although, there are some standards (i.e. FIPA-IEEE [16], KQML [17]), they are often complex to manage (i.e. too much information in a message), therefore ad-hoc communication protocols are often used, which simplifies message processing.

On the other hand, agent interaction protocols provide patterns for dialogues, so that agents can start and end an interaction with success (i.e. avoiding endless waiting for answers due to communication failures, recognising uncommitted actions of malicious agents, etc.). Interaction protocols are designed according to the coordination goal, and comprehend voting, auctions, bargaining, market-based mechanism, contract-net, and coalitions [15]. The protocol is conditioned by the number of agents involved. For example, voting is usually recommended when a great number of agents are involved in a decision.

The problem a MAS designer is faced with is to choose the appropriate coordination mechanism given the problem to be solved. The mechanism will be strongly conditioned depending on wether the environment is cooperative or competitive. In a cooperative scenario, agents collaborate in order to solve a common goal. A cooperative system offers an improvement of computational efficiency, exploiting the different competencies (abilities, capacities) of agents, and increases the information available for solving a problem (as, for example, with a medical diagnostic problem in which several teams are involved). In a competitive scenario, agents are selfish and interact because they can increase their benefits by doing so.

Thus, cooperation is usually achieved by interaction protocols such as voting and contract-net, while competitive MASs require the use of auctions, bargaining, marked-mechanisms and others. Nevertheless, there is no crisp categorisation of interaction protocols between cooperative and competitive ones.

Finally, there are some tools to facilitate the implementation of a MAS, which can be grouped as engineering or platforms tools. Engineering tools facilitate the definition of roles of agents and their interaction, following higher abstract approaches developed using UML-like methodologies. Gaia [18] and Prometheus [19] are examples of very well-known tools in this first group. Platforms deal with low level communication issues, so that agent developers can focus on the strategic behavior of agents. JADE [20] is perhaps the leading open source platform, and JACK is a popular commercial tool. Naturally, some platforms are also designed to support some engineering tools (as JACK is for Prometheus).

2.3. Multi-agent Systems and Learning

Machine learning is concerned with the improvement of intelligent agents through their experience [21]. Machine learning has been an active research topic in MAS, and it faces the following problems [15]:

- Learning about other agents
- Learning for coordination improvement
- Learning about communication.

First, agents are interested in improving their skills once they have interacted with other agents. In a cooperative scenario, agents can learn their role so that all agents complement each other, achieving a trade off between quality and cost [22]. In a competitive scenario, agents can learn about their opponent in order to improve their strategy. Second, as the results of agents interactions emerge, agents can work towards better coordination to improve their global behavior. From such improvement, conflicts on shared resources can be minimised. Lastly, learning about communication processes leads to improvement and a reduction in communication load.

The most popular techniques employed in MASs for learning are reinforcement learning and genetic algorithms (GA). The former is concerned with the learning of control policies by experimentation in a given environment [23]. Learning is assessed by a reward function that assigns a numerical value (pay off) to the actions taken by the agents. In the case of a MAS, the environment includes other agents, so therefore agents improve their global behavior by sharing learnt policies [24]. In contrast, GAs are adaptive algorithms that solve problems by following a computational model of natural evolutionary systems [23]. Candidate solutions to problems are coded as chromosomes of a population, which is then evolved according to genetic operators (crossover, mutation) until an improved solution is found. Haynes et al. [25] were the first ones to apply this technique to discover cooperative coordination mechanism in agents.

2.4. Distributed Case-based Reasoning

Case-based reasoning has been introduced in recent years for coordination learning and reasoning in what is known as distributed case-based reasoning [26, 8]. In [8] the authors propose a classification for distributed case-based systems according to two dimensions: knowledge and processing. For knowledge, there are single or multiple case bases. For processing, two kinds of approaches can be distinguished: single or multiple agents.

Multi-case-based reasoning (MCBR) works by enriching a case-based local with cases from other bases with different task or execution environments [27]. MCBR research then focuses on strategies to decide when to access case bases and how to apply their cases.

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[28, 29]. Thus, in addition to the local-global principle, a social policy should be added to CBR. While the local-global principle [30] 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 case bases or agents.

In [31] some schemas for agent collaboration on problem solving are proposed, including several voting approaches. Moreover, several schemas for case exchange based on the machine learning dimension of case-based reasoning are provided.

The approach presented in this paper is simple, with the aim being to support collaborative prognosis, as well as to bring decision support systems

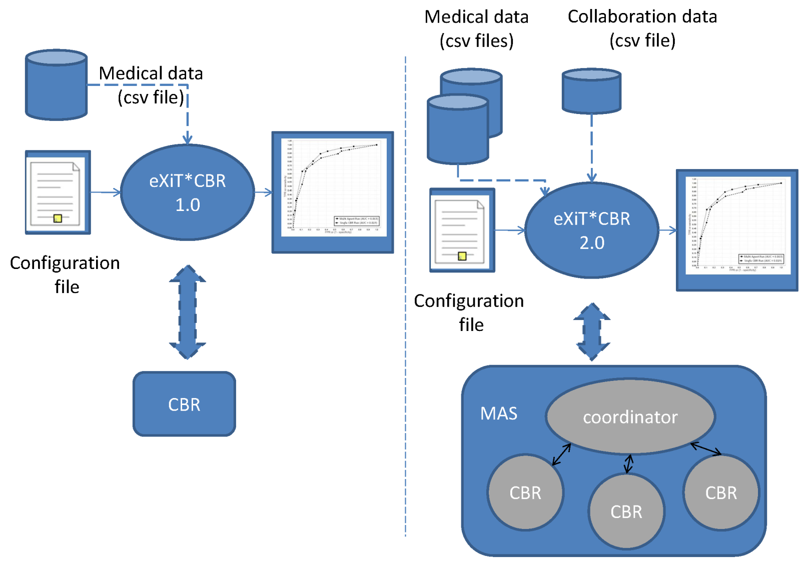


Figure 2: eXiT*CBR. Left: version 1.0, tool for designing isolated CBR systems. Medical data includes vocabulary, cases and attribute weights. Right: version 2.0, tool for designing multi-agent cooperative CBR systems. Several vocabularies, cases and weights can be provided; optionally, information of agent relevance (collaborative data) can be provided, or otherwise learned.

closer to actual clinical practice. Further research should consider other distributed case-based reasoning techniques. Thus, we focus on multiple agents, each agent with a single case base.

3. eXiT*CBR.v2 Architecture

eXiT*CBR.v2 has been projected to support the design of distributed case-based reasoning under a cooperative schema. Like its predecessor, it is domain specific for medical prognosis, designed to deal with the particular requirements of this domain as explained in [4]. In a nutshell, the tool supports development of and experimentation with new case-based reasoning systems. It does this by providing a user-friendly navigation interface which facilitates the analysis of experiments. It is adapted to the information physicians use to work (receiver operator curves, ROC), and supports experiment reproducibility by storing and tracking experiment information [5]. In this paper we discuss new features of the v2 tool that complement the cooperative nature of medical teams. They are: a cooperative coordination mechanism, a learning mechanism for coordination, and MAS validation methods.

*3.1. eXiT*CBR Basics*

The tool is designed for conducting experiments which predict the likelihoods of a patient illness under case-based reasoning methodology. Targeted users include health care and medical staff working in collaboration. Users can apply the tool to analyse the impact on prognosis and to consider the inclusion/exclusion of certain variables, collaborators and derivable information (such laboratory tests, radiology proves).

Physicians are required to provide the knowledge in CSV files that can be managed by SPSS tools and the like. Knowledge contained in the CSV file includes the vocabulary, case-base, and attribute relevance [32]. First, vocabulary consists of attribute-value pairs, allowing numerical, categorical and textual data representation. The vocabulary is provided together with cases, in the headers (first rows) of the CSV file.

Second, the case base is provided in the CSV file, so that columns represent attributes, and rows represent cases. Since the tool is concerned with prognosis, one of the attributes is labelled as the class, with value 0 or 1 to indicate a negative result (the person has not suffered the illness) or positive (the person is ill)[1]. The use of case-based reasoning is, given some person data, obtain the prognosis. The novelty of version 2 is that one or multiple case bases may be used, enabling EHR handling through multiple case-based agents.That is to say, when each EHR fragment is handled by an agent (multi-agent system), the context of the EHR fragment's interpretation can be maintained. EHR fragments are provided in CSV files, one fragment per agent.

And third, attribute relevance (weights) to be used in similarity measures can be specified as special rows of the CSV files.

In addition to these classical knowledge repositories, when dealing with a distributed system there is a need for information on the social policies that govern agent coordination, and on how different EHR data contribute to the final prognosis. Different EHR components are represented by independent CSV files, and thus, agents in the distributed CBR system. The tool offers the possibility of using an external file (i.e. a weights file) for expressing the

[1]This values are exchangeable.

relevance of each agent in the final result. The weights file also follows a CSV format to facilitate its manipulation by editing tools commonly used by physicians.

About the methods for use at every CBR stage, namely retrieve, reuse, revise, and retain, the architecture of eXiT*CBR enables the incorporation of user-defined methods while the tool, as it is, provides some methods by default. These methods can be selected and parameterised from a friendly interface, which permits the configuration file to be set up for the experiments to be carried out. Thus, from the configuration file, a distributed CBR system is generated (see Figure 2). Next, results of the experimentation are graphically shown to the user.

Retrieve methods involve similarity measures, some which are local (Hamming, Euclidean) and others global (e.g., average, weighted average). They can be selected in the configuration file. At the current tool stage, simple reuse methods (as the one described in [33]) are provided that do not require particular knowledge. The inclusion of reuse knowledge, as simple rules, such as those proposed for diabetes in [34], could be considered in the near future. Regarding revision, in prognosis there is only two possible situations: the prognosis is right or it is not. As prognosis involves future, it is difficult to assess whether it is right or not when a system is running in a real environment. However, eXiT*CBR is devoted to experimentation and used to test a user CBR system configuration according to some validation methodology (see Section 3.4) in which the outcome of a problem is known in advance (target). Therefore, it is possible to check if the current configured CBR system provides right answers, by comparing the CBR system outcome with the target. This is the method provided by default in eXiT*CBR. The user, however can include in the tool other methods when required. Finally, there is no retain method provided by the tool. The automatic incorporation of cases in medical applications, without the supervision of a professional, is still a matter of discussion in this domain. eXiT*CBR can be used once and again for testing the CBR system resulting of changing the case base or other system features that could be updated in the retain stage (as for example, attribute relevance). Hence, the tool provides methods by default for the stages of CBR as typically they are used in medical applications [35].

Finally, knowledge engineers can support physicians in experiments to determine even more suitable parameters and methods, so as to exploit all of CBR and MAS potentials. Engineers can add methods to the tool, as it follows a modularity approach. Hybridisation with other reasoning methods

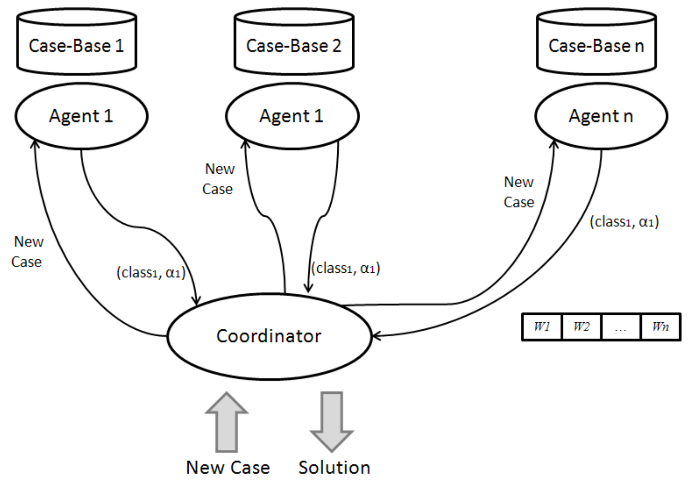


Figure 3: Agent voting schema

as well as with data mining tools is also possible. See [5] for further details.

3.2. Cooperative Coordination Mechanism

Cooperative case-based reasoning is deployed by assigning a different case base to each agent, which is guided by the data organisation provided by physicians. Organisational structure can be defined according to two different approaches [15]:

- Functional decomposition by expertise, so that data from the endocrinologist or the cardiologist is kept separately, while prognosis with a given patient is shared.
- Spatial decomposition by information source or decision point, where some overlapping of data may occur. Exchanges of experiences with different patients are used to asses a new prognosis.

Both organisations are valid mechanisms in a medical domain, and can be uniformly dealt with in eXiT*CBR.v2. Thus, there are as many CBR agents as departments or sources of information. CBR agents are able to interact with each other, they provide their own opinion (prognosis and confidence), then, a coordinator agent makes a final decision based on a weighted voting schema (see Figure 3).

When a new case C needs to be solved, the coordinator agent delivers the case to every CBR agent. Each agent j computes its own prognosis following a case-based reasoning behavior. As a result, each agent sends its own outcome back to the coordinator (e.g., 0 for negative or healthy cases and 1 for positives or illness samples) and a confidence δ in its prognosis. Both, prognosis and δ are the outcome of the CBR retrieve and revision stages. Observe than in isolated systems, δ is not provided but it is in the MAS approach. The δ value is determined as a function of the cases most similar to C and the ratio between positive and negative solutions of these most similar cases is also taken into account (see [33] for further details). All the agents follow the same retrieve and reuse methods, but they use different knowledge (vocabulary, cases or attribute relevance, depending on the organizational structure).

The coordinator agent determines the solution to the case based on the solutions of the different agents. For that purpose, the coordinator keeps a weight w_j concerning the reliability on each agent j (some other authors consider these weights as trusts [36]). The weights have either been entered by the users or learned by the tool (see next Section). Using agent weights and confidences in the prognosis, the coordinator agent collects all the evidence in relation to positive evidence (v^+) and negative cases (v^-) according to the following weighted voting schema:

$$v^+ = \frac{\sum_{class_j=+} w_j * \delta_j}{\sum_{class_j=+} w_j} \tag{1}$$

$$v^- = \frac{\sum_{class_j=-} w_j * \delta_j}{\sum_{class_j=-} w_j} \tag{2}$$

where w_j is the agent weight and δ_j is the agent's confidence in its prognosis. If v^+ is greater than v^- the coordinator classifies the case as positive (illness), a lesser result will be classified as negative (healthy).

Although Equations 1 and 2 are similar to the ones usually used in the retrieval phase, they aggregate information to work towards multi-agent system outcomes. These equations are related to the social component of distributed case-based reasoning, and operate at a higher level than the local-global principle discussed in [30] (see also Section 2.4).

3.3. Coordination Learning Using Genetic Algorithms

Weights quality is directly related to prognosis accuracy; consequently, the process of assigning a weight to each agent is not trivial. In terms of EHR, physicians may assume that some data related to a certain illnes are more informative than others, or that the data reliability depends on the health care staff that provide it. To facilitate this task, eXiT*CBR.v2 provides a feature to learn agents weights using a GA [23].

Weight learning is performed by the coordinator agent, which interacts with all of the case-based agents. In this sense, it is important to observe that weight learning focuses on the improvement of the coordination mechanism, and tries to model a trust value for each case-based agent. For example, some physicians prefer the information coming from one radiologist as opposed to another, because they trust the former's experience much more. Information reliability is handled by the agent (people, hospital units) and not at the case level. Thus, weight learning should be considered as a learning facility which is complementary to the one provided by case-based reasoning methodology. In the future, a case-based system can be added inside the coordinator to fine tune the trust obtained by the GA. The GA solves the cold start problem.

The GA implementation we made consists of the following steps:

1. Create a randomly generated population of phenotypes
2. Calculate the fitness function for each phenotype in the population
3. Sort the phenotype population from best to worst
4. For each phenotype
   (a) Select a phenotype to form a pair
   (b) With a probability p_c cross them over to form a pair of offspring
   (c) With a probability p_m mutate the pair of offspring
5. Mash the best phenotypes of each population in order to create a new one.
6. Repeat step 2 until the population error is stacked.

In our implementation, each phenotype is composed of a set of weights (one for each agent). The fitness function consists of building and testing the distributed CBR system that follows the phenotype cr_x and obtaining an evaluation rate on a cases set. It is then defined as follows:

$$fitness(cr_x) = (1 - error(cr_x))^3 \tag{3}$$

$$error(cr_x) = \frac{\sum_{j=1}^{l} |real_j - predicted_j|}{l} \tag{4}$$

Where l is the number of cases in which the GA is tested, $real_j$ is the real class for case j, and $predicted_j$ is the CBR classification result. As a result, at the end of the GA run, the phenotype with a higher fitness function contains the best set of weights.

GAs usually overfits the cases set [37]. To reduce the incidence of overfitting, the process of using a GA to determine the weights is repeated several times with different cases sets. Thus, after m runs of the GA on a number of n CBR agents, we get a collection of m possible weights for each agent. The m different results obtained are then averaged through multi-criteria decision methods (MCDM) (see Figure 4). These techniques significantly increase system performance and robustness [38]. eXiT*CBR.v2 offers the option of using different MCDM based on different statistics (means, voting, errors, etc) for combining weights obtained in the GA runs on different datasets, namely:

- Mean value: the weights obtained are the mean weights of the different GA runs.
- Error based: the weights are calculated using the prediction error obtained during the GA runs. Agents with a lower prediction error are those with a higher weight.
- Rated ranking: This approach ranks the agents in each GA from the highest to lowest weight. Agents that obtain a higher mean ranking are those with higher weights.
- Voted ranking. As with the previous method, in this approach the agents are ranked according to the weight obtained in each GA run, then, final weights are assigned according to the most repeated ranking position in the GA runs.

3.4. MAS Cross-Validation Methods

As stated in the introduction, the aim of eXiT*CBR.v2 is to facilitate the development of CBR systems that are composed by a user-selected set of methods and their parameters. In order to evaluate the accuracy and other measures of the created system, a validation method is required to test them.

Stratified cross-validation is a very well-known technique that is used when small data sets are available [39, 40]. This is often the case with medical

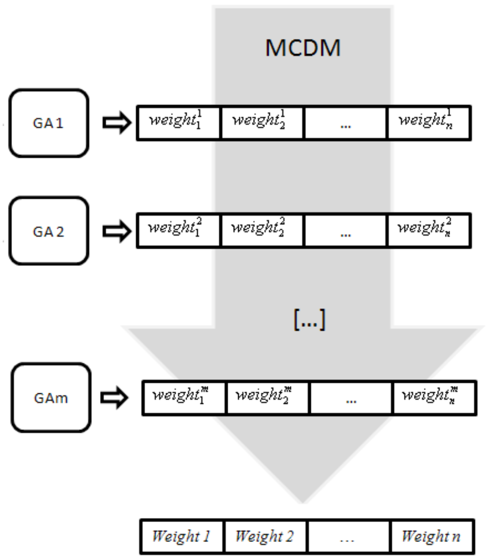


Figure 4: Weight learning schema.

applications. Under the assumption that there is a set C of previous cases available for training the CBR system, stratified cross-validation consists of generating different partitions $P_1, \ldots, P_n$ in C, running the CBR system in each of the partitions, obtaining different results according to the particular evaluation measures (such as recall, precision, fault-positives, etc. [1]), and finally the process concludes by averaging the results across all the partitions. Each partition P_i consists of two parts: training cases and test cases, with the training cases composed of many more cases than the test cases. Cases are randomly assigned as training or test cases in each partition, without repetition and maintaining the class distribution. Usually, the methods are parameterised by either a k constant or a p percentage that indicates the difference between the amount of training and test cases in the partitions. For example, a $p = 10\%$, means that the size of test cases is 10% of the total size of P_i; while $k = 10$ means that there will be 10 partitions.

There is a natural extension of the stratified cross-validation method - from isolated systems to distributed environments - by adding an additional level, which we call *groups*. There are as many groups as agents, and each group G_i contains the cases known by the agent A_i. Thus, all groups can contain the same cases, but different attributes per case (functional decomposition). Conversely, groups can have different cases, but each of them will have the same attributes (spatial decomposition).

eXiT*CBR.v2 currently offers the possibility of implementing both approaches, which allows the user to test the feasibility of building a distributed system and also compare the benefits of an isolated one (as shown in the case study of Section 4).

3.4.1. Folders for Spatial Decomposition

In this approach, all agents receive cases with the same contents (as in [31]). Using cross-validation in this way, it is possible to enhance the collaboration of experts in the same field who have different past experiences. This is quite close to evidence-based reasoning, typical in medicine. Evidence-based medicine looks for the best answer to medical questions by tracking down, with maximum efficiency, the best evidence (from clinical examination, diagnostic laboratory, published literature, or other sources) [41], then, physicians collaborate in a peer-to-peer way to conclude final medical outcome. With our tool, none of the agents have in its case base the case to be tested, rather they have different case bases that cover part of the domain (also known as case-based bias [31]). For each test case, each agent reports its result according to its case-base, then prognosis is computed as per the voting schema shown in Equations 1 and 2.

Given a partition P_i, as many group folders G_i^j as agents are generated (see Figure 5). Each group contains part of the training cases of P_i. Groups can be disjoints or not, meaning that agents may have different past experiences, with or without some degree of overlapping. Test cases of a partition are used to query all agents.

3.4.2. Folders for Functional Decomposition

In a functional decomposition, agent expertise is not overlapped. For example, several physicians can contribute to the gathering of patient information relating to an illness according to their speciality, whether it be an endocrinologist, a radiologist, etc. This information can include tests and examinations performed on the patient as well as the interpretation of these tests according to the physician's expertise. In this situation, information is often kept in relational data-bases (such as the one depicted in Figure 7, left) with different authorisations views. From the database, partition folders are generated so that each group contains all the cases, but the information per case is partial, according to which segment/view of the database the agent is allowed to access (see Figure 6).

When testing the system, all of the agents receive the same test case. None of them will have an identical case in their case-base, but all of them share the same experiences (i.e. different data from the same patients). On this basis, they will output their prognosis, and their combined contribution will lead to the final result.

4. Sample Tool Usage

In this section we provide the reader with an example of our tool's capabilities. The application chosen in this article for illustrative purposes is in breast cancer prognosis. eXiT*CBR.v2 is being used for medical researchers trying to set up an application tool to provide support for clinical practice. As such, they should start from cases coming from a previous clinical study.

When predicting likelihood of a patient suffering breast cancer, information from several departments is required. This information is gathered in a relational data base, such as the one shown in Figure 7, left. The database consists of 871 cases: 628 healthy women and 243 women with breast cancer. The MAS approach naturally follows the same organisation as that of the data-base, as shown in Figure 7 right, resulting in a MAS with eight specialised case-based agents: an epidemiologist agent, a toxic habits agent, a dietetic habits agent, an adolescent habits agent, a gynecological history agent, a pathological history agent, a radiological history agent and a neoplastic history agent.

Note that in a clinical study data may be made available in a centralised way, though when deploying the system in a real-life scenario the data may belong to different departments that would like the information to remain private. As stated previously, the issue of patient data ownership remains unsettled, and is bound by legal considerations[4, 7]. Collaboration can take place at a high level, providing opinions, but rarely sharing patient details. Thus, eXiT*CBR.v2 allows the distributed CBR system to be configured as it would be deployed in a real scenario, so as to provide support for an oncologist making a final prognosis. The question is, how do the different specialisations influence the final decision?

As a first approach, the system designer could try an experiment in which all the agents collaborate to the same degree (with the same weight in the voting schema proposed in Equations 1 and 2). For this purpose, the designer

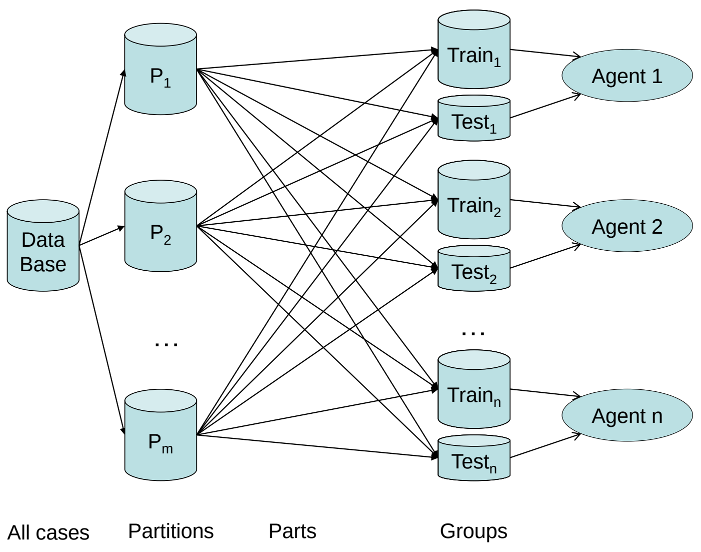


Figure 5: Stratified cross-validation for spatial MAS organisations.

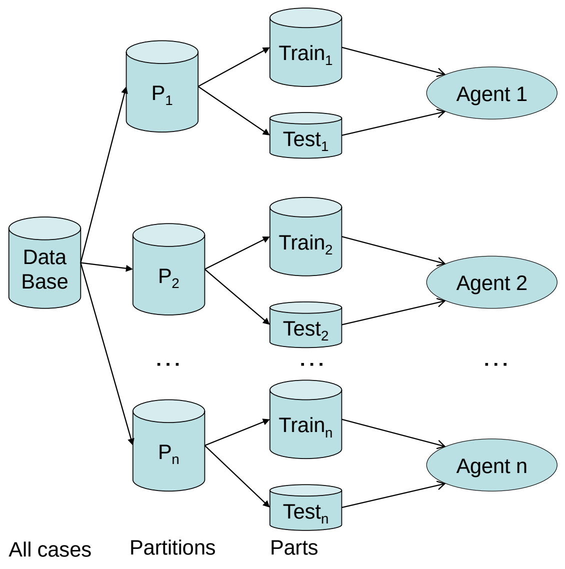


Figure 6: Stratified cross-validation for functional MAS organisations.

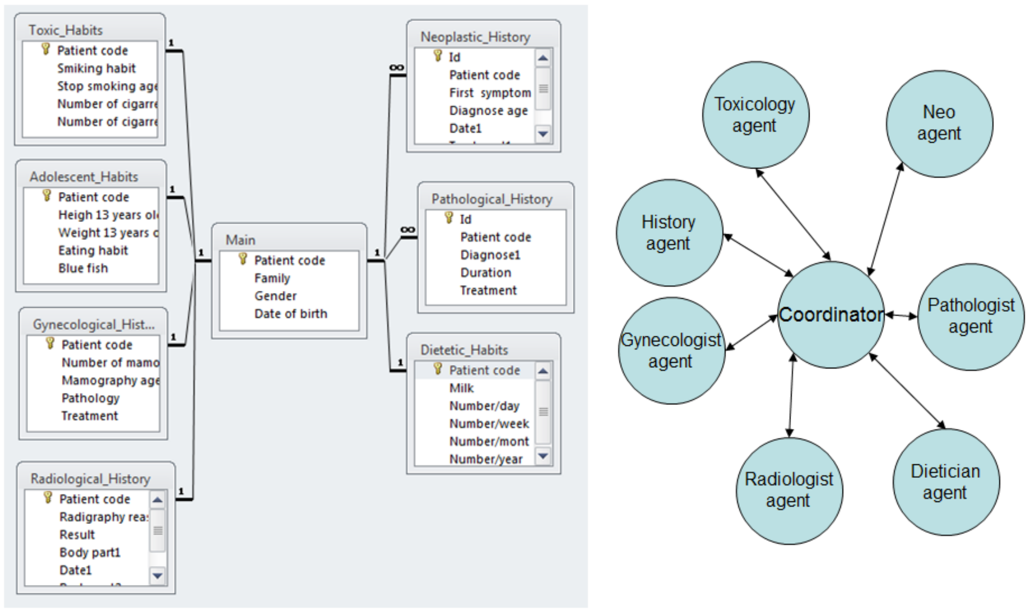
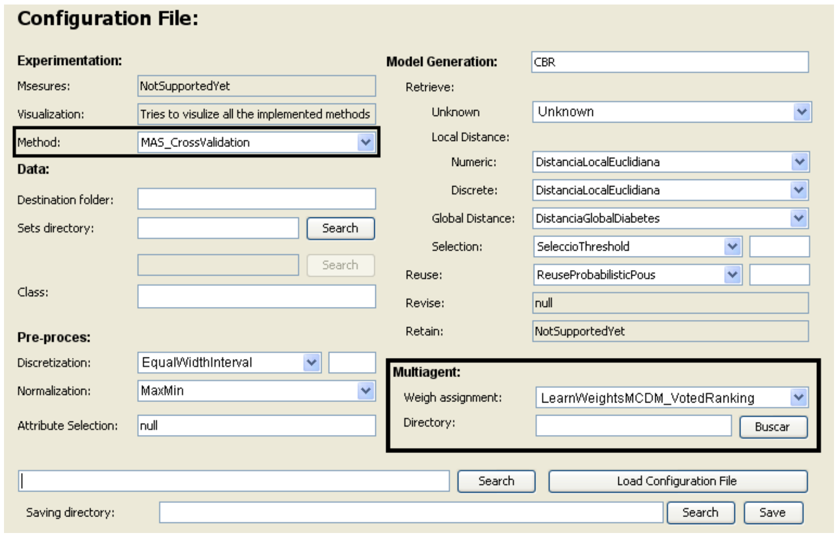


Figure 7: Relational data-base for a functional organisation of a MAS

defines the MAS system according to a simple user interface (see Figure 8) [2]. The parameters to be provided are organised in the following blocks:

- **Experimentation**: parameters in this block are used to define the type of experiment to be performed: batch, cross-validation, MAS-cross-validation.
- **Data**: parameters of this block are used to define the source of the database, the attribute that represents the illness (class) and the directory where the results will be stored.
- **Pre-process**: in this section, the user can operate the data sets in order to normalise or to discretise the data if necessary.
- **Model Generation**: this block contains the parameters relevant to the different methods available for each CBR stage.

[2]Further details can be found in the tutorial provided in http://exitcbr.udg.edu/



Configuration File:

Experimentation:	
Msesures:	NotSupportedYet
Visualization:	Tries to visulize all the implemented methods
Method:	MAS_CrossValidation
Data:	
Destination folder:	
Sets directory:	
Class:	
Pre-proces:	
Discretization:	EqualWidthInterval
Normalization:	MaxMin
Attribute Selection:	null
Model Generation:	CBR
Retrieve:	
Unknown	Unknown
Local Distance:	
Numeric:	DistanciaLocalEuclidiana
Discrete:	DistanciaLocalEuclidiana
Global Distance:	DistanciaGlobalDiabetes
Selection:	SeleccioThreshold
Reuse:	ReuseProbabilisticPous
Revise:	null
Retain:	NotSupportedYet
Multiagent:	
Weigh assignment:	LearnWeightsMCDM_VotedRanking
Directory:	
Saving directory:	

Figure 8: eXiT*CBR v2 parameter interface.

- **Multiagent**: when executing a MAS experiment, the user must either specify the file and directory where agents' weights are stored or indicate that the weights will be learned. In this last case, the learning methodology must be specified and, if necessary, also the MCDM.

The first and last blocks are highlighted in Figure 8, where the new functionalities explained for the first time in this paper are included.

As a result, the user obtains a visualisation of system behavior in the form of a ROC curve, (See Figure 9), which displays the relation between true and false positive ratios, and the area under the curve (AUC) value, which explains the quality of the classifier. The result is not satisfactory since an ideal classifier would have an AUC of 1.

Next, the system designer can decide to use the learning facility to assign weights to agents. To do so, a new distributed CBR system has been set up, but with the change of a single parameter on the user interface. Instead of "LoadWeightsFromFile" the user has written "LearnWeightsMCDM_*MCDMalgorithm*" .

New results can be overlapped in eXiT*CBR.v2 thanks to the experiment navigation facility. As a result, the user can obtain the graphic shown in Figure 10, where it is possible to observe how the usage of collaborative CBR agents improves the performance of the system, increasing the AUC up

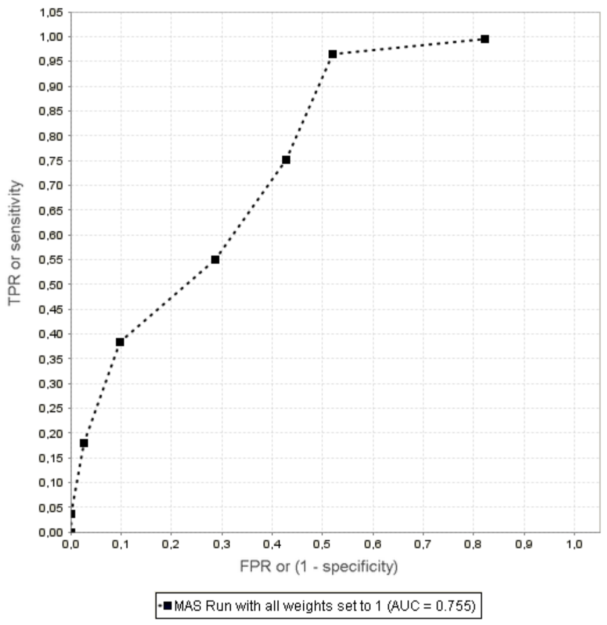


Figure 9: ROC for the first MAS-CBR experiment. Functional cross validation with 50 data sets. Area Under the Curve of 0.755

to 0.863.

Finally, the distributed system is compared against a centralised CBR. For this purpose, the CBR designer need only change the parameter of the interface called ”experimentation method”. Now, the ROC output of the experiment has an AUC of 0.819 while the MAS execution had an AUC of 0.863. The navigation tool facilitates the comparison between experiments, facilitating the user task of deciding when to run a single CBR or when a MAS is a better solution.

In all cases we have followed a functional approach for the cross-validation procedure where 90% of the data has been used for training and the remaining 10% for testing.

In conclusion, we have illustrated how the user can change from a multi-agent collaborative approach to a single one in an easy way, learning weights with a single change of the eXiT*CBR.v2 parameters, to display the effects of the changes in the resulting CBR system.

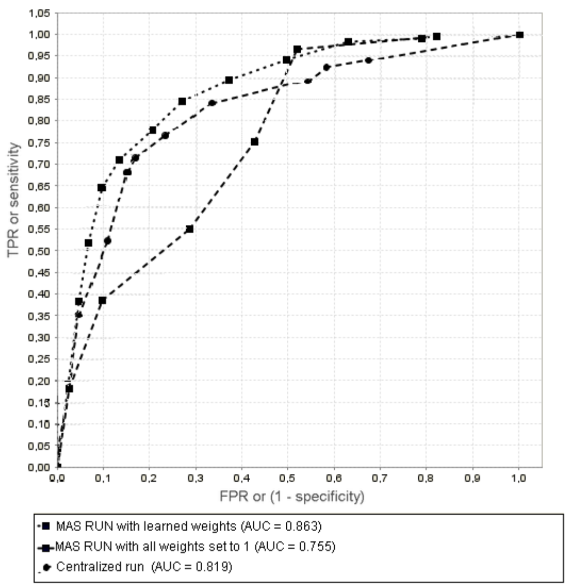


Figure 10: ROC outputs comparison of the experiments performed.

5. Comparison with similar tools

There are now a number of tools similar to eXiT*CBR.v2 which have been developed by other research groups. The software closest to our application is probably jCOLIBRI [2], developed by the Universidad Complutense de Madrid (UCM). In terms of the multi-agent capability focus of this paper, jCOLIBRI has been recently extended in a multi-agent approach called D2ISCO [42]. Although the authors present the tool for distributed CBR as an extension of their previous jCOLIBRI tool, they are offering a specific tool for recommender system prototyping. Thus our approach, which addresses the need to constrain the scope to a particular task (as prognosis) or domain (i.e., medical), seems the only way to provide useful tools for supporting systems development. Another important difference in our tool is the fact that we are providing multi-agent learning mechanisms as a starting point for dealing with agents weights. These weights are related to trust values in D2ISCO, evolved through agents interaction. In our approach, weights cannot be evolved due to the constraint imposed by the domain, rather they are learnt. The GA we are proposing is a method available by default, but it will be possible to add many other approaches in the future.

Other related software for developing CBR systems in distributed approaches is Noos [43]. Noos is an agent platform that has been used to test the feasibility of distributed case-based reasoning systems [31], but it is not conceived as a specific distributed CBR toolkit, which we offer with eXiT*CBR.v2. Elsewhere, more mature coordination mechanisms than the voting schema presented in this paper have been proposed, such as the AMAL protocol for argumentation processes[44] (see [8], for example, for a broad survey). Our specific aim is to support collaborative prognosis, by extending the existing eXiT*CBR tool, but future work should also consider study the methods in [44] so as to render them useful to physicians for clinical practice.

Other publications have made reference to tools for developing MASs, such as those explained in the Section 2, as well as other tools that provide integration kits (see, for example, CBR Works [45] which is implemented in CORBA [46]) so that different CBR systems can cooperate. However, the tool is not designed per se to test MAS CBR systems.

An example of where case-based reasoning has been applied in MASs for agent coordination can be seen with CAKE (Collaborative Agent-based Knowledge Engine [47]). CAKE supports collaborative business by integrating workflow technology, agents and humans. In such a context, high flexibility for business processes, tasks and agents is required, and case-based reasoning is used to meet the requirements of a specific situation.

There are other popular case-based reasoning tools, but unlike eXiT*CBR.v2, they do not support the collaborative work. Among them, it is interesting to highlight CBRShell [48] which also provides an agent-based algorithm for weight learning. As with our tool, cases are also provided via a comma separator value file. However, it lacks of modularity, and a single similarity method is provided only to solve classification problems. eXiT*CBR.v2 allows the definition of several methods for the different stages of case-based reasoning.

Among other available tools that also include case-based reasoning technology are Orenge [49] and AIS [50]. Orange is a commercial retrieval engine which supports maintenance issues related to the case base and is able to deal with case generalisation. One of the main drawbacks of Orenge is the proprietary language required for representing the knowledge (vocabulary, similarity measures, reuse rules). AIS is a commercial semantic middleware for organisational search and classification problems, supporting marketing activities in enterprises as well as costumer information data mining analysis. Some collaborators of AIS offer a free, alternative tool called myCBR [51]. Based exclusively on CBR, it has more flexibility for defining similarity measures, and includes a semantic module. For a detailed comparison of other tools against the non-agent version of eXiT*CBR, see [5].

Regarding domain specificity, Orange [52] is a data mining tool which allows the user to install some additional widgets to deal with bioinformatic problems, as well as microarray and genomic data. We think that Orange presents some complements to our case-based reasoning tool, which should be studied in future work to find appropriate synergies and connections. However, unlike eXiT*CBRv2, Orange, as well as other well-known free available data mining tools such as Weka [53], do not consider collaborative decision making.

The application of GA to case-based reasoning has been studied to the retrieval phase for feature and instance learning, as in [54, 55, 56]. We understand that in the future we need to complement our approach with these studies and consider the synergies between feature learning and classifier weight learning, together with local similarity measures, such as those studied in [57].

6. Conclusions and Future Work

Collaborative medical decision making is a reality today, and thus we need to provide physicians with tools that take into account such a collaborative working mode. In this paper we have presented eXiT*CBR.v2, a distributed-case-based reasoning tool which works towards supporting the development and experimentation of medical decision-making in a distributed environment. Thus, we are empowering the previous version of the tool (eXiT*CBR.v1) with facilities for CBR systems to interact in open, collaborative environments. Like its predecessor, eXiT*CBR.v2 is focused on medical prognosis, while keeping the original user-friendly interface designed for testing medical applications, to therefore minimise possible disruption in the work of current v1 users who wish to adopt the next version of the tool. The new tool includes a collaborative mechanism for agents under a voting schema, methods that support learning for coordination and cross-validation for spatial and functional MAS organisations.

We illustrate the use of the tool through several experiments carried out with a breast cancer database, and we show how easy it is to compare distributed approaches that maintain naturally distributed clinical organisation, compared to centralised systems. As a future work, we intend to include other methods to generate executable CBR systems for exploitation purposes that could be valid even in mobile platforms. The addition of other coordination mechanisms should be also explored, as well as the extension to other related tasks (diagnosis).

Acknowledgements

This research project has been partially funded through the projects labeled TIN2008-04547, DPI2011-24929 and CTQ2008-06865-C02-02 and the grants UDG-BR10/18 and FPU-AP2009-2831.

7. Bibliography

[1] J. Comunity, Drools 5 - the business logic integration platform, `http://www.jboss.org/drools` (Accessed 6 2010).

[2] B. Díaz-Agudo, P. A. González-Calero, J. A. Recio-García, A. A. Sánchez-Ruiz-Granados, Building CBR systems with jCOLIBRI, Sci. Comput. Program. 69 (2007) 68–75.

[3] N. C. Perry, M. W. Wiggins, M. Childs, G. Fogarty, Can reduced processing decision support interfaces improve the decision-making of less-experienced incident commanders?, Decision Support Systems 52 (2012) 497–504.

[4] K. Cios, G. Moore, Uniqueness of medical data mining, Artificial Intelligence in Medicine 26 (2002) 1–24.

[5] B. López, C. Pous, A. Pla, P. Gay, J. Sanz, J. Brunet, eXiT*CBR: A framework for case-based medical diagnosis development and experimentation, Artificial Intelligence in Medicine 51 (2011) 81–91.

[6] SPSS software, `http://www-01.ibm.com/software/analytics/spss/` [Accessed: 14/06/2012].

[7] D. Kalra, Electronic health record standards, in: IMIA Yearbook of Medical Informatics, IMIA and Schattauer GmbH, 2006, pp. 136–144.

[8] E. Plaza, L. McGinty, Distributed case-based reasoning, Knowl. Eng. Rev. 20 (2005) 261–265.

[9] A. Aamodt, E. Plaza, Case-based reasoning: Foundational issues, methodological variations, and system approaches, AI Communications 7 (1994) 39–59.

[10] I. Gilboa, L. Samuelson, D. Schmeidler, A unified model of induction, Tech. rep., MIT Economics, `http://economics.mit.edu/files/4558` [Accessed: 14/06/2012] (2009).

[11] E. A. M. L. Abdrabou, A.-B. M. Salem, Case-based reasoning tools from shells to object-oriented frameworks, in: ICCOMP'08, World Scientific and Engineering Academy and Society (WSEAS), 2008, pp. 781–786.

[12] M. Wooldridge, An Introduction to MultiAgent Systems, 1st Edition, John Wiley & Sons, 2002.

[13] N. R. Jennings, Coordination techniques for distributed artificial intelligence, John Wiley & Sons, Inc., New York, NY, USA, 1996, Ch. 7, pp. 187,210.

[14] T. W. Malone, K. Crowston, The interdisciplinary study of coordination, ACM Comput. Surv. 26 (1994) 87–119.

[15] G. Weiss (Ed.), Multiagent systems: a modern approach to distributed artificial intelligence, MIT Press, Cambridge, MA, USA, 1999.

[16] N. Islam, G. A. Mallah, Z. A. Shaikh, FIPA and MASIF standards: a comparative study and strategies for integration, in: Proceedings of the 2010 National Software Engineering Conference, NSEC '10, ACM, New York, NY, USA, 2010, pp. 7:1–7:6.

[17] T. Finin, R. Fritzson, D. McKay, R. McEntire, KQML as an agent communication language, in: Proceedings of the third international conference on Information and knowledge management, CIKM '94, ACM, New York, NY, USA, 1994, pp. 456–463.

[18] F. Zambonelli, N. R. Jennings, M. Wooldridge, Developing multiagent systems: The Gaia methodology, ACM Trans. Softw. Eng. Methodol. 12 (2003) 317–370.

[19] F. Bergenti, M. P. Gleizes, F. Zambonelli (Eds.), Methodologies and Software Engineering for Agent Systems. The Agent-Oriented Software Engineering handbook, Kluwer Publishing, 2004.

[20] F. Bellifemine, A. Poggi, G. Rimassa, JADE: a FIPA2000 compliant agent development environment, in: Proceedings of the fifth international conference on Autonomous agents, AGENTS '01, ACM, New York, NY, USA, 2001, pp. 216–217.

[21] S. J. Russell, P. Norvig, J. F. Candy, J. M. Malik, D. D. Edwards, Artificial intelligence: a modern approach, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.

[22] G. Weiß, M. Nickles, M. Rovatsos, F. A. Fischer, Specifying the intertwining of cooperation and autonomy in agent-based systems, J. Network and Computer Applications 30 (3) (2007) 1196–1215.

[23] M. Mitchell, An introduction to genetic algorithms, A Bradford book, MIT Press, 1996.

[24] M. Tan, Multi-agent reinforcement learning: Independent vs. cooperative agents, in: Proceedings of the Tenth International Conference on Machine Learning, Morgan Kaufmann, 1993, pp. 330–337.

[25] T. Haynes, R. Wainwright, I. Sen, Evolving cooperation strategies, in: Proceedings of the First International Conference on Multi–Agent Systems, MIT Press, 1995, pp. 45–0.

[26] M. V. Nagendra Prasad, E. Plaza, Corporate memories as distributed case libraries, in: Proc. 10th Banff Knowledge Acquisition for Knowledge-based Systems Workshop, volume 2, 1996, pp. 1–19.

[27] D. B. Leake, R. Sooriamurthi, Automatically selecting strategies for multi-case-base reasoning, in: Proceedings of the 6th European Conference on Advances in Case-Based Reasoning, ECCBR '02, Springer-Verlag, London, UK, UK, 2002, pp. 204–233.

[28] L. McGinty, B. Smyth, Collaborative case-based reasoning: Applications in personalised route planning, in: Aha, DW & Watson, I (eds.): Proc. of the Fourth International Conference on Case-Based Reasoning, ICCBR, Springer-Verlag, 2001, pp. 362–376.

[29] C. Hayes, P. Cunningham, M. Doyle, Distributed CBR using XML, Tech. Rep. Computer Science Technical Report

TCD-CS-1998-06, The University of Dublin, Trinity College, http://hdl.handle.net/2262/13005 [Accessed: 14/06/2012] (1998).

[30] R. Bergmann, Experience Management Foundations, Development Methodology, and Internet-Based Applications, LNCS 2432, Springer, 2002, Ch. 4. Assessing Experience Utility.

[31] S. Ontañón, Ensemble case based learning for multi-agent systems, Ph.D. thesis, Universitat Autonoma de Barcelona (2005).

[32] M. M. Richter, Knowledge containers, http://pages.cpsc.ucalgary.ca/ mrichter/Papers/Knowledge%20Containers.pdf [Accessed: 14/06/2012] (2006).

[33] C. Pous, P. Gay, A. Pla, J. Brunet, J. Sanz, T. R. Cajal, B. López, Modeling reuse on case-based reasoning with application to breast cancer diagnosis, in: Proceedings of the 13th International Conference on Artificial Intelligence: Methodology, Systems, and Applications, AIMSA '08, Springer-Verlag, 2008, pp. 322–332.

[34] P. Herrero, An insulin boulus calculator based on case-based reasoning, in: Diabetes Technology Meeting, Springer-Verlag, 2011.

[35] I. Bichindaritz, S. Montani, L. Portinale, Special issue on case-based reasoning in the health sciences, Applied Intelligence 28 (2008) 207–209.

[36] A. Birk, Boosting cooperation by evolving trust, Applied Artificial Intelligence 14 (2000) 769–784.

[37] L. A. Becker, M. Seshadri, Comprehensibility and overfitting avoidance in genetic programming for technical trading rules, Tech. rep., Worcester Polytechnic Institute (May 2003).

[38] B. López, C. Pous, P. Gay, A. Pla, Multi criteria decision methods for coordinating case-based agents, in: Proceedings of the 7th German conference on Multiagent system technologies, MATES'09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 54–65.

[39] S. Borra, A. Di Ciaccio, Measuring the prediction error. A comparison of cross-validation, bootstrap and covariance penalty methods, Comput. Stat. Data Anal. 54 (2010) 2976–2989.

[40] J. Demsar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.

[41] R. C. Brownson, J. G. Gurney, G. H. Land, Evidence-based decision making in public health, J Public Health Management Practice 5 (1999) 86–97.

[42] S. González-Sanz, J. A. Recio-García, B. Díaz-Agudo, D2ISCO: Distributed deliberative CBR systems with jCOLIBRI, in: Proceedings of the 1st International Conference on Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems, ICCCI '09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 321–332. doi:978-3-642-04441-0_28.

[43] E. Plaza, J. L. Arcos, F. Martin, Cooperative Case-Based Reasoning, Vol. 1221, Spriger-Verlag, 1997, Ch. 7, pp. 180–201.

[44] S. Ontañón, E. Plaza, An argumentation-based framework for deliberation in multi-agent systems, in: Proceedings of the 4th international conference on Argumentation in multi-agent systems, ArgMAS'07, Springer-Verlag, 2008, pp. 178–196.

[45] S. Schulz, CBR-Works - A state-of-the-art shell for case-based application building, in: Proceedings of the 7th German Workshop on Case-Based Reasoning, GWCBR'99, Wrzburg, Springer-Verlag, 1999, pp. 3–5.

[46] Common object request broker architecture (CORBA), `http://www.omg.org/spec/CORBA/` [Accessed: 14/06/2012].

[47] R. Bergmann, A. Fressmann, K. Maximini, R. Maximini, T. Sauer, Case-based support for collaborative business, in: Proceedings of the 8th European conference on Advances in Case-Based Reasoning, ECCBR'06, Springer-Verlag, Berlin, Heidelberg, 2006, pp. 519–533.

[48] AIAI, CBR Shell, `http://www.aiai.ed.ac.uk/project/cbr/CBRDistrib/` [Accessed: 14/06/2012].

[49] T. Roth-Berghofer, Developing maintainable case-based reasoning systems: Applying SIAM to empolis Orenge, in: M. Nick, K.-D. Althoff (Eds.), CEUR Workshop Proceedings Volume 67, CEUR-WS.org, 2003.

[50] AIS, `http://www.attensity.com/` [Accessed: 14/06/2012].

[51] myCBR, `mycbr-project.net` [Accessed: 14/06/2012].

[52] Orange, `http://orange.biolab.si/` [Accessed: 14/06/2012].

[53] I. Witten, E. Frank, Data mining: practical machine learning tools and techniques, 2nd edition, Morgan Kaufmann, San Francisco, CA, USA, 2005.

[54] P.-C. Chang, C.-Y. Lai, R. Lai, A hybrid system by evolving case-based reasoning with genetic algorithm in wholesaler's returning book forecasting, Decision Support Systems 42 (2006) 1715–1729.

[55] H. Ahn, K. jae Kim, I. Han, Hybrid genetic algorithms and case-based reasoning systems, in: J. Zhang, J.-H. He, and Y.Fu (Eds.): CIS, LNCS 3317, Springer, 2004, pp. 922–927.

[56] J. Jarmulak, S. Craw, R. Crowe, Genetic algorithms to optimise CBR retrieval, in: EWCBR '00: Proceedings of the 5th European Workshop on Advances in Case-Based Reasoning, Springer-Verlag, 2000, pp. 136–147.

[57] A. Stahl, T. Gabel, Local similarity measures using evolution programs to learn, in: Proceedings of the Fifth International Conference on Case-Based Reasoning, Springer, 2003, pp. 537–551.