Self-Adaptive MAS for Biomedical Environments

Juan F. De Paz, Sara Rodríguez, Javier Bajo and Juan M. Corchado

Departamento de Informática y Automática, Universidad de Salamanca
Plaza de la Merced s/n, 37008, Salamanca, España
{fcofds, srg, jbajope, corchado}@usal.es
Department of Computer Science and Automation, University of Salamanca Plaza de la Merced s/n, 37008, Salamanca, Spain

Abstract. The application of information technology in the field of biomedicine has become increasingly important over the last several years. This paper presents an intelligent dynamic architecture for knowledge data discovery in biomedical databases. The core of the system is a type of agent that integrates a novel strategy based on a case-based planning mechanism for automatic reorganization. This agent proposes a new reasoning agent model, where the complex processes are modeled as external services. The agents act as coordinators of Web services that implement the four stages of the case-based planning cycle. The multi-agent system has been implemented in a real scenario to classify leukemia patients. The classification strategy includes services to analyze patient's data, and the results obtained are presented within this paper.

Keywords: Multiagent Systems, Case-Based Reasoning, microarray, Case-based planning

1. Introduction

The continuous growth of techniques for obtaining cancerous samples, specifically those using microarray technologies, provides a great amount of data. Microarray has become an essential tool in genomic research, making it possible to investigate global genes in all aspects of human disease [8]. Expression arrays [9] contain information about certain genes in a patient's samples. These data have a high dimensionality and require new powerful tools. Usually, existing systems are focused on working with very concrete problems or diseases, with low dimensionality for the data, and it is very difficult to adapt them to new contexts for diagnosing different diseases. This research presents an entirely new perspective that focuses on the concept of Intelligent Organizations, proposing an architecture capable of modeling biomedical organizations through multi-agent systems to analyze biomedical data.

This paper presents an innovative solution to model decision support systems in biomedical environments, based on a multi-agent architecture which allows integration with Web services and incorporates a novel planning mechanism that makes it possible to determine workflows based on exising plans and previous results. The Multiagent System centers on obtaining a self-adaptive biomedical organizational model, making it possible to represent laboratory workers within a virtual environment and the interactions that take place, in order to carry out daily classification tasks. The core of system is a CBP-BDI (Case-based planning) (*Belief* Desire Intention) agent [3] specifically designed to act Web services coordinator, making it possible to reduce the computational load for the agents in the organization and expedite the classification process. CBP-BDI agents [3] make it possible to formalize systems by using a new planning mechanism that incorporates graph theory and Bayesian networks as a reasoning engine to generate plans. The system was applied to case studies, consisting of the classification of leukemia patients and brain tumors from microarrays, and the multiagent system developed incorporates novel strategies for data analysis and microarray data classification. Microarray has become an essential tool in genomic research, making it possible to investigate global gene expression in all aspects of human disease [8].

The next section describes the main characteristics of the proposed multiagent system and briefly explains its components. Section 3 presents a case study consisting of a distributed multi-agent system for cancer detection scenarios. Finally section 4 presents the results and conclusions obtained.

2. Multiagent System for Expresion Analysis

Nowadays, having software solutions at one's disposal that enforce autonomy, robustness, flexibility and adaptability of the system to develop is completely necessary. The dynamic agents organizations that auto-adjust themselves to obtain advantages from their environment seems a more than suitable technology to cope with the development of this type of systems. The integration of multi-agent systems with SOA (*Service Oriented* Architecture) and Web Services approaches has been recently explored [14]. Some developments are centred on communication between these models, while others are centred on the integration of distributed services, especially Web Services, into the structure of the agents. Ricci et al. [15] have developed a java-based framework to create SOA and Web Services compliant applications, which are modelled as agents. Communication between agents and services is performed by using what they call "artifacts" and WSDL (Web Service Definition Language). We have used the FUSION@ architecture [12] as a reference, which not only provides communication and integration between distributed agents, services and applications.

The approach presented in this paper is an organizational model for biomedical environments based on a multi-agent dynamic architecture that incorporates agents with skills to generate plans for analysis of large amounts of data. The core of the system is a novel mechanism for the implementation of the stages of CBP-BDI mechanisms through Web services that provides a dynamic self-adaptive behaviour to reorganize the environment. Moreover, the system provides communication mechanisms that facilitate integration with SOA architectures. The multiagent system was initially designed to model the laboratory environments oriented to the processing of data from expression arrays. To do this, the system defined specific agent types and services. The agents act as coordinators and managers of services, while the services are responsible for carrying out the processing of information by providing replication features and modularity. Agents are available to run on different types of devices, so different versions were created to suit each one. The types of agents are distributed in layers within the system according to their functionalities, thus providing an organizational structure that includes an analysis of the information and management of the organization, and making it possible to easily add and eliminate agents from the system. The agent layers constitute the core and define a virtual organization for massive data analysis, as can be seen in Figure 1. Figure 1 shows four types of agent layers:

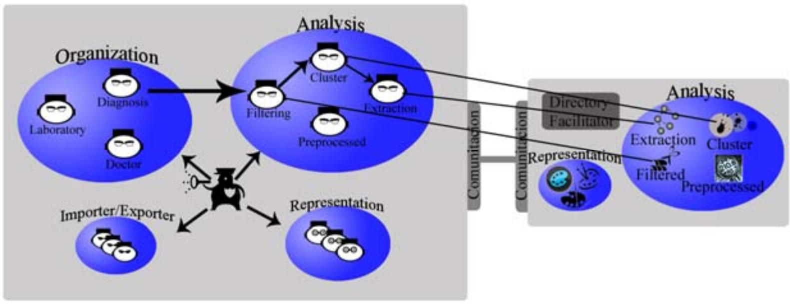

Fig. 1. Multiagent System Architecture

- Organization: The agents will be responsible for conducting the analysis of information following the CBP-BDI [3] reasoning model. The agents from the organizational layer should be initially configured for the different types of analysis that will be performed. Because these analyses vary according to the available information and the search results.
- Analysis: The agents in the analysis layer are responsible for selecting the configuration and the flow of services that best suit the problem to solve. They communicate with Web services to generate results. The agents of this layer follow the CBP-BDI [3] reasoning model. The workflow and configuration of the services to be used is selected with a Bayesian network and graphs, using information that corresponds to the previously executed plans. The agents at this layer are highly adaptable to the case study to which is applied. Specifically, the microarray case study includes those agents that are required to carry out the expression analysis, as shown in figure 1.
- Representation: These agents are in charge of generating the tables with the classification data and the graphics for the results.
- Import/Export: These agents are in charge of formatting the data in order to adjust them to the needs of agents and services.
- The Controller agent manages the agents available in the different layers of the multiagent system. It allows the registration of agents in the layers, as well as their use in the organization.

On the other hand, the services layer is divided into two groups:

- Analysis Services: The analysis services are services used by analysis agents for carrying out different tasks. The analysis services include services for pre-processing, filtering, clustering and extraction of knowledge. Figure 1 illustrates how these services are invoked by the analysis layer agents in order to carry out the different tasks corresponding to microarray analysis.

- Representation Services: They generate graphics and result tables.

Within the services layer, there is a service called Facilitator Directory that provides information on the various services available and manages the XML file for the UDDI (Universal Description Discovery and Integration). To facilitate communication between agents and services the architecture integrates a communication layer that provides support for the FIPA-ACL and SOAP protocols.

Figure 1 shows the connections between the diagnosis agent (in the organization layer) with the agents in the analysis layer and the services. The connections represent a plan. A diagnosis incorporates a filtering process, carried out by an analysis agent that selects the sequence of services for the plan. Then, a clustering agent selects the optimum service. Finally, the knowledge extraction obtains the relevant probes.

Nowadays, there exist different possibilities to services planning and composition. One of the most important is services composition using HTN (Hierarchical Task Network) and HTN planners as SHOP2 [20]. These systems don't provide a planning mechanism that make use of past experiences, so they have a lack of adaptation an learning abilities. Another techniques are based on Quality of Service [21] that make use of heuristics to obtain an optimum composition. However, the quality of each of the services is not independent of the others.

2.1. Coordinator CBP-BDI Agent

The coordinator agent is the core of the system, since provides the ability for self-organization. The agents in the organization layer have the capacity to learn from the analysis carried out in previous procedures. They adopt the model of reasoning CBP, a specialization of case-based reasoning (CBR) [2]. CBP is the idea of planning as remembering [3]. In CBP, the solution proposed to solve a given problem is a plan, so this solution is generated taking into account the plans applied to solve similar problems in the past [13]. The problems and their corresponding plans are stored in a plans memory. A plan P is a tuple $<S,B,O,L>$, S is the set of plan actions, O is an ordering relation on S allowing to establish an order between the plan actions, B is a set that allows describing the bindings and forbidden bindings on the variables appearing in P, L is a set of casual links.

The CBP-BDI agents stem from the BDI model [16] and establish a correspondence between the elements from the BDI model and the CBP systems. The BDI model adjusts to the system requirements since it is able to define a series of goals to achieve based on the information that has been registered with regards to the world. Fusing the CBP agents together with the BDI model and generating CBP-BDI agents makes it possible to formalize the available information, the definition of the goals and actions that are available for resolving the problem, and the procedure for resolving new problems by adopting the CBP reasoning cycle.

The CBP-BDI agent type presented in this paper acts as coordinator of services. The terminology used is the following: The environment M and the changes that are produced within it, are represented from the point of view of the agent. Therefore, the world can be defined as a set of variables that influence a problem faced by the agent

$$M = \{\tau_1, \tau_2, \cdots, \tau_s\} \text{ with } s < \infty \qquad (1)$$

The beliefs are vectors of some (or all) of the attributes of the world taking a set of concrete values

$$B = \{b_i / b_i = \{\tau_1^i, \tau_2^i, \cdots, \tau_n^i\}, n \le s \quad \forall i \in N\}_{i \in N} \subseteq M \qquad (2)$$

A state of the world $e_j \epsilon E$ is represented for the agent by a set of beliefs that are true at a specific moment in time t. i represents a belief of the N.

Let $E = \{e_j\}_{j \epsilon N}$ set of status of the World if we fix the value of t then

$$e_j^t = \{b_1^{jt}, b_2^{jt}, \cdots b_r^{jt}\}_{r \in N} \subseteq B \quad \forall j, t \qquad (3)$$

The desires are imposed at the beginning and are applications between a state of the current world and another that it is trying to reach

$$d : \underset{e_0}{E} \underset{\rightarrow}{\rightarrow} \underset{e^*}{E} \qquad (4)$$

Intentions are the way that the agent's knowledge is used in order to reach its objectives. A desire is attainable if the application *i*, defined through n beliefs exists:

$$i : \underset{(b_1, b_2, \cdots\cdots\cdots, b_n, e_0)}{Bx\overset{n)}{Bx\cdots x}BxE} \underset{\rightarrow}{\rightarrow} \underset{e^*}{E} \qquad (5)$$

In our model, intentions guarantee that there is enough knowledge in the beliefs base for a desire to be reached via a plan of action. We define an agent action as the mechanism that provokes changes in the world making it change the state,

$$a_j : \underset{e_i}{E} \underset{\rightarrow}{\rightarrow} \underset{a_j(e_i) = e_j}{E} \qquad (6)$$

Agent plan is the name we give to a sequence of actions that, from a current state e_0, defines the path of states through which the agent passes in order to reach the other world state.

$$p_n : \underset{e_0}{E} \underset{\rightarrow}{\rightarrow} \underset{p_n(e_0) = e_n}{E} \qquad (7)$$

$$p_n(e_0) = e_n = a_n(e_{n-1}) = \cdots = (a_n \circ \cdots \circ a_1)(e_0) \quad p_n \equiv a_n \circ \cdots \circ a_1$$

Based on this representation, the CBP-BDI coordinator agents combine the initial state of a case, the final state of a case with the goals of the agent, and the intentions with the actions that can be carried out in order to create plans that make it possible to reach the final state. The actions that need to be carried out are services, making a plan an ordered sequence of services. It is necessary to facilitate the inclusion of new services and the discovery of new plans based on existing plans. Services correspond to the actions that can be carried out and that determine the changes in the initial problem data. Each of the services is represented as a node in a graph. The presence of an arch that connects to a specific node implies the execution of a service associated with the end node. Figure 2 provides a graphical representation of a service plans. As shown, the first graph has only one path and contains nodes that are not connected. The path defines the sequence of services from the start node until the end node. The plan described by the graph is defined by the sequence *(S_7 o S_5 o S_3 o S_1)(e_0)*. e_0 represents the original state that corresponds to Init, which represents the initial problem description e_0. Final represents the final state of the problem e^*.

CBP-BDI agents use the information contained in the cases in order to perform different types of analyses. As previously explained, an analysis assumes the construction of the graph that will determine the sequence of services to be performed. The construction process for the graph can be broken down into a series of steps that are explained in detail in the following sub-sections:

1. Generate the directed graph with the information from the different plans.
2. Generate a TAN (Tree Augmented Naive Bayes) classifier for the cases with the best and worst output respectively, using the Friedman-Godsmidtz [17] algorithm.
3. Calculate the execution probabilities for each service with respect to the classifier generated in the previous step.
4. Adjust the connections from the original graph according to a metric.
5. Construct the graph

2.1.1. Constructing a directed graph

The different plans are represented in the graphs. The plans represented in graphical form are joined to generate one directed graph that defines the new plans based on the minimization of a specific metric. For example, given the graphs shown in figure 2, a new graph is generated that joins the information corresponding to both graphs.

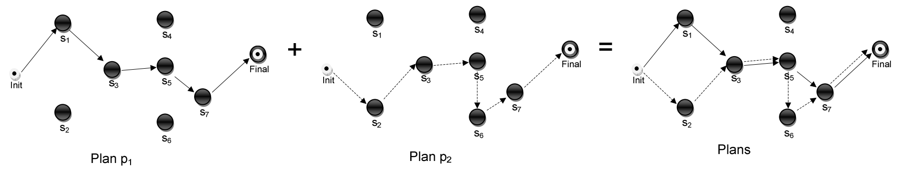

Fig. 2. Composition of the graphs

The dual connection of the nodes is indicated only to represent the existence of a connection between the two graphs, although it is not actually necessary to represent more than one connection per arc. Each of the arcs in the graph for the plans has a corresponding weight according to which it is possible to calculate the new route to be executed. This value is estimated based on the efficiency of the plans recovered as indicated in section 2.1.4. When constructing the graph of plans, the weights are estimated based on the existing plans by applying a bayesian network. The entry data to the bayesian network is broken down into the following elements: Plans with a high efficiency are assigned to class 1 and plans with a low efficiency are assigned to class 0. The Bayesian network is calculated for each of the classes according to the recovered plans, following the Friedman-Goldsmidtz [17] algorithm.

2.1.2. TAN classifier

The TAN classifier is constructed based on the plans recovered that are most similar to the current plan, distinguishing between efficient and inefficient plans to generate the model. Thus, by applying the Friedman-Goldsmidtz [17] algorithm, the two classes that are considered are efficient and inefficient. The Friedman-Goldsmidtz [17] algorithm makes it possible to calculate a Bayesian network based on the dependent relationships established through a metric. The metric proposed by Friedman is defined as follows:

$$I(X;Y \mid Z) = \sum_{x \in X} \sum_{y \in Y} \sum_{z \in Z} P(x,y,z) \cdot \log\left[\frac{P(x,y \mid z)}{P(x \mid z) \cdot P(y \mid z)}\right] \quad (8)$$

Based on the previous metric, the maximal tree is constructed.

2.1.3. Services Probabilities

Once the TAN model has been calculated for each of the classes, we proceed to calculate the probability of execution for each of the services. These probabilities influence the final value of the weights assigned to the arcs in the graph. The probabilities are calculated according to the TAN model. Assuming that the set of random variables can be defined as $U = \{X_1, X_2, \ldots, X_n\}$, we can assume that the variables are independent. The probabilities are represented by $P(x_i \mid \pi_{x_i})$ where x_i is a value of the variables X_i and $\pi_{x_i} \in \Pi_{X_i}$ where π_{x_i} represents one of the parents for the node X_i. Thus, a Bayesian network *B*, defines a single set probability distribution over *U* given for

$$P(X_1, X_2, \ldots, X_n) = P(X_n \mid X_{n-1}, \ldots, X_1) \cdot P(X_{n-1}, \ldots, X_1) =$$
$$\prod_{i=1}^{n} P(X_n \mid X_{n-1}, \ldots, X_1) = \prod_{i=1}^{n} P(X_i \mid \Pi_{X_i})$$

2.1.4. Considering the connections

Using the TAN model, we can define the probability that a particular number of services may have been executed for classes 1 and 0. This probability is used to determine the final value for the weight with regards to the quality of the plans recovered. Assuming that the probability of having executed service *i* for class *c* is defined as follows *P(i,c)* the weight of the arcs is defined according to the following formula. The function has been defined in such a way that the plans of high quality are those with values closest to zero.

$$c_{ij} = P(j,1) \cdot I(i,j,1) \cdot t_{ij}^{1} - P(j,0) \cdot I(i,j,0) \cdot t_{ij}^{0} \quad (9)$$

$$t_{ij}^{1} = \frac{\sum_{p \in G_{ij}^{1}, s \in G^{1}} (1 - (q(p) - \min(q(s)))) + 0.1}{\# G_{ij}^{1}} \quad (10)$$

$$t_{ij}^{0} = \frac{\sum_{p \in G_{ij}^{0}, s \in G^{0}} q(p) - \min(q(s)) + 0.1}{\# G_{ij}^{0}} \quad (11)$$

where:

- *I(i,j,c)* is the probability that service *i* for class *c* is executed before of service j
- *P(j,c)* is the probability that service *j* for class *c* is executed. The value is obtained based on the Bayesian network defined in the previous step.
- G^{s}_{ij} is the set of plans that contain an arc originating in *j* and ending in *i* for class *s*.

- G^s is the set of plans for class *s*.
- *q(p)* is the quality of plan p that also defined the execution time for the plan. The significance depends on the measure of optimization in the initial plan.
- $\#G^s_{ij}$ the number of elements in the set.
- c_{ij} is the weight for the connection between the start node *j* and the end node *i.*

2.1.5. Graph construction

Once the graph for the plans has been constructed, the minimal route that goes from the start node to the end node is calculated. In order to calculate the shortest/longest route, the Dijkstra algorithm is applied since there are implementations for the order *n*log n*. To apply this algorithm, it is necessary to add to each of the edges the absolute value of the edge with a higher negative absolute value, in order to remove from the graph those edges with negative values. The route defines the new plan to execute and depends on the measure to maximize or minimize.

3. Case Study: A Decision Support System for Patients Diagnosis

The multiagent architecture presented in this paper has been used to develop a decision support system for the classification of leukemia and brain tumors patients, and three case studies were established. The first case study uses data from patients suffering from leukemia and focuses on the classification of the type of leukemia. The second case study also analyzes the data from leukemia patients, but in this case focuses on the type of CLL leukemia and attempts to classify the patients in the three existing subtypes. Finally, the goal of the third case study is to classify patients based on the type of brain tumor. The data for leukemia patients was obtained with a HG U133 plus 2.0 chip and corresponded to 212 patients affected by 5 different types of leukemia (ALL, AML, CLL, CML, MDS) [19]. The second case study also used the HG U133 plus 2.0 chip. Finally, third case study [18] used data from the Affymetrix U95Av2 GeneChips including 4 different types of brain tumors [18].

3.1. Services Layer

The services implement the algorithms that allows the analysis expression of the microarrays [1] [19]. There are four types of services:

Preprocessing Service: This service implements the RMA (Robust Multi-array Average) [5] algorithm and a novel control and errors technique. During the Control and Errors phase, all probes used for testing hybridization are eliminated.

Filtering Services: Eliminate the variables that do not allow classification of patients by reducing the dimensionality of the data. Three services are used for filtering: (i) Variables with low variability have similar values for each of the individuals, so they are not significant for the classification process. (ii) All remaining variables that follow a uniform distribution are eliminated. The contrast of assumptions followed uses the Kolmogorov-Smirnov [6] test. (iii) The linear correlation index of Pearson is calculated and correlated variables are removed. (iv) Delete the probes which don't have significative changes in the density of individuals.

Clustering Service: It addresses both the clustering and the association of a new individual to the group more appropriate. The service used is the ESOINN (Enhanced self-organizing incremental neuronal network) [4]. Additional services in this layer are the Partition around medoids (PAM) [10] and dendrograms [11]. Classification is carried out bearing in mind the similarity of the new case using the naive bayes.

Knowledge Extraction Service: The extraction of knowledge technique applied has been CART (Classification and Regression Tree) [7] algorithm.

3.2. Agents Layer

The agents in the analysis layer implement the CBP reasoning model and, for this, select the flow for services delivery and decide the value of different parameters based on previous plans made. A measure of efficiency is defined for each of the agents to determine the best course for each phase of the analysis process. In the analysis layer, at the stage Preprocessed only a service is available. The efficiency is calculated by the deviation in the microarray. At the stage of filtering, the efficiency of the plan p is calculated by the relationship between the proportion of probes and the resulting proportion of individuals falling ill.

$$e(p) = \frac{s}{N} + \frac{i'}{I} \tag{12}$$

Where s is the final number of variables, N is the initial number of probes, i' the number of misclassified individuals and I the total number of individuals. In the phase of clustering and classification the efficiency is determined by the number of misclassified individuals. Finally, in the process of extracting knowledge at the moment, efficiency is determined by the number of misclassified individuals.

In the organization layer, the diagnosis agent chooses the agents for the expression analysis [1]. The diagnosis agent establishes the number of plans to recover from the plans memory for each of the agents and the agents to select from the analysis layer.

4. Results and Conclusions

This paper has presented the a self-adaptive multiagent architecture and its application to three real problems. The characteristics of this novel architecture facilitate a organizational-oriented approach where the dynamics of a real scenario can be captured and modelled into CBP-BDI agents. The tests were oriented both to evaluate the efficiency and the adaptability of the approach. The first experiment consisted of evaluating the services distribution system in the filtering agent for the case study that classified patients affected by different types of leukemia. According to the identification of the problem described in table 1, the filtering agent selected the plans with the greatest efficiency, considering the different execution workflows for the services that are in the plans. Table 1 shows the efficiency obtained for the service workflows that provided the best results in previous experiences. The values in the table indicates the application sequence for the services within the plan, a blank cell indicates that a service is not invoked for that specific plan.

Based on the plans shown in table 1, a new plan is generated following the procedures indicated in section 2.1. The filtering agent in the analysis layer selects the configuration parameters between a specific set of pre-determined values, when it has been told to explore the parameters. Otherwise, for a specific plan, it selects the values that have provided better results based on the measure of the previously established efficiency (12). If there is no plan with all the services that are going to be used, it selects the plan with the greatest efficiency that contains the greatest number of services equal to the current plan for selecting the different parameters.

Table 1. Efficiency of the plans

Variability (z)	Uniform (α)	Correlation (α)	Cutoff	Efficiency	Class
1	2	3	4	0.1401	1
1	2		3	0.1482	1
1				0.1972	0
	1		2	0.1462	1
		1		0.2036	0
	1			0,1862	0
			1	0,1932	0
		1	2	0,186	0

Table 2. Efficiency of the plans

Case study	Variability (z)	Uniform (α)	Correlation (α)	**Cutoff**	Efficiency
Leukemia	**1**	**2**	**3**		**0.1277**
CLL Leukemia	1	2	3	**4**	0.1701
Brain	1		2		0,1532

Once the service distribution process and the selection of parameters for a specific case study have been evaluated, it would appear convenient to evaluate the adaption of this mechanism to case studies of a different nature. To do so, we once again recover the plans with the greatest efficiency for the different workflows and case studies, and proceed to calculate the Bayesian network and the set of probabilities associated with the execution of services as mentioned in sections 2.1.2 and 2.1.3. Once the graph plans have been generated, a more efficient plan is generated according to the procedures indicated in section 2.1.5, with which we can obtain the plan that best adjusts to the data analysis. Table 2 shows the plans generated by the filtering process that best adjusts to the different case studies.

In Figure 3 it is possible to observe the performance of the agents at the organization and analysis layers. 11 plans were conducted based on manual planning and the results were compared with the automatic analysis provided by the multiagent system. In the manual planning a human expert configures the service's parameters, such as if the RMA will use interquantile normalization, or the sequence to execute the services. Each of the agents of the organization layer selects the agents from the analysis layer and, each of these agents in turn selects the services and configuration parameters. The different kinds of agent from the analysis layer can be seen at the bottom of Figure 3 (the name of the agents from the organization layer is indicated in the right of the graphics). In each chart the efficiency measure used is shown. The surface for the CBP-BDI agent is the highest efficiency according to the definitions.

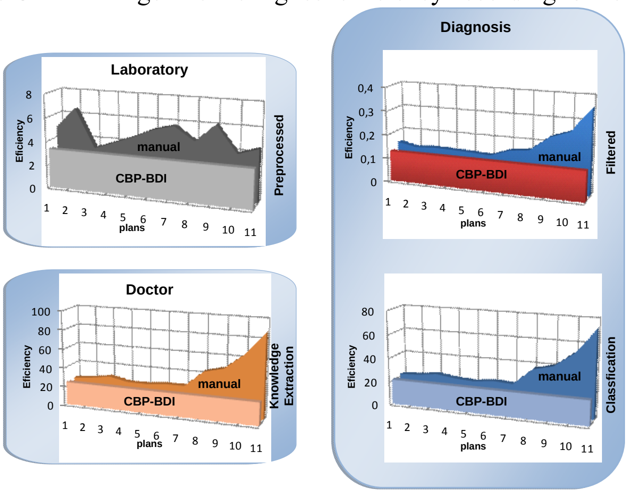

Fig. 3. Performance Comparison between the manual and the automatic planning

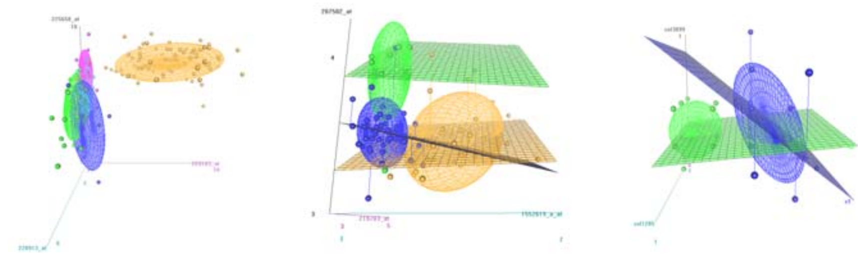

Fig. 4. Patients with CLL leukemia, CLL leukemia subtypes and patients with anaplastic and oligodendrogliomas tumors.

With regards to the classification process, we were able to obtain promising results for each of the case studies. As shown in figures 4a 4b 4c, the probes recovered by the knowledge extraction agent are those that provide the relevant information that makes it possible to classify new individuals. In the first image, we can see the 3 probes that best characterize the patients with CLL leukemia. In the second image, we can see the 3 subtypes of leukemia. Finally, the last image represents the patients with anaplastic and oligodendroglioma tumors. The multi agent system simulates the behavior of experts working in a laboratory, making it possible to carry out a data analysis in a distributed manner, as normally done by experts. The system distributes the functionality among Web services, automatically calculates the expression analysis and allows the classification of patients from the microarray data. Our approach improves the performance provided by the manual procedure for selecting workflow analyses.

References

[1] Lander,E. et al: Initial sequencing and analysis of the human genome. Nature. 409 860-921 (2001)
[2] Kolodner, J.: Case-Based Reasoning. Morgan Kaufmann. (1993)
[3] Glez-Bedia, M. and Corchado, J.: A planning strategy based on variational calculus for deliberative agents. Computing and Information Systems Journal. 10 (1) 2-14 (2002)
[4] Furao, S., Ogura, T. and Hasegawa, O.: An enhanced self-organizing incremental neural network for online unsupervised learning. Neural Networks. 20 893-903 (2007)
[5] Irizarry, R., Hobbs, B., Collin, F., Beazer-Barclay, Y., Antonellis, K., Scherf, U. and Speed, T.: Exploration, Normalization, and Summaries of High density Oligonucleotide Array Probe Level Data. Biostatistics. 4 249-264 (2003)
[6] Brunelli, R.: Histogram Analysis for Image Retrieval. Pattern Recognition. 34 1625-1637 (2001)
[7] Breiman, L., Friedman, J., Olshen, A. and Stone, C.: Classification and regression trees. Wadsworth International Group. Belmont, California. (1984)
[8] Quackenbush, J.: Computational analysis of microarray data Nature Review Genetics. 2 (6) 418-427 (2001)
[9] Affymetrix http://www.affymetrix.com/support/technical/datasheets/hgu133arrays_datasheet.pdf
[10] Saitou, N. and Nie, M.: The neighbor-joining method: A new method for reconstructing phylogenetic trees. Molecular Biology and Evolution. 4 406-425 (1987)
[11] Kaufman, L. and Rousseeuw, P.: Finding Groups in Data: An Introduction to Cluster Analysis. Wiley Series in Probability and Statistics. (1990)
[12] Tapia, D.I., Rodriguez, S., Bajo, J. and Corchado, J.M.: FUSION@, A SOA-Based Multi-agent Architecture. International Symposium on Distributed Computing and Artificial Intelligence, Advances in Soft Computing. 50 99-107 (2008)
[13] Corchado, J.M., Bajo, J., De Paz, Y. and Tapia, D.I.: Intelligent Environment for Monitoring Alzheimer Patients, Agent Technology for Health Care. Decision Support Systems. 44 (2) 382-396 (2008)
[14] Ardissono, L., Petrone, G. and Segnan, M.: A conversational approach to the interaction with Web Services. Computational Intelligence, Blackwell Publishing. 20 693-709 (2004)
[15] Oliva, E., Natali, A., Ricci, A., and Viroli, M.: An Adaptation Logic Framework for {J}ava-based Component Systems. Journal of Universal Computer Science. 14 (13) 2158-2181 (2008)
[16] Bratman, M.: Intention, Plans and Practical Reason. Harvard U.P., Cambridge. (1987)
[17] Friedman, N., Geiger, D. and Goldszmidt, M.: Bayesian Network Classifiers. Machine Learning. 29 131-163 (1997)
[18] Nutt, C.L., Mani, D.R., Betensky, R.A., Tamayo, P., Cairncross, J.G., Ladd, C., Pohl, U., Hartmann, C., McLaughlin, M.E., Batchelor, T.T., Black, P.M., von Deimling, A., Pomeroy, S.L., Golub, T.R., Louis, D.N.: Gene expression-based classification of malignant gliomas correlates better with survival than histological classification. Molecular Biology and Genetics. 63 (7) 1602-1607 (2003)
[19] Corchado, J.M., De Paz, J.F., Rogríguez, S. and Bajo, J.: Model of experts for decision support in the diagnosis of leukemia patients. Artificial Intelligence in Medicine. 46 (3) 179-200 (2009)
[20] Sirin E., Parsia B., Wu, D., Hendler J., Nau D.: HTN planning for Web Service composition using SHOP2. Web Semantics: Science, Services and Agents on the World Wide Web, 1 (4) 377-396 (2004)
[21] Ko, J.M., Kim, C.O., Kwon, I.H.: Quality-of-service oriented web service composition algorithm and planning architecture. The Journal of Systems and Software.81 2079–2090 (2008)