Modelo de planificación dinámica para la extracción de conocimiento en expresiones genéticas

VNiVERSiDAD
D SALAMANCA

Marzo de 2010

JUAN FRANCISCO DE PAZ SANTANA

Tesis

Informática y Automática

Universidad de Salamanca

Director

Dr. Juan Manuel Corchado Rodríguez

Dr. Javier Bajo Pérez

La memoria titulada “Modelo de planificación dinámica para la extracción de conocimiento en expresiones genéticas” que presenta D. Juan Francisco De Paz Santana para optar al Grado de Doctor en Informática y Automática por la Universidad de Salamanca ha sido realizado bajo la dirección del profesor Dr. Juan Manuel Corchado Rodríguez, profesor titular del Departamento de Informática y Automática, y el profesor Dr. Javier Bajo Pérez, profesor Encargado de Cátedra de la Universidad Pontificia de Salamanca.

Salamanca, Marzo de 2010

Los directores

Fdo: Dr. Juan Manuel Corchado Rodríguez
Profesor Titular de Universidad
Informática y Automática
Universidad de Salamanca

Fdo: Dr. Javier Bajo Pérez
Profesor Encargado de Cátedra
Escuela Universitaria de Informática
Universidad Pontificia de Salamanca

El doctorando

Fdo: Juan Francisco De Paz Santana
Profesor Ayudante
Informática y Automática
Universidad de Salamanca

Resumen

La aplicación de las tecnologías de la información a los entornos biomédicos ha adquirido una gran importancia durante los últimos años. En este trabajo se presenta el modelo “Intelligent Biomedic Organizations (IBO)”, una arquitectura dinámica para la extracción del conocimiento en base de datos biomédicas. Se trata de un modelo organizativo especialmente diseñado para dar soporte al personal médico en sus tareas diarias, así como para establecer un innovador sistema inteligente para la clasificación y predicción de patrones a partir de grandes volúmenes de datos. IBO se fundamenta en una arquitectura multiagente con capacidades para la integración con servicios Web. El núcleo del sistema son agentes que incorporan estrategias de razonamiento basado en casos para la reorganización y planificación automática. El modelo propone un nuevo modelo de computación distribuida mediante los servicios Web. Los agentes actúan como coordinadores de los servicios Web que implementan que implementan los diferentes algoritmos del ciclo de razonamiento. Los agentes se han implementado bajo casos de estudios reales y se han incorporado nuevas técnicas estadísticas para la realización de filtrados, cluster para el análisis de los datos. Al final del trabajo se muestran los resultados obtenidos. Se hace un estudio tanto de las técnicas aplicadas como del mecanismo de planificación y su integración en la arquitectura multiagente.

Abstract

The application of information technology in the field of biomedicine has become increasingly important over the last several years. This works presents the Intelligent Biomedic Organizations (IBO) model, an intelligent dynamic architecture for knowledge data discovery in biomedical databases. It involves an organizational model specially designed to support medical personnel in their daily tasks and to establish an innovative intelligent system to classify and predict with huge volumes of information. IBO is based on a multi-agent architecture with web service integration capability. 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. In this sense, 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, and the classification strategy includes services such as a novel neural network and statistical methods to analyze the patient's data. The results obtained are presented within this work and demonstrate the effectiveness of the proposed organizational model.

Agradecimientos

Deseo expresar mi más sincero agradecimiento a las personas que me han apoyado durante la realización de esta tesis doctoral

Al director Dr. Juan Manuel Corchado Rodríguez, que me ha guiado durante el proceso de desarrollo de la tesis y me ha proporcionado los contactos necesarios para iniciar la investigación.

Al director Dr. Javier Bajo Pérez que más que director ha sido un amigo que siempre me ha ayudado mucho en todo lo referente a la investigación. Ha participado muy activamente en la revisión de la tesis y en la elaboración de artículos para la difusión de los resultados. Siempre ha estado disponible con independencia de las fechas, día de la semana o su propia carga de trabajo para ayudar, contribuir y aguantarme en los malos ratos.

A la Dra. Yanira del Rosario De Paz Santana (mi hermana) que me ha ayudado siempre en la investigación: en lo referente a publicaciones, conceptos matemáticos y me ha aguantado los ratos de mal humor.

Al Dr. Dante Israel Tapia Martínez con el que he colaborado muy activamente en la investigación y ha hecho que sea más llevadero y ameno el trascurso de la elaboración de la tesis doctoral.

A Cristian Iván Pinzón, con el que también he realizado investigaciones en diversos campos y también ha hecho mucho más llevadero todo el proceso.

Al Dr. Miguel Rocha, Hugo, Rafael, Pedro, Danilo y demás personas del grupo de bioinformática de la Universidad de Minho, por darme la oportunidad de pasar un verano trabajando con ellos en Portugal. Me sentí como en casa a pesar de no entender nada las primeras semanas.

A mis padres, especialmente a los dos. Mi padre siempre fue el que el más me motivó para realizar la tesis y le habría hecho mucha ilusión estar presente, mi madre siempre estuvo ahí cuando me hacía falta.

A mis otros hermanos y demás familiares.

A mis amigos de Plasencia y/o Salamanca, especialmente a Alfonso, Oscar, Toñi y Ángeles con los que he pasado tan buenos ratos desde la infancia. Tampoco me puedo olvidar de amigos, muy buenos amigos, que hice durante la carrera, especialmente a Alberto y Ángel compañeros de prácticas, de buenos y de malos ratos.

Finalmente, a todos los miembros del grupo de investigación BISITE.

Contenido

Lista de figuras

Capítulo I

Introducción

VNiVERSiDAD
D SALAMANCA

1 Introducción

Durante los últimos años se han producido grandes avances en el campo de la Biomedicina. La incorporación de técnicas computacionales y de inteligencia artificial a la medicina ha conducido a un notable progreso en la prevención y detección de enfermedades. Una de las áreas de la medicina que se encuentra en pleno desarrollo y en la que resulta fundamental la aplicación de técnicas que faciliten el tratamiento automático de datos y la extracción de conocimiento es la genómica. La genómica se ocupa del estudio de los genes, de su catalogación, estructura y de la forma en interaccionan entre ellos. Se distinguen diferentes campos de estudio dentro de la genómica. Uno de ellos es la transcriptómica que se encarga de estudiar los ácidos ribonucleicos (ARN) mediante técnicas como el análisis de expresión. Estas técnicas estudian las cadenas de ARN identificando el nivel de expresión de cada uno de los genes estudiados. Consiste en la hibridación de una muestra de un paciente y la coloración del material celular con un tipo especial de colorante. Se obtienen así diferentes niveles de luminiscencia que pueden ser analizados y representados como un array de datos. Tradicionalmente, los métodos y herramientas estadísticas desarrollas están preparadas para trabajar con información distinta a los microarrays, debido a que el número de elementos de estudio es considerablemente superior al número de variables mientras que en los microarrays ocurre lo contrario. Los microarrays permiten obtener información de alrededor 5.5 millón de características por cada análisis por lo que es necesarios desarrollar o adaptar técnicas para que se puedan aplicar en estos casos [Corchado *et al.,* 2009].

Se plantea entonces la necesidad de crear un sistema multiagente (SMA), capaz de analizar los datos procedentes de microarrays, concretamente los arrays de expresión, de modo que permita incorporar el conocimiento extraído en los sucesivos análisis. El análisis de los arrays de expresión recibe el nombre de análisis de expresión. Un análisis de expresión consiste en tres etapas: normalización y filtrado; clustering y clasificación; extracción del conocimiento. Estas etapas se llevan a cabo a partir de los valores de luminiscencia encontrados en las sondas. En la actualidad el número de sondas que contienen los arrays de expresión se ha incrementado de modo considerable por lo que se hace necesario [Affymetrix] el uso de nuevas métodos y técnicas que permitan analizar la información de manera más eficiente. Se hace necesario desarrollar nuevas técnicas que permitan analizar grandes volúmenes de datos, que sean capaces de extraer la información relevante y eliminar la información que carezca de importancia para el proceso de clasificación. El conocimiento adquirido resulta de gran importancia para posteriores clasificaciones. Existen distintas técnicas de inteligencia artificial como redes neuronales artificiales [Sawa, Ohno-Machado, 2003] [Bianchia *et*

al., 2007], redes bayesianas [Baladandayuthapani *et al.,* 2007], lógica borrosa [Avogadri, Valentini, 2009] que han sido aplicadas en el análisis de microarrays. Estas técnicas se pueden aplicar de manera conjunta en las diversas etapas de un análisis de expresión, pero es necesario incorporar conocimiento de los sucesivos análisis realizados en los diferentes casos para incluirlo en los sucesivos análisis.

Así, se plantea la necesidad de integrar un sistema de razonamiento basado en casos (CBR) en una arquitectura multiagente que permita analizar y extraer la información sobre las sondas que caracterizan a los individuos. El sistema de razonamiento utilizado será Case Base Reasoning (CBR), basa su funcionamiento en el conocimiento almacenado en memoria, que está compuesta por los casos que se han ido sucediendo en el tiempo [Kolodner, 1993], con lo que resulta muy adecuado para resolver el problema en consideración. Los sistemas CBR analizan y obtienen soluciones mediante algoritmos de búsqueda, recuperación, comparación y adaptación a las etapas del ciclo de razonamiento recuperación, reutilización revisión y aprendizaje. La implementación de estos algoritmos en las diferentes etapas del ciclo de razonamiento posibilita la realización de un análisis de expresión, facilitando la creación de estrategias muy similares a los procesos seguidos en los laboratorios médicos, que permiten hacer clasificaciones por medio de recuperación de información de análisis previos, facilitando la detección y eliminación de sondas relevantes e irrelevantes en análisis anteriores.

En este capítulo, se define la hipótesis y los objetivos que se pretenden alcanzar en este trabajo de investigación. En esta memoria se presenta un estudio de arquitecturas multiagentes y la integración de diferentes técnicas que permiten dotar a los agentes del sistema de capacidades como el aprendizaje o la autonomía en el análisis de expresión. Se describe el alcance del proyecto, las principales características del modelo propuesto, así como los componentes y conceptos tecnológicos en los que se apoyará el sistema, marcando aspectos relacionados con el desarrollo de agentes y sistemas multiagente.

1.1 Hipótesis y Objetivos

En esta memoria se plantea la necesidad de la creación de un sistema de planificación automático que permita generar flujos de acciones óptimos para el análisis de datos de microarrays. Para llevar a cabo dicha tarea, se plantea la posibilidad de usar una arquitectura multiagente distribuida para gestionar la planificación automática de los algoritmos aplicados en los análisis de expresión. Se plantea que una arquitectura multiagente es adecuada para llevar a cabo un análisis de expresión, ya que proporciona la infraestructura necesaria para crear sistemas flexibles, autónomos, con capacidad de aprendizaje que facilitan la gestión de forma automática el proceso de planificación y la extracción del conocimiento. En el marco de esta investigación para el análisis de

microarrays, la opción estimada como más eficiente es la combinación de las arquitecturas multiagentes con los sistemas de razonamiento basados en casos [Kolodner, 1993] y especializaciones como los sistemas de planificación basados en casos [Corchado *et al.*, 2008b].

El objetivo fundamental de esta investigación es el desarrollo de un modelo de planificación automático que facilite la resolución de problemas dinámicos mediante la generación de planes en tiempo de ejecución. La planificación automática se adaptará para el análisis y la extracción de conocimiento en datos procedentes de microarrays. El sistema de planificación automático deberá ser capaz de generar el flujo de ejecución de acciones que mejor se adapta al problema actual.

También será necesario identificar la arquitectura y modelo adecuado para llevar a cabo la planificación automática. Los sistemas multiagentes (SMA) son una arquitectura adecuada que la planificación distribuida y automática ya que permiten integrar diferentes modelos de razonamiento que permiten generar modelos de planificación automática en base a la información obtenida planificaciones anteriormente llevadas a cabo.

Además del sistema de planificación automática, se llevará a cabo un estudio de las diferentes técnicas existentes empleadas en los análisis de expresión. Estas técnicas engloban el preprocesado, filtrado, cluster/clasificación y extracción del conocimiento, por lo que se llevará a cabo un estudio de las alternativas existentes y se desarrollarán nuevas en determinados casos.

El modelo debe proporcionar un mecanicismo de extracción del conocimiento con capacidades de adaptación que proporciona el conjunto de patrones que caracterizan a determinados individuos. El sistema se limitará a la extracción de patrones de relevancia para su posterior análisis por parte del personal adecuado.

Finalmente, el modelo creado, debe ser un modelo de planificación genérico fácilmente adaptable a otros entornos y problemas aparte del análisis de los microarrays.

1.2 Motivación

La tecnología actual ha avanzado de modo considerable en el campo de la genómica y ha permitido adquirir gran cantidad de información a partir de pruebas realizadas en los laboratorios. Los microarrays, se han convertido en una herramienta fundamental para la investigación en el campo de la genómica, permitiendo estudiar la influencia de determinados genes en las enfermedades de los pacientes [Quackenbush, 2001]. En la actualidad existen diversos tipos de microarrays tales como los CGH arrays [Shinawi, Cheung, 2008], los arrays de expresión [Affymetrix] etc. Los arrays de expresión

contienen información sobre los genes de los pacientes, mientas que los CGH arrays contienen información sobre las ganancias o pérdidas en determinadas regiones cromosómicas. La información proporcionada por los arrays de expresión llega a alcanzar varios millones de datos por cada una de las pruebas según sea el tipo de array. Esta gran cantidad de información requiere de técnicas de minería de datos que puedan trabajar con ella de modo eficiente ya que normalmente están orientadas a trabajar con grandes cantidades de información pero no de variables por lo que se hace necesario explorar las técnicas más adecuadas.

Los sistemas de análisis de datos permiten automatizar tareas que permiten ayudar a la toma de decisiones en diversos campos de la medicina. Muchos de estos sistemas integran técnicas de inteligencia artificial para intentar simular el comportamiento de las personas pero están normalmente orientados a realizar tareas muy concretas en entornos muy restringidos [Chua *et al.*, 2008] [François *et al.*, 1992].

En la actualidad, existen diversas aproximaciones para la ayuda a la toma de decisiones, entre estos sistemas se incluyen myGrid [Stevens *et al.*, 2004] [Vittorini *et al.*, 2008], Gene-CBR [Riverola *et al.*, 2006]. En estas aproximaciones, es el usuario el responsable de crear la secuencia de acciones para resolver un problema específico. MyGrid basa su funcionalidad en el uso de servicios web implementados según OGSA (Open Grid Services Architecture) [Foster *et al.*, 2002]. La arquitectura OGSA facilita la resolución de problemas complejos de modo distribuido mediante arquitecturas SOA [Erl, 2005] y las arquitecturas Grid, pero no poseen capacidades de adaptación ni de descubrimiento de nuevas planificaciones en los análisis. Sin embargo, existen nuevas líneas de investigación con capacidades de adaptación como son los sistemas de razonamiento basados en casos (CBR) [Kolodner, 1993] [Kolodner, 1983a] [Kolodner, 1983b] [Joh, 1997], estos sistemas, resuelven nuevos problemas teniendo en cuenta experiencias pasadas [Kolodner, 1993] [Leake, Kendall-Morwick, 2008] y permiten su integración dentro de los agentes permitiendo de este modo incorporar mecanismos de adaptación en los sistemas multiagentes. Sin embargo, los sistemas de ayuda a la toma de decisiones basados en CBR presentan deficiencias cuando se trabaja con grandes cantidades de información como es el caso de los microarrays.

Los sistemas multiagentes [Wooldridge, Jennings, 1995] son una alternativa a las arquitecturas SOA y Grid en la computación distribuida. El comportamiento en estas arquitecturas se descompone en una serie de entidades denominadas agentes que poseen una serie de capacidades y habilidades. Estas capacidades y habilidades se pueden implementar de diferentes modos, anteriormente se han nombrado los CBR aunque existen otros mecanismos que permiten incorporar adaptabilidad a estos tipos de sistemas. Así, se tienen planificación basados en casos (Case-Based Planning) [Glez-Bedia, Corchado, 2002] y los planificación basados en casos que integran el modelo BDI (Belief Desire Intention) [Glez-Bedia, Corchado, 2002]. El modelo planificación basados en casos con modelos BDI consiste en la fusión de ambos conceptos mediante la formalización de los problemas siguiendo el modelo BDI y usando como mecanismo de razonamiento modelos de planificación. El modelo de planificación basado en casos es

una extensión del modelo CBR que permite incorporar diversos mecanismos para la generación de planes en base a los planes anteriores de modo que la combinación de ambos modelos se ajusta a las necesidades del problema. Por tanto, sería interesante el uso de una arquitectura multiagente que permita extraer conocimiento de datos procedentes de análisis de expresión. Los sistemas multiagente permiten incorporar servicios web, facilitando de este modo la adaptabilidad de los agentes en base a los servicios disponibles permitiendo de este modo una mayor capacidad evolución en función de las planificaciones llevadas a cabo con anterioridad. La separación de la funcionalidad del motor de razonamiento facilita del mismo modo la reutilización de las técnicas de análisis aplicadas y el uso de técnicas ya implementadas, así como la modificación del comportamiento de forma separada a los servicios. Además, la separación de la capa de procesamiento permite a los agentes una mayor disponibilidad ya que la posibilidad de bloqueo de los agentes se reduce de modo considerable.

Los sistemas multiagente permiten modelar el entorno de trabajo de un laboratorio simulando las acciones y los análisis llevados a cabo en análisis de expresión genética. El proceso de análisis de los microarrays recibe el nombre de análisis de expresión [Lander *et al.*, 2001] [Simon, 2009] y consiste en una serie fases: recuperación de los datos, preprocesado, análisis estadístico e interpretación de los resultados. Estas fases se pueden agrupar en tres etapas: normalización y filtrado, clustering y clasificación, y extracción del conocimiento. Estas etapas se pueden integrar dentro del ciclo de razonamiento implementado por un planificador basado en casos: recuperación, adaptación, revisión y aprendizaje permitiendo de modo sencillo integrar un análisis de expresión dentro de un sistema de planificación basado en casos. Para cada una de estas etapas de un análisis de expresión, existen diversas técnicas que se pueden aplicar de modo aislado o bien en combinación, el problema radica en seleccionar las técnicas a aplicar y el orden de las mismas [Riva *et al.*, 2005] [Lee *et al.*, 2005]. La selección de las técnicas y su orden de aplicación representan un plan, un planificador basado en casos permite seleccionar para cada una de las etapas el conjunto de técnicas que mejor se adaptan a cada uno de los problemas y su orden. Finalmente, la formalización del sistema se lleva a cabo mediante un BDI y es a partir de esta formalización sobre la que se aplica el motor de razonamiento de un planificador basados en casos, de este modo, se puede incorporar el razonamiento de un planificador basado en casos con el modelo BDI en la arquitectura de los agentes del sistema para llevar a cabo las planificaciones que permitan generar los flujos de análisis.

1.3 Metodología y plan de trabajo

El proceso seguido para llevar a cabo la tesis se ha fundamentado en la metodología Action Research. La metodología identifica el problema para así formular una hipótesis. Se parte de unos conceptos definidos dentro de un modelo cuantitativo de la realidad.

Posteriormente, se realiza una recopilación, organización y análisis de información, continuando con el diseño de una propuesta enfocada a solucionar el problema. Finalmente, se formulan las conclusiones respectivas tras evaluar los resultados obtenidos de la investigación. Para seguir esta metodología, es necesario definir una serie de actividades que permitan alcanzar los objetivos planteados así como demostrar la hipótesis. Las actividades planteadas son las siguientes:

1. Definición del problemática: planteamiento del problema junto con el entorno que lo define para poder establecer los objetivos y la hipótesis de la investigación.
2. Revisión del estado del arte: análisis de la problemática y soluciones en entornos similares que se han llevado a cabo por parte de otros investigadores. El proceso de revisión se centrará tanto las técnicas aplicadas en análisis de expresión como en las alternativas y métodos para llevar a cabo la planificación automática. La revisión del estado del arte debe ser un proceso continuo a lo largo de la investigación.
3. Proposición de modelos y validación del cumplimiento de los objetivos a medida que se concretan los diferentes componentes. Los modelos se descomponen en una serie de componentes para facilitar el proceso de validación y mejorar así el proceso de investigación.
4. Estudio de los resultados obtenidos mediante la comparación con otros procedimientos y así determinar si se han alcanzado los objetivos y la hipótesis inicialmente planteada.
5. Publicación de los resultados obtenidos a lo largo de la investigación tanto en congresos como en revistas. La publicación en congresos es de gran importancia porque permite la asistencia a conferencias que facilitan el intercambio de ideas de primera mano.

Las actividades anteriores se ejecutan iterativamente a lo largo del proceso de investigación, es por ello que se puede considerar un proceso iterativo e incremental al igual que metodologías usadas en ingeniería del software como el proceso unificado.

1.4 Estructura de la memoria

Para probar la hipótesis de partida y alcanzar los objetivos establecidos, se ha estructurado esta memoria en siete capítulos.

En los dos primeros capítulos se llevará a cabo un estudio del estado del arte de la expresión genética y técnicas de inteligencia artificial distribuida. En el capítulo 2, se hará una visión general de la problemática en los análisis de expresión haciendo una revisión de las técnicas usadas para el análisis de microarrays. Se dará una visión general sobre el funcionamiento de algunos tipos de microarrays y los procedimientos seguidos

para realizar las diferentes etapas de análisis de expresión. Las etapas de un análisis de expresión son: recuperación de los datos, preprocesado, análisis estadístico e interpretación de los resultados, por tanto, el capítulo 2 se centrará en el estudio de las técnicas empleadas para cada una de estas etapas.

En el capítulo 3 de la memoria se hace una revisión de mecanismos de planificación e integración en entornos distribuidos dentro de la inteligencia artificial distribuida. En este capítulo, se hará un especial hincapié en lo referente a la planificación automática mediante el estudio de diferentes modelos de planificación como son los CBR, planificador basado en casos bajo el modelo BDI y su integración en arquitecturas distribuidas como son los sistemas multiagentes.

En el capítulo 4 se expondrá la arquitectura propuesta y mecanismos seguidos para llevar a cabo la planificación automática mediante el uso del modelo de un planificador basado casos con el modelo BDI y la incorporación de redes bayesianas, teoría de grafos y procesos estadísticos para la generación de planes eficientes de modo automático.

El capítulo 5 muestra el caso de estudio bajo el que se aplica la arquitectura propuesta. Se concertarán las diferentes técnicas usadas para llevar a cabo el análisis de expresión. Estas técnicas se integran en el modelo propuesto y ya son propias del caso de estudio. Dentro de este capítulo se analizarán los resultados obtenidos para el modelo y las técnicas propuestas.

El capítulo 6 mostrará las conclusiones alcanzadas a lo largo de la investigación.

Para finalizar, se ha añadido un capítulo en inglés a modo de resumen del trabajo realizado. Este capítulo se añade con motivo del cumplimiento de la reglamentación para el doctorado europeo.

Capítulo II

Expresión genética

VNiVERSiDAD
D SALAMANCA

Uno de los objetivos del sistema de planificación es la generación de un modelo que permita llevar a cabo un análisis de expresión de modo automático. Para poder llevar a cabo este modelo, es necesario primero realizar un estudio de las técnicas usadas en los análisis de expresión para su posterior integración en el sistema. Este capítulo se centra en las diferentes alternativas existentes para llevar a cabo el preprocesado, filtrado, cluster/clasificación y extracción del conocimiento.

Introducción

En los últimos años se han producido grandes avances en el campo de las ciencias biomédicas orientados al estudio de características genéticas. Estos avances se han producido fundamentalmente en la tecnología usada para llevar a cabo los análisis. Las nuevas tecnologías de la información han proporcionado nuevas técnicas para extraer información de las características genéticas de pacientes, lo que permite un mejor estudio de determinadas enfermedades. Los avances en el análisis genético vienen de la mano de los chips conocidos como microarrays. Estos chips permiten realizar un estudio detallado de los diferentes niveles de expresión de genes y secuencias de ADN, permitiendo la búsqueda de patrones que caracterizan determinadas enfermedades. Normalmente, el uso de microarrays está asociado con las enfermedades relacionadas con la herencia genética y, por tanto, su principal aplicabilidad se produce en el estudio de enfermedades relacionadas con diferentes tipos de cáncer, aunque también se puede usar para genotipado de enfermedades infeccionas como puede ser el VIH o la hepatitis.

El material genético de los diferentes organismos se codifica mediante una serie de nucleótidos que componen el ADN, de tal forma que la secuencia de nucleótidos permite diferenciar a unos organismos de otros. En esencia, un nucleótido está compuesto por el azúcar presente (desoxirribosa), una base nitrogenada que suele ser A, C, T, G (Adenina, Citosina, Timina, Guanina) y un fosfato que actúa como enlace de los nucleótidos. Lo único que diferencia a un nucleótido en la secuencia de ADN es la base nitrogenada. Por tanto, la secuencia de ADN se suele expresar en función de ella. En los seres vivos, el ADN se encuentra distribuido en dos hebras enlazadas mediante una serie de puentes de hidrógeno que unen determinadas combinaciones de bases nitrogenadas. Las hebras se distribuyen a modo de doble hélice unidas por parejas de nucleótidos. Las

parejas que se pueden encontrar son las siguientes: A-T, C-G. Los microarrays se fundamentan en base a este principio tan básico de las uniones entre nucleótidos.

El ARN (Ácido RiboNucleico) posee similitudes con el ADN aunque, a diferencia del ADN, el azúcar presente es la ribosa y la base Timina se sustituye por Uracilo (U). Además, se trata de un polímero lineal. Dentro del ARN se distinguen varios tipos en base a la funcionalidad: ARNt de transferencia, ARNr ribosómico, ARNm mensajero [Kokko, 2006].

El ADN de las células contiene la información necesaria para construir las diferentes proteínas. Las proteínas son las responsables del funcionamiento de las diferentes células y el comportamiento de los diferentes órganos. Las proteínas son una secuencia de aminoácidos, donde cada aminoácido se codifica como una secuencia de tres bases denominadas codones. Las proteínas se construyen a partir de la información del ADN de las células que es transmitida mediante el ARN.

En este capítulo, se va a realizar un estudio del estado del arte de las diferentes técnicas que se emplean para el análisis de expresión genética. En un primer apartado, se realizará una visión general de la tecnología usada para la extracción de la información para posteriormente realizar un estudio de las diferentes técnicas que se pueden aplicar en este campo. El estudio de las técnicas será de relevancia para determinar las alternativas existentes para llevar a cabo análisis de expresión genética y su inclusión en mecanismos de planificación automática.

2.1 Microarrays

Un microarray consiste en una secuencia de moléculas de ADN sobre un sustrato sólido distribuidas a modo de una matriz de dos dimensiones. Se trata de fragmentos de ADN que se caracterizan por ser: i) secuencias cortas llamadas oligonucleótidos; ii) o bien tratarse de secuencias más largas denominadas cDNA (ADN complementario, sinterizado a partir del mRNA); iii) o bien productos de PCR (replicación in vitro de secuencias de ADN mediante la reacción en cadena de la Polimerasa). Estos fragmentos de ADN constituyen las sondas que se distribuyen a lo largo del chip, de modo que cada una de las sondas contiene una secuencia de ADN sin complementar, es decir, sólo contiene una hebra de las dos que forman el ADN. El ADN/ARN a analizar se tiñe con una serie de fluorescencias y se introduce en el chip, de modo que se produce una hibridación que consiste en la unión de las bases del ADN introducido junto con las bases de las muestras existentes en las sondas. Posteriormente, se produce una lavado del chip y se eliminan los fragmentos de ADN/ARN introducidos que no presentaban fragmentos complementarios. Finalmente, en base a la fluorescencia proporcionada por el tinte presente en cada sonda, se puede determinar el nivel de marcado de cada una las sondas mediante el uso de un escáner [López *et al.*, 2002]. En la figura (Fig 1)

[Affymetrix c], se muestra de modo gráfico las sondas y el ADN/ARN con diferentes fluorescencias introducido en el chip.

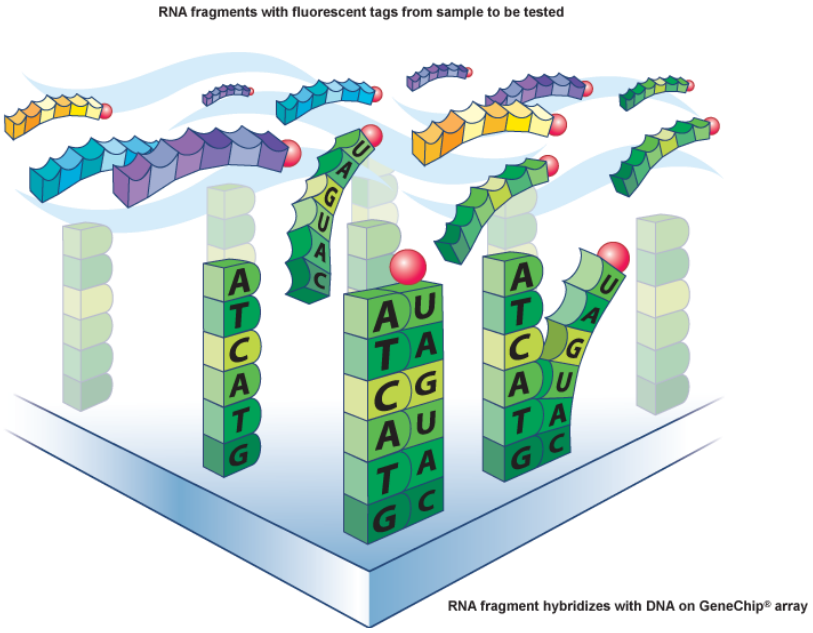

Fig 1. Hibridación

Según la clasificación proporcionada por el informe de [López *et al.*, 2002], en base al material utilizado para la fabricación se pueden tener los siguientes tipos de chips:

- Gene Chips: oligonucleótidos (arrays de expresión)
- cDNA Array: cDNAs
- Protein Chip: Proteínas
- Tissue Chip: Tejidos

Las secuencias de ADN a imprimir en los chips se extraen de diferentes bases de datos como GeneBank, dbEST [Olson, 2006]. Estas bases de datos almacenan aquellas secuencias que se consideran básicas para el análisis. El número de nucleótidos existentes en cada sonda varía en función del tipo de chip. Así, los genechips [Affymetrix] disponen de unas decenas de valores por cada una de las sondas, mientras que los cDNA array [Shinawi, Cheung, 2008] contienen algunos centenares de valores.

2.1.1 Arrays de expresión

Los arrays de expresión están compuestos por una serie de oligonucleótidos distribuidos en sondas a lo largo de la superficie del chip. Las sondas representan un conjunto de nucleótidos de especial interés extraídos de bases de datos que se imprimen sobre la base del chip y sobre las que se realiza la hibridación. Para la generación de los arrays de expresión existen dos alternativas [Quackenbush, 2001]:

- Arrays en dos colores
- Arrays en una sola coloración

Los arrays en dos colores usan dos muestras de ARN que se tiñen de diferentes colores, de modo que cuando se produce la hibridación, se genera una serie de luminiscencias que se escanean y se crea el fichero de microarrays. Lo normal, es tomar como referencia un paciente sin una determinada enfermedad, denominado muestra de control, y etiquetarlo con un determinado color. Por otro lado, las muestras tomadas de pacientes afectados por la enfermedad en consideración se tiñen de otra coloración. Por tanto, en base a las coloraciones finales, se pueden detectar las zonas similares y diferentes de ambos pacientes.

En los arrays con una sola coloración, la muestra del paciente se colorea con un determinado tinte y, posteriormente, se introduce en el chip y se realiza la hibridación. El resultado es una imagen con diferentes escalas de coloración en función de los enlaces creados entre los nucleótidos y las respectivas coloraciones. En la figura (Fig 2) extraída del artículo [Quackenbush, 2001] se muestra el proceso de hibridación seguido para ambos tipos de arrays descrito anteriormente. En la parte de arriba se tienen los arrays formados por dos coloraciones y en la parte inferior los arrays formados por una sola coloración.

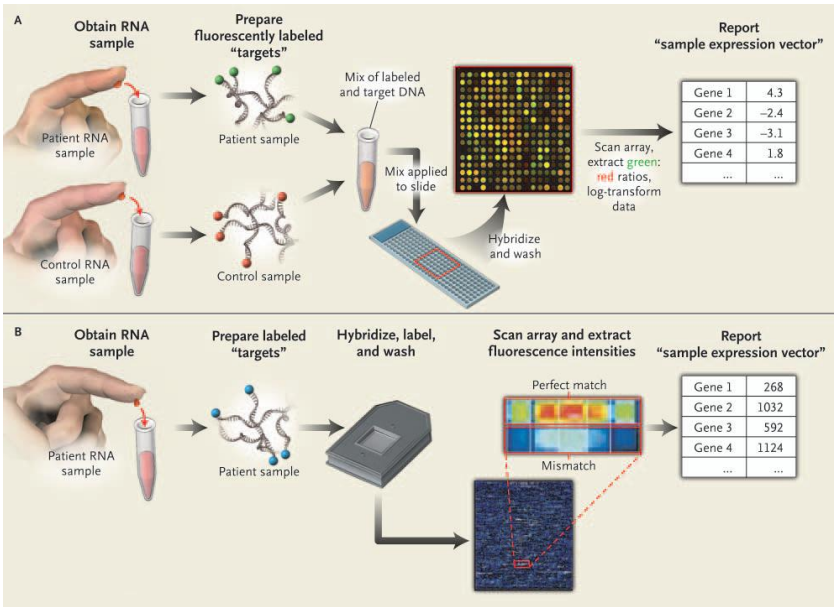

Fig 2. Proceso de hibridación

Concretamente, el modelo seguido para los arrays de expresión por el fabricante Affymetrix (el fabricante más importante de este tipo de chips) es el basado en una sola coloración siendo, por tanto, el resultado final de un análisis de expresión similar al mostrado en la imagen (Fig 3). De esta forma, los datos a tratar serán un conjunto de valores que indican las intensidades de coloración de las diferentes sondas existentes en la superficie del chip. En la figura (Fig 3) [Affymetrix c], se muestra el resultado obtenido una vez escaneado el chip y medidos los diferentes valores de luminiscencia para cada una de las sondas.

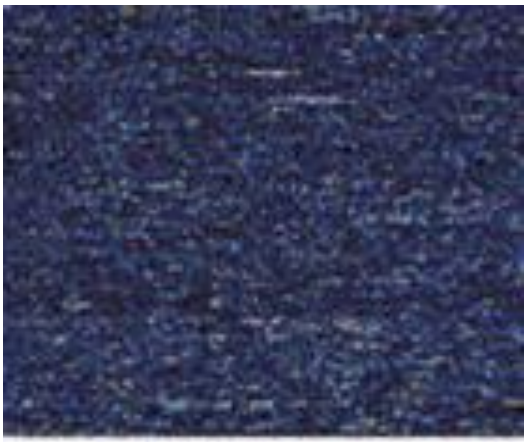

Fig 3. Microarray hibridado

2.2 Análisis de expresión

El proceso de análisis de los arrays de expresión se denomina análisis de expresión [Lander *et al.*, 2001] y consiste en las siguiente etapas: recuperación de los datos, preprocesado, análisis estadístico e interpretación de los resultados. Olson [Olson, 2006] propone un flujo de ejecución similar al indicado por Lander, en el que las etapas se descomponen en una normalización de los datos, análisis estadísticos para la extracción de sondas diferenciables, clasificaciones y finalmente extracción de patrones de relevancia para el estudio de la importancia médica. En otros trabajos como los de [Simon, 2009], se establece un flujo de trabajo similar en los análisis de los microarrays, las diferentes fases descritas son: preprocesado, búsqueda de genes, localización de patrones en cluster, clasificadores para nuevos elementos y finalmente interpretación de los resultados. Como se puede ver, el flujo de trabajo a llevar a cabo es muy similar con independencia de los autores y trabajos siendo lo que varía de modo considerable el conjunto de técnicas aplicadas a cada una de estas etapas.

En los siguientes apartados, se analizan diferentes técnicas que se pueden usar en los análisis de expresión. Las etapas que se han considerado han sido: preprocesado, filtrado, cluster y extracción del conocimiento. Estas etapas se han obtenido a partir de aquellas seguidas en los trabajos anteriormente citados [Lander *et al.*, 2001] [Olson, 2006] [Simon, 2009].

2.3 Preprocesado

El preprocesado [Simon, 2009] [Noriega *et al.*, 2009] es la etapa inicial que se lleva a cabo en un análisis de expresión. Normalmente, los chips de microarrays contienen varias sondas con la misma información replicada en diferentes zonas del chip. Por tanto, para una misma secuencia de nucleótidos se obtienen diferentes medidas de intensidades que se tienen que transformar y convertir en una única medida. De modo paralelo a la secuencia de búsqueda de un gen, se introduce una secuencia que posee la base central cambiada por su complementaria. La primera de las secuencias se denomina Perfect Match (PM), mientras que la segunda de ellas MissMatch (MM). Además de la replicación de las sondas, también existen problemas debido a los dispositivos hardware que obtienen los datos. Así, para un mismo chip los valores de intensidades leídos pueden variar en función del escáner usado y por tanto, se debe de realizar un reajuste de los datos previo uso. Este proceso se denomina normalización. El preprocesado de la información se encarga de solucionar estos problemas, generando un valor por cada conjunto de sondas y normalizando las medidas de luminiscencia

proporcionadas por los chips, de modo que se elimine la influencia de los componentes hardwares utilizados.

Existen varias técnicas usadas para el preprocesado de la información. Se trata de la fase más estandarizada dentro de los análisis de expresión. Las técnicas más usadas en trabajos como [Kibriya *et al.*, 2007] [Noriega *et al.*, 2009] [Ungera, 2005] son:

- Robust Multi-array Average (RMA) [Irizarry *et al.*, 2003]
- Probe Logarithmic Intensity Error (PLIER) [Affymetrix b]
- Microarray Affymetrix Suite 5.0 (MAS 5.0) [Hubbell, 2002]
- dChip [Man, Qin, 2007] [Noriega *et al.*, 2009]

El funcionamiento de estas técnicas varía ligeramente unas de otras, así RMA sólo se basa en la en los valores de PM para el cálculo de las intensidades realizando una regresión de los valores, PLIER se basa tanto en el MM como en el PM para eliminar ruidos de fondo y define una función objetivo a minimizar, MAS 5.0 que también se basa en los valores del PM-MM pero realizar un preprocesado en función del test de Tukey a partir de los datos del PM-MM, finalmente, se tiene MAS 5.0 que funciona de modo similar a RMA mediante regresión lineal.

Existen diversos estudios sobre la idoneidad de cada uno de los métodos [Man, Qin, 2007] [Noriega *et al.*, 2009] [Affymetrix b]. Los resultados de estos estudios se pueden resumir en la siguiente tabla (Tabla 1). En la tabla, se muestran las ventajas y desventajas de cada uno de los métodos.

	Ventajas	Desventaja
MAS 5.0	Cada microarray es independiente de los otros microarrays existentes en el conjunto	No es tan sensible a pequeños cambios en la expresión de sondas como RMA y PLIER
PLIER	Tiene la capacidad de detector pequeñas variaciones en los datos	Los microarrays presentan más varianza en la intensidad de la señales
RMA	Minimiza la varianza en los microarrays	Comprime los resultados obtenidos para valores de intensidades bajos
dChip	Fácilmente reproducible los resultados para los arrays y varianza estable	Puede tener más de un valor posible en caso de que valores de intensidades no concuerden

Tabla 1. Comparación de las técnicas de preprocesado

La aplicación de alguna de alguna estas técnicas es de uso obligado en cualquier sistema de análisis puesto que sin ellas los resultados varían en función de las herramientas usadas y los resultados no son fiables. Por tanto, en el marco de la investigación se tendrá en cuenta su incorporación como paso previo a realizar cualquier tipo de análisis de datos.

2.4 Técnicas de reducción de la dimensionalidad

Las técnicas de reducción de la dimensionalidad se emplean para reducir el número de variables existentes mediante la generación de un nuevo conjunto de variables que recogen en gran medida la información de las variables originales, o bien mediante la eliminación de variables que no disponen información relevante. En el caso de la reducción de la dimensionalidad, las técnicas más extendida son las componentes principales (ACP) [Costache *et al.*, 2009] aunque existen otras técnicas orientadas al uso de variables categóricas como es el caso del Análisis de Correspondencias Múltiples [Salvador, 2003]. Entre las principales técnicas de reducción de dimensionalidad se tienen:

- Análisis de Componentes Principales (ACP) [Costache *et al.*, 2009]
- Análisis factorial [Wang, 2009]
- Multidimensional Scaling (MDS) [Cox, Ferry, 1993]
- Análisis de Correspondencias [Salvador, 2003]

De modo resumido, un ACP permite transformar un conjunto de variables correlacionadas en un conjunto nuevo de variables independientes entre sí que son combinaciones lineales de las originales. La reducción de la dimensionalidad suele ser muy elevada aunque es necesario determinar el criterio de parada en la selección de variables. El análisis factorial es muy similar al ACP a diferencia de que en lugar de buscar maximizar la varianza final recogida por el nuevo conjunto de variables trata de reflejar las correlaciones entre las variables originales. El análisis de correspondencias se comporta de modo diferente a las técnicas anteriores y no es de importancia para el estudio ya que está orientado para trabajar con variables categóricas por lo que no se ajusta al caso de estudio. Finalmente, se tiene MDS que se diferencia de modo considerable de ACP o el análisis factorial, ya que lo que trata es de generar a partir de la matriz de distancias original una nueva matriz de dimensión inferior en el que la relación de orden de cercanía entre los elementos de la matriz o bien la distancias de la nueva matriz varíen en lo mínimo posible con respecto a la matriz original.

En la investigación en la que nos centramos, podría ser interesante sobre todo ACP, porque a pesar de permitir reducir la dimensionalidad de los datos de modo considerable cuando las variables presentan correlaciones, en ocasiones, dificulta el análisis final de los datos. Normalmente, el resultado final obtenido no puede ser interpretado de modo directo puesto que las variables creadas suelen venir expresadas en función de las variables originales y no existe una relación directa entre ellas. En ocasiones, el uso de ACP para la representación de los datos en baja dimensionalidad resultó ser considerablemente peor que otras técnicas como MDS, ya que a simple vista era complicado detectar agrupaciones de los datos a pesar de que la variabilidad regida era alta.

El análisis factorial debido a la incorporación de las rotaciones se puede seleccionar en cierto modo la relación entre las variables originales y los factores. Así, se pueden establecer los valores de los pesos de manera que estén muy correlacionados con unas pocas variables e incorrelados con el resto. Al poder estimar de forma más sencilla las relaciones entre las variables originales y los factores, los resultados son más fácilmente interpretables y, además, las representaciones son más intuitivas que en el caso del ACP.

Finalmente, MDS da muy buenos resultados para representar los datos en baja dimensionalidad pero no es recomendable usarlo para trabajar con los datos generados para realizar análisis posteriormente. Sólo se debe usar para representar los datos una vez se ha realizado el procesamiento. En caso de querer trabajar con los datos es más aconsejable usar ACP o análisis factorial.

Todas estas técnicas presentan como principal problema que es difícil trabajar con los datos generados puesto que generan un conjunto de variables completamente diferentes por lo que se pierde la referencia con las variables originales. Por ello, no es muy aconsejable su aplicación en etapas tempranas del análisis aunque si se pueden usar para realizar representaciones en baja dimensionalidad aunque la variabilidad recogida final sea baja.

Como alternativa a las técnicas de reducción de la dimensionalidad, se pueden aplicar técnicas orientadas a la detección y eliminación de sondas que carecen de información relevante. Estas técnicas se suele fundamentar en el uso de test estadísticos de diferentes características. Estos test estadísticos se agrupan en paramétricos en caso de hacer suposiciones sobre la distribución de los datos, y no paramétricos en caso de no hacer suposiciones sobre su distribución. El test de ANOVA es uno de los más usados en el campo de la bioinformática [Pavlidis, 2003] [De Haan *et al.*, 2009], aunque hay que tener en cuenta que no siempre se pueden tomar como ciertas las suposiciones sobre la distribución de los datos.

Las técnicas estadísticas más usadas en problemas de bioinformática se limitan a determinar similitudes de variables entre diferentes grupos existentes. Estas técnicas permiten determinar de forma sencilla si una variable presenta características similares para cada uno de los grupos, o bien si la variable presenta características diferenciadoras permitiendo diferenciar elementos de cada uno de los grupos mediante la información proporcionada por ella.

Para llevar a cabo esta tarea, existen contrastes paramétricos y no paramétricos. Los contrastes paramétricos requieren hacer un cierto número de suposiciones sobre los datos en cuanto a su distribución que no siempre se cumplen, por lo que es necesario en muchas ocasiones usar contrastes no paramétricos que no hacen suposiciones sobre la distribución de los datos. Las pruebas más usadas suelen ser la ANOVA [Cvijović *et al.*, 2005] por parte de los contrastes paramétricos y el contraste de Kruskal Wallis [Kruskal, Wallis, 1952] por parte de las paramétricas. También existen otros test propios del campo de la bioinformática como son el PAM [Tibshirani *et al.*, 2002] y el SAM [Tusher

et al., 2000]. Aparte de ANOVA y Kruskal Wallis, se tienen el T-test [Ardanuy, Martín, 2003] y el U-test [Mann, Whitney, 1947] que permiten comparar igualdad de variables pero sólo para dos grupos, T-test para prueba paramétrica y U-test para no paramétrica.

En el caso de U-Test o T-test, la prueba permite determinar la igualdad de distribución de los datos y en caso de no poder determinar que la distribución de los datos es igual, determina que se rechaza la H_0 lo que no significa que las variables no tengan la misma distribución.

En ANOVA no siempre se puede suponer que los datos siguen distribuciones normales y es necesario aplicar test estadísticos para comprobarlo. Algunos test que se puede aplicar son el del Kolmogorov-Smirnov [Brunelli, 2001] o el de Shapiro-Wilk [Jurečkováa, Picek, 2007]. En el caso de los microarrays en pocas ocasiones se da la situación de que las sondas sigan distribuciones normales, por lo que no se puede aplicar ANOVA, aunque se aplica en muchos estudios [Pavlidis, 2003] [De Haan *et al.*, 2009] sin llevar acabo test de normalidad sobre las variables. El test de Kruskal Wallis se aplica cuando no se cumple alguna de las condiciones de ANOVA. El test se usa habitualmente para determinar si las muestras de los grupos siguen la misma distribución o bien para determinar la igualdad de medianas.

Otras técnicas como el SAM o PAM presenta problemas en cuanto a la selección de parámetros ya que el número de genes seleccionados varía de modo considerable según su selección aunque suele ser bastante bajo tal y como se puede ver en el estudio [Pan, 2002]. Además, la selección de los genes se fundamenta en técnicas no paramétricas por lo que se puede aplicar con independencia de la distribución de los datos. Además, estas medida no presentan un gran fundamento matemático/estadístico ya que no presenta ningún contraste y la selección de genes varía en función de la constante s_0. No obstante, su uso está muy extendido en la bioinformática.

Como se puede ver, todas estas técnicas están orientadas a la extracción de sondas relevantes una vez se haya realizado la clasificación final por lo que inicialmente no son útiles para la eliminación de sondas que carecen de importancia para realizar análisis de datos. Por ello, se hace necesaria la exploración de nuevas técnicas que permitan la eliminación de sondas como paso posterior al preprocesado para así permitir trabajar con un número inferior de variables. Poder reducir la dimensionalidad, es fundamental para poder aplicar técnicas de procesado más complejas y de mayor carga computacional que ralentizarían el funcionamiento de otras técnicas como puede ser el caso de redes neuronales.

2.5 Técnicas de Clustering

El análisis de clustering es una rama del análisis estadístico multivariante usado para la detección de patrones en la clasificación de elementos. El análisis de clustering se emplea en gran variedad de campos como la bioinformática [Corchado *et al.,* 2009] [Kerr *et al.*, 2008], rutas de vigilancia [Carbó *et al.*, 2005] [García *et al.*, 2005] etc. Los métodos empleados para realizar el clustering difieren considerablemente en función del tipo de datos y el volumen de información de la que se dispone. Tradicionalmente, las técnicas de clustering se descomponen en las siguientes categorías [Rung *et al.*, 2009 [Kerr *et al.*, 2008] [Akhbardeh *et al.*, 2008]:

- Jerárquicas, que incluye por ejemplo: dendogramas [Saitou, Nie, 1987], AGNES [Kaufman, Rousseeuw, 1990], DIANNA [Kaufman, Rousseeuw, 1990].
- Redes neuronales, como los Self-Organized Maps [Kohonen, 1982], GCS [Fritzke, 1995] [Hodge, Austin, 2001] o ESOINN [Furao *et al.*, 2007].
- Métodos basados en particiones mediante la minimización de funciones objetivo como son k-means [Hartigan, Wong, 1979] y PAM [Kaufman, Rousseeuw, 1990] [Van der Laan *et al.*, 2002] (Partition Around Medoids), CLARA [Kaufman, Rousseeuw, 1990].
- Modelos probabilísticos tales como EM [Xu, 1997] (Expectation-maximization) y FANNY [Kaufman, Rousseeuw, 1990].

Tradicionalmente, los diferentes métodos tratan de minimizar la distancia existente entre los individuos de las agrupaciones. En determinados algoritmos es necesario establecer el número de cluster de antemano, o bien de seleccionarlo una vez se ha finalizado el algoritmo. Las redes neuronales permiten realizar, en determinados casos, una selección automática del número de cluster en función de los elementos existentes. Normalmente, requieren de una fase de adaptación previa de las neuronas a los datos de entrada que permita generar las conexiones entre los elementos y, en determinados casos, los grados de conectividad de las neuronas son prefijados de antemano.

Las técnicas empleadas para la realización de cluster varían según el tipo de problema con el que se esté tratando. Así, en el caso de la bioinformática, las técnicas de clustering más extendidas son las jerárquicas y métodos basados en la minimización de funciones objetivo, debido a su simplicidad en la realización de cálculos. Los métodos jerárquicos como los dendogramas [Saitou, Nie, 1987], no requieren establecer el número de cluster de antemano y se puede determinar a partir de la representación gráfica obtenida. Los métodos basados en particiones optimizan una determinada función objetivo y requieren establecer el número de cluster de antemano [Xu, 1997]. El algoritmo k-means presenta problemas con los puntos atípicos. Para resolver esta situación, otros métodos jerárquicos como el PAM, seleccionan los centroides dentro de

los elementos a agrupar para evitar el problema de los datos atípicos. Tanto los métodos jerárquicos como los que minimizan una determinada función objetivo, presentan deficiencias a la hora de reconocer agrupaciones de individuos debido a que estos algoritmos suelen asociar cada elemento en base a las distancias a un determinado elemento. Esto conlleva a que no sean capaces de detectar correctamente agrupaciones. Las redes neuronales artificiales (RNA) [Pavón *et al.*, 2008], [Pavón *et al.*, 2007a] [Pavón *et al.*, 2007b] tienen la capacidad de adaptarse a las superficie de los datos pero normalmente requieren de un tiempo extra para poder llevar a cabo esta tarea.

2.5.1 Métodos basados en particiones

Los métodos basados en particiones consisten en dividir los datos de entrada en una serie de cluster de modo que se optimice una determinada función objetivo que se ha definido previamente. La función objetivo depende normalmente de una medida de distancia, por lo que los resultados finales dependen de la medida definida. En general, se trata de métodos iterativos cuyo resultado final depende de la iniciación del algoritmo, por lo que los resultados no son siempre reproducibles y consistentes. Los métodos basados en particiones más comunes y extendidos son K-medias [Hartigan, Wong, 1979] y PAM [Kaufman, Rousseeuw, 1990] [Van der Laan *et al.*, 2002].

Los métodos basados en particiones requieren de la definición de una función objetivo que se trata de maximizar o minimizar. La definición de dicha función depende de la medida de distancia que se esté usando por lo que los resultados finales dependen en gran manera de dicha medida de distancia. Además, se trata de métodos iterativos cuyo resultado final depende de los datos de iniciación. El procedimiento de cluster PAM, permite realizar clustering de modo similar a K-medias pero debido al uso de medioides en lugar de centroides permite un mejor funcionamiento que k-medias, al ser más robusto frente a datos atípicos.

2.5.2 Métodos Jerárquicos

Los métodos jerárquicos crean agrupaciones de los datos mediante estrategias aglomerativas y divisorias:

- Aglomerativo o bottom-up: Parten de un conglomerado por cada uno de los cluster y los van fusionando hasta tener un solo conglomerado o alcanzar una

condición de parada. Ej. Dendogramas [Saitou, Nie, 1987], AGNES [Kaufman, Rousseeuw, 1990],

- Divisorio o top-down: Empiezan por un solo grupo y lo van dividiendo hasta que cada conglomerado contiene un solo elemento o bien se alcanza una condición de parada. Ej. DIANA [Kaufman, Rousseeuw, 1990].

Los dendogramas y AGNES son una técnica muy útil para realizar cluster cuando existe una clara diferencia entre los elementos de los conglomerados. Sin embargo, en casos en los que datos son bastante homogéneos, los resultados y la selección del punto de corte del árbol es complicada y la tasa de error suele ser demasiado alta. La eficiencia del método depende de la medida de distancia seleccionada, por lo que los resultados varían considerablemente en función de la selección de una u otra. A diferencia de otros métodos, los resultados son consistentes para una misma configuración, siendo por tanto posible la replicación de los resultados.

DIANA tiene las mismas limitaciones que otras técnicas como los dendogramas. La selección del número final de cluster se tiene que realizar manualmente resultando complicado en muchas ocasiones. La selección se hace en base a las distancias mayores existentes dentro de cada grupo. Además, tampoco permite detectar agrupaciones con formas atípicas como son líneas paralelas, esquemas concéntricos etc. Los resultados obtenidos por medio de esta técnica son muy similares a los dendogramas por lo que no es de especial interés su estudio para el objetivo de este trabajo de tesis.

2.5.3 Redes Neuronales

Las redes de principal interés para este trabajo de tesis son las redes neuronales de aprendizaje no supervisado entre las que destacan las redes basadas en mallas. Entre las redes basadas en mallas, las más habituales son los mapas autorganizados de Kohonen (SOM). Las redes SOM [Kohonen, 1982] tienen una variante de método de aprendizaje que basa su comportamiento en métodos similares a las Neural Gas (NG) [Martinetz, Schulten, 1991]. Las rede SOM crean una maya que se ajusta automáticamente a un área específica. La gran desventaja, es que el número de las neuronas que se distribuyen a lo largo de la superficie y el grado de conectividad son prefijadas de antemano. La red Growing Cell Structure (GCS) [Fritzke, 1995] no establece el número neuronas y el grado de conectividad pero establece la dimensionalidad de las mayas. Esto complica la fase de separación entre elementos de un grupo y su distribución a través de la superficie. Existen otras redes neuronales como la SOINN [Shen, 2006] (Self-Organizing Incremental Neuronal Network), y la ESOINN [Furao *et al.*, 2007] (Enhanced Self-Organizing Incremental Neural Network) que no establecen tampoco la dimensionalidad de las mallas. A diferencia de la SOINN, la ESOINN no diferencia dos etapas de entrenamiento por lo que no es necesario determinar el modo o las condiciones bajo las cuales se pasa

de una etapa a la siguiente. No obstante, en la red ESOINN es necesario realizar un ajuste de las neuronas a la superficie de los datos que se quiere agrupar por lo que requiere de una fase de ajuste previa división de las neuronas. Las redes ART (Adaptive Resonance Theory) [Carpenter, Grossberg, 1987] se pueden considerar una alternativa para realizar clustering en lugar de los mapas autorganizados. Son redes no supervisadas que facilitan la detección automática de cluster y en sus últimas versiones, permiten la incorporación de patrones de entrenamiento continuos. La gran desventaja que presentan estas redes es la selección de los patrones de monitoreo [Akhbardeh *et al.*, 2008] para determinar el número de cluster. Otra desventaja, es que la extracción del conocimiento es más complicada que en las redes basas en mallas, siendo más complejo de interpretar el aprendizaje.

Si nos centramos un poco en el funcionamiento de cada una de las redes, podemos ver que la red SOM presenta un comportamiento similar a otros algoritmos como PAM. A diferencia del PAM los centroides se representan por pesos en las conexiones de neuronas, y el comportamiento es menos determinista ya que la actualización de las neuronas se realiza de modo aleatorio en función de los patrones de entrada seleccionados y las actualizaciones se realizan en base a las relaciones de vecindad establecidas. Las relaciones de vecindad varían según el grado de conectividad establecido entre las neuronas, así la red puede presentar forma de malla o forma de circunferencia. Por tanto, estas redes presentan las mismas deficiencias que técnicas como el PAM vistas anteriormente.

La red neuronal ESOINN permite crear mallas que se distribuyen a lo largo de la superficie de modo automático, a partir de las cuales se crean las diferentes agrupaciones en base a las distancias de las conexiones presentes. A diferencia de la SOM, la red neuronal permite tener grados de conectividad variable por cada una de las neuronas. Con respecto a la GCS como principal diferencia presenta que los lados de los polígonos que forman las mallas son de dimensionalidad variable y no se fijan de antemano a un valor, lo que facilita el paso de adición y borrado de neuronas a la malla puesto que no es necesario mantenerlo constante. Los resultados finales obtenidos por las GCS y la ESOINN son muy similares y se pueden ver en los artículos de los autores [Fritzke, 1993] [Furao *et al.*, 2007].

Como se puede ver, las redes neuronales presentan ventajas en el funcionamiento con respecto a los métodos vistos anteriormente, ya que permiten la detección de cluster de modo automático y permiten ajustarse a cualquier superficie de datos. Por el contrario, los métodos basados en particiones o los jerárquicos presentan problemas a la hora detectar cluster en superficies que presentan distribuciones de datos atípicas como formas alargadas. Estos métodos basan su funcionamiento principalmente en las distancias con respecto a un elemento tomado como representante de un conjunto por lo que suelen detectar de modo correcto hiperesferas.

2.5.3.1 Redes Neuronales de aprendizaje supervisado

Las redes mostradas anteriormente se enmarcan dentro de las redes de aprendizaje no supervisado. El aprendizaje no supervisado consiste en la realización del aprendizaje de la red sin especificar en ningún momento las salidas para llevarlo a cabo. Por el contrario, existen otras redes de aprendizaje supervisado en las que el entrenamiento se realiza indicando tanto los valores de entrada como los de de salida para posteriormente realizar una clasificación.

En las redes neuronales de aprendizaje supervisado es difícil determinar la naturaleza de las representaciones internas generadas por la red para responder ante un problema determinado. A diferencia de los modelos estadísticos clásicos, no es tan evidente conocer en una red la importancia que tiene cada variable predictoria sobre la salida del modelo. Sin embargo, esta percepción acerca de las RNA como un compleja "caja negra" no es del todo cierta. De hecho, han surgido diferentes intentos por interpretar los pesos de la red neuronal de los cuales el más ampliamente utilizado es el denominado análisis de sensibilidad [Fu, Chen, 1993] [Theron, De Paz, 2006].

El análisis de sensibilidad está basado en la medición del efecto que se observa en una salida debido al cambio que se produce en una entrada. Así cuanto mayor efecto se observe sobre la salida, mayor sensibilidad podemos deducir que presenta respecto a la entrada. No obstante, desde el punto de vista práctico, es más interesante ver cómo le afectan las entradas a una determinada salida para los diferentes patrones, en lugar de comparar la influencia de las entradas con las salidas. Esto es debido a que en numerosas ocasiones, lo que se quiere realizar es determina la pertenencia de un individuo a un determinado grupo.

A partir de los datos obtenidos por un análisis de sensibilidad, se pueden determinar las neuronas que afectan en mayor medida a cada una de las neuronas de salida. En la figura (Fig 4), se puede ver una representación de un análisis de sensibilidad descrito en el trabajo [Theron, De Paz, 2006]. Las barras verticales representan cada uno de los casos, la amplitud de los rectángulos en valor obtenido para el análisis de sensibilidad de modo que si su valor es positivo se representa con perspectiva hacia fuera y si es negativo hacia dentro. Los valores mostrados en la parte superior indican el valor de salida real y el valor obtenido por la red, la escala de colores de azul a rojo indican la magnitud del valor de salida. Los colores de las barras se usan para diferenciar los diferentes atributos. En la parte inferior, se tienen unas coordenadas paralelas que representan a modo de líneas el valor de de cada individuo para las diferentes variables. Una descripción más detallada se puede ver en el trabajo de Theron y De Paz [Theron, De Paz, 2006].

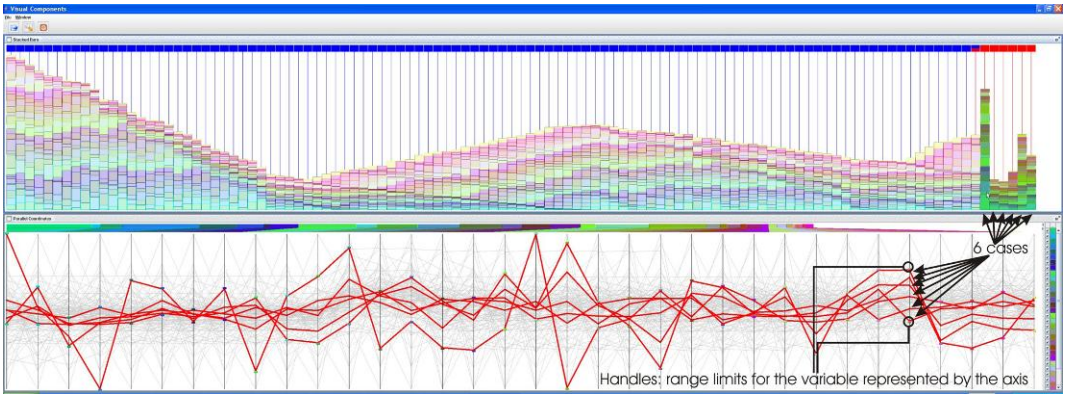

Fig 4. Representación gráfica de un análisis de sensibilidad

En teoría un análisis de sensibilidad permite obtener el grado de influencia en la salida de las diferentes entradas de una red neuronal. No obstante, los resultados prácticos obtenidos continúan siendo complejos de analizar aún a pesar de realizar representaciones gráficas.

Un entrenamiento erróneo de la red neuronal puede provocar también problemas en los resultados obtenidos. Así, en el caso de no barajar los patrones de entrenamiento en cada iteración puede conllevar a entrenamientos anómalos que aunque produzcan resultados factibles, no reflejen bien las relaciones de entrada con las salidas de modo real.

2.5.4 Métodos probabilísticos

Los métodos probabilísticos como Expectation Maximization (EM) [Bolstada *et al.*, 2007] [Dellaert, 2002] o Fuzzy Analysis (FANNY) [Kaufman, Rousseeuw, 1990] asignan los diferentes elementos en base a probabilidades. Los métodos calculan las probabilidades de pertenencia a cada uno de los conjuntos y se distribuyen los elementos maximizando o minimizando determinadas funciones.

Estos métodos presentan como principal ventaja, que permiten dar ciertos grados de pertenencia de los elementos a conjuntos. Por tanto, se puede medir en cierto modo el posible grado de corrección de las clasificaciones realizadas. Los métodos más característicos son EM y FANNY que se describen a continuación de forma abreviada.

Las agrupaciones llevadas a cabo mediante EM son bastante buenas para el caso de variables categóricas aunque el tiempo de cálculo suele ser superior a otras técnicas no basadas en redes neuronales. Para el caso de variables continuas, es más aconsejable

usar otras técnicas como dendogramas o redes neuronales que suelen dar mejores resultados.

Por el contrario, en FANNY la maximización de la función es eficiente si el número de variables no es muy elevado. En caso contrario el proceso de cálculo se vuelve muy pesado y no resulta eficiente. El problema de rendimiento es similar al encontrado con el Travelling Salesman Problem (TSP) [Dantzig *et al.*, 1954] en su resolución mediante programación lineal.

2.6 Técnicas de Extracción del Conocimiento

Debido a la capacidad de cálculo existente hoy en día, los ordenadores son capaces de realizar procesos de clasificación complejos que en muchas ocasiones son difíciles de justificar. Por ello, que se hace necesario el uso de técnicas que proporcionen información sobre los procesos de clustering y clasificación, de forma que se permita extraer la información necesaria para explicar los resultados obtenidos mediante las técnicas de clustering, permitiendo de este modo crear procesos de clasificación fácilmente entendibles por las personas humanas.

Las técnicas de extracción de conocimiento son variadas, aunque de modo genérico se podrían agrupar de la siguiente manera:

- Teoría de los Conjuntos Aproximados (Rough Set)
- Árboles de decisión
- Análisis de sensibilidad
- Redes bayesianas

Los conjuntos aproximados, árboles de decisión y redes bayesianas se aplican a cualquier tipo de problemas mientas que los análisis de sensibilidad se aplica a las redes neuronales que utilizan el algoritmo de retropropagación en el aprendizaje.

El objetivo de los conjuntos aproximados y los árboles de decisión, es proporcionar a un experto humano información sobre la clasificación realizada por el sistema mediante la generación de reglas que servirán de apoyo para la toma de decisiones. Es necesario señalar, que las técnicas de extracción de conocimiento no persiguen sustituir la capacidad de juicio y experiencia de un experto humano en la evaluación de un diagnóstico, pero sí complementarla y añadir una sistemática o procedimientos comunes de análisis.

2.6.1 Teoría de Conjuntos Aproximados

Esta teoría tiene sus orígenes en los trabajos de Pawlak [Pawlak, 1982], [Pawlak, 1997]. Como visión global, esta teoría permite obtener a partir de una agrupación previa, una tabla de decisión facilite la clasificación de nuevos individuos en base a una serie de atributos. La teoría de conjuntos aproximados basa su funcionamiento en un marco de trabajo en el que el conocimiento es impreciso, lo que supone que para unos mismos atributos se pueden dar clasificaciones diferentes para el problema.

Esta técnica presenta como principal problema que está orientada a trabajar con atributos cualitativos y no cuantitativos. El hecho de no poder trabajar con atributos cardinales, sería subsanable, si fuera posible el uso de atributos ordinales realizando una discretización previa de los valores, pero por la propia definición de las clases de equivalencia se ve que no es posible el uso de esta técnica en dichas situaciones. Sería necesario, un replanteamiento de la definición de las clases de equivalencia para poder utilizarla en estos casos.

Otro de los principales problemas que presenta esta técnica de extracción del conocimiento, es la alta computación que requieren los algoritmos para la extracción de los reductos a partir de los subconjuntos, especialmente, cuando se tratan casos con más de 100 atributos. Es por ello, que esta técnica se hace poco abordable en muchas situaciones si se quieren obtener los diferentes reductos posibles. No obstante, el principal obstáculo es la imposibilidad de usar atributos ordinales para la obtención de reglas con desigualdades como la siguiente:

$$a_1 > V_{a_1} \wedge \ldots \wedge a_n < V_{a_n} \rightarrow (a_1^p > V_{a_1^p}, \ldots, a_s^p > V_{a_s^p})$$

Donde a_i representa el atributo i y V_{ai} el valor para ese atributo.

2.6.2 Reglas de decisión

Son algoritmos que permiten obtener, a partir de una serie de grupos predefinidos, la información que permite clasificar a un individuo en un grupo u otro. Estos métodos, facilitan explicar las agrupaciones llevadas a cabo mediante diferentes algoritmos, pudiendo ser usados con diferentes técnicas de cluster. La generación de las reglas permite generar explicaciones en métodos considerados como cajas negras, como es el caso de las redes neuronales artificiales. Las redes neuronales artificiales, como se ha visto anteriormente en la sección (2.5.3), pueden ser usadas para la generación de cluster, pero presentan como principal desventaja frente a otros métodos como pueden ser el PAM [Kaufman, Rousseeuw, 1990] o los métodos jerárquicos, la imposibilidad de explicar de manera sencilla las agrupaciones realizadas. En los modelos jerárquicos o en

modelos basados en particiones, es bastante sencillo explicar el resultado final, pero de igual modo resulta complejo reducir la información disponible a la realmente relevante que se ha tenido en cuenta llevar a cabo la clasificación. El objetivo de las técnicas generadoras de reglas de decisión pretende resolver esta situación.

Existen diversos tipos de reglas, por un lado se tendrían los árboles de decisión, y por otro lado las reglas de decisión que generan modelos de reglas no jerárquicos. La diferencia es que en los árboles de decisión, se puede generar a partir de las reglas que los componen un representación del conocimiento en forma de árbol, puesto que los antecedentes de las reglas poseen elementos comunes, mientras que en las reglas de decisión tradicionales, los antecedentes no tienen por qué tener elementos comunes.

Dentro de las reglas de decisión, existen diversos algoritmos muchos de ellos están implementados en software de minería de datos como Weka. Entre los algoritmos existentes que han sido probados están RIPPER [Cohen, 1995], M5 [Holmes *et al.*, 2007], One-R [Holte, 1993] y PART [Frank, Witthen, 1998]. En algunas pruebas realizadas, se han comportado de manera muy similar a los árboles de decisión y han generado las mismas reglas para algún caso de estudio.

RIPPER se trata de un algoritmo con carga computacional en la búsqueda de los literales en cada una de las reglas, puesto que realiza una búsqueda en amplitud de todas las alternativas. El algoritmo se puede extender fácilmente con el simple hecho de incluir en la categoría negativos a todos los elementos que no pertenecen a la clase positiva. De inicio está orientado al uso de variables categóricas aunque se puede extender al uso de variables continuas aunque ya no es tan eficiente. A pesar de todos estos inconvenientes, el funcionamiento del algoritmo es mucho mejor que el caso de los rough set, se ha probado para varios miles de variables y 100 individuos y ha finalizado sin problemas en un tiempo de computación bastante bajo, apenas unos segundos.

2.6.3 Árboles de decisión

Son un caso particular de las reglas de decisión que permiten además representar las reglas en forma de árbol [Quinlan, 2002]. Es una técnica de extracción del conocimiento ampliamente usada en diferentes ámbitos, desde la simple estadística, hasta la bioinformática [Corchado *et al.,* 2009]. Se trata de una técnica bastante evolucionada y con una diversa variedad de propuestas. Entre ellos es posible enumerar a CLS (Concept Learning System) [Hunt *et al.*, 1966], ID3 (Induction Decision Trees) [Quinlan, 1979], CART (Classication and Regression Trees) [Breiman *et al.*, 1984] [Timofeev, 2004], OC1 (Oblique Classier 1) [Murthy, 1994], ASSISTANT [Cestnik *et al.*, 1987] o C4.5, C5.0/See5 [Quinlan, 1993], que han alcanzado una notable repercusión dentro del ámbito de la biomedicina.

Estas técnicas, permiten extraer el conocimiento en problemas en los que se dispone de una colección de casos, que contienen un conjunto de atributos que se consideran independientes y un atributo que se considera dependiente. Los atributos pueden ser de diversos tipos, pudiendo ser continuos o discretos, o bien nominales y ordinales. La meta es determinar el conjunto de atributos independientes que implican un determinado valor para el atributo dependiente.

Los árboles se componen de dos tipos de nodos, los nodos hojas y los nodos de test. Los nodos hojas, se corresponden con un nodo final del árbol que contiene la información del grupo final en el que los casos son clasificados, mientras que los nodos de test, contienen una regla lógica que determina el subárbol a seleccionar según el valor que tome. En caso de ser cierto, se selección la rama de la izquierda y en caso de no cumplirse la condición se selecciona la rama de la derecha. De este modo, los casos se van disgregando según el valor de los nodos de test aplicando la estrategia de divide y vencerás.

Un ejemplo de árbol de decisión, se puede ver en la siguiente figura (Fig 5). Se puede ver que se parte de un nodo raíz y en cada uno de los nodos intermedios se tiene la regla de decisión. Finalmente, se tiene la clasificación final a la que se llega y el número de individuos clasificados de cada tipo. Este árbol se ha generado mediante CART (Classication and Regression Trees) [Breiman *et al.*, 1984].

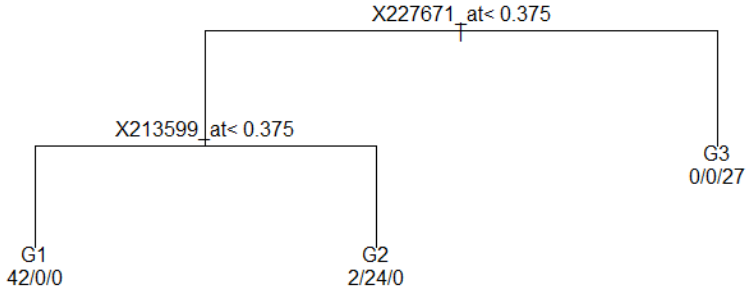

Fig 5. Árbol de decisión generado por CART

De modo general, los árboles de decisión se construyen a partir de la maximización de funciones que tienen en cuenta la tasa de acierto y fallo de los nodos clasificados en función de los atributos seleccionados y la condición establecida.

Los árboles de decisión son una técnica que permite extraer de forma sencilla las reglas que permiten explicar las clasificaciones de los individuos. Las reglas se pueden expresar mediante antecedentes y consecuentes o bien se puede representar la información en forma de árbol lo que facilita la comprensión. Las últimas variantes de los algoritmos, permiten la inclusión de condiciones de decisión más variables permitiendo incorporar atributos de tipo numérico bien se tratante de de valores continuos u ordinales. No obstante, el uso de atributos numéricos, requiere en la generación de las reglas de decisión sobrecarga en gran medida el procesamiento, por lo

que es necesario discretizar los valores para poder aplicar estos procedimientos cuando el número de atributos es elevado.

Si no se lleva a cabo la discretización de los valores, la selección del punto de corte para las reglas de decisión se convierte en una etapa demasiado pesada para el procesamiento. No obstante, tanto CART como J48 se comportan de manera bastante eficiente cuando el número de atributos es elevado, por lo que es realmente útil cuando se tiene este problema. Otras técnicas, como los rough set, no permiten incorporar atributos continuos u ordinales y tampoco pueden manejar los volúmenes de información que manejan los árboles de decisión.

Además, las técnicas empleadas para la selección de atributos, se pueden aplicar para la extracción de sondas relevantes. Las fórmulas aplicadas a la selección de atributos se pueden aplicar para la extracción de atributos de relevancia ya que permiten extraer las sondas que mejora caracterizan a los grupos.

El principal problema que presentan estas técnicas es que su carga computacional suele ser elevada por lo que no se puede aplicar de forma directa a un gran volumen de información de información por lo que no son aplicables de modo directo sin haber realizado un filtrado previamente. Además del filtrado, tal y como se ha comentado anteriormente, es necesario realizar una discretización de los valores para facilitar la selección de las reglas de los nodos.

2.6.4 Redes Bayesianas

Una red bayesiana es un modelo probabilístico multivariado que relaciona a un conjunto de variables aleatorias mediante una grafo dirigido acíclico que permite inferencia bayesiana para la estimación de probabilidades de variables no conocidas a partir de variables conocidas. Las redes bayesianas constan de los siguientes elementos:

- Un conjunto de nodos, uno por cada variable aleatoria.
- Un conjunto de arcos dirigidos que conectan los nodos.
- Cada nodo contiene la distribución de probabilidad condicional que lo relaciona con el nodo padre.

Se fundamentan en teorema de Bayes [Duda *et al.*, 1973]. El teorema de Bayes permite calcular la probabilidad de un suceso a priori A_i sabiendo que ha ocurrido a posteriori el suceso B. Se define de la siguiente manera.

$$P(A_i \mid B) = \frac{P(B \mid A_i) \cdot P(A_i)}{P(B)} = \frac{P(B \mid A_i) \cdot P(A_i)}{\sum_{j=1}^{n} P(B \mid A_j) \cdot P(A_j)}$$

Donde:

- $P(A_i)$ es la probabilidad de que ocurra el suceso A_i
- $P(B \mid A_i)$ es la probabilidad de que ocurra el suceso B sabiendo que ha ocurrido A_i
- $P(A_i \mid B)$ es la probabilidad de que haya ocurrido anteriormente el suceso A_i sabiendo que posteriormente se ha dado B
- $P(B)$ es la probabilidad de ocurra el suceso B.
- Los sucesos A_i son excluyentes

Inicialmente, las redes bayesianas se construían a mano mediante el conocimiento de un experto. Actualmente, se han desarrollado procedimientos para realizar el proceso de modo automático o semiautomático. Para definir la red de Bayes [Duda *et al.*, 1973], es necesario establecer el conjunto de variables que forman parte del problema que se está tratando.

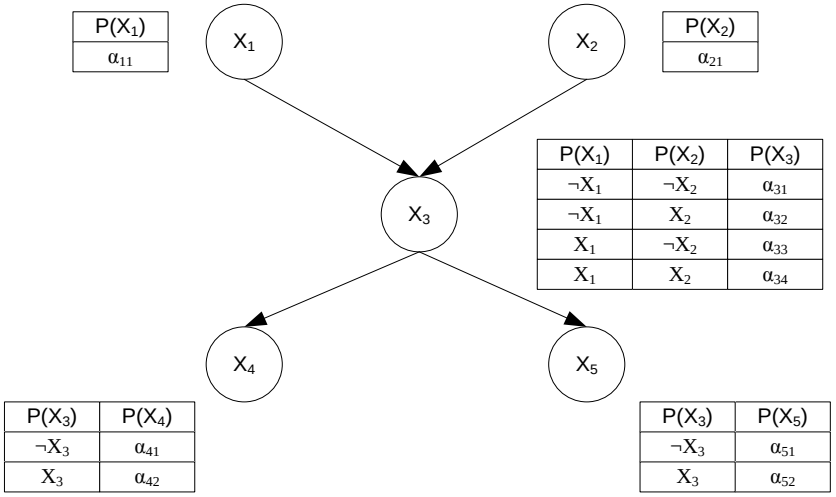

Fig 6. Red Bayesiana

Siguiendo este procedimiento, se podría calcular la probabilidad de que ocurran las diferentes combinaciones de sucesos, consiguiendo así determinar la probabilidad de las diferentes combinaciones.

En el caso de querer establecer una prioridad a priori a través de la red bayesiana, conocido lo que ha ocurrido a posteriori se puede realizar siguiendo la siguiente ecuación:

$$P(X \mid e) = \alpha P(X, e) = \alpha \sum_{y} P(X, e, y)$$

Donde X representa una variable de un suceso que se quiere calcular a priori, e un conjunto de variables conocidas a posteriori, α es una constante de modo que la suma de los diferentes sucesos de la variable X sea 1.

Siguiendo el procedimiento anterior y la definición de la probabilidad de un determinado suceso $P_B(X_1, X_2,..., X_n)$, se puede calcular la probabilidad de que ocurra el suceso $P(X \mid e)$ suponiendo que X se corresponde con las diferentes alternativas del suceso X_1 y e con los sucesos X_4, X_5

$$P(X \mid e) = \alpha\langle P(X_1 \mid X_4 X_5), P(\neg X_1 \mid X_4 X_5)\rangle =$$
$$\alpha < \sum_{x_2 \in X_2} \sum_{x_3 \in X_3} P(X_1, x_2, x_3 X_4 X_5), \sum_{x_2 \in X_2} \sum_{x_3 \in X_3} P(\neg X_1, x_2, x_3 X_4 X_5) >=$$
$$\alpha < \sum_{x_2 \in X_2} \sum_{x_3 \in X_3} P(X_1) P(x_2) P(x_3 \mid X_1 x_2) P(X_4 \mid x_3) P(X_5 \mid x_3),$$
$$\sum_{x_2 \in X_2} \sum_{x_3 \in X_3} P(\neg X_1) P(x_2) P(x_3 \mid X_1 X_2) P(X_4 \mid x_3) P(X_5 \mid x_3) >$$

Una vez se obtenida la expresión anterior, se sustituyen las probabilidades. El valor de α se establece de modo que $\alpha(P(X_1 \mid X_4 X_5) + P(\neg X_1 \mid X_4 X_5)) = 1$

Dentro de las redes bayesianas se puede obtener un clasificador con una estructura en forma de árbol que se denomina TAN (Tree Argumented Navie Bayes), en forma de red multiconectada denominada BAN.

En el caso del clasificador de Bayes simple, los cálculos se simplifican debido a que se supone la independencia entre los sucesos dada una determinada clase, por tanto, la probabilidad de que se de una determinada clase dado un conjunto de atributos viene definido de la siguiente manera:

$$P(X \mid e) = P(X) \cdot P(e \mid X) / P(e) =$$
$$P(X \mid e) = P(X) \cdot P(e_1 \mid X) \cdot \ldots \cdot P(e_n \mid X) / P(e)$$

Donde X representa las diferentes clases de un atributo y e los diferentes valores de los atributos. Gráficamente, se representa por una red, en la que los atributos no se encuentran interconectados entre sí, y sólo se encuentran conectados con el atributo que representa las diferentes clases identificadas por X. El clasificador de Bayes simple no suele dar buenos resultados puesto que la hipótesis de que los atributos son independientes no suele ser cierta.

Las redes bayesinas suponen una alternativa bastante interesante a los árboles de decisión debido a que permiten obtener una probabilidad en las predicciones y aportan más información que los árboles o las reglas de decisión. No obstante, el procedimiento de construcciones de las mismas es bastante costoso siendo por tanto el principal problema cuando se trabaja con un número elevado de variables. El proceso de

construcción requiere el cálculo de probabilidades que son necesarias estimar en base a al estudio de la frecuencia de los datos existentes.

El cálculo de las probabilidades a partir de las frecuencias de los datos se complica a la hora de realizar los cálculos con variables continuas puesto que es necesario realizar una discretización de los datos o bien aplicar lógica difusa para su cálculo. Sólo se permite el uso de algunos tipos de variables continuas con distribuciones gaussianas y relaciones lineales [Sucar, 2009].

Desde el punto de vista de la bioinformática, más concretamente en los microarrays, no es muy utilizado debido a la gran cantidad de información disponible, ya que necesitaría mucho tiempo para la realización de los cálculos y además no resulta sencilla su interpretación. No obstante, se pueden aplicar en otros campos como la planificación para la catalogación de actividades como eficientes e ineficientes.

2.6.5 Técnicas de clasificación

Las técnicas de clasificación permiten asociar casos a grupos existentes. El comportamiento de estos algoritmos es similar a los existentes en clustering pero mucho más sencillo en la mayoría de los casos. También, se pueden considerar como técnicas de clasificación a los árboles de decisión, reglas de decisión, redes bayesianas puesto que permiten realizar la misma funcionalidad por lo que se podría considerar su estudio en un mismo apartado, no obstante, se ha optado por separarlo puesto que las técnicas de clasificación básicas no proporcionan la posibilidad de la extracción del conocimiento.

Las técnicas existentes se pueden agrupar en los siguientes tipos en base a la naturaleza de los algoritmos empleados:

- Modelos probabilísticos: Naive Bayes [Duda *et al.*, 1973]
- Lógica borrosa: K-NN (K-Nearest Neighbours), NN (K-Nearest Neighbours) [Aha, Kibler, 1991].
- Definición de funciones: Sequential Minimal Optimization (SMO) [Platt, 1999]

A continuación, se describen de modo resumido las diferentes alternativas existentes en el proceso de clasificación.

2.6.5.1 Modelos probabilísticos

Los modelos probabilísticos de establece la probabilidad de pertenencia de un caso a cada uno de los grupos en función los atributos. Se fundamentan en el estudio del valor de cada atributo del nuevo caso con respecto a la frecuencia que aparece en cada uno de los cluster existentes, de modo que se asigna una probabilidad de pertenencia de un elemento a cada una de los grupos. Entre los clasificadores probabilísticos se tiene el clasificador de Naive Bayes.

El clasificador de Naive Bayes permite realizar clasificaciones de datos basándose en la aplicación del teorema de Bayes [Duda *et al.*, 1973] expuesto en el apartado (2.6.4). El clasificador de Naive Bayes, permite asociar un determinado elemento a una clase en base a las diferentes características que presenta. No obstante, para realizar dicha tarea, se realiza la suposición de que las características son independientes entre sí.

De modo muy resumido, se estima la probabilidad de asignar un determinado caso a una determinada a cada una de las clases en función de los valores de los atributos. A partir de los valores obtenidos, se asigna a la clase que proporciona el valor máximo.

En diversas pruebas realzadas, cuando se trabaja con un número elevando de variables y un número bajo de casos, el clasificador presenta claras deficiencias en cuanto a lo genérico del modelo. Posee una tasa de acierto muy elevada para los casos con los que se ha entrenado el modelo pero no es capaz de predecir de manera muy correcta casos previos. Se ha testeado el modelo en diversos datos de microarrays y se no se han obtenido buenos resultados a la hora de realizar la clasificación de nuevos individuos. No obstante, otras técnicas como los árboles de decisión, tampoco proporcionaban buenos resultados para estos casos, por lo que tampoco se puede descartar como una técnica a usar dentro de un sistema de clasificación de pacientes.

2.6.5.2 Lógica Borrosa

Las técnicas de lógica borrosa son posiblemente las más sencillas e inmediatas de usar. Se fundamentan en el uso de medidas de distancias o matrices de disimilaridad para determinar los casos más similares previamente clasificados. Una vez establecida una jerarquía de similitud con los casos previamente clasificados, se procede a recuperar un determinado número de individuos por orden de similitud. A partir de los individuos recuperados, se estudia la proporción existente de cada uno de los grupos y así se establece el grado de pertenencia del nuevo caso a cada uno de los grupos. Alguno de

los algoritmos que siguen esta técnica K-NN (K-Nearest Neighbours) o NN (K-Nearest Neighbours) [Aha, Kibler, 1991].

2.6.5.3 Definición de funciones

En la actualidad es una alternativa muy extendida dentro de los clasificadores. Consiste en la definición de funciones que permiten separar las diferentes agrupaciones. El caso más simple es la definición de funciones lineales pero el uso de funciones lineales no permite separar cualquier tipo de problema por lo que es necesario aplicar algoritmos que permitan transformar problemas no separables linealmente en problemas que sí lo son. Es por ello por lo que surge Support Vector Machine (SVM) [Vapnikand, Lerner, 1963]

Support Vector Machine (SVM) es una técnica de aprendizaje supervisado aplicada a la clasificación y regresión de elementos. SVM tiene aplicaciones en diversos campos como son la química [Li *et al.*, 2009] [Ivanciuc, 2007], modelado y simulación [Yang *et al.,* 2008], minería de datos [Lessmann, Voß, 2009] o text mining [Lo, 2008]. El algoritmo representa una extensión a los modelos no lineales [Vapnikand, Lerner, 1963], que inicialmente se desarrolló para la clasificación en problemas separables linealmente y básicamente consistía en encontrar la recta o hiperplano (en más de dos dimensiones) que permitiera separar a los elementos de un conjunto. SVM también permite separar clases de elementos que no son separables linealmente y para ello mapea el espacio de coordenadas inicial en un espacio de alta dimensionalidad mediante el uso de funciones. Debido a que la dimensionalidad del nuevo espacio puede ser muy alta, no es practicable el cálculo de hiperplanos que permiten realizar la separabilidad lineal. Para ello, se usan una serie de funciones no lineales denominadas núcleos que permiten realizar estas operaciones de modo eficiente.

Para el cálculo del clasificador, existen métodos como el Sequential Minimal Optimization (SMO) [Platt, 1999] que permiten calcular de modo eficiente la función clasificadora de modo iterativo. Al permitir el cálculo de modo iterativo facilita la acotación temporal del algoritmo pudiendo acotar temporalmente la solución. El algoritmo SVM se ajusta muy bien a los datos y posee una tasa de acierto muy alta para con el juego de casos de entrenamiento, no obstante, en las pruebas realizadas los resultados no han sido muy satisfactorios a la hora de realizar clasificaciones en juegos de datos nuevos y los resultados no mejoraban los de otros clasificadores como los árboles de decisión.

Por otro lado, implementaciones como SMO permiten realizar el proceso de modo iterativo facilitando la acotación temporal frente a otras técnicas como los árboles de decisión. Los árboles de decisión necesitan un formateo previo de los datos en caso de trabajar con muchas variables para que el funcionamiento sea eficiente. Es necesario

convertir las variables de continuas a discretas y categorizarlas en niveles para que el funcionamiento sea eficiente.

2.7 Conclusiones

En este capítulo se han visto técnicas que se aplican en las diferentes etapas de un análisis de expresión. Estas técnicas de modo aislado, no proporcionan unos resultados satisfactorios por lo que es necesario aplicarlas en conjunto para la generación de resultados satisfactorios. Por tanto, es necesario incluirlas en un sistema que automatice el proceso facilitando así la realización de los diferentes análisis. La selección de las diferentes técnicas de modo que se pueda predecir la eficiencia de un problema es de gran importancia. Los análisis de expresión se suelen caracterizar por la carga computacional y realizar la selección de planes en base a la selección manual de planes puede resultar muy costoso. Por ello, sería necesario crear un sistema que permita seleccionar flujos de ejecución que permitan de modo automático predecir la eficiencia de planes mediante la selección de las acciones y su ordenación.

En el siguiente capítulo se estudiaran las alternativas existentes para la integración de las técnicas vistas en una arquitectura distribuida con capacidades de planificación y aprendizaje.

Capítulo III

Agentes y Sistemas Multiagentes

VNiVERSiDAD
D SALAMANCA

En el capítulo anterior se han tratado las diferentes técnicas que se pueden emplear en un análisis de expresión. Estas técnicas se deben aplicar en una secuencia determinada para conseguir realizar una extracción del conocimiento de modo eficiente. Las técnicas es necesario integrarlas en un sistema que controle la ejecución de las diferentes etapas y que sea capaz de aprender de las sucesivas ejecuciones del sistema. Por todo ello, surge la necesidad de estudiar las alternativas existentes en la inteligencia artificial distribuida (IAD) que permitan realizar una planificación y control de las tareas a realizar. Dentro de la IAD un campo de especial interés es son los sistemas multiagentes.

Introducción

La Inteligencia Artificial Distribuida integra los conceptos de dos campos de conocimiento: la Inteligencia Artificial (IA) y los Sistemas Distribuidos. El campo de IA intenta comprender las entidades inteligentes, y está íntimamente relacionada con áreas como la filosofía y la psicología, pero a diferencia de ellas la Inteligencia Artificial intenta construir entidades inteligentes [Russell, Norvig, 1995]. El campo de los Sistemas Distribuidos estudia las propiedades de conjuntos de procesadores autónomos que no comparten memoria primaria, pero sí cooperan comunicándose por medio del envío de mensajes sobre una red de comunicación.

A partir de los conceptos de Inteligencia Artificial y Sistema Distribuido, la Inteligencia Artificial Distribuida (IAD) puede ser definida como un campo del conocimiento que estudia e intenta construir conjuntos de entidades autónomas e inteligentes que cooperan para desarrollar un trabajo y se comunican por medio de mecanismos basados principalmente en el envío y recepción de mensajes.

Desde que surgió, la IAD se ha interesado, entre otros temas, por estudiar el modelo y comportamiento de varios agentes que cooperan entre sí para la resolución de un problema o desarrollo de una tarea. Los tres ejes fundamentales que se han estudiado en IAD son [Labidi, Lejouad, 1993]:

- Los Sistemas Multiagente (SMA) [Tapia *et al.*, 2008] [Tapia *et al.*, 2006]: esta rama de la IAD estudia el comportamiento de agentes inteligentes que resuelven un problema de manera cooperativa.
- Resolución Distribuida de Problemas (RDP) [Sommaruga, 1993]: esta rama de la IAD trabaja con las formas de dividir un problema, para asignar las partes a un conjunto de entidades independientes y cooperantes, para que en grupo hallen la solución.
- La Inteligencia Artificial en Paralelo (IAP) [Labidi, Lejouad, 1993]: esta rama de la IAD se centra en el desarrollo de lenguajes y algoritmos paralelos para sistemas concurrentes en IAD.

Algunos autores ha intentado proporcionar una clara división de la IAD en diferentes campos [Sommaruga, 1993]. El resultado de esta división se puede ver en la siguiente figura (Fig 7).Tal y como se puede ver, se muestra la división de la IAD en los diferentes campos, una de las ramas son los sistemas multiagentes de los que derivan los agentes cooperativas.

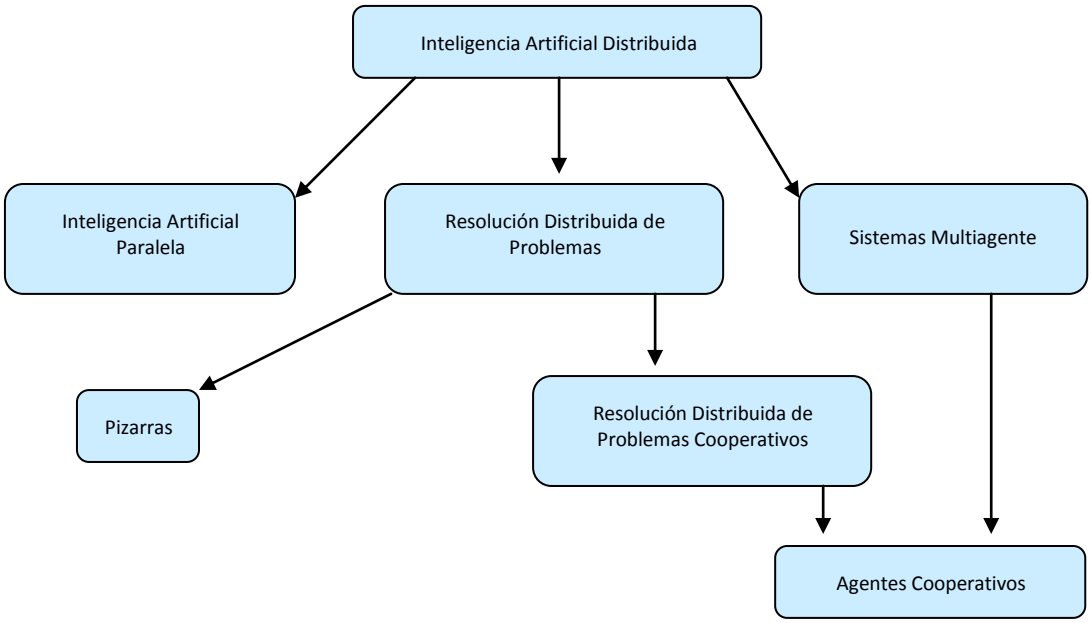

Fig 7. Clasificación de la IAD

El término de agente es difícil de definir, generalmente se pueden definir como una entidad que posee una serie de atributos de importancia en un determinado dominio [Wooldridge, Jennings, 1995].

Definiciones más formales de agentes pueden ser la proporcionada por [Labidi, Lejouad, 1993] que define un agente como: entidad física o abstracta que puede percibir su ambiente a través de sensores que es capaz de evaluar tales percepciones y tomar

decisiones por medio de mecanismos de razonamiento sencillos o complejos, comunicarse con otros agentes para obtener información y actuar sobre el medio en el que se desenvuelve a través de ejecutores. La más usada en la actualidad según [Wooldridge, 2002] define un agente como un sistema computacional que se sitúa en algún entorno y es capaz de actuar de forma autónoma en dicho entorno para alcanzar sus objetivos de diseño.

En definitiva, los agentes inteligentes pueden ser considerados como entidades que tratan de emular procesos de razonamiento o comportamientos propios de sociedades humanas. Se pueden tener agentes para la ayuda a los usuarios, para realizar determinadas tareas, supervisión, filtrar información etc.

La expansión del uso de agentes viene de la mano de la expansión de Internet y la necesidad de delegar en la tarea de búsqueda de información que se realice de forma automática y que sean capaces de realizar toma de decisiones. En la siguiente figura (Fig 8), se muestran las diferentes áreas principales que forman la denominada teoría de los agentes y las relaciones entre ellas.

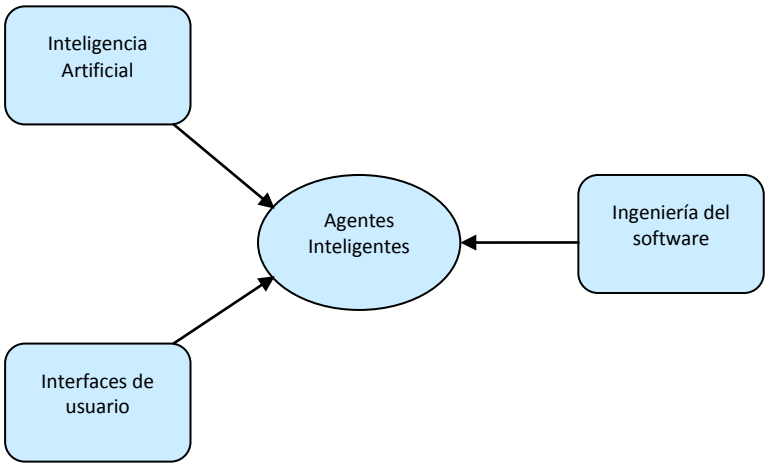

Fig 8. Áreas de conocimiento relacionadas con la teoría de los agentes

Hay que tener claro la diferencia entre los sistemas multiagentes y los sistemas basados en agentes. Un sistema basado en agentes, es aquel que utiliza el concepto de agente como mecanismo de abstracción pero podría ser implementado con estructuras software tradicionales, sin embargo un sistema multiagente es aquel que se diseña e implementa pensando en que estará compuesto por varios agentes que interactuarán entre sí, de forma que juntos permitan alcanzar la funcionalidad deseada [Bussman, Müller, 1993]. En este caso, se exige un esfuerzo de abstracción en el diseño, identificar mecanismos de aprendizaje, coordinación, negociación, etc.

En el capítulo de expresión genética, se han visto diferentes técnicas que se pueden aplicar en los análisis de expresión. Tal y como se ha comentado, las técnicas de

análisis de datos se deben de aplicar en conjunto para poder llevar a cabo el análisis de modo eficiente. El problema se puede descomponer en una serie de agentes encargados de realizar determinadas etapas de un análisis de expresión de modo que se integren en un sistema multiagente con capacidades organizativas que facilitará la obtención del resultado final. Además, el concepto de agente facilita la creación de sistemas adaptables facilitando de este modo su reorganización interna a planes de mayor eficiencia según modelos de planificación que se verán en este capítulo.

En los siguientes apartados, se verán los conceptos de agentes y sistemas multiagentes y las arquitecturas existentes para la implementación de comportamientos inteligentes en el funcionamiento los agentes. Los comportamientos inteligentes permitirán la adaptabilidad del sistema a los cambios experimentados en el entorno y en base a los análisis llevados a cabo anteriormente.

3.1 Agentes

Anteriormente, ya se ha definido el concepto de agente [Wooldridge, 2002] y como se dijo, un agente es capaz de percibir el entorno mediante una serie de sensores y en base a estas percepciones, se van a llevar a cabo una serie de acciones que han sido obtenidas mediante diversos mecanismos de razonamiento. Este funcionamiento se puede ver representado en la figura (Fig 9). Se puede ver como el agente percibe del entorno a través de unos sensores para posteriormente ejecutar una serie de acciones a través de los efectores.

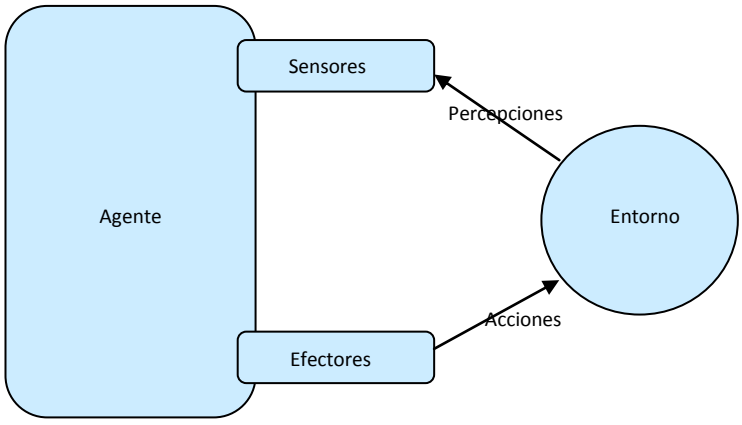

Fig 9. Agente percepciones y acciones en un entorno

Puesto que el concepto de agente no está muy claramente definido, se ha llegado a la conclusión de que la mejor forma para definir el concepto de agente es el de

definir una serie de características que una entidad debe poseer (todas o algunas de ellas) para poder ser considerada agente [Wooldridge, 2002]:

- Adaptabilidad: Capacidad de aprender y mejorar con la experiencia.
- Autonomía: Capaz de proporcionar un respuesta en base a las percepciones y al conocimiento que ha ido adquiriendo a lo largo del tiempo.
- Comportamiento Colaborativo: Capacidad de colaborar con otros agentes para llegar a una meta.
- Racionalidad: Los agentes tienen una serie de objetivos específicos que siempre intentan llevar a cabo.
- Reactividad: El agente reacciona antes los estímulos que le llegan de su entorno.
- Proactivo: Acepta solicitudes y es responsable de decidir cómo, dónde y cuándo satisfacerlas (iniciativa).
- Iniciativa: Actúa en base a los estímulos y reacciona en base a sus objetivos que debe cumplir.
- Movilidad: Capacidad de migrar entre diferentes sistemas de una red.
- Benevolencia: No poseen objetivos contradictorios.
- Continuidad Temporal: Posibilidad de persistencia a lo largo de grandes períodos de tiempo.
- Carácter: Tiene personalidad y estados emocionales. No obedece ciegamente, puede modificar, solicitar aclaraciones o rehusarlas.
- Flexible: No ejecuta procedimientos preplaneados, planifica sus acciones dinámicamente.

Existen diferentes clasificaciones de los agentes según sus propiedades y la información que almacenan. La clasificación más extendida es la siguiente [Corchado *et al.*, 2005]

- Agentes de reflejo simple: Este tipo de agente no contiene internamente estados y sus procesos o acciones que realiza son respuestas a la entrada de percepciones, a esta conexión entre percepciones y acciones se las denomina reglas de condición-acción.
- Reflejo con estado interno: En ocasiones no es posible tomar una decisión a partir de una percepción porque los sensores no proporcionan toda la información y es necesario poseer capacidad de memoria para distinguir entre estados diferentes del mundo. Las percepciones se interpretan a partir de:
    - Estado en el que se encuentra
    - Evolución del entorno
    - La forma en la que influyen las acciones en el entorno
- Agentes basados en metas: Además de los estados, los agentes necesitan almacenar cierto tipo de información sobre sus metas. Estas metas van a detallar las situaciones a las que se desea llegar. De este modo, el agente

puede combinar las metas con la información de los resultados (acciones) que emprenda y, de esta manera, poder elegir aquellas acciones que permitan alcanzar la meta.

- Agentes basados en utilidad: Los objetivos no bastan para tener un comportamiento similar al de los humanos. Un agente basado en la utilidad utiliza un criterio para estimar el grado de satisfacción de un estado para el agente que le sirve para escoger entre diferentes acciones válidas. La utilidad no deja de ser una simple función que asocia a cada estado un valor que se pretende maximizar.

Existen otras clasificaciones atendiendo a diferentes criterios como la movilidad de los agentes [Nwana, 1996], que los clasifica entre estáticos y móviles.

Los agentes inteligentes están altamente determinados por el ambiente que los rodea, por lo que se puede señalar los tipos de ambientes ante los que tiene que hacer frente:

- Accesibles y no accesibles: Si los sensores de un agente le permiten tener accesos al estado total del ambiente se dice que este es accesible a tal agente. Un agente es realmente accesible si los sensores detectan todos los aspectos relevantes a la elección de una acción.
- Deterministas y no deterministas: Si el estado siguiente de un ambiente se determina completamente mediante el estado actual, así mismo como las acciones escogidas por el agente, nos encontramos ante un ambiente determinista.
- Episódicos y no episódicos: En este ambiente la experiencia del agente se divide en "episodios". Cada episodio consta de un agente que percibe y actúa. La calidad de su actuación dependerá del episodio mismo dado que los episodios subsecuentes no dependerán de las acciones producidas en episodios anteriores.
- Estáticos y dinámicos: Si existe la posibilidad de que el ambiente sufra modificaciones mientras que el agente se encuentra deliberando, se dice que tal ambiente se comporta en forma dinámica en relación con el agente.
- Discretos y continuos: Si existe una cantidad limitada de percepciones y acciones distintas y claramente discernibles, se dice que el ambiente es discreto caso contrario es continuo.

3.2 Sistemas multiagentes

Tradicionalmente la IA se encargaba de resolver problemas de forma aislada, bien sea mediante algoritmos, conocimiento experto o métodos basados en el aprendizaje,

siendo la interacción entre los diferentes sistemas bastante escasa o incluso nula. A raíz de la aparición de las redes de ordenadores y el procesamiento en paralelo, surge la posibilidad de distribuir el razonamiento entre diferentes procesos que cooperan para la consecución de una determinada tarea lo que dio lugar a lo que se denomina la Inteligencia Artificial Distribuida [Decker, 1987]. No obstante, aunque los procesos cooperen para la consecución de una determinada tarea, cada uno de ellos tendrá una serie de objetivos que debe cumplir.

Al igual que ocurría con el término agente, es difícil establecer una definición para el concepto de sistema multiagente, por lo que se opta también por describirlos mediante una serie de propiedades que deben tener estos sistemas y clasificarlos en base a las mismas. Es el caso de la clasificación en ocho dimensiones propuesta por Sridharan [Sridharan, 1986].

- Modelo computacional: Es el nivel de distribución.
- Granularidad: Nivel de descomposición del problema.
- Paralelismo.
- Organización del sistema: Si la estructura del sistema es cambiante en lo que se refiere a la a la creación y eliminación de agentes de forma dinámica.
- Autonomía
- El resultado se puede obtener a partir de la síntesis, descomposición o cualquier mezcla de ambas.
- Recursos disponibles en el sistema.
- El nivel de interacción de los agentes.

En general un sistema multiagente es un sistema formado por múltiples agentes autónomos que muestran las siguientes características [Jennings *et al.*, 1998]:

- Los agentes son incapaces de resolver un problema por sí mismo, es decir, necesitan de la ayuda de otros para conseguir sus metas.
- No hay un sistema de control global que gestione las interacciones.
- La información está descentralizada.
- La computación es asíncrona

3.3 Técnicas de planificación

Las técnicas de planificación en la IA han variado desde los inicios. Los problemas de planificación consisten en la búsqueda de una secuencia de acciones que permiten resolver una determinada tarea. Inicialmente, la generación de las acciones se conseguía mediante planificaciones y acciones localizadas [Weld, 1994]. El primero de los enfoques es eficiente cuando las acciones interactúan entre sí, y el segundo de los casos cuando

las acciones no interfieren unas entre otras. El primero paso necesario para llevar a cabo la planificación es la definición del problema para lo que es necesario

- Descripción del mundo en un lenguaje formal
- Descripción de las metas
- Descripción de las posibles acciones

Para describir estos elementos, es necesario establecer un lenguaje que permita recoger la información del entorno mediante el uso de literales. A partir de los literales definidos, se definen una serie de reglas que permiten alterar la información de los literales. Una vez definido este conjunto de elementos, es necesario establecer un algoritmo que permita realizar la búsqueda de la secuencia de acciones que permiten alcanzar las metas.

Según el tipo de algoritmo, Weld [Weld, 1994] propone la siguiente clasificación según el tipo de planificación que se esté llevando a cabo:

- Búsqueda en espacio de estados
- Búsqueda en el espacio de planes
- Esquemas de acciones
- Razonamiento basado en planes

Las técnicas de búsqueda en espacios de estados generan un grafo que posee de nodos los diferentes estados posibles. Se interconectan los nodos para los que existen acciones que los relacionan. A partir de dicho grafo se busca el estado final y se construye el plan. Se usan algoritmos como ProgWS [Weld, 1994], RegWS [Weld, 1994] [Waldinger, 1977] que buscan a partir de la estructura de nodos la ruta que permite alcanzar el estado final. El proceso de búsqueda consiste en algoritmos de backtracking que localizan el camino que une el estado inicial y final, en el trabajo de [Weld, 1994] se puede ver el pseudocódigo usado por ProgWS y RegWS algoritmos. En la búsqueda a través del espacio de estados, los algoritmos pueden seguir dos procesos diferentes bien una estrategia progresiva partiendo desde el estado inicial o bien una regresiva en la que se parte del estado final y se busca alcanzar una estado inicial. Los algoritmos ProWS y RegWS son progresivos y regresivos respectivamente.

Las técnicas de búsqueda en espacios de planes son similares al caso anterior pero los nodos en lugar de contener literales que describen los estados, contiene reglas aplicadas. Las conexiones entre los nodos representan refinamiento de los planes mediante la adicción de una serie de reglas entre el estado origen y destino [Weld, 1994]. En los algoritmos usados hay que hacer una distinción entre planificación de orden total y ordenación parcial. En la planificación de orden total posee una secuencia de planes totalmente ordenada mientras que en la planificación de orden parcial en el que sólo se establecían una ordenación de determinados subplanes que se consideraban esenciales. Un ejemplo de algoritmo de búsqueda usado es POP (Partial Order Planner), se trata de un algoritmo de planificación parcial que parte del estado final y realiza una

búsqueda de backtracking hasta localizar un estado en el que todas las precondiciones son ciertas. El pseudocódigo se puede encontrar en [Weld, 1994].

Los esquemas de acciones definen un conjunto de acciones generales mediante el uso de variables. Las reglas también pueden incorporar precondiciones, efectos condicionados, cuantificadores universales, efectos… para establecer el conjunto de acciones permitidas en función de valor de las variables. Para poder aplicar estos algoritmos es necesario definir inicialmente el lenguaje de especificación y posteriormente los algoritmos que permiten definir el proceso de razonamiento. Los lenguajes de especificación deben permitir representar la información necesaria para poder determinar la secuencia de pasos que permite alcanzar el estado final. Así, para poder llevar a cabo la planificación, es necesario definir los siguientes elementos:

- Conjunto de fórmulas atómicas. Contienen la información de los literales y representan el conjunto de hechos que son importantes
- Conjunto de acciones definidas
- Valor inicial para los literales
- Valor final que se quiere alcanzar para los literales

Tal y como se puede ver, la información recogida es muy similar a la indicada en el trabajo de [Weld, 1994] mostrada en el apartado (3.3). Para recoger esta información, es necesario establecer un lenguaje que permite describir los literales y las acciones y a su vez un algoritmo que permita a partir del lenguaje establecer los planes adecuados.

Los lenguajes de especificación permiten describir la información necesaria para lleva a cabo las planificaciones. Entre los lenguajes de planificación más importantes se tiene a STRIPS [Fikes, Nilsson, 1971] (Stanford Research Institute Problem Solver) y extensiones del mismo como es ADL [Pednault, 1989] (Action Description Language). En ADL se añaden elementos como tipos de datos, precondiciones, postcondiciones, operadores lógicos o cuantificadores universales. A partir de los lenguajes de especificación se definen los algoritmos de planificación que permiten recuperar la secuencia de acciones que permiten recorrer el estado final e inicial. Los algoritmos más usados inicialmente fueron UCPOP [Penberthy, 1992] que es una variante de POP y SNLP, posteriormente surgieron otros algoritmos como Prodigy, TLplan o algoritmos como SHOP (Simple Hierarchical Ordered Planner) [Nau *et al.*, 1999], SHOP2 [Nau *et al.*, 2003] aplicados a la planificación jerárquica más conocida como HTN (Hierarchical Task Network). La planificación basada en HTN a diferencia de STRIPS define tareas bajo diferentes niveles de abstracción. Las tareas más básicas se corresponden con acciones mientras que las complejas se corresponden con secuencia de acciones. Esta descomposición facilita la búsqueda de soluciones por parte de algoritmos de planificación.

El razonamiento basado en planes consiste en crear a partir de planificaciones anteriores llevadas a cabo un nuevo plan que permita resolver un problema similar. Para ello, es necesario definir unos mecanismos de recuperación de planes y adaptación que permiten seleccionar los planes más similares al problema actual y adaptarlo según las

diferencias. Los sistema de planificación más usados que siguen este modelo son los CBR Case Base Reasoning) [Kolodner, 1983] [Kolodner, 1983b] [Joh, 1997] y modelo de planificación basado en casos [Corchado *et al.*, 2008b], se verán más en detalle en los apartados siguientes.

3.4 Arquitecturas de agentes

De modo genérico una arquitectura en informática se puede definir como una estructura lógica y física de una serie de componentes. En el campo de los agentes toma un significado similar, así una arquitectura de agentes específica cómo se descomponen los agentes en un conjunto de módulos que interactúan entre sí para lograr la funcionalidad requerida.

Los sistemas de planificación utilizan modelos de representación del conocimiento y razonamiento de tipo simbólico y su modo de actuación está definido por la necesidad de satisfacer unos objetivos básicos, para lo que elaboran un plan. Estos sistemas tienen la desventaja de que requieren un elevado tiempo de respuesta. Este es un gran inconveniente cuando se utilizan en problemas de tiempo real, ya que los algoritmos de planificación no siempre responden en un tiempo prudencial a las demandas del sistema. Esto es así debido a que los principios básicos de la planificación son indecidibles, lo que hace que no constituyan una opción del todo viable para los agentes. Estas críticas dirigidas fundamentalmente hacia el modelo simbólico utilizado, llevaron a la búsqueda de alternativas que utilizasen otro modelo de representación o razonamiento, como los reactivos o híbridos.

A continuación se presentan tres arquitecturas de agente que se diferencian en el modelo de razonamiento que utilizan [Wooldridge *et al.*, 1995].

- Deliberativas
- Reactivas
- Híbridas

3.4.1 Deliberativas

Esta arquitectura utiliza modelos de representación simbólica del conocimiento y suelen estar basadas en la teoría clásica de planificación partiendo de un estado inicial en el que existen una serie de planes y un estado final al que se quiere llegar. La transición del

estado inicial al final se hace siguiendo un sistema de planificación que permite determinar los planes a ejecutar para así conseguir los objetivos [Maes, 1989].

Los agentes intencionales se pueden implementar utilizando una arquitectura deliberativa. Estos agentes están basados en una serie de creencias e intenciones que utilizan para razonar [Jennings, 1993]. Entre las arquitecturas intencionales la más ampliamente extendida es la que basa su implementación en el modelo BDI (Belif, Desire, Intention) [Rao, Georgeff, 1995] [Bajo *et al.*, 2006]. El modelo BDI posee una descripción simbólica del problema mediante la especificación de los siguientes elementos: creencias la información que posee el agente del entorno, deseos las metas y objetivos que quiere alcanzar y finalmente las intenciones las acciones que los agentes pueden llevar a cabo. En el apartado (3.4.4) se detallará más en profundidad el modelo BDI.

3.4.2 Reactivas

Las arquitecturas reactivas cuestionan la viabilidad del paradigma simbólico y proponen una arquitectura que actúa siguiendo un enfoque conductivista, con un modelo estímulo-respuesta [Brooks, 1990] [Keith *et al.*, 1997]. Las arquitecturas reactivas no tienen un modelo que permita establecer una representación simbólica del mundo como elemento central de razonamiento y no utilizan razonamiento simbólico complejo, sino que siguen un procesamiento ascendente (bottom-up), para lo cual mantienen una serie de patrones que se activan con ciertas condiciones de los sensores y tienen un efecto directo en los actuadores. La opción de mantener o no una representación explícita del modelo, no es una discusión específica del campo de los agentes sino de la inteligencia artificial en general; de hecho las primeras arquitecturas de agentes reactivos se basan en los planificadores reactivos. Las principales arquitecturas reactivas son:

- Reglas situadas: la implementación más sencilla de reactividad consiste en definir el comportamiento con reglas del tipo si situación-percibida entonces acciones-específicas.
- Arquitecturas de subsunción (subsumption) [Brooks, 1990] [Brooks, 1991] y autómatas de estado finito: permiten gestionar problemas de mayor complejidad que las reglas. Las arquitecturas de subsunción están compuestas por capas que ejecutan una determinada conducta (p. ej. explorar, evitar un obstáculo, etc.). La estructura de cada capa es la de una red de topología fija de máquinas de estados finitos. Las capas mantienen una relación de inhibición sobre las capas inferiores (inhibir entradas de los sensores y acciones en los actuadores). El control no es central, sino dirigido por los datos en cada capa.

- Tareas competitivas: un agente debe decidir qué tarea debe realizar de entre varias posibles, seleccionando la que proporciona un nivel de activación mayor. Se basa en una aproximación ecológica de la resolución distribuida de problemas, simulando como por ejemplo en el sistema MANTA [Drogoul et al., 1995], en el que cada agente es una hormiga y decide qué acción debe ejecutar para cumplir sus objetivos. El problema se resuelve sin comunicación entre los individuos, estableciendo un criterio de terminación del problema. Por ejemplo, los problemas clásicos de búsqueda (misioneros y caníbales, mundo de los bloques, etc.) se interpretan como agentes (cada misionero, cada bloque, etc.) que pueden realizar movimientos y se fija una condición global de terminación.
- Redes neuronales: la capacidad de aprendizaje de las redes neuronales también ha sido propuesta en algunas arquitecturas formadas por redes que son capaces de realizar una función concreta, como, por ejemplo, evitar colisiones.

3.4.3 Híbridas

Dado que algunos investigadores opinan que para la construcción de agentes no es del todo acertado utilizar una arquitectura totalmente deliberativa, o totalmente reactiva, se han propuesto sistemas híbridos que pretenden combinar aspectos de ambos modelos.

Los más claros exponentes de estas arquitecturas son PRS [Georgeff, Lansky, 1987], TouringMachines [Ferguson, 1992] e INTERRAP [Müller, Pischel, 1994] [Müller, Pischel, 1994] [Müller, 1996]. Estas arquitecturas combinan módulos reactivos con módulos deliberativos. Los módulos reactivos se encargan de procesar los estímulos que no necesitan deliberación, mientras que los módulos deliberativos determinan qué acciones deben realizarse para satisfacer los objetivos locales y cooperativos de los agentes. Por su propia naturaleza estas arquitecturas son propicias para una estructuración por capas, que puede ser:

- Vertical, sólo una capa tiene acceso a los sensores y actuadores.
- Horizontal, todas las capas tienen acceso a los sensores y a los actuadores.

Un enfoque obvio será construir un agente con dos (o más) subsistemas: uno deliberativo, conteniendo un modelo simbólico del mundo, que desarrolla planes y toma decisiones tal y como propone la IA simbólica; y uno reactivo, que es capaz de reaccionar a sucesos que ocurren en su entorno sin comprometerse en el razonamiento complejo. Frecuentemente, al componente reactivo se le da algún tipo de precedencia sobre el deliberativo, para que dar una respuesta rápida a sucesos importantes. Este tipo de estructura conduce a la idea de una arquitectura de capas, de la que

TouringMachines [Ferguson, 1992] e INTERRAP [Müller, Pischel, 1994] [Müller *et al.*, 1994] [Müller, Pischel, 1994] [Müller, 1996] son buenos ejemplos. En las arquitecturas híbridas, los subsistemas de control del agente se organizan en una jerarquía, con capas más altas que barajan información a niveles crecientes de abstracción. Así, por ejemplo, la capa más baja podría enviar datos directamente del sensor a las salidas efectoras, mientras que la capa superior trata con metas a largo plazo. Un problema clave en estas arquitecturas es qué clase de estructura control se incrusta en los subsistemas del agente, para dirigir las interacciones entre las diversas capas.

3.4.4 Arquitecturas Intencionales: Modelo BDI

Posiblemente ha sido el modelo más difundido y el más estudiado dentro de los modelos de razonamiento de agentes de las arquitecturas deliberativas. Las nociones de complejidad y cambio tienen un gran impacto en la forma en que se construyen los sistemas computacionales y por tanto en los agentes. Los agentes y en particular los agentes BDI incorporan los componentes esenciales y necesarios para enfrentarse con el mundo real [Georgeff, Rao, 1998].

Muchas de las aplicaciones de sistemas informáticos son algorítmicas y trabajan con información exacta. Pero la mayoría de las aplicaciones del mundo real requieren sistemas más complejos, sistemas capaces de relacionarse con un entorno cambiante y con un cierto grado de incertidumbre. Los agentes y los sistemas multiagentes tienen por tanto que ser capaces de proporcionar soluciones a este tipo de problemas. El modelo BDI proporciona soluciones en entornos dinámicos, inciertos; en los que el agente o los agentes sólo tienen una visión parcial del problema (el acceso a la información está limitado) y posiblemente manejen un número limitado de recursos (recursos informáticos finitos). Las creencias, los deseos, las intenciones, y los planes son una parte fundamental del estado de ese tipo de sistemas (Fig 10) [Rao, Georgeff, 1995].

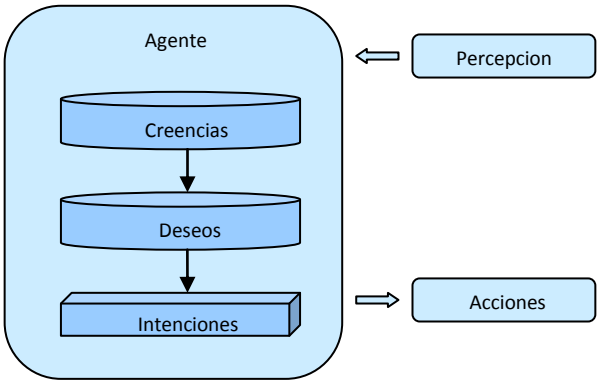

Fig 10. Agente BDI

A continuación se describen las principales aptitudes mentales de los agentes expuestas anteriormente [Rao, Georgeff, 1995]:

- Creencias: Son aptitudes cognitivas o informacionales. Es la información que el agente conoce y que se almacena del entorno, y que será actualizada a medida de que se vayan obteniendo nuevas percepciones del entorno. Estas percepciones pueden ser mediante una serie de sensores instalados en el sistema, o bien mediante la interacción con otros agentes o usuarios. Mediante la información almacenada el sistema deberá de ser capaz de comportarse de forma adecuada en un entorno cambiante.
- Deseos: Son aptitudes motivacionales. El agente debe almacenar la información relacionada con los objetivos que desea alcanzar y las prioridades de cada uno de ellos. En los sistemas tradicionales, el objetivo a alcanzar suele ser único, pero en los SMAs suelen ser varios, por lo que existe la necesidad de priorizar para determinar los que se deben cumplir con mayor prioridad puesto que alcanzarlos todos puede ser incompatible.
- Intenciones: Son aptitudes intencionales o deliberativas de los agentes. Representan las formas en que el agente actúa para cumplir los objetivos, es la parte premeditada o deliberativa. Las intenciones del sistema pueden variar a medida que cambian las creencias, ya sea por la llegada de nueva información del entorno o por el aprendizaje de actuaciones anteriores.

Los agentes BDI se modelan utilizando una estructura basada en la lógica de mundos posibles denominada árbol temporal con múltiples futuros y un solo pasado [Rao, Georgeff, 1991]. Cada nodo del árbol es una situación y las ramas del árbol pueden verse como las opciones disponibles para el agente en cada momento del tiempo. Para esta estructura de mundos posibles se definen varias relaciones de accesibilidad. Esto es, para cada situación, se definen una serie de mundos accesibles en base los deseos, las creencias y de las intenciones [Rao *et al.*, 1995]. La idea es que el agente cambie de mundo accesible por sus creencias a otro accesible por sus objetivos aplicando una serie de intenciones.

Deben existir una serie de relaciones entre las creencias, los deseos y las intenciones del agente para que éste pueda cumplir los objetivos para los que ha sido creado [Rao, Georgeff, 1995]:

- Compatibilidad entre creencias y objetivos. Desde el mundo en que se encuentra el agente debe de ser accesible otro mundo en base a las creencias en el que dicho objetivo es cierto.
- Compatibilidad entre objetivos e intenciones. Antes de adoptar una fórmula como intención, debe adoptarla previamente como deseo.
- Las intenciones conducen a acciones. No se pospone la ejecución de intenciones que son acciones primitivas.
- Relación entre creencias e intenciones. El agente conoce y cree en sus propias intenciones.

- Relación entre creencias y objetivos. El agente conoce sus objetivos o deseos.
- Finalización. Una vez iniciada la ejecución de una intención, ésta llega a su fin en algún instante de tiempo.

De este modo, las intenciones actuales del agente guían o influyen en sus decisiones sobre futuras intenciones. Dependiendo de cómo afecten las intenciones pasadas a las futuras, se identifican varios tipos de agentes [Rao, Georgeff, 1991]:

- Ciego. El agente mantiene sus intenciones hasta que sabe que las ha alcanzado. Por lo tanto, si es necesario rechazará las creencias o deseos que contradigan sus compromisos.
- Firme. El agente mantiene sus intenciones mientras crea que tiene opciones de alcanzarlas.
- Imparcial. El agente mantiene sus intenciones mientras éstas se corresponden con sus deseos, es decir, mientras el deseo o deseos que dieron lugar a esa intención no cambian.

3.4.4.1 BDI con capacidades avanzadas de razonamiento

La idea básica para implementar sistemas BDI con capacidades de razonamiento consiste en simular el aprendizaje deliberativo humano. Una de las posibilidades es utilizar un modelo de razonamiento fundamentado en el aprendizaje en base a experiencias anteriores en la resolución de problemas similares [Kolodner, 1983] [Kolodner, 1983b] [Joh, 1997]. El de razonamiento basado en casos (Case Base Reasoning (CBR)) [Laza *et al.*, 01] resuelve problemas mediante la adaptación de soluciones que fueron propuestas previamente para resolver problemas similares [Riesbeck, Schank, 1989].

Los sistemas CBR analizan y obtienen soluciones mediante algoritmos de búsqueda, recuperación, comparación y adaptación de un problema a una determinada situación. Para hacer esto, se basan en el conocimiento obtenido a partir de experiencias pasadas. Un caso se define como una experiencia pasada compuesta de: descripción del problema, la solución y el resultado obtenido [Laza, Corchado, 2001]. Los sistemas CBR se caracterizan por ejecutar ciclos de razonamiento CBR. En la figura (Fig 11), se puede ver el ciclo básico de un sistema CBR. En una primera fase se recuperan los casos similares (aquellos que tienen una descripción de problema similar al caso actual). En la fase de reutilización se adaptan las soluciones de los casos recuperados para generar a partir de ellas la solución para el caso actual. Una finalizada la fase de reutilización, se procede con la fase de revisión en la que se evalúa la solución propuesta para determinar si fue adecuada. La última fase es la de aprendizaje y almacenamiento,

en la que se almacena la nueva experiencia en la memoria de caso y en la base de conocimiento.

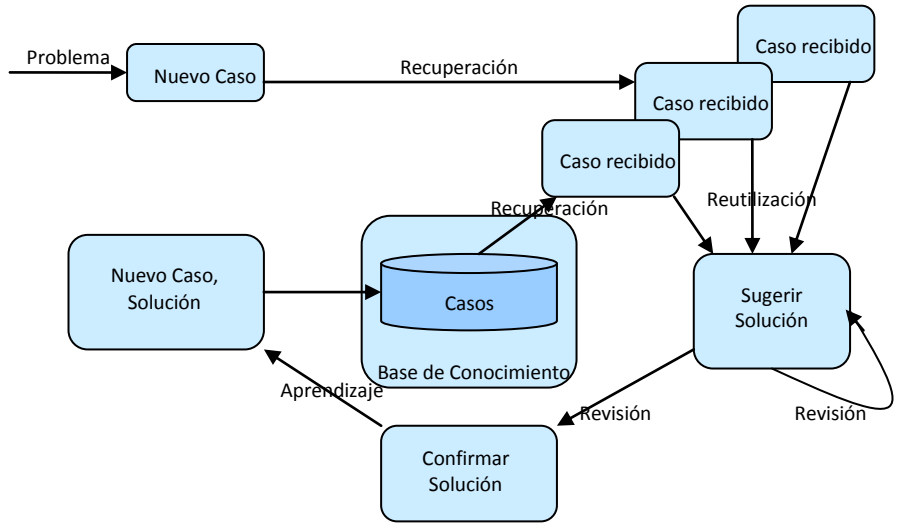

Fig 11. Ciclo CBR

El ciclo CBR va a incluir cuatro pasos que se van a repetir de forma cíclica y en la secuencia vista en la figura (Fig 11): recuperación, reutilización, revisión y aprendizaje [Kolodner, 93] [Aamodt, Plaza, 1994] [Watson, Marir, 1994]. Básicamente, el algoritmo al recibir un nuevo caso, busca en la base de casos y selecciona aquellos que son más similares al del problema actual, los casos son adaptados para generar una posible solución. Finalmente, la solución es revisada y en caso de ser óptima se añade a la base de casos.

La automatización de las capacidades de un sistema CBR ha permitido establecer la relación entre los casos de un CBR con las creencias, los deseos y las intenciones de un los agentes BDI. Como se vio en el apartado (3.1), los agentes deben de disponer de una serie de capacidades para que se considere que realmente se está trabajando con agentes. Así, por ejemplo, debe ser autoadaptable a las condiciones y autónomo. Estas capacidades se pueden conseguir integrando en el comportamiento del agente el motor de razonamiento de un CBR. Como resultado de esta integración se generan los agentes denominados CBR-BDI. Los agentes CBR-BDI [Bajo *et al.*, 2007a] [Bajo *et al.*, 2007b] [Bajo *et al.*, 2007c] [Corchado *et al.*, 2008a] [Corchado *et al.*, 2008b] tienen la capacidad de adaptarse al entorno en función de los análisis llevados a cabo anteriormente permitiendo de este modo una evolución del sistema mediante la continua integración y adaptación de experiencias en las sucesivas iteraciones el ciclo CBR llevadas a cabo.

Tal y como se puede apreciar, el ciclo de razonamiento CBR sólo establece las fases pero en no su implementación, es por ello, necesario explorar diferentes alternativas en función del problema. En el capítulo de expresión genética se han visto diferentes técnicas usadas en los análisis de expresión que se pueden incorporar en el

ciclo de razonamiento aunque quedaría por definir el modelo seguido para la recuperación, adaptación, revisión y aprendizaje que se verá en la arquitectura propuesta.

3.4.4.1.1 Formalización de un sistema CBR-BDI

La generación del modelo CBR-BDI permite integrar dentro de la arquitectura deliberativa BDI el motor de razonamiento de un CBR, para ello, es necesario establecer una relación entre la información usada por el CBR y la arquitectura BDI de modo que ambos puedan interaccionar.

La estructura de un sistema CBR se ha diseñado entorno al concepto de caso. Cada uno de los casos va a estar formado por una terna formada por la descripción del problema, la solución y el resultado obtenido [Laza, Corchado, 2001]. Esta descomposición de un CBR se puede ver en la figura (Fig 12).

CBR	**Sintaxis**
Caso: Problema, solución, resultado **Problema**: estado_inicial **Solución**: {acción, [estado_intermedio]}* **Resultado**: estado_final	{}: Secuencia []: Opcional *: 0 a n repeticiones +: 1 a n repeticiones |: disyunción BDI
BDI	
Creencias: {estado_final}|{estado acción, estado} **Intenciones**: {creencias}+ **Deseos**: {estado_final}	

Fig 12. Caso CBR vs BDI

En la figura (Fig 12) para el caso del CBR, se muestran los componentes: el problema define la situación del entorno en un momento dado, la solución es el conjunto de estados del entorno como consecuencia de las acciones que son llevadas a cabo y el resultado muestra la situación del entorno una vez que el problema ha sido resuelto.

En la figura (Fig 12) para el caso del BDI, se definen las creencias, los deseos y las intenciones de un agente. Los cambios entre estados después de llevar a cabo una acción se considera una creencia (los agentes recuerdan las acciones que son llevadas a cabo bajo una determinada situación y el resultado obtenido). Los objetivos que un agente quiere alcanzar a partir de un estado en un ambiente conocido son lo que se conoce como deseos. Las intenciones, como se dijo anteriormente, son la serie de planes que debe seguir para cumplir los objetivos, no son más que una serie creencias que han sido ordenadas. Así, en función de la información mostrada, se puede ver que existe una relación entre la descripción del caso del CBR y las creencias del modelo BDI.

3.4.4.1.2 Ciclo de vida de un sistema CBR

Si se profundiza un poco más en el modelo de razonamiento de un CBR, las etapas de razonamiento de un CBR se pueden describir de la siguiente manera:

- Recuperación: En esta fase se recuperan de la base de conocimiento, aquellos casos que son similares. Para determinar la similitud entre los casos, se utiliza una determina métrica
- Reutilización: Durante esta fase se obtiene una primera solución en base a los casos recuperados y el caso del problema. Esta solución se obtiene siguiendo la misma secuencia de acciones del pasado modificándolas para adaptarlas al problema actual. Habrá dos alternativas dependiendo de los casos, la primera sería partir de un caso similar en el que el estado de partida fuera muy similar, de esta manera, el conjunto de acciones a llevar a cabo para el nuevo problema serían las mismas que para el caso recuperado. La segunda opción, sería construir la secuencia a acciones a partir de una mezcla de todas las acciones de los casos recuperados de la fase anterior.
- Revisión: Determina si los resultados obtenidos en la fase anterior son correctos o no. Esto es, determinar si el estado final puede ser alcanzado según la secuencia de acciones que se van a tomar [Corchado *et al.*, 2000].
- Aprendizaje: En esta fase, se almacenan la secuencia de estados y las acciones tomadas en la base de casos, siempre y cuando, el resultado obtenido haya sido satisfactorio.

3.4.4.2 Planificación basada en casos

A partir del modelo CBR descrito, surgen modelos orientados a generación de planes [Bajo *et al.*, 2007a] [Bajo *et al.*, 2007b] [Bajo *et al.*, 2007c] [Bajo *et al.*, 2007e] [Corchado *et al.*, 2008b] como especialización del CBR. Un modelo de planificación aplicado a la generación de planes es similar a un CBR pero está orientado a trabajar con planes en lugar de casos. Un caso se define del siguiente modo:

- Estado inicial: conjunto de variables
- Metas: conjunto de variables
- Operadores: conjunto de acciones que se pueden aplicar a un plan

Los planes son similares a los casos pero su contenido viene definido por los siguientes campos:

- Conjunto de operaciones

- Secuencia en que son aplicadas las acciones
- Conjunto de acciones permitidas bajo diferentes estados
- Conjunto de aplicaciones que permiten cambiar entre conjunto de acciones

A partir de la definición del planificador basado en casos, se puede establecer la relación con el modelo CBR para generar el modelo de planificación híbrido que integra los BDI [Corchado *et al.*, 2008b] a modo de especialización del CBR-BDI visto anteriormente. De este modo, es posible dotar a los agentes deliberativos BDI, con mecanismos de planificación basados en casos, derivados de los sistemas CBR, diseñados específicamente para la elaboración de planes.

Este tipo de sistemas, comienzan identificando los roles y las metas de los agentes, siguiendo la pauta de los sistemas CBR en el diseño y la implementación de la arquitectura de los agentes, facilitando el aprendizaje y la adaptación, lo que conlleva a un mayor grado de autonomía del que se encuentra en la arquitectura BDI. Los agentes planificadores con el modelo BDI, al utilizar CBR como mecanismo de razonamiento, son capaces de aprender a partir de un conocimiento inicial, interactuar de forma independiente con el entorno, usuarios y otros agentes dentro del sistema, y adaptarse a las necesidades del entorno. La estructura básica de un agente deliberativo con capacidades de planificación se muestra en la siguiente figura (Fig 13). En la parte central de la figura, se tiene el mecanismo de razonamiento de un agente CBR, en la fase de recuperación se accede a la base de planes y se recuperan los planes similares, una vez recuperados se seleccionan los planes que mejor se ajustan al problema para posteriormente en la fase de reutilización adaptarlos al agente BDI con las creencias y lo objetivos y generar el nuevo plan. Finalmente, se pasa por la fase de revisión en la que se determina la adecuación del plan generado y en caso de ser una solución satisfactoria se almacena en la base de planes.

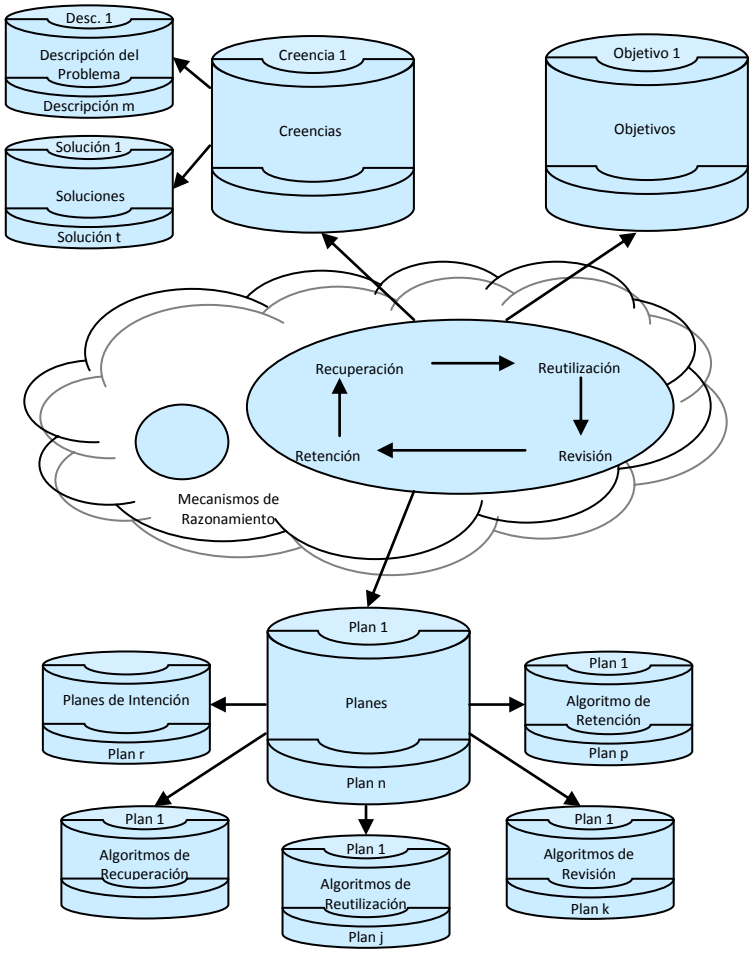

Fig 13. Agente planificador basado en casos que integra el modelo BDI

3.5 Conclusiones

El modelo de planificación basado en casos presenta ventajas con respecto a un sistema CBR-BDI ya que se centra en las planificaciones frente al modelo CBR-BDI que se centra más en la definición del problema. No obstante, en ambos se presenta el mismo problema ya que sólo establecen una arquitectura genérica que debe ser implementada pero en ninguno de los casos define como implementarla. La implementación de los diferentes algoritmos debe ser particular de cada uno de los problemas, por ello, en el capítulo de expresión genética, se introdujeron las diferentes técnicas empleadas para llevar a cabo un análisis de expresión.

Según la hipótesis definida, es necesario definir un sistema de planificación automático para la selección del modelo de análisis que mejor se ajusta a los datos. Los mecanismos de planificación basado en casos se ajustan a este proceso por lo que serán de especial interés a la hora de llevar a cabo el mecanismo de planificación usado en el sistema multiagente.

En el capítulo siguiente, se procederá a describir la arquitectura multiagente del sistema junto con el mecanismo planificación basado en casos con el modelo BDI empleado para llevar a cabo el sistema de planificación automático.

Capítulo IV

Arquitectura Propuesta

VNiVERSiDAD
D SALAMANCA

En los capítulos anteriores se han introducido las técnicas usadas para llevar a cabo análisis de expresión y las alternativas existentes en la inteligencia artificial para la creación de sistemas autoadaptables con capacidades de aprendizaje. En este capítulo se introducirá la arquitectura genérica propuesta y el modelo de planificación seguido para la generación del mecanismo de planificación automático.

Introducción

IBO (Intelligent Biomedic Organizations) es un modelo organizativo orientado a entornos biomédicos que incorpora agentes con capacidades de generación de planes para el análisis de grandes cantidades de información. El núcleo e IBO es un novedoso mecanismo para la implementación de etapas de un planificador basado en casos con el modelo BDI a través de la implementación de servicios web que proporciona una capacidad autoadaptiva en diferentes entornos. Además, IBO proporciona mecanismos de comunicación para facilitar la integración con la arquitectura SOA.

IBO se ha diseñado para modelar entornos biomédicos orientados al análisis de arrays de expresión. Para ello, IBO define una serie de agentes y servicios adecuados a estas necesidades. Los agentes actúan como coordinadores y controladores de los servicios, mientras que los servicios son los responsables de llevar a cabo el procesamiento de la información facilitando características como la replicación y la modularidad. Hoy en día, existen diversas plataformas que facilitan el desarrollo de agentes y permiten su ejecución en numerosos dispositivos. Los tipos de agentes se distribuyen en capas dentro de IBO, de acuerdo a sus funcionalidades, proporcionando así una estructura organizativa que incluye análisis de información y gestión de la propia organización, de modo que es posible añadir y eliminar agentes al sistema de forma sencilla.

La capa de agentes constituye el núcleo de IBO y define la organización virtual para el análisis de datos masivos, tal y como se puede ver en la figura (Fig 14). La figura muestra 4 tipos de agentes.

- Organización: Los agentes de la organización se ejecutan en los dispositivos de los usuarios o servidores. Los agentes instalados en los dispositivos de los usuarios harán de puente entre los dispositivos y los agentes del sistema

que realizan el análisis de los datos. Los agentes instalados en los servidores serán los encargados de realizar el análisis de la información según el modelo de planificación basado en casos con el modelo BDI [Glez-Bedia, Corchado, 2002]. Los agentes de la capa de organización se deben configurar inicialmente para los diferentes tipos de análisis que se quieren llevar a cabo. Estos análisis varían en función de la información disponible y los resultados buscados, por lo que es preciso establecer una configuración previa del flujo de la capa de análisis.

- Análisis: Los agentes de la capa de análisis son responsables de seleccionar la configuración y el flujo de servicios que mejor se ajustan a las necesidades del problema. Se comunican con los servicios web para generar los resultados. Los agentes siguen el modelo de planificación basado en casos bajo el modelo BDI [Glez-Bedia, Corchado, 2002]. La selección del los flujos y configuración de los servicios a utilizar, se realiza mediante la generación de redes bayesinas y grafos utilizando información correspondiente a los planes ejecutados anteriormente. Los agentes de esta capa son altamente adaptables al caso de estudio al que se ha aplicado IBO. Concretamente, para el caso de estudio de los microarrays se incluyen los agentes necesarios para llevar a cabo el análisis de expresión que se pueden ver en la figura (Fig 14).
- Representación: Los agentes de esta capa son los encargados de generar las tablas y los gráficos con los resultados.
- Importadores/Exportadores: Estos agentes son los encargados de formatear los datos para ajustarlos a las necesidades de los agentes.
- El agente controlador controla los agentes que están disponibles en las diferentes capas del sistema multiagente. Esto permite el registro de agentes en las diferentes capas y su uso en la organización.

Por otro lado, la capa de servicios se ha dividido en dos grupos según la funcionalidad, al igual que ocurría con la capa de agentes.

- Servicios de análisis. Los servicios de análisis se usan por los agentes de análisis para llevar a cabo diferentes tareas. Dentro de los servicios de la capa de análisis se disponen de servicios para realizar el preprocesado, filtrado, clustering y extracción del conocimiento, estos servicios se usan en la capa de análisis para llevar a cabo el análisis de los microarrays.
- Servicios de representación generan gráficos y tablas de resultados.

Dentro de la capa de servicios, se tiene un servicio Directory Facilitator que proporciona la información sobre los diferentes servicios disponibles y gestiona el fichero XML de UDDI (Universal Discovery Description and Integration). Para facilitar comunicación entre agentes y servicios la arquitectura integra una capa de comunicación que proporciona compatibilidad FIPA-ACL y SOAP (Simple Object Access Protocol) [Bauer, Huget, 2003].

Los agentes de las diferentes capas, se intercomunican entre sí para generar el plan final de análisis de los datos. Así en la figura (Fig 14), se muestran los diferentes agentes del sistema distribuidos por capas y las conexiones que puede realizar un agente con los diferentes agentes y servicios del sistema. Así, se puede ver que el agente Diagnosis de la capa de organización, para llevar a cabo su cometido, selecciona agentes de la capa de análisis en una determinada secuencia. A su vez los agentes de la capa de análisis, seleccionan los servicios necesarios para llevar a cabo el estudio de los datos. Así, por ejemplo, el agente de filtrado de la capa de análisis selecciona dentro de los servicios de la categoría de filtrado los servicios y flujo que son adecuados para los datos. Los demás agentes de la capa de análisis siguen el mismo comportamiento que el agente de filtrado.

Los agentes de la capa de organización y análisis heredan el son agentes con planificación basado en casos bajo el modelo BDI por lo que todos ellos, disponen de la capacidad de generar planes de modo automático en base a los planes previos existentes en el sistema. Cada uno de los agentes maneja su propia memoria de planes en la que almacena experiencias pasadas relacionadas con la tarea asociada al agente. La memoria de planes es actualizada cada vez que una tarea global se lleva a cabo.

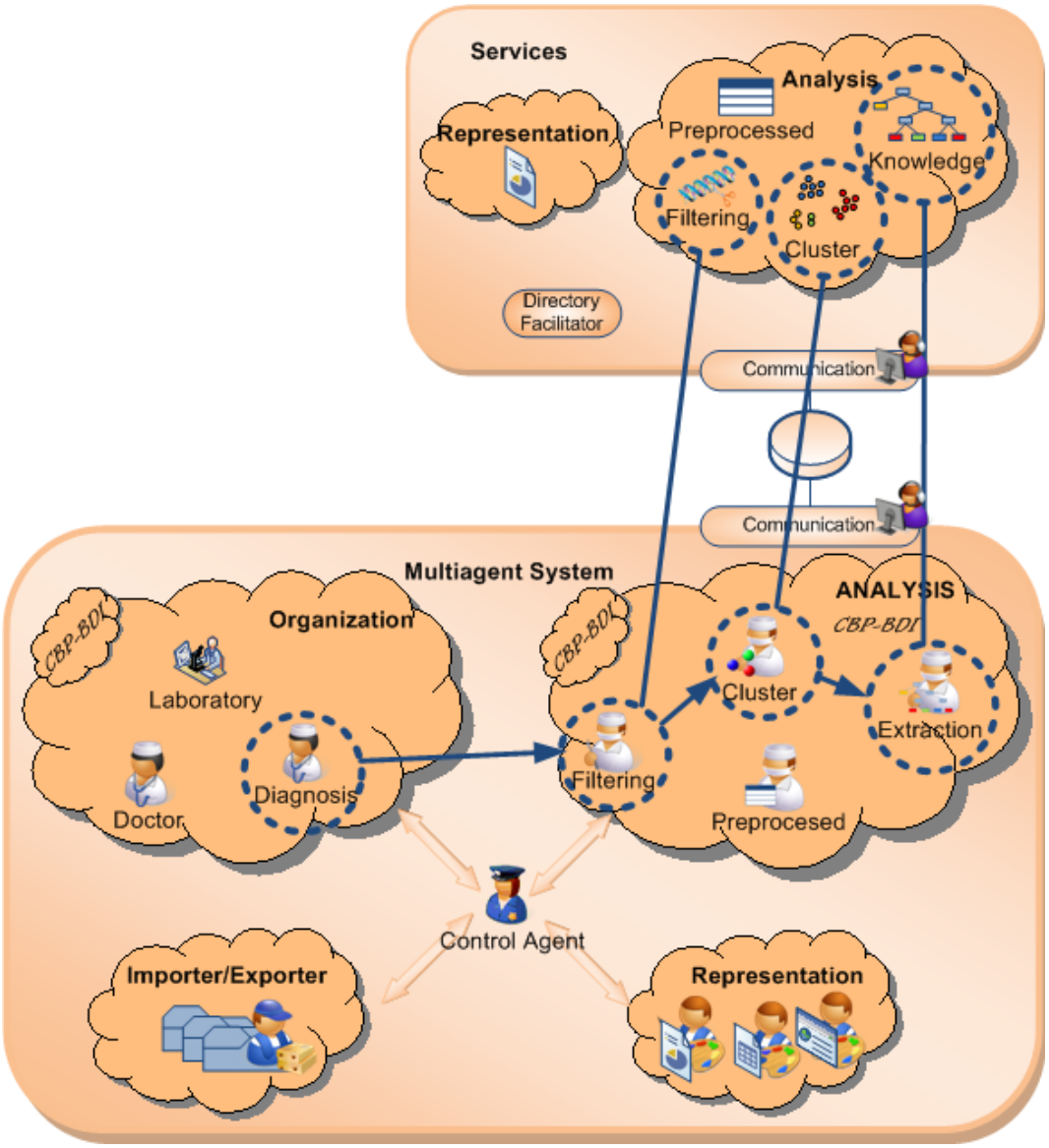

Fig 14. Arquitectura de IBO

4.1 Metodologías para el modelado de Organizaciones IBO

Uno de los principales problemas en el desarrollo de una arquitectura basada en sistemas multiagentes es que actualmente no existente un estándar claro o

metodologías desarrolladas para definir los pasos de análisis y diseño que son necesarias llevar a cabo para el desarrollo de los sistemas. Hay presentes un número de metodologías: Gaia [Wooldridge *et al.*, 2000], INGENIAS [Pavón *et al.*, 2007a] [Pavón *et al.*, 2007b], TROPOS [Giorginia *et al.*, 2005], MESSAGE [EURESCOM, 2001]. La opción elegida fue usar la metodología Gaia en combinación con el lenguaje de modelado AUML (Agent UML) AUML [Corchado, Laza, 2003] [Odell *et al.*, 2004] y de este modo tener las ventajas de ambos [Bajo *et al.*, 2009].

IBO proporciona un mecanismo de modelado en el que se lleva a cabo el análisis y diseño de la arquitectura mediante el uso de Gaia y AUML. Siguiendo la metodología de Gaia [Wooldridge *et al.*, 2000], en la fase de análisis se describe los diferentes roles que existen en el sistema. En la figura (Fig 15), se muestra el modelo de roles del rol Planner. Un rol puede ser visto como una descripción abstracta de una entidad, que va a tener una serie de responsabilidades y una serie de permisos. Las responsabilidades representan la funcionalidad de un determinado rol. Se pueden dividir en dos categorías: las responsabilidades liveness y safety. Liveness representan las responsabilidades que cada agente tiene, estas responsabilidades se definen mediante un flojo de actividades y protocolos. En este caso la responsabilidad del agente es crear un plan. Los permisos están asociados con los recursos disponibles y las limitaciones de acceso. Acciones son las actividades que el agente lleva a cabo.

Name		Create Plan
Description		Genera un plan a partir de la información de los planes existentes
Protocols Activities		InformPlan CalculateGraph CalculateBayesianNetwork CalculateWeight CalculatePlan
Permissions		
Responsibilities	Liveness	CreatePlan: RequestEfficientPlans · RequestInefficientPlans · CalculateGraph · CalculateBayesianNetwork · CalculateWeight · CalculatePlan
	Safety	Neither void plan

Fig 15. Modelo de Roles de Planner

La información recogida por los modelos de roles se recoge en el modelo de agentes Gaia mostrado en la Figura (Fig 16). Esta figura muestra los diferentes roles que pueden llevar a cabo el agente Diagnosis mostrado en el centro de la imagen. Los tres nodos hojas representan los diferentes roles: CreatePlan, Create WorkFlow, Update Plan.

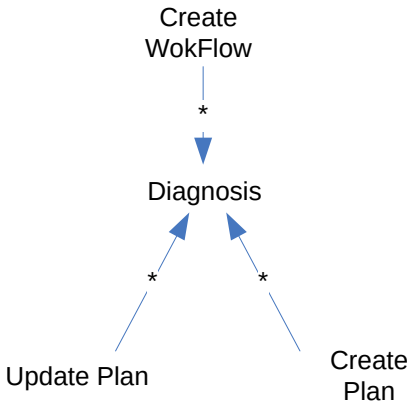

Fig 16. Modelo de agentes del agente Diagnosis

El análisis y diseño de alto nivel mediante la metodología Gaia permite modelar el sistema en términos de organización, pero proporciona un nivel de detalle demasiado bajo. Por ello, es necesario utilizar el lenguaje de modelado AUML para obtener un diseño detallado. De esta forma, para cada tipo de agente se obtienen sus capacidades y servicios. En la Figura (Fig 17), se muestra un ejemplo de diagrama de clases de AUML para el agente diagnóstico. En dicho diagrama, se pueden observar los diferentes servicios y capacidades que proporciona el agente. Los servicios se corresponden con funcionalidades que ofrece el agente a otros agentes o usuarios, mientras que las capacidades muestran las funcionalidades que el agente puede desempeñar. Como se puede apreciar en la Figura (Fig 17), el agente diagnóstico es un agente con capacidad de planificación basado en el modelo BDI por lo que dispone de capacidades de planificación de modo automático, dispone de las capacidades de planificación y replanificación a partir de la información de planes almacenados en la base de casos y la información proporcionada por el nuevo plan a llevar a cabo.

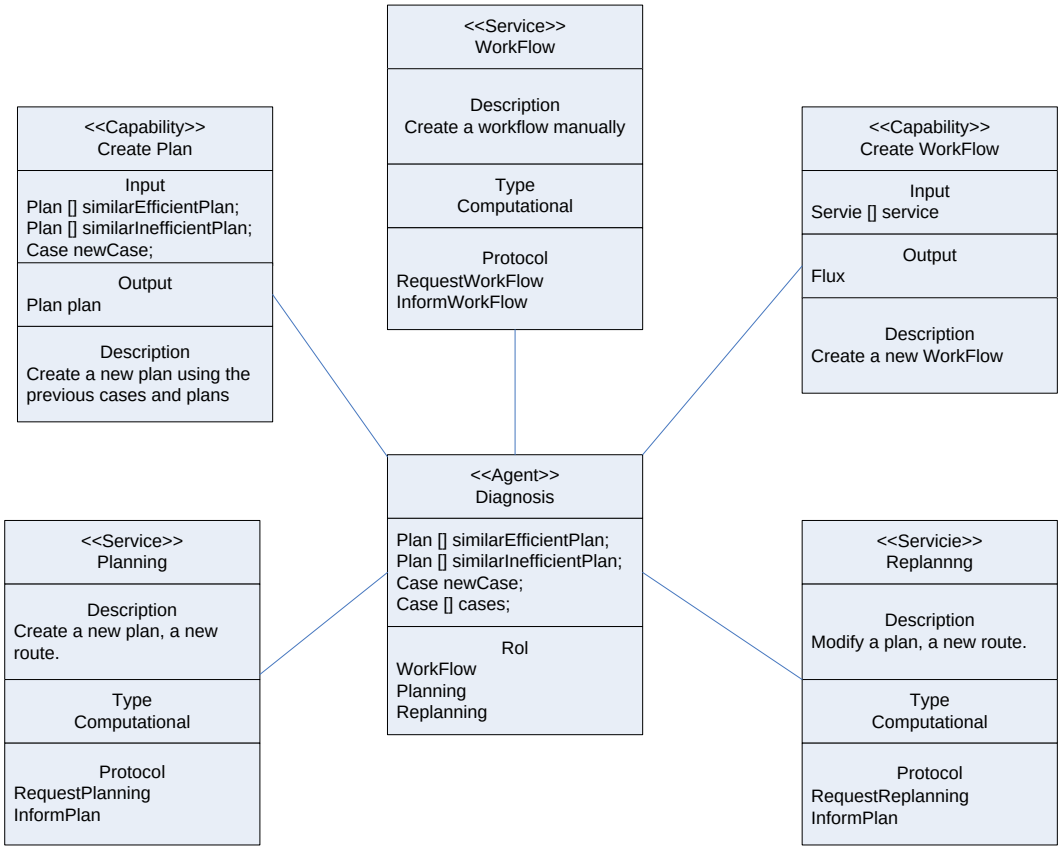

Fig 17. Diagrama de clases AUML del agente Diagnosis

A continuación, se presentan en detalle los 2 elementos más innovadores en IBO, como son un tipo de agente con capacidades de planificación basado en el modelo BDI especializado en tareas de coordinación y un mecanismo de distribución dinámica de servicios.

4.1.1 Agente coordinador con capacidades de planificación

Los agentes en la capa de organización y en la capa de agentes tienen la capacidad de aprender de los análisis llevados a cabo en etapas anteriores. Para ello, los agentes adoptan el modelo de planificación basado en casos que es una especialización del modelo de razonamiento basado en casos (CBR) [Kolodner, 1993]

El agente planifica en base a los planes llevados a cabo anteriormente (Glez-Bedia and Corchado 2002). En los planificadores basados en casos, las soluciones vienen

dadas por planes, los planes se generan teniendo en cuenta los planes anteriores aplicados a problemas similares y se almacenan finalmente en una memoria para problemas posteriores.

Un plan P es una tupla *<S,B,O,L>*:

- S es un conjunto de acciones
- O es una relación de orden sobre S que permiten establecer un orden entre las acciones en un plan
- B es un conjunto que permite describir conexiones permitidas y prohibidas entre las acciones de un plan
- L conjunto de enlaces que permite cambiar de un conjunto de acciones S a otro S'

El razonamiento basado en casos (CBR) es un tipo de razonamiento basado en experiencias pasadas [Kolodner, 1993]. El término principal cuando se trabaja con CBR es el concepto de caso. Un caso se puede definir como una experiencia pasada que está compuesta por tres elementos: la descripción del programa, que describe la situación inicial del problema; una solución, que proporciona una secuencia de acciones que se pueden llevar a cabo con el objetivo de resolver un problema; un estado final, que describe los estados alcanzados una vez que la solución es aplicada. Un CBR maneja casos (experiencias pasadas) para resolver nuevos problemas. Los casos son manejados mediante un ciclo de razonamiento que se compone de cuatro fases secuenciales: recuperación, reutilización, revisión y aprendizaje. Cuando llega al sistema una nueva descripción de un problema se inicia la fase de recuperación que procede mediante algoritmos de similitud para recuperar de la memoria de casos cuya descripción del problema es más similar a la actual. Una vez que los casos más similares son recuperados se adaptan y se genera una nueva solución, normalmente la nueva solución se lleva a cabo y se revisa por un experto de modo que si el resultado fue satisfactorio la nueva solución se almacena en la memoria de casos para futuras planificaciones.

Tal y como se ha dicho, un modelo de planificación basado en casos es un CBR especialmente diseñado para la generación de planes (secuencia de acciones) [Corchado *et al.*, 2008a] [Corchado *et al.*, 2008b], las soluciones a los problemas vienen dadas por planes. Las soluciones se generan a partir de planes aplicados a problemas similares de igual modo que se hacía en los CBR. En este caso, la descripción del problema (estado inicial) y la solución (situación cuando se alcanza un estado final) son representadas por medio de creencias, el estado final alcanzado y la secuencia de acciones se representan como planes.

Los agentes con capacidades de planificación basados en el modelo BDI parten del modelo BDI y establecen una correspondencia entre los elementos del modelo BDI y de los sistemas de planificación basado en casos. El modelo BDI se ajusta a los requisitos del sistema puesto que permite definir una serie de metas a alcanzar a partir del conocimiento registrado acerca del mundo. La fusión de los agentes con capacidades de planificación junto con el modelo BDI, generan agentes con capacidades de planificación

basados en el modelo BDI que permiten formalizar tanto la información disponible, la definición de las metas y acciones disponibles para resolver el problema así como el procedimiento para resolver nuevos problemas mediante la adopción del ciclo de razonamiento.

Los agentes con la arquitectura BDI tienen sus orígenes en el razonamiento filosófico. Los agentes son capaces de decidir en cada momento la acción a ejecutar en función a sus objetivos. El proceso de razonamiento se descompone en dos fases, en una inicial, se establecen las metas y en una segunda se define cómo alcanzarlas. Una representación basada en acciones requiere una arquitectura que permita adquirir conocimiento del entorno. En esta sección se explica el procedimiento seguido para adquirir este conocimiento a través del modelo BDI [Kolodner, 1983] [Kolodner, 1983b] [Joh, 1997] [Corchado *et al.*, 2008b].

El entorno M y los cambios que se producen dentro de él se representan por medio de un conjunto de variables.

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

Las creencias son vectores de valores correspondientes a un subconjunto o la totalidad de las variables del entorno.

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

Un estado del mundo $e_j \in E$ se representa en el agente como un conjunto de creencias que son ciertas en un determinando instante de tiempo t.

Sea $E = \{e_j\}_{j \in N}$ el conjunto de estados del mundo, fijando el valor de t el estado j viene definido por

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

Los deseos son establecidos al inicio de la aplicación. Son aplicaciones entre un estado del mundo y el estado que se quiere alcanzar.

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

Las intenciones es el modo en que el conocimiento de los agentes es manejado para alcanzar los objetivos. Un deseo es alcanzable si existe una aplicación i definida a través de n creencias de modo que

$$i : \underset{(b_1, b_2, \cdots, b_n, e_0)}{BxBx \overset{n)}{\cdots} xBxE} \rightarrow \underset{e^*}{E} \quad (5)$$

En nuestro modelo, las intenciones garantizan que existe suficiente conocimiento en la base de creencias de modo que un deseo se puede alcanzar mediante un plan de acciones.

Las acciones se definen como los mecanismos que provocan cambios en el mundo generando cambios de estado.

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

Un plan de un agente es una secuencia de acciones que van desde el estado actual e_0, define la ruta de estados a través de las cuales el agente pasa para alcanzar otro estado del mundo.

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

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

Partiendo de esta representación, los agentes con capacidades de planificación basados en el modelo BDI en IBO hacen coincidir el estado inicial de un caso con las creencias iniciales, el estado final de un caso con las metas del agente y las intenciones con las acciones que se pueden llevar a cabo para crean planes que permitan alcanzar el estado final. En IBO, las acciones a realizar son servicios, de tal forma que un plan es una secuencia ordenada de servicios. Es necesario facilitar la inclusión de nuevos servicios y el descubrimiento de nuevos planes en base a los planes existentes. En IBO, los servicios se corresponden con las acciones que se pueden llevar a cabo y que determinan cambios en los datos de entrada del problema. Cada uno de los servicios describe un nodo de un grafo de modo que cada plan viene representado por un camino en el grafo. La presencia de un arco que llega a un determinado nodo implica la ejecución del servicio asociado al nodo de llegada. Por tanto, la secuencia de nodos representa la secuencia de acciones/servicios y el orden en el que se llevan a cabo, de modo que cada plan se puede representar como una ruta en un grafo. Cada uno de los nodos del grafo lleva asociado un conjunto de variables con sus correspondientes valores, formando el conjunto de creencias que describe a cada uno de los estados del grafo. Adicionalmente, a los nodos correspondientes a los servicios, se tienen un par de nodos ficticios correspondientes a un nodo de origen y a otro de fin. Los nodos de origen y fin son necesarios para poder establecer el servicio inicial de un plan y para poder establecer el punto final de un determinado plan. En la figura (Fig 18), se muestra plan en grafo de los servicios. Tal y como se puede ver, el grafo sólo dispone de un camino y contiene nodos no conectados, el camino define la secuencia de servicios desde el nodo de inicio hasta el nodo fin. El plan descrito por el grafo viene definido por la siguiente secuencia $(s_7 \circ s_5 \circ s_3 \circ s_1)(e_0)$. e_0 representa el estado de origen que se corresponde con Init. El estado init representa la descripción inicial del problema, el e_0, mientras que Final representa el estado final e^*.

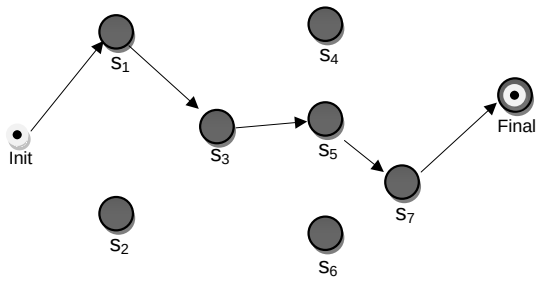

Fig 18. Ruta del plan en el grafo inconexo.

De este modo, un agente con capacidades de planificación basado en el modelo BDI trabaja con planes que contienen información asociada con las acciones que deberían llevar a cabo, es decir, cada agente de la capa de análisis define su propia memoria de planes que contiene la información que necesita. La información requerida por cada uno de los agentes en la capa de análisis depende de su funcionalidad por lo que cada agente tiene la responsabilidad de definir su propia memoria de planes. Determinados servicios requieren ejecutar acciones tales como la composición de servicios, mientras que otros agentes sólo necesitan seleccionar el servicio que mejor se ajusta a sus necesidades sin tener que llevar a cabo ninguna composición. La tabla (Tabla 2) proporciona un ejemplo de la descripción de la estructura de casos para el filtrado de un agente. Como se muestra, el agente de filtrado tiene en cuenta el número de casos, variables, tipo de optimización buscada y la descripción del problema.

Variable	tipo
numeroCasos	Integer
numeroVariables	Integer
optimización	Integer
calidad	Real
problema	Integer

Tabla 2. Caso agente filtrado

Adicionalmente, la información de los planes se define por la secuencia de acciones/servicios en la que son aplicados. La tabla (Tabla 3) representa la información contenida en los planes de filtrado, tal y como se puede ver es una lista de servicios. Finalmente, se define la estructura que contendrá la información para cada uno de los servicios ejecutados. La tabla (Tabla 4) muestra la estructura de información definida para el agente de filtrado, la estructura contiene el número de variables, nombre de los servicios y la lista de parámetros a usar por los servicios.

Variable	Tipo
servicio	ArrayList of Services

Tabla 3. Servicios del caso agente filtrado

Variable	Tipo
numberesVariablesFinal	Int
servicios	String
parametros	ArrayList of parameter

Tabla 4. Servicios

Los agentes con capacidades de planificación basados en el modelo BDI usan la información contenida en los casos para mejorar los análisis de los datos llevados a cabo. Tal y como se ha explicado, un análisis de los datos consiste en la construcción de un grafo que determina la secuencia en que los servicios son llevados a cabo. En la figura (Fig 19) se detalla el proceso de construcción del grafo, el proceso se divide en una serie de etapas que se explican detalladamente en las siguientes secciones para el caso del agente de filtrado.

1. Extraer el conjunto de casos similares al caso actual con mejor y peor rendimiento.
2. Generar el grafo dirigido con la información de los diferentes planes
3. Generar el clasificador TAN para los casos con mejor rendimiento y peor rendimiento respectivamente mediante el Friedman-Goldsmidtz [Friedman *et al.*, 1996]. Calcular las probabilidades de ejecución de cada servicio en función del clasificador generado anteriormente
4. Ponderar las conexiones del grafo original en función de una métrica
5. Construcción del grafo

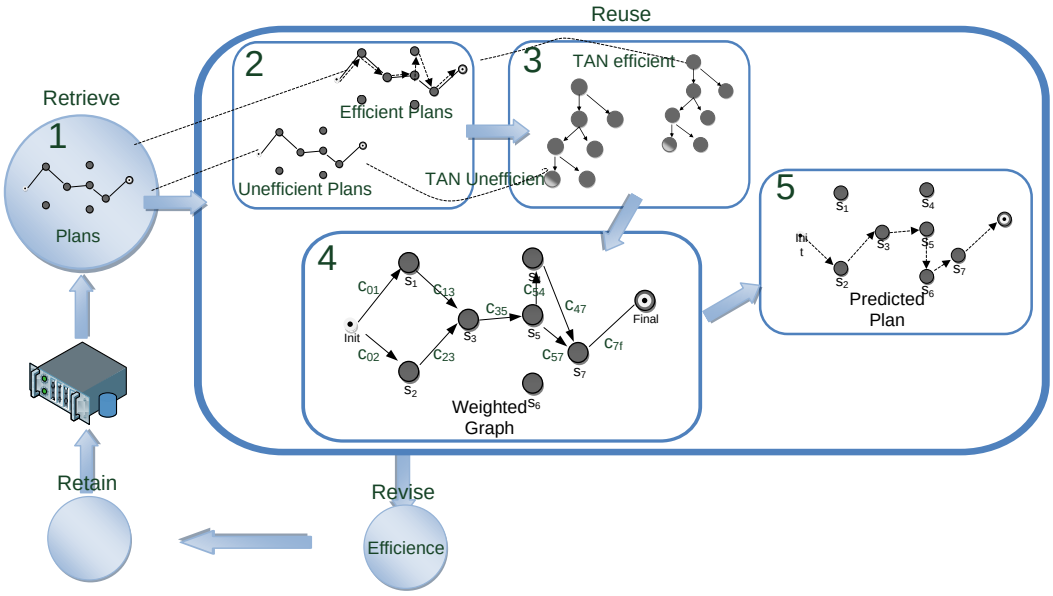

Fig 19. Composición de grafos

4.1.1.1 Casos similares

En la fase de recuperación, se seleccionan los planes de mayor eficiencia y menor eficiencia que se han aplicado. Los microarrays se componen de sondas que representan variables que marcan el nivel de significación de determinados genes, el conjunto de estas variables definen el estado inicial e_0 según se muestra en la ecuación (3).

La recuperación de los planes se realiza de dos formas diferentes en función del caso de estudio. Para recuperar los casos se tiene en cuenta si se ha realizado algún análisis previo para un caso de estudio de las mismas características. En caso de tener algún estudio previo, se seleccionan los planes correspondientes al mismo caso de estudio.

En caso de no disponer planes correspondientes al mismo caso de estudio o en caso de que el número de planes sea insuficiente, se recuperan los planes correspondientes al caso de estudio más similar. La selección del caso de estudio más similar se realiza en base a la distancia del coseno aplicada sobre el siguiente conjunto de variables:

1. Número de sondas
2. Número de casos
3. Coeficiente de variación Pearson [Kuo *et al.*, 2003] de los e_0.

El número de casos eficientes e ineficientes seleccionados se fija de antemano.

4.1.1.2 Construcción del grafo dirigido

Los diferentes planes se representan en grafos como el mostrado en la figura (Fig 18). Los planes representados en forma de grafos se unen para generar un grafo dirigido que permite definir los nuevos planes en base a la minimización de una determinada métrica. Así, dados los grafos mostrados en la figura (Fig 20), se genera un nuevo grafo que fusiona la información correspondiente a ambos grafos.

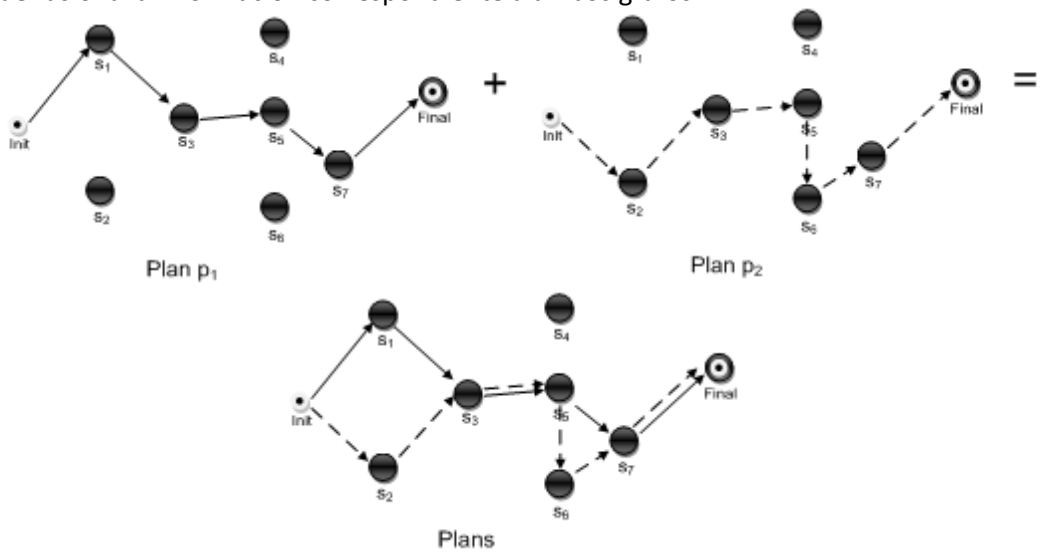

Fig 20. Composición de grafos.

Las dobles conexiones de los nodos se han indicado únicamente para representar la existencia de una conexión en los dos grafos, pero en realidad no es necesario representar más de una conexión por arco. Cada uno de los arcos del grafo de planes lleva asociado un peso y, en función del peso, se calcula la nueva ruta a ejecutar que define el nuevo plan obtenido a partir de los planes recuperados. El valor de los pesos se estima a partir de la eficiencia de los planes y los planes recuperados tal y como se indica posteriormente en el apartado (4.1.1.5).

La generación de los nuevos planes se lleva a cabo mediante la construcción del grafo de planes mostrado en la figura (Fig 20). Los pesos se estiman a partir de los planes existentes mediante la aplicación de una red bayesiana. Los datos de entrada a la red bayesiana se descomponen en los siguientes elementos:

- Planes con alta eficiencia. Se asignan a la clase 1.

- Planes de baja eficiencia. Se asignan a la clase 0.

A partir de los planes recuperados se calcula la red bayesiana para cada una de las clases. La red bayesiana se construye siguiendo el algoritmo de Friedman-Goldsmidtz [Friedman *et al.*, 1996]. Al disponer de dos clases diferentes, se generarán por tanto dos redes bayesianas, una para cada una de las clases.

4.1.1.3 Clasificador TAN

El clasificador TAN se construye a partir de los planes recuperados más similares al plan actual, haciendo una separación entre planes eficientes y no eficientes en la generación del modelo. Así, al aplicar el algoritmo de Friedman-Goldsmidtz [Friedman *et al.*, 1996]. Las dos clases tenidas en cuenta son: eficiente y no eficiente. El algoritmo de Friedman-Goldszmidt permite calcular una red bayesiana a partir de las relaciones de dependencia establecidas mediante una métrica. La métrica tiene en cuenta las relaciones de dependencia entre dos variables en función de la variable clasificadora. En este caso, la variable clasificada es la eficiencia y las demás variables indican si un servicio está o no disponible. La métrica definida por Friedman se define del siguiente modo.

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

Los valores de las probabilidades se estiman en función de las frecuencias de los datos. EL algoritmo seguido se descompone en las siguientes etapas:

- Calcular el valor de $I(X,Y \mid C)$ para las diferentes variables/atributos X, Y que se encuentren interconectadas en el grafo original, la clase C varía entre casos similares eficientes y no eficientes.
- Construir el grafo no dirigido completo
  - Establecer como nodos los diferentes atributos/variables.
  - Establecer en las conexiones como pesos los valores obtenidos en el primer paso. Para los arcos que no se disponga de conexiones establecer el valor 0.
- Crear el árbol maximal a partir del algoritmo de Kruskal [Campos, Ricardo, 2008].
- Transformar el árbol no dirigido en un dirigido. La conexión inicial y la selección del siguiente nodo a conectar marcan el sentido de las conexiones.
- Finalmente, se construye el modelo TAN, para ello, se añade un nodo que representa a la clase C y un arco que conecta a C con cada uno de los atributos.

4.1.1.4 Probabilidades de los servicios

Una vez calculado el modelo TAN para cada una de las clases, se procede a calcular la probabilidad de ejecución de cada uno de los servicios. Estas probabilidades influirán en el valor final de los pesos asignados a los arcos del grafo. Las probabilidades se calculan a partir del modelo TAN.

En la figura (Fig 21) se tiene un modelo de un clasificador para una red multiconectada. El cálculo de las probabilidades para un árbol se realiza de modo similar al mostrado a continuación.

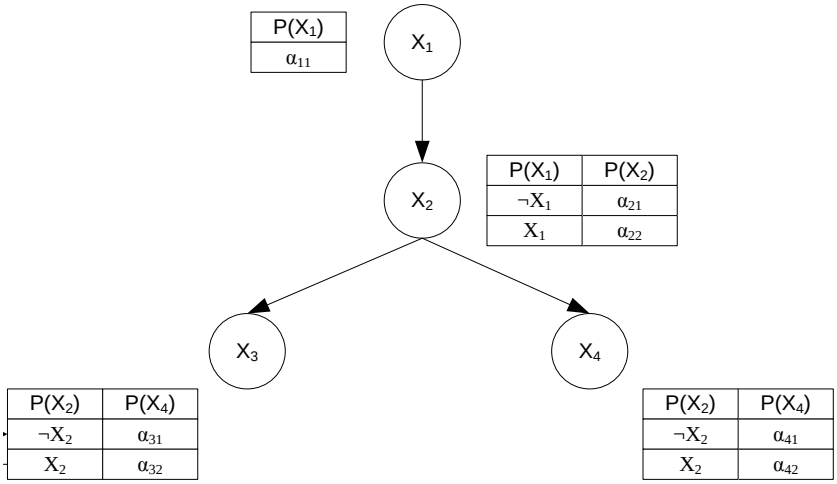

Fig 21. Red Bayesiana

Suponiendo que el conjunto de variables aleatorias se puede definir de la siguiente manera $U = \{X_1, X_2, ..., X_n\}$, se parte de la suposición de que las variables son independientes entre sí. Una red bayesiana para U se define como una tupla formada por dos elementos $B = < G, T >$ donde G representa el grafo dirigido acíclico en el que los nodos son las variables y las conexiones entre los nodos de T contiene las probabilidades de conexión entre las variables y sus posibles valores, las probabilidades se representan por $P_B(x_i \mid \pi_{x_i})$ donde x_i es un valor de la variables X_i y $\pi_{x_i} \in \Pi_{X_i}$ donde π_{x_i} representa a uno de los padres del nodo X_i. Así, una red bayesiana B, define una distribución de probabilidad conjunta única sobre U dada por

$$P_B(X_1, X_2, ..., X_n) = P_B(X_n \mid X_{n-1}, ..., X_1) \cdot P(X_{n-1}, ..., X_1) =$$

$$\prod_{i=1}^{n} P_B(X_n \mid X_{n-1}, ..., X_1) = \prod_{i=1}^{n} P_B(X_i \mid \Pi_{X_i})$$

Como se puede ver, $\Pi_{X_i} \equiv X_{n-1}, ..., X_1$. Posteriormente, es necesario ordenar las variables en relación con la aparición en el grafo dirigido acíclico. Así, en la figura (Fig

21) y dadas las tablas de probabilidad indicadas, la probabilidad de que ocurra $X_4, \neg X_3, X_2, X_1$ viene dada por

$$P(X_4, \neg X_3, X_2, X_1) =$$
$$P(X_4 \mid \neg X_3, X_2, X_1) \cdot P(\neg X_3, X_2, X_1) =$$
$$P(X_4 \mid \neg X_3, X_2, X_1) \cdot P(\neg X_3 \mid X_2, X_1) \cdot P(X_2, X_1) =$$
$$P(X_4 \mid \neg X_3, X_2, X_1) \cdot P(\neg X_3 \mid X_2, X_1) \cdot P(X_2 \mid X_1) \cdot P(X_1)$$

Teniendo en cuenta las relaciones de independencia entre las variables se puede simplificar las anteriores probabilidades de la siguiente manera:

$$P(X_4 \mid \neg X_3, X_2, X_1) = P(X_4 \mid X_2)$$

La probabilidad de que ocurra el suceso X_4 es independiente de que ocurra los sucesos X_3, X_1 por tanto, se puede reducir a lo que haya ocurrido en el único suceso que le precede en el árbol, que es X_2. De este modo, el conjunto de padres del nodo X_4 viene representado por $\Pi_{X_4} = \{X_2\}$. De igual modo, simplificando las expresiones anteriores, se tiene que:

$$P(X_4, \neg X_3, X_2, X_1) = P(X_4 \mid X_2) \cdot P(\neg X_3 \mid X_2) \cdot P(X_2 \mid X_1) \cdot P(X_1)$$

Sustituyendo en la expresión anterior por los valores reales se tiene lo siguiente:

$$P(X_4, \neg X_3, X_2, X_1) = \alpha_{42} \cdot (1 - \alpha_{31}) \cdot \alpha_{21} \cdot \alpha_{11}$$

4.1.1.5 Ponderación de las conexiones

A partir del modelo TAN, se define la probabilidad de que se haya ejecutado un determinado servicio para las clases 1 y 0. Esta probabilidad se usa para determinar el valor final del peso en función de la calidad de los planes recuperados. Suponiendo que la probabilidad de haber ejecutado el servicio i para la clase c viene definido de la siguiente manera $P(i, c)$, el peso de los arcos se define siguiendo la siguiente fórmula. La función se ha definido de modo que los planes con mayor calidad poseen valores positivos más cercanos a cero.

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

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

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

Donde:

- $I(i, j \mid 1)$ la probabilidad de ejecutarse el servicio i para la clase 1
- $P(j,1)$ la probabilidad de que se ejecute el servicio j para la clase 1. El valor se obtienen a partir de la red bayesiana definida en el paso anterior.
- $I(i, j \mid 0)$ la probabilidad de ejecutarse el servicio i para la clase 0
- $P(j,0)$ la probabilidad de que se ejecute el servicio j para la clase 0. El valor se obtiene a partir de la red bayesiana definida en el paso anterior.
- G_{ij}^{1} es el conjunto de planes que poseen un arco con origen en j y destino en i para la clase 1.
- G^{s} es el conjunto de planes de clase s
- $q(p)$ es la calidad del plan p. q también puede definir el tiempo de ejecución del plan. El significado depende de la medida a optimizar en el plan inicial.
- $\#G_{ij}^{1}$ cardinal del conjunto
- C_{ij} es el peso de la conexión entre el nodo origen j y el nodo destino i

4.1.1.6 Construcción del grafo

Una vez construido el grafo de planes, se procede a calcular la ruta mínima que lleva desde el nodo origen al nodo destino. Para el cálculo de la ruta más corta/larga, se aplica el algoritmo de Dijkstra ya que existen implementaciones de orden n*log n. Para poder aplicar dicho algoritmo es necesario sumar a cada uno de los arcos el valor absoluto del arco con mayor valor absoluto negativo para eliminar así los arcos con valores negativas del grafo. La ruta define el nuevo plan a ejecutar y depende de la medida a maximizar o minimizar.

4.1.2 Distribución de servicios en IBO

Los servicios de IBO se descomponen en categorías en base a funcionalidad que llevan a cabo. Los servicios se descomponen de modo similar a los agentes de la capa de análisis, de manera que cada agente de la capa de análisis conoce los servicios disponibles para llevar a cabo las diferentes tareas.

El conocimiento de los servicios disponibles es útil para llevar a cabo la selección manual del flujo de análisis de los datos. Para la planificación automática no es necesario disponer de dicha información, puesto que la información ya se encuentra disponible, asociada a los diferentes casos almacenados en el sistema. En el caso de estudio de microarrays, los servicios se descomponen en las categorías necesarias para llevar a cabo el análisis de expresión: preprocesado, filtrado, cluster-clasificación, extracción del conocimiento. Los diferentes servicios se explicarán posteriormente en el caso de estudio, puesto que su implementación depende de modo directo del caso de estudio que se está tratando.

4.2 Estructura de los agentes

En los apartados anteriores, se ha descrito de modo global el funcionamiento del sistema sin centrarse en ningún agente en concreto, se ha descrito el mecanismo de planificación usado y cómo es usado para llevar a cabo la planificación final. Sin embargo, no todos los agentes del sistema requieren del uso de un planificador de estas características o incluso parte de su funcionalidad se encuentra predefinida de antemano en base a las características de los problemas. Cada caso de estudio tiene sus peculiaridades que se definen en base a los resultados que se quieren adquirir por tanto, la configuración global para la obtención de los resultados es necesario establecerla de antemano.

En un problema de análisis de expresión, lo resultados finales suelen estar relacionados con la recuperación de sondas y la asociación de las sondas a diferentes tipos de patologías, sin embargo, en un análisis arrays CGH, el caso de estudio se suele centrar en seleccionar las regiones cromosómicas que presentan ganancias o pérdidas con respecto a otros grupos para la selección de las regiones más importantes. Aunque a priori el resultado final ha de ser similar, el proceso de cálculo es muy diferente puesto que las etapas a llevar a cabo varían mucho en función del tipo de análisis. Así, en un análisis de expresión no es necesario realizar una segmentación de los datos mientras que en análisis de CGH es de vital importancia realizarlo. Sin embargo, en un análisis de expresión es muy común realizar un filtrado de los datos que viene a ser similar a la etapa de segmentación, por tanto, a pesar de ciertas diferencias, los problemas también

presentan analogías aunque en cierto modo sigue siendo necesario establecer de antemano la visión global de las etapas a llevar a cabo antes de realizar el procesamiento. Una vez especificado de modo global el flujo de ejecución cada uno de los elementos del sistema sí es responsable de seleccionar las técnicas que mejor se adaptan en función de los resultados obtenidos anteriormente. En el capítulo de análisis de expresión se han descrito numerosas técnicas que se pueden aplicar en diversas etapas.

La memoria de casos se encuentra distribuida entre los diferentes agentes pero existe una memoria común a todos ellos que se describe en función de la tabla (Tabla 5). El primero de los campos almacena el ID del caso. El segundo de los campos almacena el conjunto de intensidades asociadas al microarray. El tercer y cuarto campo almacena la información referente a la clasificación asignada y a la clasificación final. La existencia de esta memoria de casos compartidos es simplemente por eficiencia para reducir el tamaño de las memorias.

Campo	Tipo
Id	**Integer**
Intensidades	**ArrayList<Real>**
Clasificación	**String**
Clasificación Real	**String**

Tabla 5. Estructura global de las creencias

A parte de esta memoria, otros agentes almacenan su propia memoria en función de las necesidades de cada uno de ellos, tal y como se mostraba en la planificación del agente con capacidades de planificación basado en el modelo BDI tablas (Tabla 2, Tabla 3, Tabla 4).

En los subapartados siguientes, se describirá cada una de las capas de modo más expuesta en la arquitectura mostrada en la figura (Fig 14) de un modo más concreto.

4.2.1 Capa de Organización

La capa de organización engloba los diferentes agentes que replican la estructura organizativa seguida para llevar a cabo un análisis de expresión. Un análisis de expresión se suele descomponer en varias etapas, en una primera etapa, el personal de laboratorio toma una serie de muestras de los pacientes para llevar a cabo el proceso de hibridación, una vez obtenidas las muestras, las introduce en los chips de microarrays, se

realiza la hibridación y finalmente se produce un escaneo de los niveles de intensidades para posteriormente interpretar los resultados. Durante el proceso de hibridación se pueden producir errores y determinadas muestras pueden no contener los valores deseados por tanto, se hace necesario que el personal verifique que el proceso se ha llevado a cabo satisfactoriamente antes de introducir los datos en los análisis. Además, antes de analizar los datos es necesario homogeneizar los datos para reducir las diferencias de medida procedentes de los diferentes aparatos usados para medir las intensidades (miroarrays y escáner).

Una vez realizada la primera etapa en la que se obtienen las diversas medidas, el siguiente paso que se llevaría a cabo sería el análisis de los datos. Los datos normalmente son analizados por personal cualificado en dicha tarea y no suele ser realizado comúnmente por médico ya que son necesarios conocimientos estadísticos para llevar a cabo dichos análisis. Los conocimientos estadísticos son necesarios para evitar que se apliquen análisis erróneos como podría ser la aplicación de una ANOVA [Pavlidis, 2003] [De Haan *et al.*, 2009] sobre datos que no son normales, o la selección de un número de variables incorrectos al aplicar PCA, por razones similares, es necesario que este trabajo lo realice personal cualificado.

Finalmente, una vez el personal cualificado ha realizado el análisis de los datos, se procede a extraer la información relevante de un caso de estudio que será analizada por el personal especializado para extraer las correspondientes conclusiones. En este caso, el personal cualificado son los médicos y serían los encargados de determinar si los resultados extraídos son importantes o no. No obstante, se le pueden facilitar diferentes mecanismos para la ayuda a la toma de decisiones sobre la relevancia de determinadas variables.

En la capa de organización, se ha intentado simular el comportamiento descrito anteriormente. Se ha replicado la información del personal explicado en la arquitectura multiagente tal y como se muestra en la figura (Fig 14), no obstante, la funcionalidad de cada uno de los agente varía en función de las necesidades. En los siguientes apartados se indica la funcionalidad y estructura de cada uno de ellos.

4.2.1.1 Laboratorio

El agente de laboratorio de la capa de Organización es el encargado de tomar las muestras correctas para su posterior selección y análisis. Además, es el encargado de realizar la homogenización de los valores para evitar variabilidad de los datos fruto de la realización de los análisis por diferente personal.

El agente Laboratorio, al igual que el resto de los agentes de la capa de organización, no implementa la funcionalidad en su comportamiento sino que realiza

una toma de decisiones sobre la técnica a usar en función de las diferentes alternativas existentes. Las diferentes alternativas usadas comúnmente en análisis de expresión son PLIER, RMA, MAS 5 que se han explicado brevemente en el apartado (2.3) en el que se trataban las diferentes técnicas de preprocesado existentes. Puesto que la variabilidad de las técnicas usadas para este apartado no es muy grande, la selección de la misma no es un aspecto que sea necesario automatizar puesto que el uso de estas técnicas está muy estandarizado y es la que se usa comúnmente. Sin embargo, la decisión de la selección de la técnica no radica en dicho agente, el agente Laboratorio delega esta decisión a los agentes de la capa de análisis con los que interacciona para llevar a cabo los resultados. En este caso, se configura el agente para que se comunique con el agente preprocesado de la capa de análisis que es el responsable de la toma de decisiones del proceso a aplicar. De este modo, el agente Laboratorio desacopla su funcionalidad tanto de la toma de decisiones de la selección del método más adecuado como de la programación algorítmica de los mismos.

Así, la estructura interna del agente viene marcado por el modelo que se muestra en el siguiente pseudocódigo:

planificación basada en casos con el modelo BDI

Input: Nuevo Caso

1. retrieve: vacía
2. reuse: llama al agente correspondiente de la capa de análisis.
3. revise: vacía
4. retain: vacía

En este caso, la memoria de casos se define en función de lo mostrado en la siguiente tabla (Tabla 6). La información de la memoria de casos se basa en la definición global de la memoria para evitar el tamaño excesivo de la misma. Los datos almacenados en este caso son los IDs de los casos tratados, y el identificador del plan seguido por el agente de la capa de análisis que ha llevado el procesamiento.

Campo	Tipo
listaIDCasos	**ArrayList<Integer>**
idPlan	**Integer**

Tabla 6. Creencias del caso agente Laboratorio

4.2.1.2 Doctor

El agente doctor es el encargado de recuperar los resultados obtenidos y realizar las interpretaciones oportunas de los mismos, este agente recibe la información proporcionada por el agente diagnóstico y analiza los resultados. El agente tiene muy poca funcionalidad porque se limita a mostrar los resultados y el es usuario/experto el que los interpreta y establece si los resultados han sido buenos o no. No obstante, este proceso es de vital importancia para mantener la eficiencia de los casos. En función de la de la veracidad de los resultados, los demás agentes calculan la eficiencia que es determinante para la selección de flujos de ejecución en planificaciones siguientes. Este agente sólo accede a la memoria de casos global (Tabla 5) y la actualiza. De modo automático al actualizar la memoria de casos los agentes de la capa de organización reciben el aviso de dicha actualización y avisan a los agentes de la capa de análisis para que actualicen su memoria de casos.

El comportamiento del agente se muestra en el siguiente fragmento.

planificación basada en casos con el modelo BDI

Input: Reglas, Casos, Cluster, Nuevo Caso

5. retrieve: vacía
6. reuse: vacía
7. revise:
    a. Revisa Cluster
    b. Revisa variables de las reglas
    c. Revisa asignación del nuevo caso en base a reglas y cluster
    d. Asigna la clasificación final del nuevo Caso
8. retain: actualiza la memoria de casos global

4.2.1.3 Diagnóstico

Se trata del agente con mayor funcionalidad de la capa de organización, sin embargo, delega todo el comportamiento a agentes de la capa de análisis que son los que realmente realizan el procesamiento. El agente diagnóstico, se debe de configurar de modo manual en las primeras iteraciones para indicar los resultados que se quieren obtener, los resultados varían en función del problema. Así, no es lo mismo realizar una análisis de expresión que realizar un análisis de de arrays CGH por tanto, es necesario indicar los pasos inicialmente a realizar para establecer los resultados que se quieren obtener.

En el caso de un análisis de expresión los pasos típicos que se deben realizar son: filtrado, cluster-clasificación y extracción del conocimiento. Para cada una de estas fases existen un gran número de técnicas e incluso se pueden aplicar varias de estas técnicas en conjunto para realizar una determinada fase. Así, en la fase de filtrado es normal aplicar diversos test para seleccionar las diferentes sondas de modo que los datos se filtran de modo iterativo. En la fase de clasificación-cluster al igual que en la fase de filtrado se pueden aplicar muchas técnicas, se ha dedicado el capítulo (2) para indicar las diferentes alternativas existentes y los beneficios de cada una de ellas. De igual modo, en el caso de las técnicas de extracción del conocimiento se tienen diversas alternativas, en el capítulo de análisis de expresión se explican alguna de las existentes. Para llevar a cabo las diferentes tareas de cada una de las fases, se hace uso de los diferentes agentes de la capa de análisis. En el caso de los análisis de expresión, se seleccionan los agentes de filtrado, cluster y extracción del conocimiento de modo que los resultados de cada uno de los agentes son suministrados al siguiente hasta finalizar el flujo. Cada uno de estos agentes serán los responsables de seleccionar finalmente la secuencia de actividades a realizar que mejor se adaptan al problema.

El agente diagnóstico se basa en la arquitectura de planificación basada en casos con el modelo BDI que permite establecer el flujo de ejecución de agentes para un problema en base a los planes anteriores. En este caso, el funcionamiento es muy sencillo puesto que se limita a seguir el mismo flujo de agentes que la configuración inicial realizada para el tipo de problema en concreto. Si se trata de una análisis de expresión automáticamente selecciona los agentes de filtrado, cluster y extracción del conocimiento para calcular los resultados.

El comportamiento del agente se muestra en el siguiente fragmento

Planificación basada en casos con el modelo BDI

Input: Casos Preprocesados, Nuevo Caso

1. retrieve: filtra los casos y recupera las variables importantes
2. reuse: crea los diferentes cluster y clasifica el nuevo caso
3. revise: genera el conocimiento
4. retain: vacio

4.2.2 Capa de Análisis

A diferencia de la capa de Organización la capa de análisis presenta una mayor capacidad de adaptación que se consigue gracias al modelo planificación basado en el modelo BDI. La capa de análisis contiene a los agentes encargados de seleccionar las

diferentes técnicas a aplicar para llevar a cabo el análisis de los diferentes tipos de datos. Cada uno de los agentes de esta capa, dispondrá de su propia memoria de casos en base a la cual se realizan las planificaciones para llevar a cabo los diversos análisis. Los agentes de esta capa siguen el modelo de planificación basada en casos con el modelo BDI y presentan una mayor funcionalidad que los agentes de la capa de organización ya que serán los responsables de realizar los análisis de los datos. Para llevar a cabo el análisis de lo datos harán uso de una serie de servicios siguiendo una arquitectura SOA (Servicio Orientado Arquitectura), de este modo los agentes desacoplan la funcionalidad de su comportamiento facilitando de este modo la adaptación y la implementación de nuevas técnicas en base a planificaciones pasadas.

Siguiendo una arquitectura SOA, se implementa las diferentes técnicas necesarias para llevar a cabo un análisis de expresión, los servicios son invocados por los agentes según una secuencia calculada en base a los planes y eficiencia de planes calculados anteriormente. La secuencia de los servicios y los servicios se calculan en base al modelo de planificación explicado anteriormente en la sección (4.1.1) lo que permite añadir nuevos algoritmos que una vez usados en el sistema, se integran de modo automático a la planificación permitiendo ser usados en futuras planificaciones en caso de que se estime óptimo.

Los servicios se implementan fuera de la capa de análisis y se ponen a disposición para su uso desde la capa de análisis. Los servicios se agruparán de igual modo que los agentes de la capa de análisis, tal y como se verá posteriormente, y todos ellos deberán de cumplir una serie de restricciones para que se puedan usar. De modo muy resumido, todos deben aceptar los mismos datos de entrada con la excepción de parámetros de configuración. Hay que tener en cuenta que la capa de análisis no realiza un descubrimiento de servicios por lo que no es necesario definir una ontología y establecer un motor de razonamiento que permita concatenar los servicios para conseguir alcanzar una meta. En este caso, lo importante no es alcanzar la meta, sino lo importante es alcanzar una meta que sea eficiente en cuanto a una métrica. El sistema mutligente trata de optimizar los planes en aquellos sistemas en los que alcanzar un estado final es sencillo y el problema es alcanzar un estado final óptimo.

4.2.2.1 Agente de preprocesado

No se ha prestado mucha importancia a dicho agente puesto que en este caso sólo se ha implementado una técnica para llevarlo a cabo. Anteriormente, en el capítulo de análisis de expresión se expusieron diferentes técnicas que se usaban para llevar a cabo este análisis. Debido a que no son muy variadas y puesto que la más extendida es RMA, el sistema se enfocó otras etapas llevadas a cabo en los análisis de expresión.

El comportamiento del agente se muestra en el siguiente fragmento

Planificación basado en el modelo BDI

Input: Nuevo Caso

1. retrieve: casos existentes
2. reuse: preprocesado del nuevo caso junto con los casos existentes.
3. revise: automática llevada a cabo por el algoritmo de preprocesado que marca las sondas defectuosas.
4. retain: actualiza la memoria de casos global

4.2.2.2 Agente Filtrado

El agente de filtrado es el encargado de tomar las muestras de datos y seleccionar las variables que permiten llevar a cabo una clasificación o distinción de los individuos, es decir, su objetivo es eliminar las variables que contienen información irrelevante para realizar una clasificación o una identificación de casos a partir de sus atributos. Esta tarea es de vital importancia si se está trabajando con un número elevado de variables ya que permite seleccionar la información relevante y además, permite reducir la dimensionalidad facilitando posteriormente la aplicación de algoritmos con cierta carga computacional.

Para la selección de las variables, el sistema es responsable de filtrar las variables mediante la selección de una serie de técnicas que mejor se ajusten al caso de estudio. La selección de las técnicas, se hará en función de la eficiencia obtenida en planificaciones anteriores. Cada uno de los planes de esta capa almacena la información mostrada anteriormente en las tablas (Tabla 2, Tabla 3 y Tabla 4). La estructura interna del agente se basa en el modelo planificación basada en casos con el modelo BDI explicado anteriormente en la sección (4.1.1).

Los diferentes algoritmos se implementan como servicios lo que facilita la inclusión de nuevos procedimientos y su inserción en los mecanismos de planificación. Los servicios disponibles se encuentran en la capa de servicios correspondiente al análisis y dentro del análisis en el conjunto de servicios de filtrado. Los servicios son compatibles entre sí, es decir cualquier salida de un servicio de este subconjunto se puede aplicar como entrada a otro servicio del mismo subconjunto. Esto permite no tener que realizar un descubrimiento de servicios para llegar a la meta. En este caso, el objetivo es encontrar un plan óptimo que alcance una meta no es alcanzar una meta con independencia de la eficiencia.

El agente a partir de los servicios disponibles en el conjunto de servicios de filtrado, crea un plan que consiste en un flujo de ejecución de modo que eliminar las variables que considera irrelevantes.

El comportamiento del agente se muestra en el siguiente fragmento:

Planificación basada en casos con el modelo BDI

Input: Casos Preprocesados, Nuevo Caso

1. retrieve: Recupera planes de filtrado óptimos y no óptimos realizados sobre los casos en etapas anteriores.
2. reuse: construye el grafo dirigido en base a los explicado en el apartado 4.1 y calcula la mejor ruta
3. revise: calcula la eficiencia de la etapa en base a la ecuación definida en el caso de estudio en concreto
4. retain: almacena el plan junto con la eficiencia final

4.2.2.3 Agente de cluster

La funcionalidad del agente de cluster es más sencilla que el agente de filtrado puesto que para la realización de dicha tarea, sólo es necesario aplicar una técnica y no una concatenación de varias, por lo que el sistema se limita a seleccionar la mejor de las existentes en base a las creencias. En este caso, las creencias se definen en base a la técnica empleada y la tasa de error obtenida en base a las clasificaciones previas ya conocidas por lo que es relativamente sencillo saber la eficiencia de un plan si se define en estos parámetros.

Las creencias del agente, se definen en función de los siguientes parámetros: casos asociados a un determinado plan, la técnica de cluster asociada, la tasa de acierto y finalmente el número de variables de que se disponía en cada uno de los casos. La información se muestra en la siguiente tabla (Tabla 7).

Campo	Tipo
listaIDCasos	**ArrayList<Integer>**
cluster	**Cluster**
listaClasificación	**ArrayList<Integer>**
tasa	**Real**

variables	**Entero**

Tabla 7. Creencias del caso agente Laboratorio

El agente sigue el modelo planificación basada en casos con el modelo BDI al igual que los agentes anteriores, selecciona el servicio que mejor se ajusta en base al siguiente comportamiento:

Planificación basada en casos con el modelo BDI

Input: Casos Filtrados, Nuevo Caso

1. retrieve: Recupera planes de cluster que poseen un número variables similares
2. reuse: selecciona el plan de mayor eficiencia y aplica la técnica de cluster asociada
3. revise: calcula la tasa de error
4. retain: almacena el plan junto con la tasa de error en la memoria

4.2.2.4 Agente de Extracción del conocimiento

Durante esta etapa, se pueden aplicar muchos algoritmos, algunos de ellos, los más representativos se mostraron en el capítulo de análisis de expresión aunque existen muchas más. Al igual que ocurría con la etapa de cluster, en esta etapa el sistema se limita a seleccionar la técnica que mejores resultados ha dado en base a las creencias de planes anteriores. Las creencias se definen en base al siguiente esquema:

Campo	Tipo
listaIDCasos	**ArrayList<Integer>**
extracción	**Extracción**
listaClasificación	**ArrayList<Integer>**
tasa	**Real**
variables	**Entero**

Tabla 8. Creencias del caso agente Laboratorio

El agente sigue el modelo planificación basada en casos con el modelo BDI al igual que los agentes anteriores, seleccionando el servicio que mejor se ajusta en base al siguiente comportamiento:

Planificación basada en casos con el modelo BDI

Input: Casos Filtrados, Cluster,

1. retrieve: Recupera planes de extracción que poseen un número variables similares
2. reuse: selecciona el plan de mayor eficiencia y aplica la técnica de extracción asociada
3. revise: calcula la tasa de error
4. retain: almacena el plan junto con la tasa de error en la memoria

4.3 Comunicación

La comunicación entre los diversos agentes se realiza de modo directo mediante el uso del estándar FIPA-ACL [FIPA, 2002] [Bauer, Huget, 2003] [Odell, Huget, 2003]. El estándar FIPA-ACL define una forma común para el intercambio de mensajes entre sistemas multiagentes lo que permitiría la interconexión entre diferentes sistemas. El estándar FIPA-ACL define una serie de parámetros configurables en los mensajes, en la tabla (Tabla 9).

Parámetro	*Categoría*
performative	Tipo de comunicación
sender	Emisor
receiver	Receptor
reply-to	Agente a responder
content	Contenido
language	Descripción del contenido
encoding	Descripción del contenido
ontology	Descripción del contenido
protocol	Control de la comunicación
conversation-id	Control de la comunicación

reply-with	Control de la comunicación
in-reply-to	Control de la comunicación
reply-by	Control de la comunicación

Tabla 9. Parámetros mensajes FIPA-ACL

Entre los campos más importantes, se tienen performative que indica el tipo de comunicación y suele ser usado para el filtrado y la selección de mensajes, language la definición del lenguaje usado, encodig la codificación o el formato del mensaje y finalmente la ontología que sigue, la ontología se puede usar de igual modo como un campo de filtrado. Una completa descripción de cada uno de los parámetros se puede ver en la especificación de FIPA-ACL [FIPA, 2002].

De modo independiente al contenido de los mensajes, se define el formato de los mensajes usado para el envío. En formato puede ser XML aunque existen otros diferentes según la plataforma de agentes.

La comunicación entre los agentes y los servicios se realiza siguiendo el protocolo estándar SOAP, para ellos se dispone de un módulo de comunicación que convierte el formato de los mensajes de FIPA-ACL a SOAP y viceversa. El módulo permite interconectar servicios y agentes de modo sencillo, este módulo viene implementado en algunas plataformas de agentes como JADE por lo que no siempre sería necesaria su implementación

4.4 Ejemplo de procesamiento

El funcionamiento de la plataforma varía en función del modelo de procesamiento indicado ya que presenta dos modos diferentes: uno manual y otro automático. El modelo manual es necesario en fases iniciales del sistema cuando no se dispone de casos para llevar a cabo el análisis de los datos, además, este modo permitirá incluir nuevos algoritmos al sistema incluyendo para ello planes que incorporan las nuevas técnicas a añadir. Es necesario recordar que en caso de disponer de un servicio no conectado al grafo, este nunca será incluido en ninguno de los planes de ahí la necesidad de la planificación manual tanto como punto de partida inicial como de mecanismo de inclusión de nuevos servicios.

La planificación manual se limita a aplicar sobre los planes de una creencia un flujo de ejecución determinado para un agente de la capa de análisis en concreto. El flujo se aplica a nivel de agentes de la capa de análisis puesto que son los que permiten incorporar nuevos algoritmos. La configuración de los agentes de la capa de organización es configurable pero es estática, lo que implica que nunca se realiza una

configuración automática diferente a la preestablecida. Esta limitación, se comentó anteriormente puesto que no es posible determinar de antemano la intención de los analistas de los datos ni los resultados deseados. En una planificación manual, se indican los datos del mismo modo que en una automática la diferencia de que el agente no se comporta como una agente con capacidad de planificación automática. El comportamiento en modo manual sigue el siguiente esquema (Tabla 10) para cada uno de los agentes de la capa de organización.

Agente capa de organización Input: Casos, Nuevo Caso 1. Agente capa de organización selecciona los agentes de la capa de análisis de modo manual. 2. Cada uno de los agentes selecciona los servicios que llevan a cabo las tareas del agente de la capa de análisis, al igual que en el paso anterior, este proceso se realiza de modo manual. 3. Se ejecuta el flujo de servicios seleccionado manualmente.

Tabla 10. Agente Laboratorio Manual

De este modo, cada uno de los servicios insertados en el sistema se añade de modo manual a la memoria de planes, así de este modo, se añade de modo automático el nuevo plan al sistema de planificación.

En el caso de tratarse de una planificación automática, es necesario hacer una distinción para cada uno de los agentes de la capa de organización y de análisis. En el caso de tratarse del agente laboratorio de la capa de organización, y suponiendo que en la memoria de planes se dispone de una configuración previa para el análisis de expresión, el agente se limita a seleccionar los agentes de la capa de análisis. En un funcionamiento normal, el agente de la capa de análisis seleccionado sería el agente de preprocesado y este a su vez selecciona el servicio más adecuado para llevar a cabo esta tarea. De modo similar se procedería con el resto de los agentes de capa de organización. Los diferentes servicios se verán en siguiente capítulo.

4.5 Conclusiones

En este capítulo se ha expuesto el modelo propuesto y el sistema de planificación automática para los diferentes agentes de la arquitectura. El modelo propuesto se trata de un modelo genérico sin ser aplicado en particular a ningún caso de estudio y tiene la

posibilidad de modificar el comportamiento mediante la incorporación de nuevas técnicas gracias a la planificación automática. La planificación automática además, permite estimar la eficiencia de flujos y la generación de nuevas planificaciones a partir de las experiencias pasadas tal y como se planteaba en la definición de la hipótesis y los objetivos.

En el siguiente capítulo se expondrán las diferentes técnicas introducidas en el modelo propuesto para llevar a cabo el análisis de los datos para los casos de estudios concretos.

Capítulo V

Caso de Estudio

VNiVERSiDAD
D SALAMANCA

En el capítulo anterior se ha expuesto el modelo de la arquitectura sin entrar en detalle sobre las técnicas empleadas para un determinado caso de estudio. En este capítulo se concreta el caso de estudio y las técnicas empleadas para llevar a cabo el análisis de los datos. Finalmente, se analizan los resultados obtenidos.

Introducción

La arquitectura multiagente IBO se ha usado para desarrollar un sistema de apoyo a la decisión para la clasificación de pacientes con leucemias [Corchado *et al.*, 2009]. El sistema se ha aplicado sobre datos obtenidos a partir del análisis de microarray de pacientes con las patologías indicadas. Los datos de pacientes con leucemia se obtuvieron a partir del chip HG U133 plus 2.0 y se correspondían con pacientes con 212 pacientes con cinco tipos de leucemia (ALL, AML, CLL, CML, MDS), también se aplicó el sistema a pacientes con leucemias de tipo CLL para la detección de las características de los diferentes subtipos existentes en la base de casos.

El sistema IBO se ha usado para modelar organizaciones correspondientes a diferentes casos de estudio, proporcionando soporte a la toma de decisiones para la realización de clasificaciones, agrupaciones de pacientes y la detección de las diferentes sondas que caracterizan a los diferentes tipos de enfermedades. El análisis de datos de microarrays se lleva a cabo de manera distribuida por diferente personal llevando a cabo un conjunto de tareas que se suelen descomponer en:

1. Proceso de hibridación: consiste en realizar la prueba física para la obtención del los valores de luminiscencia, básicamente consiste en la introducción del material genético (médula ósea, sangre…) del paciente en el chip de microarray, su posterior lavado y finalmente el escaneo de los valores de luminiscencia. Este proceso está preestablecido y no se puede alterar por lo que queda fuera de estudio.
2. Proceso de análisis de los datos consiste en llevar a cabo tareas con los algoritmos según sean los objetivos del problema
3. Interpretación de los resultados: una vez obtenido los resultados, se hace necesario llevar a cabo el análisis de los datos.

El sistema multiagente construido intenta reproducir y automatizar los diferentes pasos desde que los valores son escaneados hasta que los resultados son

interpretados y procesados. El objetivo del trabajo para el caso de estudio en concreto, es determinar si el sistema es capaz de extraer información y clasificar nuevos pacientes basados en clasificaciones previas llevadas a cabo. Para llevar a cabo estas tareas, se han desarrollado una serie de agentes inteligentes que distribuyen su funcionalidad mediante una serie de servicios permitiendo de este modo realizar un ajuste dinámico de su funcionalidad mediante la composición de servicios. Esta dinamicidad facilita el proceso de aprendizaje y adaptación de los agentes, permitiendo alterar su funcionalidad en función de los resultados obtenidos en procesos de clasificación previos. Los diferentes agentes y servicios se distribuyen según lo mostrado en la figura (Fig 14), se tendrá una capa de servicios que implementan la diferente funcionalidad del sistema y el sistema multiagente descompuesto en una serie de capas según la funcionalidad de los agentes.

En los siguientes apartados, se explican las diferentes técnicas aplicadas en cada uno de los agentes y servicios existentes.

5.1 Capa de servicios

Los servicios implementan algoritmos que permiten realizar análisis de expresión de microarrays [Lander *et al.,* 2001] [Corchado *et al.,* 2009]. Estos servicios son invocados por los agentes para llevar a cabo su funcionalidad. Los servicios presentan una combinación de nuevas técnicas aplicadas previamente junto con novedosas técnicas implementadas para la realización del análisis.

5.1.1 Preprocesado

Esta etapa comienza una vez que ha sido completado el proceso de laboratorio de un experimento con microarrays. El investigador obtiene varios archivos que contienen valores en bruto correspondientes a intensidades. Antes de empezar a analizar los datos, es muy importante llevar a cabo una etapa de pre-procesamiento, que permita eliminar muestras defectuosas y normalizar los datos. Tradicionalmente, esta etapa se divide en tres subetapas: Corrección de background, normalización y sumarización. En la actualidad, existe un reducido grupo de algoritmos que son utilizados por los investigadores para la realización de estas etapas [Kibriya *et al.*, 2007] [Noriega *et al.*, 2009] [Ungera, 2005], siendo los más usados MAS 5.0 (Affymetrix Microarray Suite5.0) [Hubbell, 2002], PLIER (Probe Logarithmic Intensity Error) [Affymetrix b] y RMA (Robust

Multi-array Average) [Irizarry *et al.*, 2003]. En la Tabla (Tabla 1) se muestran sus características más importantes.

De las diferentes técnicas se seleccionó para implementar como servicio web RMA porque es la técnica más usada en los diversos estudios como los de [Noriega *et al.*, 2009] recomienda usar RMA por la normalización que se lleva a cabo y por diferente software existente como Partek, además, viene implementado en herramientas de software libre para el análisis como R lo que simplifica de modo considerable la implementación del servicio. En cualquier lugar, la fase de preprocesado no ha sido un objetivo de especial interés en el desarrollo de este trabajo puesto que este campo se ha estudiado de modo considerable en diversos trabajos [Noriega *et al.*, 2009] [Kibriya *et al.*, 2007] y ya se encuentra bastante estandarizado aunque no obstante, autores como [Kibriya *et al.*, 2007] realizan estudios combinados aplicando diversos preprocesado para la obtención de genes finales.

RMA consiste básicamente en tres pasos:

- Corrección de fondo: los niveles de los datos de cada chip con corregidos independientemente del usando modelos probabilísticos.
- Normalización cuantílica: la normalizan de los datos se realice dentro de un conjunto de cuantiles de cada chip a partir de los datos obtenidos en la corrección de fondo. El objetivo de es conseguir que la distribución de las intensidades de las sondas sea igual para los arrays.
- Cálculo de la expresión: se calcula de modo independiente para cada uno de los conjuntos de sondas. Para obtener los niveles de expresión se asume que para cada uno de los conjuntos de sondas la corrección de fondo, normalización y transformación logarítmica de las intensidades sigue la expresión:

$$x_{ijn} = \mu_{in} + \alpha_{jn} + \varepsilon_{ijn} \text{ con i=1,..,I, j=1,...,J n=1,...,N } \sum_j \alpha_j = 0 \quad (12)$$

Donde α_j es el efecto de afinidad de la sonda, μ_i representa el $\log_2$ para el nivel de expresión para el array i y ε_{ij} representa el error idénticamente distribuido entre los diferentes arryas y de media cero. Para obtener la estimación de los valores se usa la media polaca [Holder *et al.*, 2001].

5.1.2 Filtrado

Los servicios añadidos en este subconjunto son los encargados de eliminar la información considerada irrelevante a la hora de realizar un análisis de datos de microarrays. La eliminación de dichas sondas es necesaria para poder llevar a cabo técnicas que requieren carga computacional y de modo que si no se realiza el correspondiente filtrado, no es posible aplicar algunas técnicas sin la reducción previa de la información. A continuación, se mostrarán las diferentes técnicas de filtrado que se han seguido para seleccionar el conjunto de variables relevantes.

5.1.2.1 Sondas de control

Las sondas de control son un conjunto de variables que contienen unos valores de luminiscencia prefijados. Estas sondas se usan para poder determinar si el proceso de hibridación se ha realizado de modo correcto. Si algunas de estas sondas contienen valores diferentes a los estipulados, la prueba es eliminada. Las sondas de control se encuentran distribuidas a lo largo de la superficie de chip de forma que son identificables a simple vista mediante representaciones gráficas. En la figura (Fig 22) se puede ver un ejemplo con las sondas de con la información identificativa del chip "Genechip HG-U133 Plus 2" apreciable a simple vista.

Fig 22. Sondas de control

5.1.2.2 Sondas Erróneas

Una vez eliminadas las sondas de control, también es necesario identificar las sondas erróneas en el proceso de hibridación. El marcado de las sondas erróneas se realiza de modo automático en el algoritmo RMA por lo que sólo es necesario eliminarlas antes de realizar un análisis. La determinación de las sondas erróneas se realiza teniendo en cuenta la replicación de las sondas. Una misma secuencia de nucleótidos se encuentra

replicada en diferentes sondas a lo largo de la superficie del chip, de modo que si se produce una variabilidad significativa, se deben tener en cuenta los valores para esa secuencia.

5.1.2.3 Baja variabilidad

Una vez eliminadas las sondas erróneas y de control se eliminan aquellas sondas cuya variación se considera irrelevante para generar la clasificación de los pacientes. La selección de estas sondas se hace mediante el cálculo de la desviación típica de cada una de las sondas. Los pasos a seguir son los siguientes

1. La desviación típica para la sonda j, se calcula como se indica en la siguiente fórmula

$$\sigma_{\cdot j} = +\sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(\overline{\mu}_{\cdot j} - x_{ij}\right)^2} \qquad (13)$$

   Donde n representa el número de individuos, x_{ij} el valor del individuo i para la sonda j, y $\overline{\mu}_{\cdot j}$ el valor medio de la sonda j.

2. Una vez calculada la desviación típica, se estandariza el valor y finalmente

$$z_i = \frac{\sigma_{\cdot j} - \mu}{\sigma} \qquad (14)$$

   Donde $\mu = \frac{1}{n}\sum_{j=1}^{n}\sigma_{\cdot j}$ y $\sigma_{\cdot j} = +\sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(\overline{\mu}_{\cdot j} - x_{ij}\right)^2}$

3. Se filtran todas aquellas sondas con valor final menor que un determinado umbral. Las sondas que cumplen la siguiente restricción son eliminadas $z < -1.0$. Si las variables siguen una distribución normal sabiendo que $P(z < -1.0) = 0.1587$ este procedimiento eliminará alrededor de un 16% de las sondas.

5.1.2.4 Distribución Uniforme

Durante este proceso, se eliminan las sondas que siguen una distribución uniforme. Las sondas que siguen una distribución uniforme presentan una homogeneidad en los

valores para los individuos por lo que no permiten generar cluster de modo sencillo a partir de los valores de la sonda. Para la selección de las sondas que siguen una distribución uniforme, se ha seguido el test estadístico de Kolmogorov-Smirnov [Brunelli, 2001] H_0: Los datos siguen una distribución uniforme; H_1: Los datos analizados no siguen una distribución uniforme. Estadísticos de contraste:

$$D = \max\left\{D^+, D^-\right\} \tag{15}$$

Donde $D^+ = \max_{1\le i\le n}\left\{\frac{i}{n} - F_0(x_i)\right\}$ $D^- = \max_{1\le i\le n}\left\{F_0(x_i) - \frac{i-1}{n}\right\}$ i es el patrón de entrada, n el número total de elementos y $F_0(x_i)$ la probabilidad de observar valores menores que i con H_0 siendo verdadera. El valor del contraste estadístico se compra con el siguiente valor:

$$D_\alpha = \frac{C_\alpha}{k(n)} \tag{16}$$

Para el caso de una distribución uniforme $k(n) = \sqrt{n} + 0.12 + \frac{0.11}{\sqrt{n}}$ y un nivel de significación $\alpha = 0.05$ $C_\alpha = 1.358$ $\alpha = 0.1$ $C_\alpha = 1.224$.

5.1.2.5 Puntos de corte

Finalmente, para detectar las sondas de interés a la hora de realizar las clasificaciones, se procede a realizar la selección de sondas que presentan cambios en las densidades de los datos. La selección de estas sondas se realiza mediante la ordenación de los valores de las sondas y el posterior cálculo de las distancias entre los valores contiguos. A partir de las distancias, se pueden detectar si existen o no puntos que permiten separar conjuntos de elementos, para ello, se detectan valores elevados y se determina el conjunto de individuos que permite separar, en caso de que permitiera una división significativa, se mantendría la variable. Para ello se sigue el siguiente procedimiento:

1. Sea I el conjunto de individuos con las sondas filtradas de los pasos anteriores, donde $x_{\cdot j}$ representa la sonda j para todos los individuos, y x_{ij} el individuo i para la sonda j
2. Seleccionar la sonda $j = 1$, $x_{\cdot j}$
3. Ordenar en orden creciente los valores de $x_{\cdot j}$

4. Calcular el valor de $x'_{ij} = x_{i+1j} - x_{ij}$
5. Determinar si la variable x'_{ij} sigue una distribución normal mediante el test de Shapiro-Wilk [Jurečkováa, Picek, 2007], sino ir al paso 9.
6. Calcular el valor de la $\bar{x}'_{\cdot j}$
7. Establecer el intervalo de confianza para la varianza, se establece del siguiente modo $\sigma'^2_{\cdot j} \in \left[\frac{(n-1)\cdot S^2}{\chi^2_{n-1,1-\alpha/2}}, \frac{(n-1)\cdot S^2}{\chi^2_{n-1,\alpha/2}} \right]$ con $\alpha = 0.05$ y $n = \#x'_{\cdot j}$ el número de elementos de $x'_{\cdot j}$
8. Establecer el conjunto de elementos de x'_{ij} que no pertenecen al conjunto $Q_j = \left\{ x'_{ij} / x'_{ij} > I^+_{\sigma'_{\cdot j}} \right\}$ donde $I^+_{\sigma'_{\cdot j}}$ es el valor del límite superior de los intervalos de confianza.
9. Ir al paso 11.
10. Seleccionar los valores que se encuentran por encima del percentil P_α de cada $x'_{\cdot j}$ y establecer el conjunto $Q_j = \left\{ x'_{ij} / x'_{ij} > P_\alpha \right\}$
11. Seleccionar la sonda j+1 si quedan sondas sin revisar e ir al paso 2.
12. Crear el nuevo conjunto de sondas $I = \bigcup x'_{\cdot j} / \exists x'_{ij} \in Q_j / i > \#x' \cdot u \wedge i < \#x' - \#x' \cdot u$
13. Finalizar y devolver el nuevo conjunto de sondas I

En la figura (Fig 23), se muestra una sonda en la que se localiza una separación entre dos valores significativos debido a que la distancia entre dos valores ordenados supera el valor del percentil establecido. Una vez determinado este punto, se comprueba si deja por encima y debajo un número mínimo de individuos. En este caso al superar un determinado umbral se mantiene la variable. Los círculos del eje X representan los individuos los valores del eje Y representa la densidad de individuos.

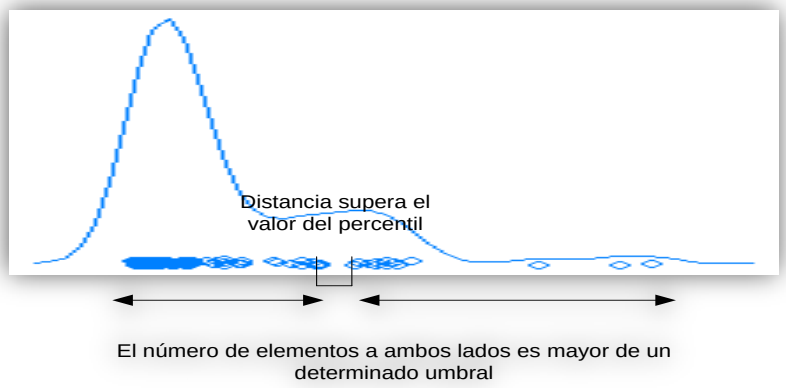

Fig 23. Sonda con puntos de corte

5.1.2.6 Correlación

Finalmente, se realiza un test de correlación lineal de Pearson y, si se detectan sondas correlacionadas directa o inversamente, se elimina una de ellas. El valor de significación seleccionado en el test es $\alpha = 0.95$.

$$r_{x_i y_j} > \alpha \tag{17}$$

Siendo: $\alpha = 0.95$ $r_{x_i y_j} = \frac{\sigma_{x_i x_j}}{\sigma_{x_i}\sigma_{x_j}}$, $\sigma_{x_i x_j} = \frac{1}{N}\sum_{s=1}^{N}\left(\bar{\mu}_{.i} - x_{si}\right)\left(\bar{u}_{.j} - x_{sj}\right)$ Donde $\sigma_{x_i x_j}$ es la covarianza entre las sondas i y j.

5.1.3 Cluster

En este apartado se han creado diversos servicios para llevar a cabo esta tarea. Las técnicas de cluster son muy diversas y varían según la dimensionalidad de los datos y el número de individuos, así, determinadas técnicas como las redes neuronales no son eficientes cuando el número de variables es muy elevado mientras que las técnicas jerárquicas y basadas en particiones estudiadas anteriormente en el apartado (2.5), permiten la creación de cluster de modo eficiente en cuanto a rendimiento independientemente del número de variables e individuos. En este apartado se indican las técnicas propias implementadas o modificaciones de previas usadas para llevar a

cabo esta tarea. Otras técnicas implementadas como PAM o Kmedias no se indican porque no se ha modificado su comportamiento.

5.1.3.1 SODTNN

La red neuronal SODTNN (Self Organized Dynamic Tree Neural Network) que se propone en este trabajo permite detectar el número de clases existentes y crear las agrupaciones a partir de las conexiones existentes en el árbol minimal construido mediante el algoritmo de Kruskal [Campos, Ricardo, 2008]. A diferencia de las redes como la ESOINN o la GCS, la red SODTNN no hace distinción entre los datos de entrada y las neuronas que se ajustan, las neuronas se corresponden en el inicio del entrenamiento con la posición de cada uno de los datos. De esta forma, es posible eliminar la fase de expansión de una neural gas para ajustarse a la superficie. Las actualizaciones de cada neurona permiten aproximar las neuronas próximas facilitando la detección de cluster y la separación de los elementos. El proceso de aprendizaje de la red, se muestra en la figura (Fig 24). En el centro se muestra el bucle principal, el bloque 2 representa el algoritmo de Kruskal, el bloque 3 el algoritmo que actualiza las posiciones de las neuronas y, finalmente, el bloque 4 es el encargado de localizar las zonas de baja densidad y separar los cluster para la generación de nuevos grupos.

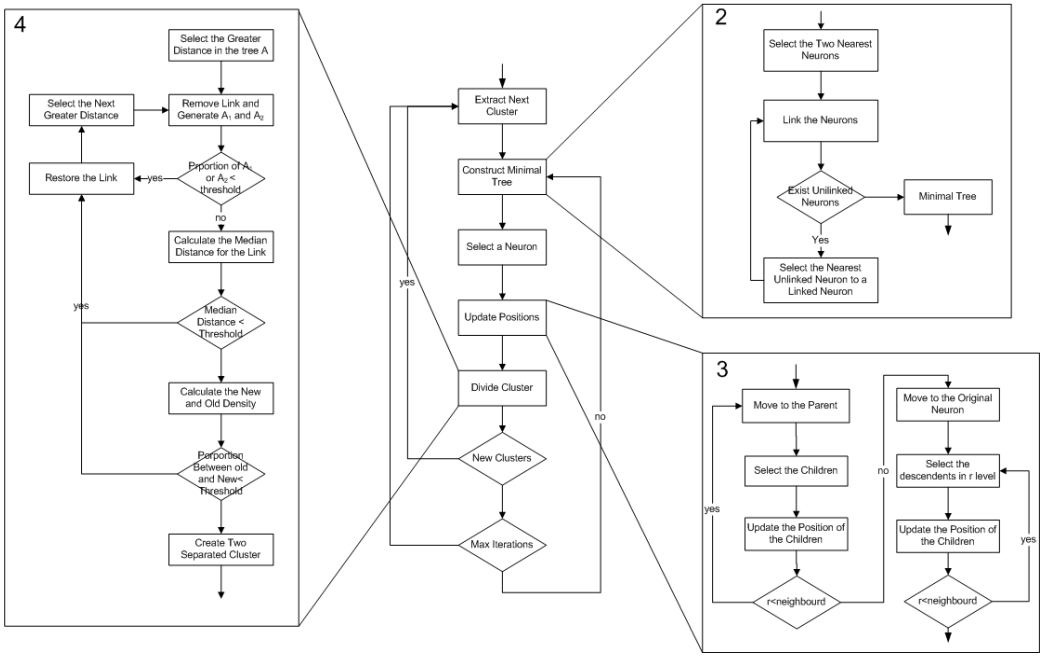

Fig 24. Algoritmo de la Red neuronal SODTNN

En los siguientes apartados se describen los pasos que se llevan a cabo en el entrenamiento de la red neuronal. La nomenclatura seguida viene definida por: *T* el conjunto de neuronas a clasificar, *A* el árbol minimal que contiene a todos los nodos de *T* donde la matriz *C* define las interconexiones entre los nodos donde el elemento $c_{ij}=1$ si el nodo $i \epsilon T$ está interconectado con el elemento $j \epsilon T$., *D* la matriz de distancias para *T*..

5.1.3.1.1 Algoritmo Densidad: bloque 4

Uno de los principales problemas que se encuentran a la hora de realizar las agrupaciones de los individuos es determinar aquellas particiones que causan un aumento significativo en la densidad de los cluster resultantes. En redes neuronales como la SOINN o la ESOINN, se lleva a cabo un estudio de la longitud de los enlaces para determinar si la longitud del enlace difiere dentro del subgrupo en el que se encuentra. Este procedimiento requiere el establecimiento de subclases dentro de cada cluster utilizando para ello una serie de funciones que determinan umbrales a partir de los que se crean estas subclases. La red neuronal SODTNN busca los puntos de corte en zonas que causan un aumento de la densidad significativo mediante la relación entre la distancia total calculada a partir de la matriz de distancias y la distancia del árbol minimal. El algoritmo para el cambio de la densidad se realiza de la siguiente manera:

1. Distancia del árbol $f^{A}(C,D)=\sum_{i,j} d_{ij} / c_{ij}=1, c_{ij} \in C, d_{ij} \in D$
2. Distancia entre las neuronas del árbol $f^{T}(D)=\sum_{i,j} d_{ij}, d_{ij} \in D$
3. Calcular la densidad final $f^{D}(C,D)=f^{T}(D) / f^{A}(C,D)$

5.1.3.1.2 Algoritmo distancia Media: bloque 4

La selección de los enlaces para la localización de zonas de baja densidad se realiza teniendo en cuenta la distancia de una neurona con respecto a su padre en el árbol *A* y la distancia media entorno a la neurona. La distancia media entorno de una neurona se calcula en función de la distancia existente en los enlaces del subárbol de profundidad igual al entorno y centrado en la neurona en estudio y el número de neuronas existentes en el subárbol. En la figura (Fig 25), se muestra el subárbol de longitud 2 para el nodo resaltado en grande y color azul. El nodo raíz del árbol se representa en rojo, las flechas representan el sentido y la magnitud de la actualización, la magnitud depende de la vecindad con respecto al nodo central del árbol y de la distancia.

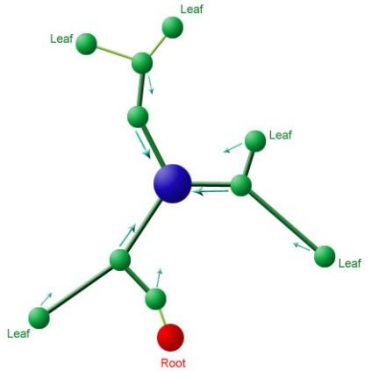

Fig 25. Subárbol de la neurona en un entorno de 2

El algoritmo seguido se describe a continuación.

1. Sea a_i la neurona del árbol para que se quiere calcular la distancia media, con i ϵ T. Sea $f^p(a_i)$ la función que determina el nodo padre de a_i, que viene definida por
$$f^p : A \to A$$
$$a_i \to f^p(a_i) = a_s \quad \text{Donde } c_{si} = 1 \text{ con } c_{si} \in C$$
2. Aplicar f^p de modo recursivo y seleccionar el nodo raíz a_r del subárbol $a_r = \overbrace{(f^p \circ \ldots \circ f^p)}^{e}(a_i)$
3. Establecer el conjunto de nodos en el entorno de a_i $A^e_{a_i} \subseteq A$ del siguiente modo $A^e_{a_i} = \{a_j \in A / \exists r \in \mathrm{N}, a_r = \overbrace{(f^p \circ \ldots \circ f^p)}^{r \le 2e}(a_j)\}$
4. Calcular la distancia media del nodo a_i $f^m(A^e_{a_i}, D) = \dfrac{\sum_{a_j \in A^e_{a_i}} d_{sj} \cdot c_{sj}}{\# A^e_{a_i}}$

5.1.3.1.3 Algoritmo de división: bloque 4

El algoritmo de división es el responsable de localizar las conexiones entre las neuronas de baja densidad para la división del cluster. Tiene en cuenta la distancia entre las neuronas y los cambios de densidades resultantes en las posibles divisiones. El procedimiento que se lleva a cabo se describe a continuación:

1. Establecer el umbral de corte de elementos α , umbral de corte de distancia β
2. Iniciar $i = 1$
3. Seleccionar i mayor distancia de $d_{jk} \in D / c_{jk} = 1$ y extraer el nodo del árbol $a_k \in A$
4. Sea A_1, A_2 los árboles resultantes al eliminar a_k y la conexión con el nodo padre $f^p(a_k)$ donde $T_1 = \{s \in T / a_s \in A_1\}$ y $T_2 = \{s \in T / a_s \in A_2\}$ con $T = T_1 \cup T_2$, $T_1 \cap T_2 = \phi$, C_1, C_2, D_1, D_2 las correspondientes matrices de enlaces y distancias.
5. Si $\#T_1 / \#T$ o $\#T_2 / \#T$ es menor que α ir al paso 13
6. Calcular la distancia media del nodo del árbol a_k siguiendo el algoritmo de la distancia media $d^m_{a_k} = f^m(A^e_{a_i}, D)$
7. Determinar si la distancia del nodo a_k del árbol con respecto a su padre es menor quela distancia media $d_{sk} \le d^m_{a_k} \cdot \beta$ donde $s \in T$ y $a_s = f^p(a_k)$ ir a paso 13
8. Determinar la densidad para T, T_1 y T_2 siguiendo el algoritmo de densidades $f^D(C, D)$, $f^D(C_1, D_1)$, $f^D(C_2, D_2)$
9. Calcular el umbral de densidad nuevo $\delta(t+1) = f^D(C_1, D_1) + f^D(C_2, D_2)$ y anterior $\delta(t) = f^D(C, D)$
10. Si el valor $\delta(t) / \delta(t+1) < 1/(\delta(0) / \delta(1) \cdot \rho)$ donde ρ es una constante ir a 12
11. Finalizar
12. Restablecer la conexión de a_k con su nodo padre.
13. Si $i < \#T$ establecer el valor de $i = i + 1$ e ir al paso 2

5.1.3.1.4 Algoritmo de actualización: bloque 3

Las neuronas de la red que definen los cluster se actualizan de forma periódica siguiendo un esquema similar al de los mapas autoorganizados de kohonen. La actualización de las neuronas permite reajustar las posiciones y conexiones para llevar a cabo separaciones entre los cluster. La red selecciona aleatoriamente una neurona de entrada y acerca a sus vecinas hacia ella, la actualización se realiza según la jerarquía del árbol. En la figura (Fig 25), se muestran unas flechas que se corresponden con el sentido y fuerza con el que las neuronas son acercadas a la neurona seleccionada. La magnitud

del vector y sentido dependen de la distancia y vecindad tal y como se indica en el siguiente algoritmo:

1. Sea $k \in T$ con $a_k \in A$ la neurona seleccionada, establecer el valor del radio de vecindad r
2. Iniciar $i = 1$, $a_s = a_k$
3. Calcular el nodo padre del nodo actual $a_t = f^p(a_s)$, obtener todos los hijos de a_t que viene definido por $A^1_{a_t}$
4. Para cada $a_j \in A^1_{a_t}$ actualizar las coordenadas de la neurona siguiendo la ecuación de los mapas autoorganizados
   $x_j(t+1) = x_j(t) + \eta(t) \cdot g(i,t) \cdot (x_s(t) - x_j(t))$
5. Donde $g(i,t)$ representa la función de vecindad y $\eta(t)$ la tasa de aprendizaje [Bajo *et al.*, 2009].
6. $$g(i,t) = Exp\left[-\frac{i}{N} \frac{\sqrt{(x_{j1} - x_{s1})^2 + \ldots + (x_{jn} - n_{sn})^2}}{\underset{i,j}{Max}\{d_{ij}\}} - \lambda \frac{i \cdot t}{\beta N} \right]$$
   $$\eta(t) = Exp\left[-\sqrt[4]{\frac{t}{\beta N}} \right]$$
7. Donde t es la iteración, N el número de elementos del conjunto $\# A$, n es la dimensión de las coordenadas, x_{ij} la coordenada j de la neurona $i \in T$, con $a_i \in A$, λ y β son constantes establecidas a 1 y 5 respectivamente.
8. Si $i < r$ establecer a $a_s = a_t$ e incrementar i
9. Actualizar siguiendo el mismo procedimiento los descendientes de $a_k \in A$ hasta una profundidad de r, $A^1_{a_k}$ ... $A^r_{a_k}$

5.1.3.1.5 Entrenamiento

El entrenamiento de la red neuronal se lleva a cabo de modo iterativo tal y como se podía ver en la figura (Fig 24). Se seleccionan de modo secuencial cada uno de los cluster de la red. El entrenamiento de la red tiene una etapa inicial tiene similitudes con mapas autoorganizados y redes neuronales como la GCS o ESOINN.

1. Iniciar el conjunto de cluster con el grupo inicial $G=\{P\}$ donde P contiene todos los nodos de la red
2. Establecer el conjunto a analizar $i=1$
3. Si $i \geq \#G$ ir a 11
4. Seleccionar el grupo a analizar $T=g_i$
5. Establecer $j=1$
6. Aplicar el algoritmo de Kruskal para construir el árbol minimal A, matriz de distancias D y la matriz de conexiones C
7. Seleccionar una neurona de T y ejecutar el algoritmo de actualización sobre el elemento de T
8. Aplicar el algoritmo dividir sobre T. Si T dividido $i=i-1$
9. Si $j<\beta N$, donde β es una constante y N el número de términos de T. Incrementar el valor de j e ir a 114
10. Si $i<\#T$ incrementar el valor de i y volver a 3
11. Finalizar

5.1.3.2 ESOINN

La red neuronal ESOINN se ha incluido como servicio con una serie de modificaciones en alguno de los pasos en su funcionamiento. La red neuronal se describe en [Furao *et al.*, 2007] por lo que en este apartado sólo se indican las modificaciones introducidas.

1. Actualizar los pesos de las neuronas siguiendo el proceso seguido anteriormente pero introduciendo una nueva definición de de la tasa de aprendizaje que se ha usado con anterioridad en otros trabajos y ha proporcionado buenos resultados [Bajo *et al.*, 2009].

$$\Delta W_{a_1} = n_1(M_{a_1})(\xi - W_{a_1})$$

$$\Delta W_{a_i} = n_2(M_{a_1})(\xi - W_{a_i}) \text{ con } a_i \in N_{a_1}$$

Donde $n_1(x)=\frac{1}{\sqrt{x}}$, $n_2(x)=\frac{1}{\sqrt{2+x^2}}$, a_i es la neurona i, N_{a_1} el conjunto de nodos vecinos de a_i, ξ patrón de entrada, M_{a_1} número de neuronas ganadoras de a_i, N_{a_1} es el conjunto de vecinos de a_i.

2. Eliminar las conexiones de mayor edad. Se estandarizan las edades y los valores que se encuentran en la región de rechazo con k>0 se eliminan. La significación establecida es $\alpha = 0.05$.

$$z_i = \frac{e_i - \mu}{\sigma} \ , \ z \equiv N(0,1) \text{ entonces } f(z) = \frac{1}{\sqrt{2\pi}} Exp\left[\frac{-z^2}{2}\right]$$

Donde $P(z < k) = \alpha / 2 \rightarrow P(z < k) = 0.975 \rightarrow \Theta(z) = 0.975$ k=1.96 Por tanto, los valores de z que son mayores que 1.96 se eliminan.

5.1.4 Extracción del conocimiento

Durante la fase de extracción de conocimiento se realiza una selección de las sondas fundamentales que caracterizan a los diferentes grupos indicados. La extracción del conocimiento se realiza por diversas técnicas al igual que el clustering, el único apartado reseñable es que se realiza una discretización de los valores previo calculo de algoritmos como CART, RIPPER o J48 ya que debido al alto número de sondas y tratarse de valores continuos el tiempo de procesado es bastante alto sino se lleva a cabo dicha tarea.

5.2 Análisis de resultados

Los resultados se han realizado en dos enfoques, en un primer paso fue necesario analizar diferentes técnicas que se aplicarían en la capa de servicios para evaluar el funcionamiento y seleccionar técnicas que den buenos resultados antes de aplicar el planificador de modo completo. El análisis separado de cada una de ellas permitió realizar una selección previa ya que algunas técnicas que se aplican no eran computacionalmente aplicables de modo directo y por tanto no se podían aplicar sin realizar algunas modificaciones previas, de ahí que se realizara esta fase inicialmente.

En una segunda parte se procedió a realizar un análisis de la planificación automática teniendo en cuenta una serie de servicios ya seleccionados. La fase que presentaba más complejidad a la hora de realizar la planificación automática, era la fase de filtrado, por tanto el análisis se centró en la misma. La complejidad proviene del hecho que es necesario seleccionar diferentes técnicas durante y ordenarlas de modo que el resultado final sea eficiente, es decir, el proceso se divide en una serie de etapas que hay que seleccionar y ordenar de modo que los resultados de una etapa se enlazan con la ejecución de la etapa siguiente. El resto de las acciones llevadas a cabo en los agentes son más sencillas de planificar puesto que se suelen componer de una sola

etapa, por tanto, el algoritmo de planificación automática sólo tiene que seleccionar la técnica más eficiente.

Para la realización de pruebas se usó de base una serie de datos obtenidos a partir del uso de microarrays. Los datos fueron proporcionados por el Instituto del Cáncer de Salamanca, obtenidos a partir de descargas directas en la web y finalmente algunos fueron proporcionados por la Universidad de Minho durante la estancia de investigación llevada a cabo durante julio-septiembre de 2009. Sobre cada uno de los conjuntos de datos se realizaron diversas pruebas, debido a que cada uno de ellos poseía mejores características para la realización de las mismas. Así, los datos proporcionados en la Universidad de Minho, se ajustaban mejor para las pruebas de extracción de conocimiento puesto que continúan un número elevado de clasificaciones. Sin embargo, los datos proporcionados por el Centro del Cáncer de Salamanca se ajustaban mejor para la realización de pruebas de filtrado y clustering ya que se trataban de datos de arrays de expresión en lugar de arrays de CGH. En el siguiente apartado se describe el conjunto de datos usados para la realización de los análisis

5.2.1 Bases de casos

El primero juego de casos se corresponde con una base de datos con 5 tipos de patologías: LAL, LAM, LLC, LMC, SMD (dónde A indica aguda, C indica crónica, L indica Linfática, M indica Mieloide). Cada una de estas patologías viene caracterizada por una sintomatología y por unos efectos que marcan su desarrollo y posibilidades de curación.

- LAL (Leucemia Aguda Linfática). Es un tipo de cáncer de la sangre y de la médula ósea, que se debe a la excesiva fabricación de linfocitos.
- LAM (Leucemia Aguda Mieloide). Es un tipo de cáncer por el que la médula ósea produce mieloblastos, glóbulos rojos o plaquetas anormales.
- LLC (Leucemia Linfática Crónica). Es un tipo de cáncer por el cual se producen demasiados linfocitos en la médula ósea. Es habitual en adultos.
- LMC (Leucemia Mieloide Crónica). Se debe al aumento de producción de glóbulos blancos en la médula ósea.
- SMD (Síndromes Mielodisplástico). Son un grupo de enfermedades de la sangre y la médula ósea por las que la médula ósea no produce suficientes glóbulos sanos. Pueden avanzar y convertirse en leucemia aguda.

En el caso de estudio que se presenta en el marco de esta investigación se dispone de 212 muestras, procedentes de análisis realizados a pacientes tanto a través de punciones en médula como de muestras sanguíneas y que han sido hibridadas y analizadas a través de arrays de expresión fabricados por Affymetrix. Para los diferentes grupos, se realizaron pruebas para la agrupación, clasificación y extracción de sondas

relevantes. Para este mismo conjunto de datos, también se realizó estudios sobre agrupaciones en el tipo de leucemias LLC.

A parte de estos datos, se disponía de una serie de datos procedentes de de GastroIntestinal Stromal Tumors (GIST), estos datos procedían de arrays CGH por lo que su análisis es muy diferente al llevado a cabo por análisis de expresión, no obstante, sirvió para comprobar el funcionamiento de diferentes clasificadores y técnicas de extracción de conocimiento sobre las diferentes características medidas. Debido a que se disponía de un número elevado de clasificaciones fue posible realizar una comparación exhaustiva de diferentes técnicas y así evaluar su funcionamiento. Más adelante, se explican los análisis llevados a cabo y las pruebas realizadas. En total se disponía de 49 muestras de estas características.

El sistema de planificación automático se evaluó sólo para los datos procedentes de arrays de expresión puesto que el modelo definido en la capa de análisis se correspondía con la arquitectura propia de un análisis de este tipo. Para el caso de los arrays de CGH los resultados a obtener suelen diferir de los arrays de expresión puesto que la información proporcionada es diferente, unos permiten analizar el grado de presencia de genes mientras que otro permite detectar anomalías en regiones cromosómicas.

A continuación en los siguientes puntos, se muestran diferentes análisis llevados a cabo sobre las diferentes bases de casos, los análisis llevados a cabo, se muestran de modo secuencial para cada uno de los conjuntos de datos. Las pruebas realizadas se llevaron a cabo en parte mediante el uso del software estadístico R [R, 2009] que facilita en gran medida la aplicación pruebas estadísticas, la aplicación técnicas de extracción del conocimiento y técnicas de clustering. El uso de dicho software facilita en gran medida el desarrollo de aplicaciones ya que permite la integración con lenguajes de programación como Java.

5.2.2 Análisis de técnicas

Tal y como se ha comentado anteriormente, en una primera iteración, se comprobó el funcionamiento de las diferentes técnicas en el sistema puesto que su validación sería complicado integradas en el sistema. La evaluación de estas técnicas se llevó a cabo mediante la aplicación de los casos de estudio indicados en el apartado anterior. Se realizó un análisis de los resultados obtenidos para las diferentes fases de un análisis de expresión. El análisis de las diferentes técnicas aplicadas en el preprocesado no se realizó puesto que RMA es la técnicas más ampliamente usada por se usó desde un inicio. Por tanto, la primera de las etapas estudiadas fue el filtrado de la información, en el filtrado de la información se implementaron las técnicas expuestas en el capítulo

anterior aunque se aplicaron más para realizar una estudio de las diferentes alternativas.

Durante la fase de filtrado, se realizó un análisis de los resultados obtenidos al aplicar las técnicas del apartado (5.1.2). El filtrado se aplicó sobre el caso de estudio de los 5 tipos de leucemias. Aplicando las técnicas indicadas, se consiguió reducir la información de las 54.000 sondas originales a 785 según la configuración final considerara óptima en base a los resultados proporcionados por las demás etapas. Los resultados detallados se pueden ver en la tabla (Tabla 11). En la cabecera de la tabla se muestran las diferentes técnicas

Variabilidad (z)	Uniforme (α)	Correlación (α)	Sondas	Errores
-1.0	0.25	0.95	2675	27
-1.0	0.15	0.90	1341	28
-1.0	0.15	0.95	1373	28
-0.5	0.15	0.90	1263	30
-0.5	0.15	0.95	1340	29
-1.0	**0.1**	**0.95**	**785**	**30**
-1.0	0.05	0.90	353	47
-1.0	0.05	0.95	357	45
-0.5	0.05	0.9	332	67
-0.5	0.05	0.95	337	67
-1.0	0.01	0.95	54	83

Tabla 11. Proceso de filtrado

De forma aislada a este proceso, se aplicó componentes principales para la reducción de la dimensionalidad pero los resultados no fueron satisfactorios puesto que las agrupaciones realizadas con estas variables eran de peor calidad.

También se aplicó el mismo procesado sobre el conjunto de pacientes con leucemias de tipo CLL para trabajar con datos más homogéneos, en este caso, el filtrado fue menos significativo y fue necesario añadir la etapa de los puntos de corte. Al añadir la nueva etapa, se consiguió reducir de modo considerable el número de sondas, se pasó de 1311 a sólo 618 aplicando esta nueva fase. En la tabla (Tabla 12) se muestran los resultados obtenidos:

	Uniforme	Puntos de corte	Correlación
CBR	1311	618	541

Tabla 12. Proceso de filtrado puntos de corte

La configuración seguida para los resultados de la tabla (Tabla 12) para los diferentes parámetros fue la misma que la resaltada en negrita en la tabla (Tabla 11).

Analizando los resultados de ambas tablas, se puede ver que las técnicas de mayor importancia para la reducción de la dimensionalidad son los filtrados de uniformidad y de puntos de corte. Otros filtrados como las correlaciones no permiten reducir de modo considerable la dimensionalidad de los datos. En la figuras (Fig 26) se muestra una sonda que sigue una distribución normal. Se muestran tres gráficos diferentes, el primero de ellos en el eje Y representa los individuos existentes mientras que el eje X representa los valores. Para cada uno de los individuos se representa un círculo en la posición del valor, tal y como se puede ver los diferentes individuos se distribuyen a lo largo de la superficie por lo que sería muy complicado crear agrupaciones con este tipo de sondas. El segundo de los gráficos es una distribución de la probabilidad uniforme, el gráfico construye una línea de puntos según la frecuencia relativa absoluta de cada probabilidad. Tal y como se puede ver, los puntos se distribuyen a lo largo de una línea por tanto la variable sigue una distribución uniforme. En la última de las figuras, se muestra un histograma, tal y como se puede ver, a excepción de los últimos valores el resto de los intervalos poseen valores similares.

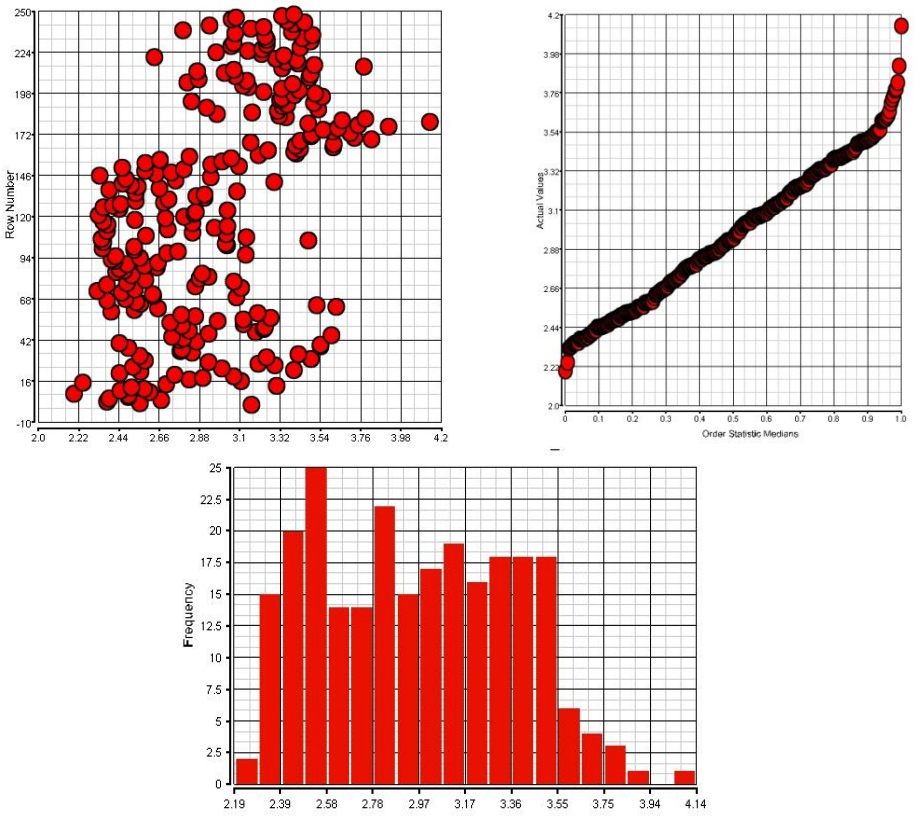

Fig 26. Sonda que sigue una distribución normal. Se representa la información de un scatter, una gráfica de probabilidad de distribución de normalidad y un histograma

En las siguientes figuras (Fig 27), se muestra una sonda que no sigue una distribución uniforme, en la primera de las imágenes se puede ver que hay dos distribuciones diferentes de los individuos, una de ellas contiene a los individuos con valores bajos mientras que el resto de los individuos poseen valores altos. En la segunda de las imágenes, se puede ver que en este caso se tienen dos líneas de distribución de los puntos por lo que la sonda no sigue una distribución uniforme. Finalmente, en la última de las imágenes se tiene un histograma, se puede ver como el intervalo situado a ambos lados de 4.39 separa dos grupos de distribución de individuos que se corresponde con lo mostrado en la primera de las imágenes.

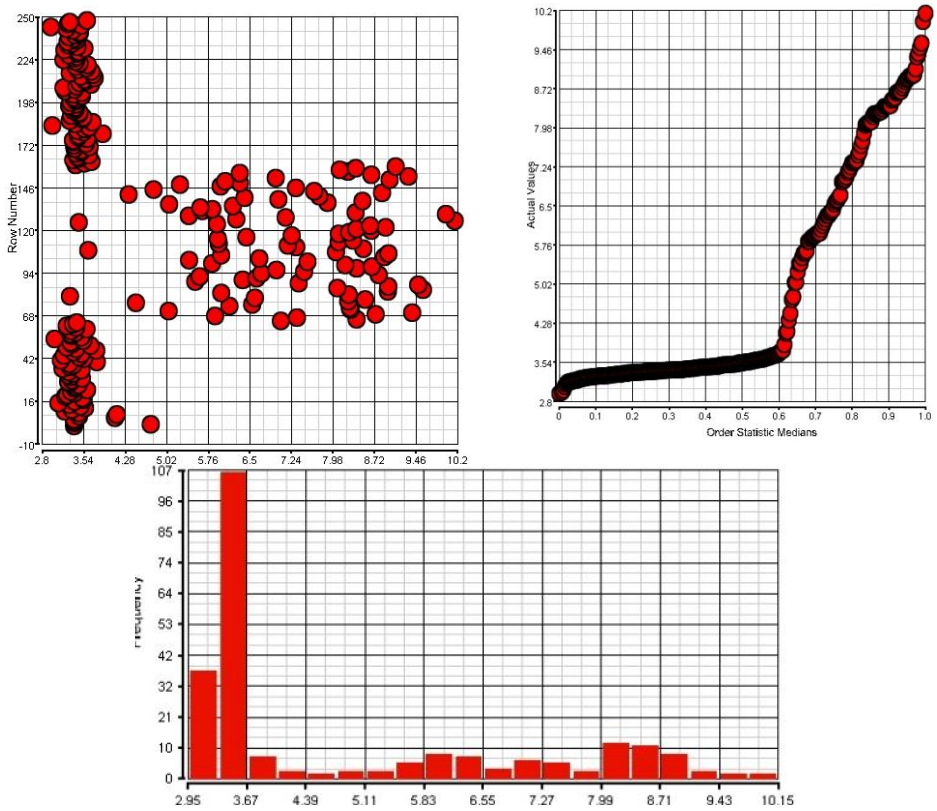

Fig 27. Sonda que no sigue una distribución normal. Se representa la información de un scatter, gráfica de probabilidad de distribución de normalidad y un histograma

En las siguientes figuras (Fig 28), se muestran dos de las sondas recuperadas durante el filtrado de las sondas. Para las dos sondas, se muestra un gráfico de intensidades y un histograma. En ambas sondas, en el gráfico de intensidades se pueden observar mínimos locales que permiten separar a conjuntos con un número de individuos significativo. Los histogramas representan la misma información pero en función de intervalos, de igual modo, se puede ver que existen intervalos con baja variabilidad.

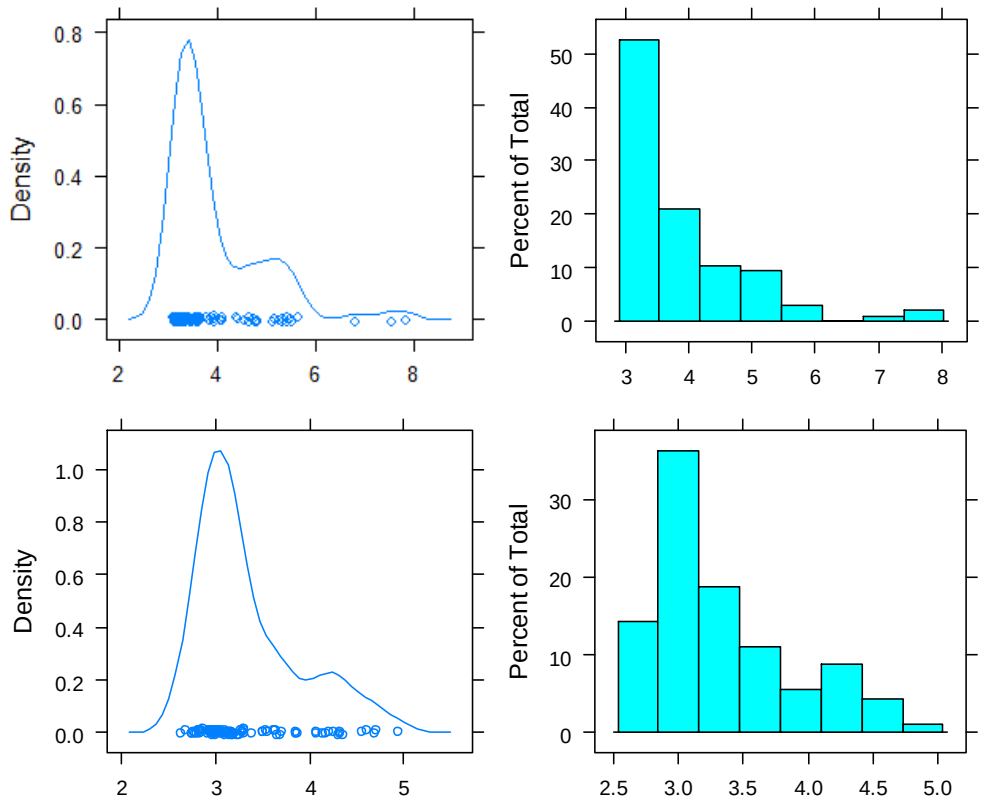

Fig 28. Sondas con puntos de corte. Se representan dos sondas y por cada una de ellas un gráfico de intensidades y un histograma

Finalmente, también se pueden eliminar variables que no aportan información y que serían claramente eliminadas o mejor dicha fusionadas como una técnica de reducción de la dimensionalidad como puede ser ACP. En este caso, sólo se estudian las correlaciones lineales tal y como se indicó en el capítulo anterior. En las figuras (Fig 29) se muestran dos sondas correlacionadas, en el primero de los dibujos se representan los diferentes valores de las dos sondas para cada uno de los individuos, una en línea roja y otra en azul, es fácil ver que los valores se encuentran correlacionados directamente. En la segunda de las imágenes, se confrontan las dos sondas en cada uno de los ejes y se representan los valores para los diferentes individuos, como se puede ver, los individuos se encuentran relacionados linealmente ya que los puntos se distribuyen a lo largo de una línea.

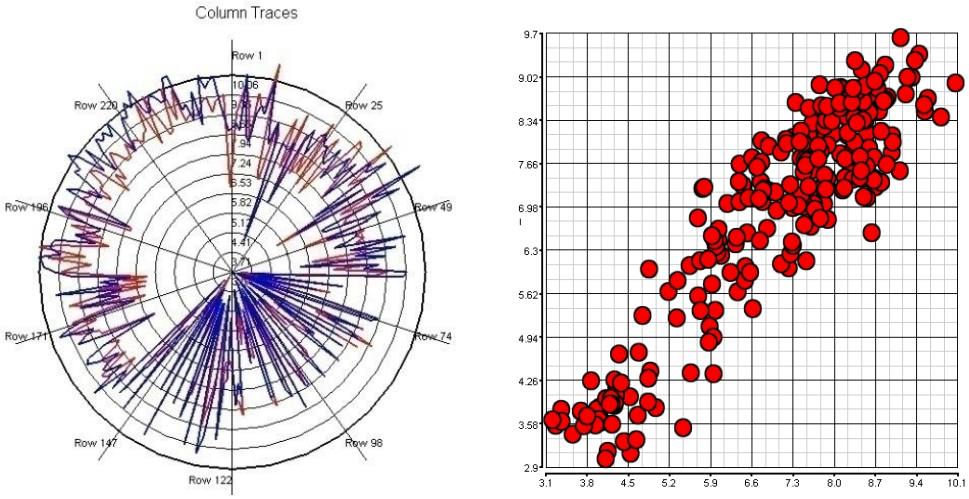

Fig 29. Sondas correlacionadas

Una vez realizado el filtrado, en el caso de los 5 tipos de leucemias, el número de variables se redujo de 54000 a 785 según los datos mostrados en la tabla (Tabla 11). Si se representan los datos de estas muestras y se calcula la distancia euclídea con respecto a las demás se genera una matriz de distancias que si se representa gráficamente, se obtiene el resultado mostrado en la figura (Fig 30). En la parte superior y a la izquierda de la imagen, se muestra una barra que indica la pertenencia de cada uno de los individuos a los diferentes grupos. Tal y como se puede observar, las sondas consideradas relevantes en la fase de filtrado, permiten diferenciar a los individuos con el simple cálculo de la distancias. Tal y como se puede apreciar, las zonas en las que los individuos coinciden, el gráfica presenta colores azules que indican una menor distancia.

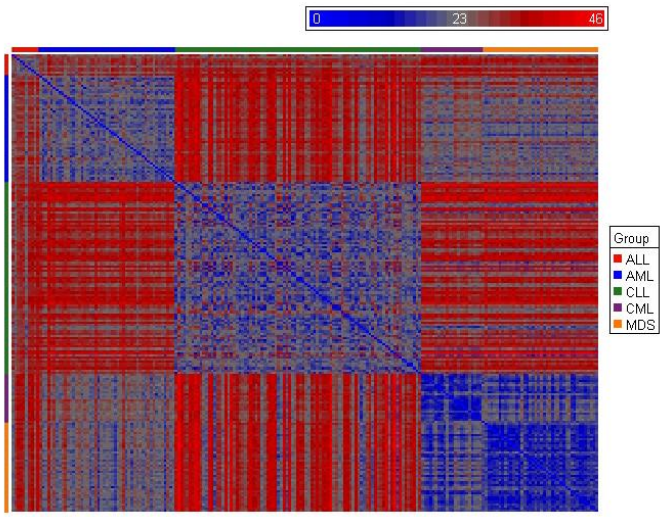

Fig 30. Representación de la matriz de distancias de los 212 individuos

Si se representan los valores de las intensidades antes de realizar el filtrado y posteriormente, se puede observar que ambas representaciones son similares. En la figuras (Fig 31) se representan los gráficos de intensidades para las 54000 sondas y en el segundo, el gráfico correspondiente para las 785 sondas una vez realizado el filtrado. Se pueden observar una serie de franjas rojas en ambas, se puede ver que permanecen después de haber realizado el filtrado y que sólo se han eliminado franjas que a priori se pueden considerar irrelevantes puesto que la mayor parte de la gráfica inicial presenta valores similares.

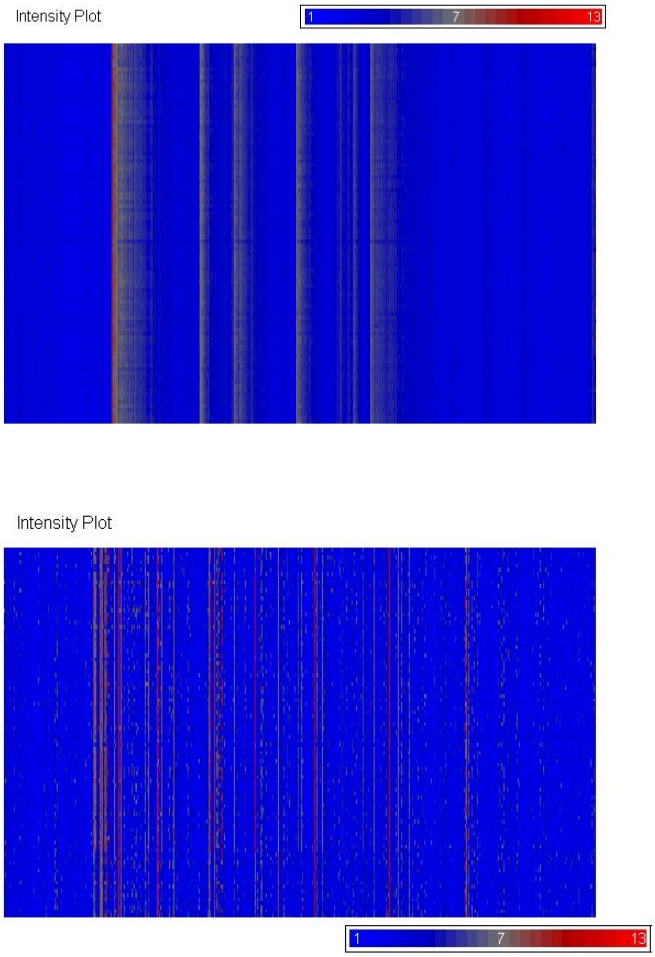

Fig 31. Gráfico de intensidades para antes de realizar el filtrado y posteriormente una vez seleccionadas las 541 sondas en pacientes con leucemias CLL.

En la imagen (Fig 32b), se muestra un gráfico de intensidades con la clasificación de los pacientes ALL. El eje X representa las sondas y el eje Y los individuos. Aunque se estén adelantando resultados de fases posteriores, se ha representado en negro los

individuos que pertenecen a las leucemias ALL y con una línea roja los individuos clasificados como ALL.

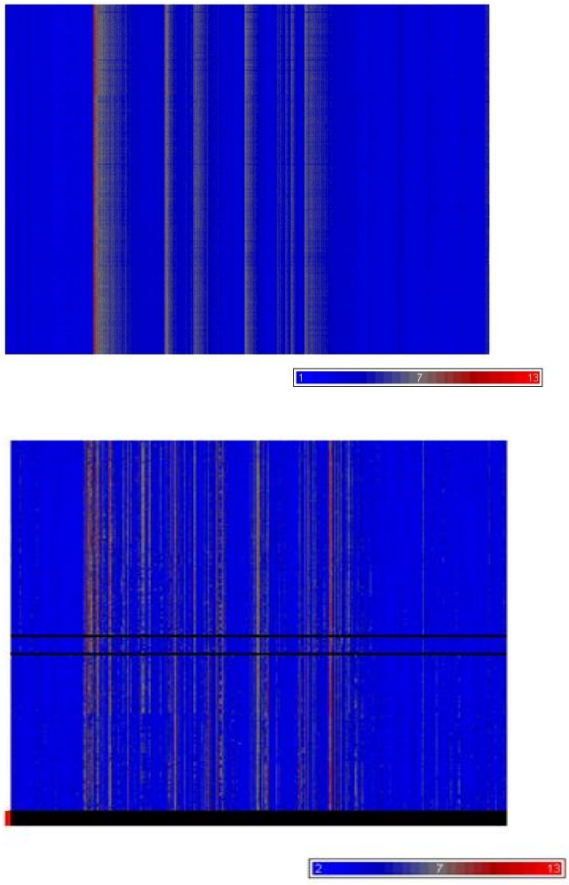

Fig 32. Gráfico de intensidades para antes de realizar el filtrado y posteriormente una vez seleccionadas las 785 sondas. En pacientes con 5 tipos de leucemia.

Si nos centramos en los individuos con leucemias de CLL, el sistema proporciona resultados similares. En la figura (Fig 33), se puede ver las imágenes son similares en cuanto a las franjas detectadas.

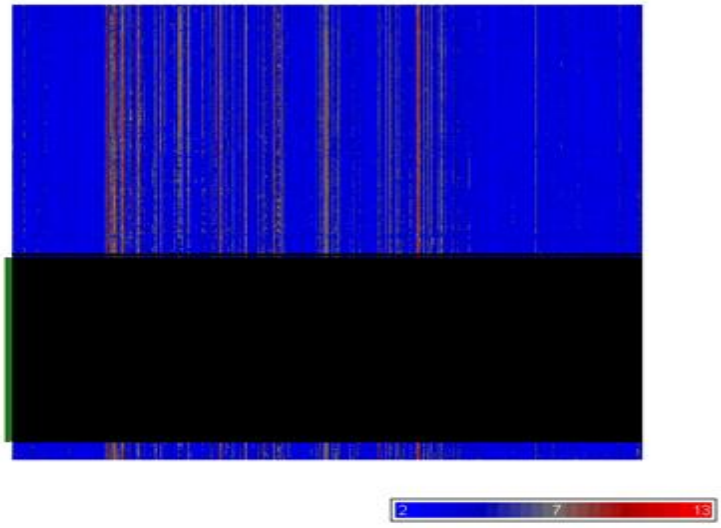

Fig 33. Clasificación de pacientes con leucemias de tipo CLL.

Una vez verificado que el proceso de filtrado proporcionaba resultados adecuados en sistema, se procedió a realizar una verificación de la siguiente etapa en el proceso de análisis de expresión. Así, se procedió a comprobar que la fase de cluster funcionaba de modo correcto. En esta etapa, se comprobó el funcionamiento de las redes neuronales comentadas en el apartado (5.1.3.1). Para ello, se procedió a realizar cluster sobre los casos preclasificados y se comprobó la tasa de acierto. La tasa de acierto para el caso de los 5 tipos de leucemias y diferentes configuraciones de la fase de filtrado se puede ver en la tabla (Tabla 11).

En la figura (Fig 32b), se puede ver una representación gráfica de los resultados obtenidos para el caso de las leucemias de tipo ALL. En la figura (Fig 33) se puede ver la clasificación de los pacientes con leucemias de tipo CLL, en negro se ha representado los pacientes con leucemias de este tipo y en verde los pacientes clasificados con este tipo de leucemia. Para los demás grupos se podría repetir el mismo proceso hasta comprobar la eficiencia total mostrada en la tabla (Tabla 11).

Finalizada la fase de cluster, se comprobó que mejoraba los resultados proporcionados con otras técnicas como los dendogramas o PAM, para realizar la prueba de estos algoritmos se usó el software R [R, 2009] para facilitar así el análisis de los datos. La red neuronal mejoraba en un 4.7% para el caso de los dendogramas y un 3% para el caso del PAM. En ambos casos, se usó el conjunto de sondas obtenido a partir de las sondas recuperadas según la configuración resaltada en negrita en la tabla (Tabla 11).

Para facilitar el estudio de las diferentes técnicas de cluster, se procedió a realizar una serie de test sobre juegos de datos más básicos que fueran fácilmente representables y así facilitar el proceso de valoración de las técnicas. Así, se procedió a

generar un conjunto de datos de modo manual de modo que siguieran diferentes distribuciones y así, se generaron los juegos de datos mostrados en la figura (Fig 34).

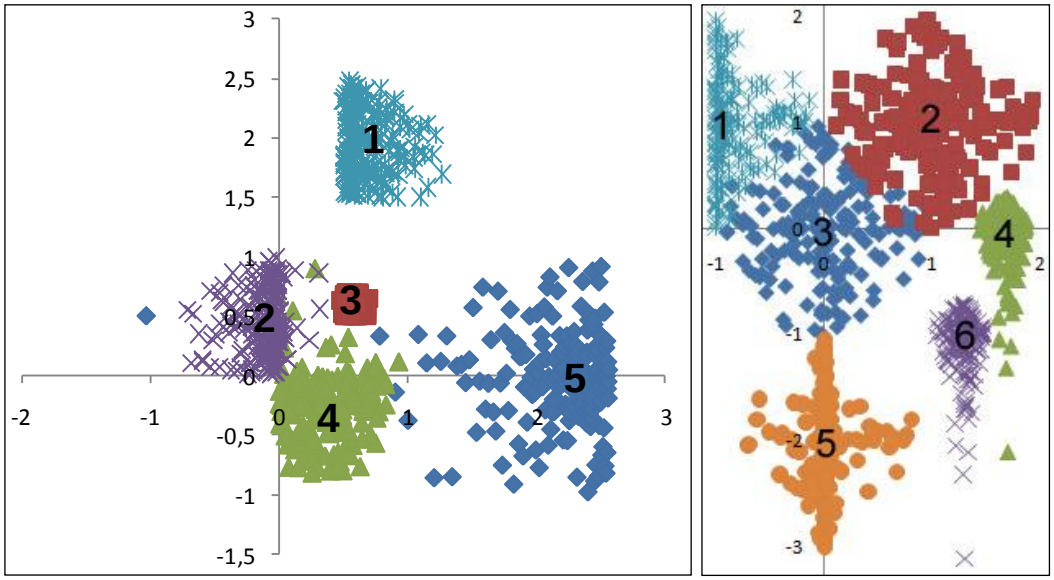

Fig 34. Juego de datos para cluster

A partir de los juegos de datos, se procedió a realizar una clasificación mediante la red neuronal mostrada en el apartado 5.1.3.1 obteniéndose las clasificaciones mostradas en la figura (Fig 35). Tal y como se puede apreciar, las clasificaciones son bastante similares al juego de datos establecido inicialmente en las figuras (Fig 34).

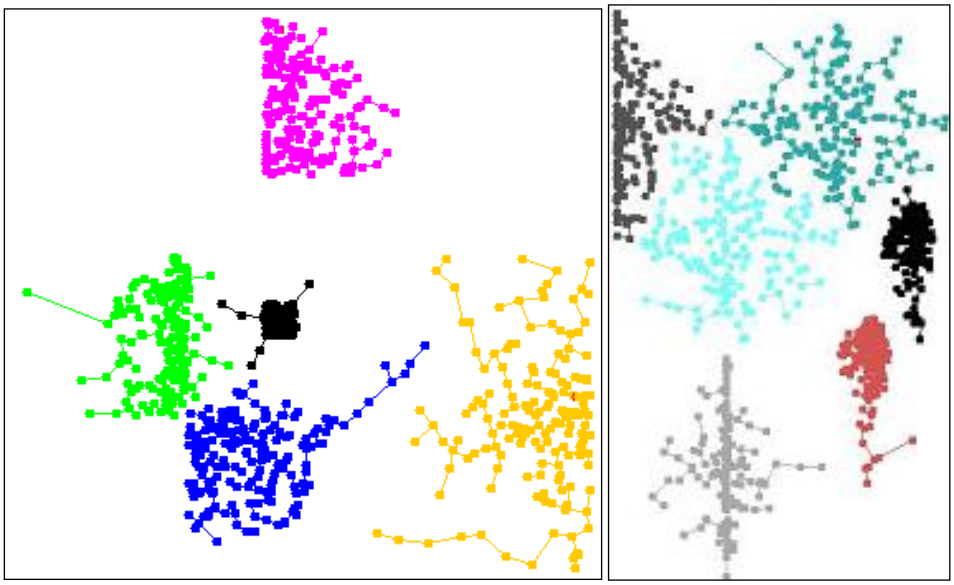

Fig 35. Clasificaciones obtenidas para la red SODTNN

Una vez comprobado el funcionamiento de la red, se procedió a testear el funcionamiento de otros algoritmos usados de modo habitual en bioinformática para el proceso de cluster. En las figuras (Fig 36) (Fig 37) se muestran los resultados obtenidos para los diferentes algoritmos. En el caso de estudio mostrado en la figura (Fig 36) PAM, FANNY, CLARA proporcionan resultados aceptables mientras que el resto no son capaces de detectar los cluster de modo correcto.

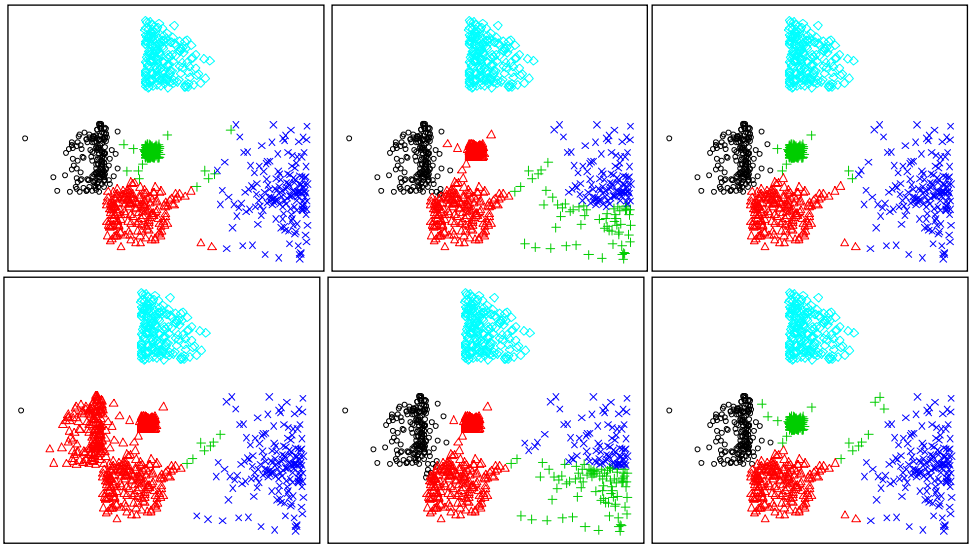

Fig 36. Secuencia de cluster generado mediante PAM, dendogramas, FANNY, AGNES, DIANNA y CLARA

En el caso de estudio mostrado en la figura (Fig 37) PAM y FANNY proporcionan resultados aceptables mientras que el resto no. Además, si analizamos la figura (Fig 34) que posee los datos originales y (Fig 35) que poseen los cluster generados por la red SODTNN, se puede ver que lo resultados obtenidos mediante PAM y FANNY no mejoran los resultados proporcionados por la red neuronal.

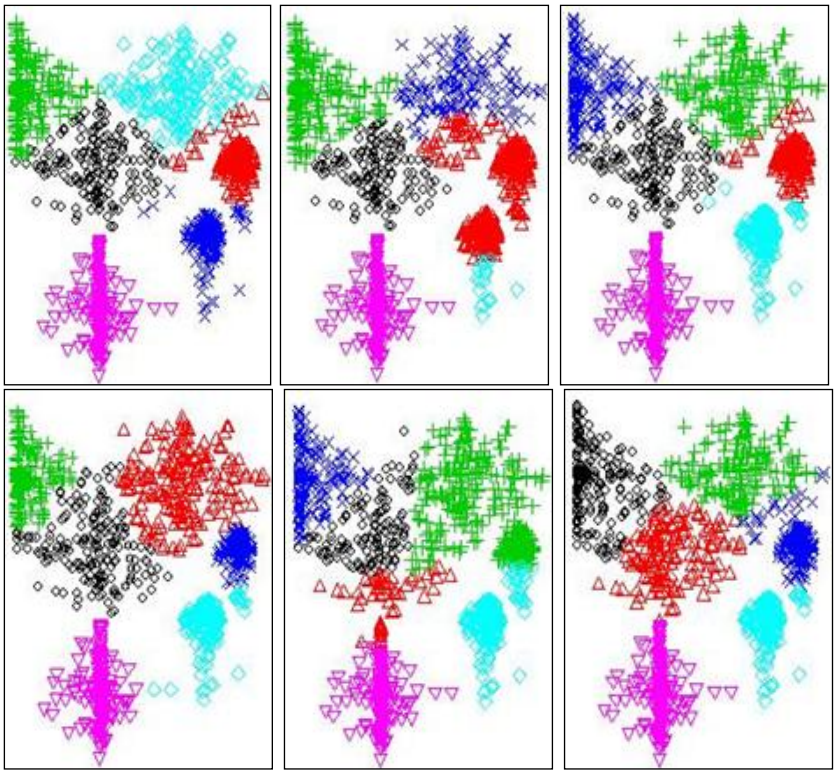

Fig 37. Secuencia de cluster generado mediante PAM, dendogramas, FANNY, AGNES, DIANNA y CLARA

Finalmente, se procedió a analizar la fase de extracción de conocimiento mediante la aplicación de las técnicas en el apartado (5.1.4). A partir de la información recogida, se procedió a realizar un estudio de la significabilidad de las sondas recuperadas mediante la representación de los datos en baja dimensionalidad mediante la aplicación de PCA y MDS para así determinar la idoneidad de las sondas recuperadas. Para llevar a cabo la extracción del conocimiento, en el sistema se hizo uso de las implementaciones proporcionadas por R y Weka [Weka, 2009] para así no tener que reimplementar los algoritmos en el sistema. En la figura (Fig 38) se muestra un fragmento de la información detallada del árbol de decisión obtenido. Cada una de las filas contiene la información de un nodo del árbol de decisión. El primer elemento representa el nombre de la sondas seguido de la condición, a continuación se muestra el número de elementos correctamente e incorrectamente clasificados seguido del tipo y finalmente la proporción de elementos para cada uno de los tipos de leucemia existentes. Finalmente, en caso de tratarse de un nodo hoja, se añade un * al final de la línea. En base a esta información se puede interpretar la siguiente figura.

```
1) root 212 123 CLL (0.047 0.23 0.42 0.1 0.2)
  2) 229103_at>=0.125 90   4 CLL (0.033 0 0.96 0.011 0)
    4) 225658_at< 0.625 5   2 ALL (0.6 0 0.2 0.2 0) *
    5) 225658_at>=0.625 85   0 CLL (0 0 1 0 0) *
  3) 229103_at< 0.125 122  73 AML (0.057 0.4 0.025 0.17 0.34)
    6) 228913_at>=0.375 44   7 AML (0.068 0.84 0.023 0.023 0.045) *
    7) 228913_at< 0.375 78  38 MDS (0.051 0.15 0.026 0.26 0.51)
     14) 225834_at< 0.625 30  12 CML (0.067 0.17 0.067 0.6 0.1)
       28) 223708_at>=0.375 8   3 AML (0.25 0.62 0 0 0.12) *
       29) 223708_at< 0.375 22   4 CML (0 0 0.091 0.82 0.091) *
     15) 225834_at>=0.625 48  11 MDS (0.042 0.15 0 0.042 0.77)
       30) 204749_at< 0.125 9   3 AML (0.22 0.67 0 0 0.11) *
       31) 204749_at>=0.125 39   3 MDS (0 0.026 0 0.051 0.92) *
```

Fig 38. Información detallada del árbol de decisión para los cinco tipos de leucemia

A partir de la figura (Fig 38) se puede transformar la información y mostrarla en forma de árbol de modo tal y como se puede ver en la figura (Fig 39). Ambas figuras almacenan la misma información pero la representación en forma de árbol simplifica el proceso de clasificación. Tal y como se puede ver, los nodos hojas contienen el grupo final clasificado y el número de individuos reales de cada tipo por el siguiente orden (ALL, AML, CLL, CML, MDS). Los nodos intermedios contienen la información de las condiciones de las sondas de modo que si es cierto, se selecciona la rama de la izquierda. Si se miran los nodos hojas, se puede ver que la tasa de acierto es bastante elevada, por lo que también se puede determinar que el proceso de filtrado funcionó correctamente. Si nos fijamos en los nodos intermedios, se observa que poseen valores comprendidos entre 0 y 1, esto es debido a que fue necesario discretizar los valores de las sondas en 5 niveles para poder aplicar CART y otras técnicas similares. Si no se realizaba la discretización el tiempo de cálculo era muy elevado.

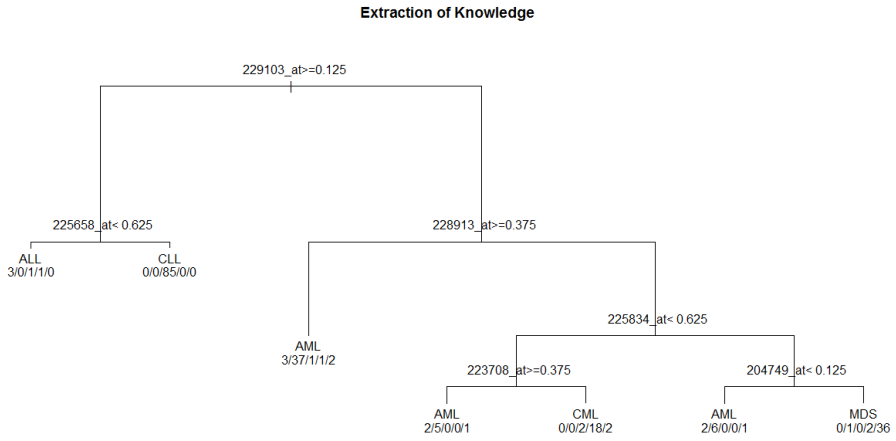

Fig 39. Árbol de decisión generado para los 5 tipos de leucemia

A partir de la información recuperada, se procedió a realizar un test para comprobar la veracidad de clasificación de las sondas proporcionadas por CART. Para ello, se procedió a representar las tres primeras sondas que proporcionaba el algoritmo para determinar si de verdad permitían clasificar a los pacientes en cuestión. En la figura (Fig 40) se muestran representadas las tres primeras sondas recuperadas mediante CART, tal y como se puede ver en la figura (Fig 40a) se pueden separar los individuos de tipo ALL y en el figura (Fig 40b) los individuos con leucemia CLL.

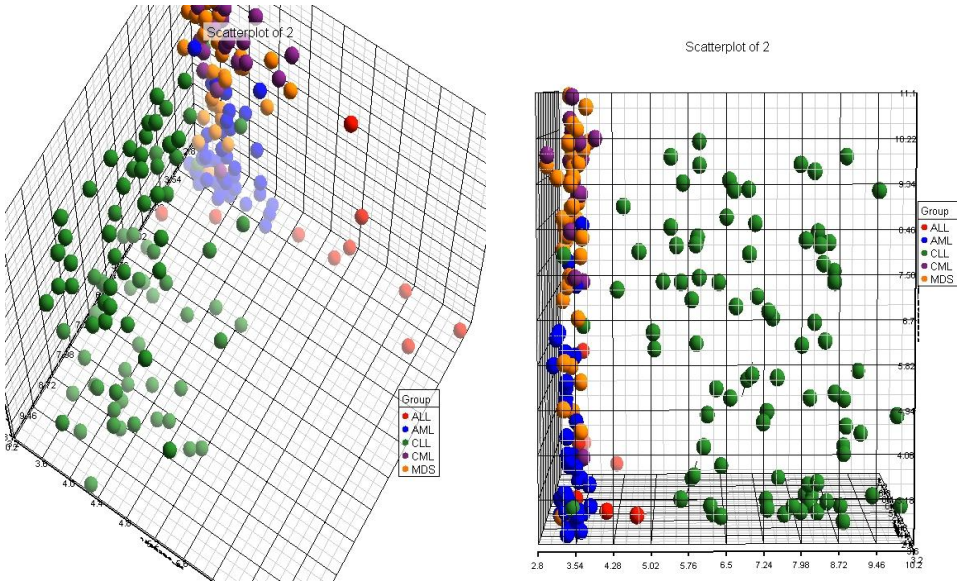

Fig 40. Representación en 3D de las tres primeras sondas recuperadas mediante CART

Además, si seleccionamos las sondas que proporcionaban más ganancia para las leucemias de tipo CLL mediante el algoritmo CART, se puede ver que permiten separar a estos individuos del resto de manera sencilla. Así, en los diagrama de cajas mostrados en la figura (Fig 41), se muestra la representación para cada una de las sondas recuperadas como relevantes para esta enfermedad. Tal y como se puede ver, la mayor parte de los individuos permanecen separados del resto de los tipos.

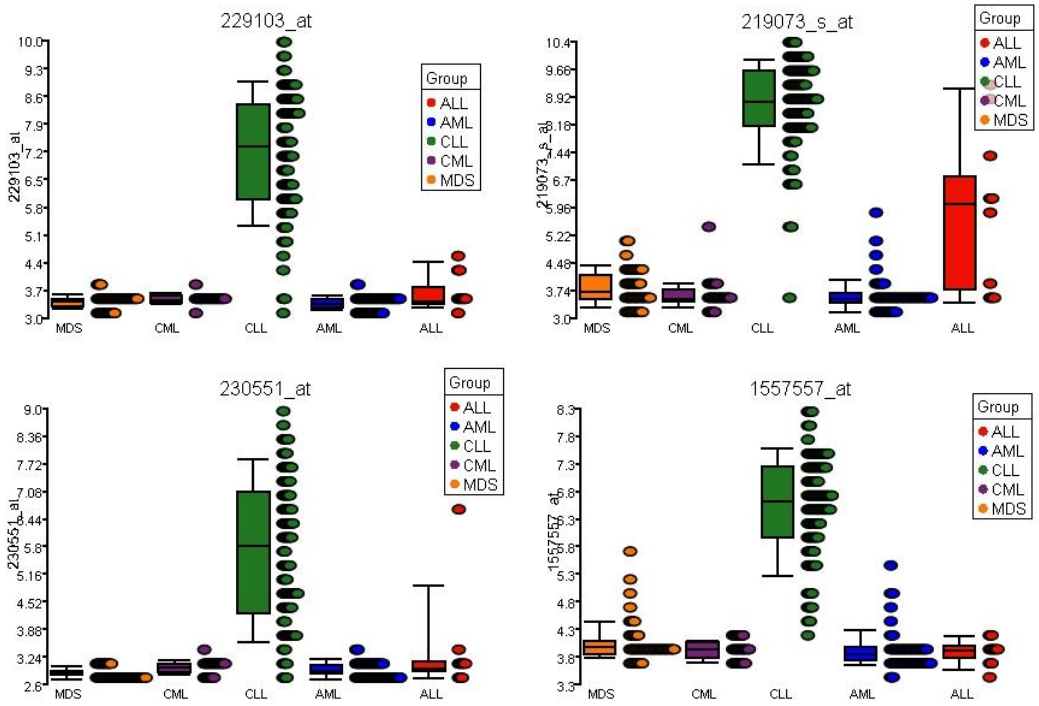

Fig 41. Diagrama de cajas para las sondas relevantes en las leucemias CLL

Si se repite el proceso para los individuos con leucemias de tipo CLL, también se pueden obtener resultados similares. Así, si se procede a aplicar J48 en lugar de CART, se puede determinar las sondas que caracterizan a la clase marcada en azul en la figura (Fig 42). En la diagonal de la figura, se tienen las diferentes sondas y la representación de sus gráficos de intensidades. El resto de las figuras compara las sondas dos a dos así, por ejemplo, la primera figura de la segunda fila compara la primera y segunda sonda de la diagonal. Tal y como se puede ver los individuos de la clase azul se separan del resto.

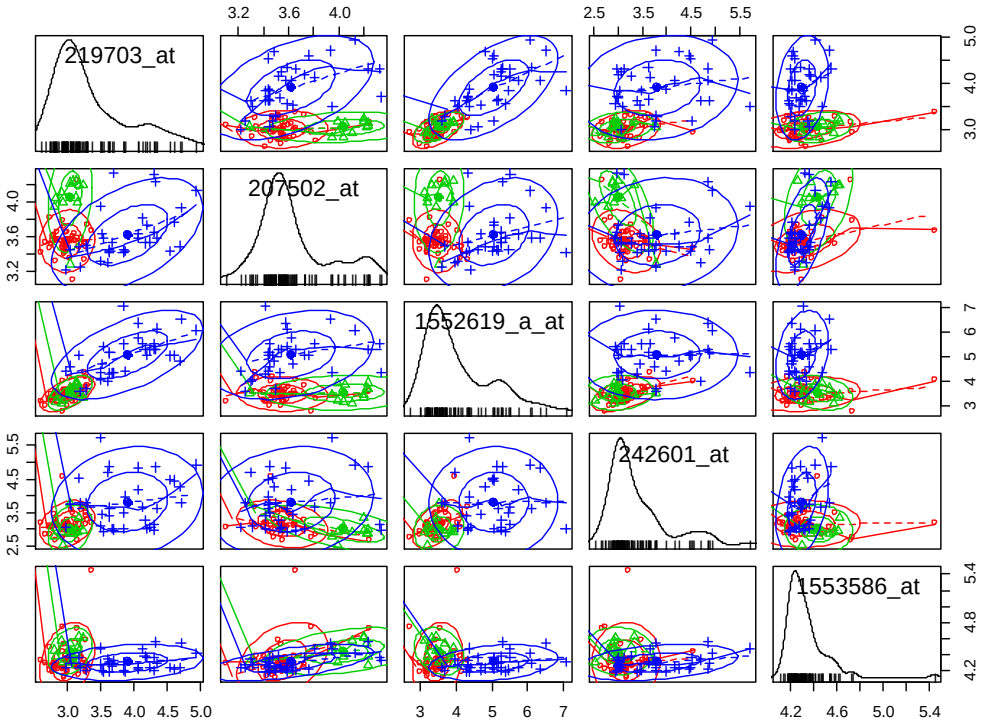

Fig 42. Sondas relevantes para el subtipo de CLL resaltado en azul

Finalmente, se realiza una representación de las tres sondas recuperadas por J48, podemos ver que los tres subtipos existentes se separan unos de otros. Para ello, se ha realizado la representación 3D mostrada en la figura (Fig 43). En la figura (Fig 43) se ha representado en colores diferentes cada uno de los tipos, además, se ha representado un plano de regresión que aproxima a los puntos. En la figura (Fig 44) se muestra la misma información pero se han añadido elipsoides que recogen al 50% de los individuos para así facilitar la separación de los mismos. Tal y como se puede ver los elipsoides permanecen separados en el espacio.

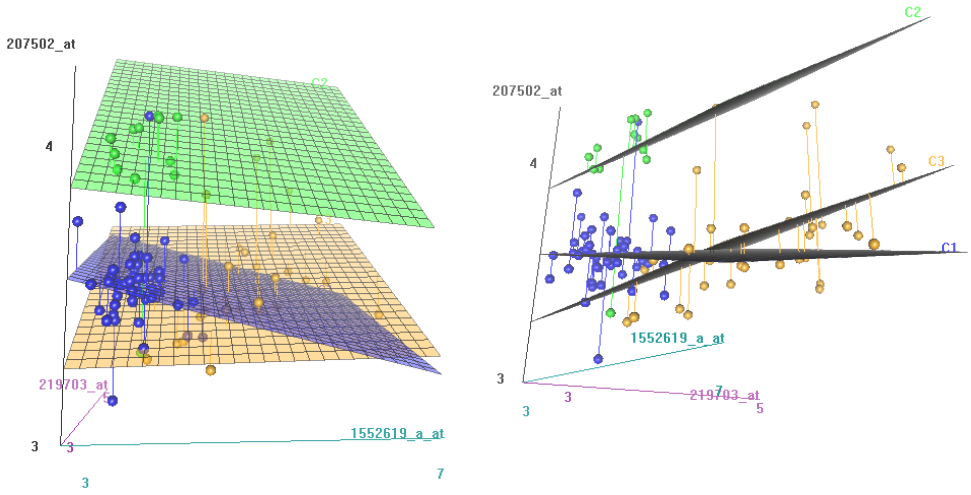

Fig 43. Representación 3D para los subtipos de CLL

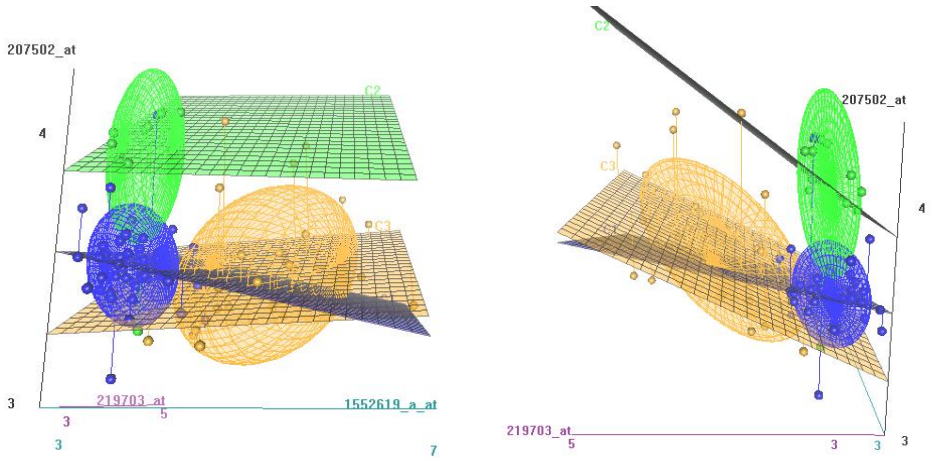

Fig 44. Representación 3D de los subtipos de CLL

Para comprobar que las representaciones obtenidas en 3D tenían cierta validez, se procedió a realizar una reducción de la dimensionalidad aplicando MDS, tal y como se puede ver en la figura (Fig 45) los individuos poseen una distribución similar a la mostrada en la figura (Fig 44) por lo que se puede suponer que la información obtenida es de relevancia.

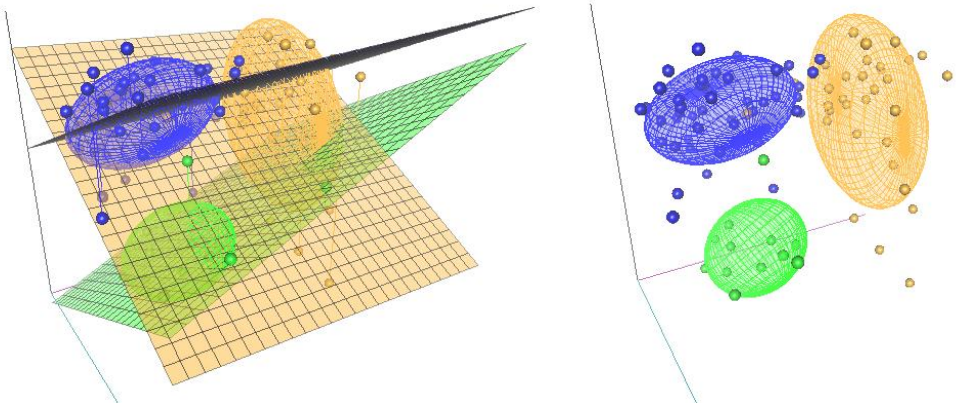

Fig 45. Representación en baja dimensionalidad aplicando MDS

Finalmente, se procedió a realizar un análisis similar aplicando RIPPER y analizando la información obtenida. Para ello, nos centramos en el caso de estudio de los subtipos de CLL. Posteriormente, se muestra un análisis más en profundidad para las diferentes técnicas de extracción de conocimiento en base a otro caso de estudio diferente.

Partiendo del ejemplo de las leucemias de tipo CLL, si procedemos a realizar una representación de los datos, de las 541 sondas recuperadas se puede obtener un resultado similar al mostrado en la figura (Fig 46). En la parte superior de la figura, se muestran los 212 individuos, uno por cada barra, cada una de las barras se encuentra subdivida en tantas partes como sondas, en este caso 541. La amplitud de cada segmento es proporcional al valor de la sonda. En la parte superior, se muestra la información del grupo real al que pertenece el paciente y el estimado, en este caso aparece la misma información puesto que no se ha realizado proceso de clasificación previo. Finalmente, en la parte de la abajo, se tiene unas coordenadas paralelas y una tabla con las sondas. Las coordenadas paralelas poseen una coordenada por cada uno de las sondas recuperadas y una línea por cada uno de los individuos. El color de las líneas varía en función de pertenencia de los individuos a las clases. Tal y como se puede ver, es muy complicado poder extraer información relevante. En la figura (Fig 48) se han añadido una serie de etiquetas la imagen para facilitar la interpretación de la misma por lo que se recomienda su visualización en caso de problemas con la interpretación.

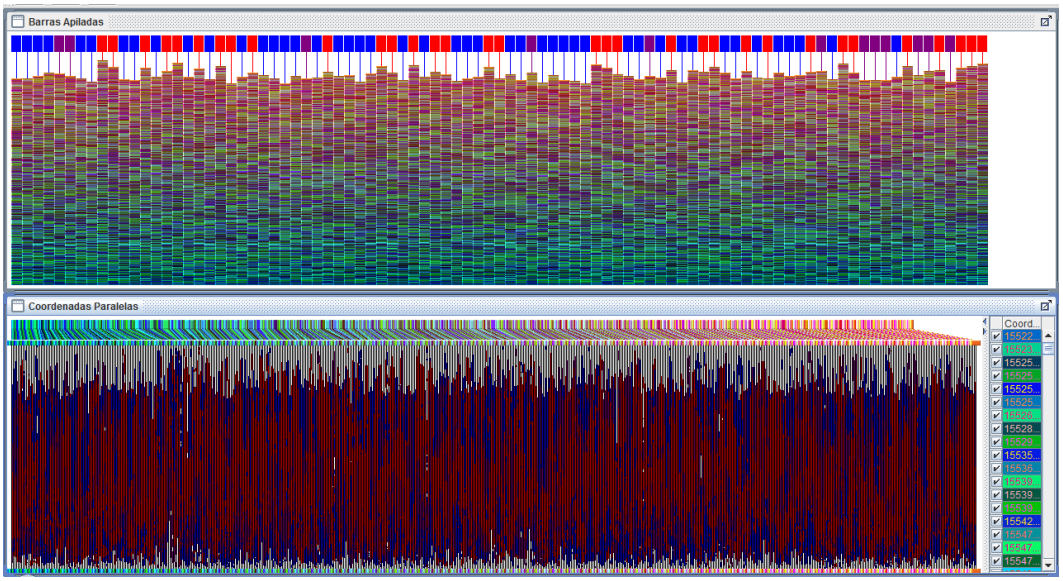

Fig 46. Representación de las sondas obtenidas tras el filtrado y la clasificación

Si aplicamos RIPPER a partir de la información de los grupos, se obtienen las reglas mostradas en la figura (Fig 47), las reglas son bastante auto explicativas por lo que no es necesario realizar explicación alguna.

(209083_at <= 0.25) and (1552280_at <= 0) => Class =C2 (10.0/1.0)
(231592_at >= 1) => Class=C2 (4.0/1.0)
(203213_at >= 0.75) => Class =C3 (29.0/0.0)
(1552619_a_at >= 0.5) => Class =C3 (2.0/0.0)
=> Class =C1 (46.0/1.0)

Fig 47. Reglas de decisión obtenidas mediante RIPPER

En la figura (Fig 48), se representa la información gráficamente de las sondas recuperadas mediante RIPPER. Esta representación facilita la interpretación de las reglas y la selección de casos que no cumplen unos determinados patrones. Los valores mostrados en las figuras son los valores reales sin discretizar, ya que antes de aplicar RIPPER al igual que ocurrió con CART y J48 fue necesario discretizar los valores para que el algoritmo fuera eficiente. En la parte superior, se pueden ver los individuos mal clasificados, para ello, sólo hay que mirar los elementos para los que no coincide el color en la parte superior. Al marcar los individuos del subtipo C3 (mostrados en rojo) se seleccionan de modo automático los rangos de las coordenadas paralelas asociados a las mismas, se dibujan las líneas con un patrón más grueso y se desactivan los individuos que no se encuentran en dicho rango. Así, se puede ver de modo sencillo los individuos que no se clasifican de modo correcto mediante las reglas de decisión creadas.

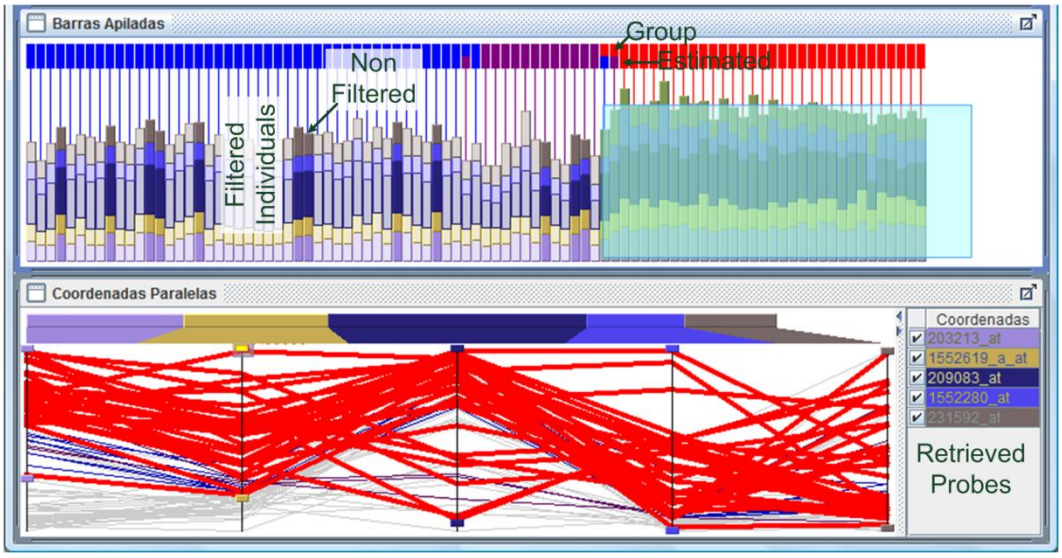

Fig 48. Clasificación de individuos con un el subtipo C3.

Una vez que los márgenes se han establecido, se puede ver que permanecen activos individuos de la clase 2 (violeta) y clase 1 (azul) por lo que si se mira las coordenadas se puede observar que la primera de ellas contiene a un individuo marcado en rojo que se encuentra seguido de otros marcados en azul y violeta por lo que hemos encontrado al elemento problemático. Así, si se modifica el control asociado a la coordenada de modo que se elimine dicho elemento, se puede ver que la clasificación para los individuos del tipo 3 se realiza de modo correcto. Así moviendo el control, se obtiene lo mostrado en la figura (Fig 49). Tal y como se puede ver, los elementos mal predichos permanecen desactivados y el resto de la clase resaltados.

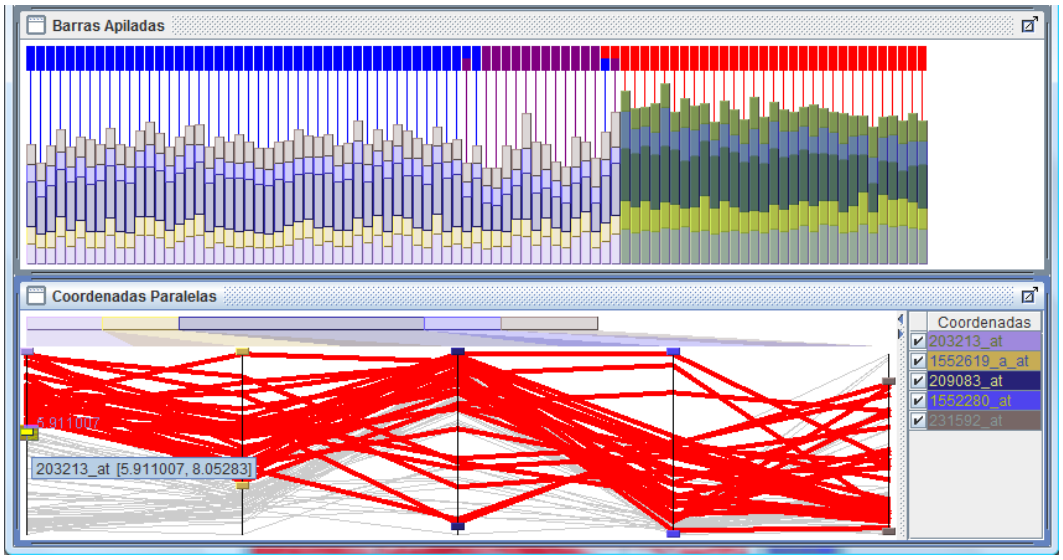

Fig 49. Clasificación de individuos con un el subtipo C3 con información interpretada

En la figura (Fig 50a) se puede ver la misma clasificación para los individuos de la clase C1 y en la (Fig 50b) para la clase C2. Los resultados para la clase 1 se pueden interpretar del mismo modo que los obtenidos para en el caso anterior mostrado en la figura (Fig 49). En la figura (Fig 50b), se procedió a realizar una selección de las coordenadas para desactivar los individuos resaltados, tal y como se puede ver, sólo un individuo de la clase 2 se desactiva debido a valores altos en la sonda 15552280_at.

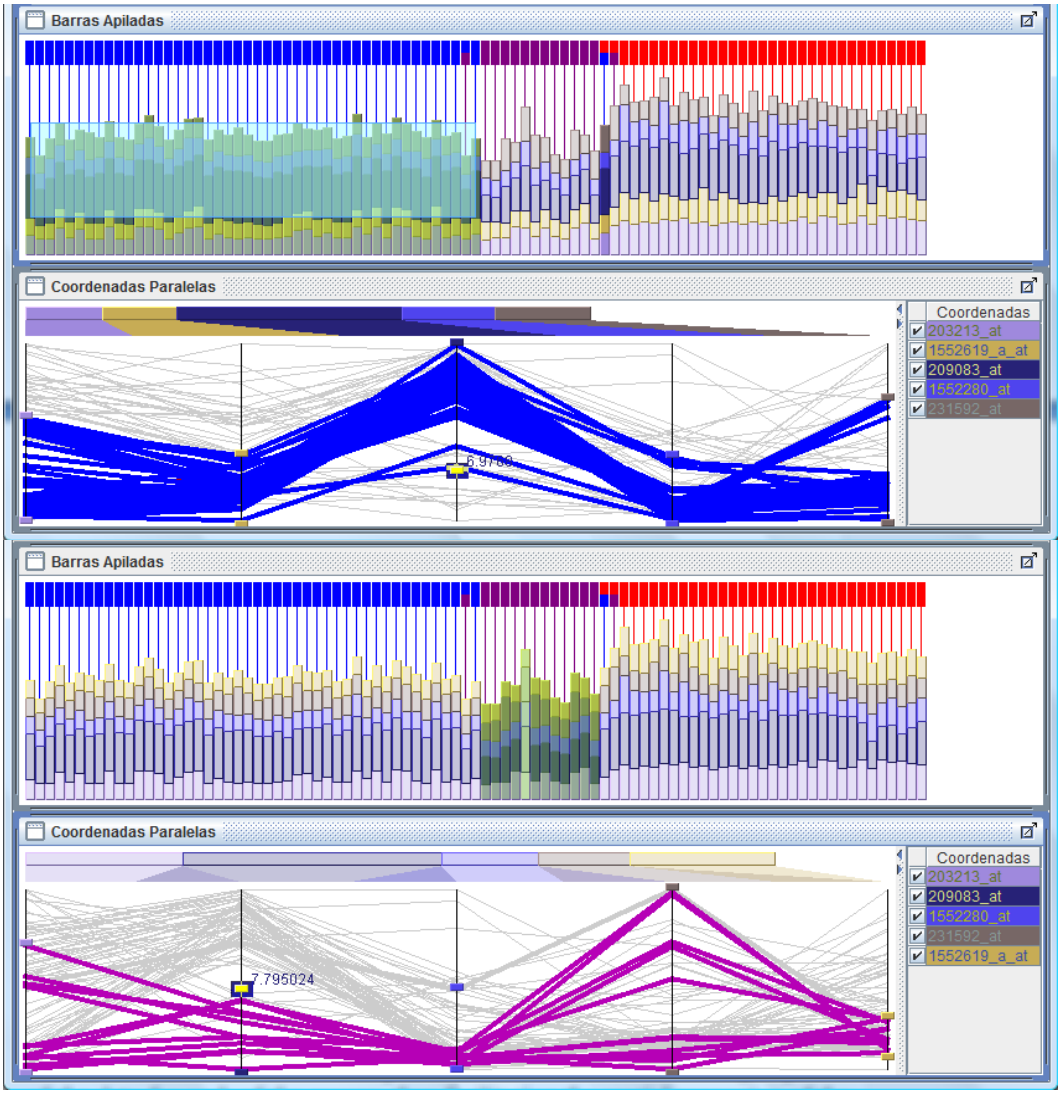

Fig 50. Clasificación de individuos con un el subtipo C1 y C2 respectivamente

Finalmente, en la figura (Fig 51), se muestra el resultado obtenido aplicando CART para los 5 tipos de leucemia. La representación es similar a lo mostrado anteriormente, se han añadido una serie de notas explicativas para facilitar la interpretación de los resultados.

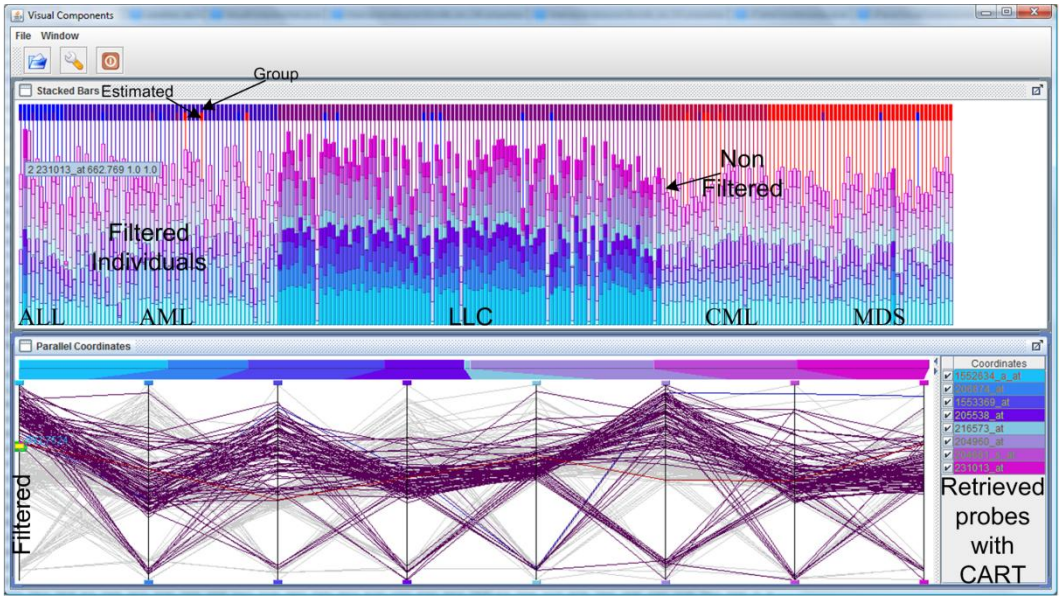

Fig 51. Representación de las reglas obtenidas mediante CART

Finalmente, para estudiar un poco las diferentes técnicas de extracción de conocimiento existente, se recurrió a R para realizar un análisis más exhaustivo procediéndose a usar un conjunto de datos de microarrays de CGH del que se disponían diferente conjunto de patologías. Se realizaron varios estudios, uno de ellos fue el tipo de mutación asociado al cáncer GIST. Se disponía de 3 mutaciones en 43 pacientes y se aplicaron los siguientes algoritmos con el número de aciertos indicados:

- Redes bayesianas: 24
- Naive Bayes retocado: 26
- Naive Bayes: 21
- AdaBoostM1: 23
- Bagging: 23
- Decision Stump: 26
- J48: 21
- K vecinos más cercanos: 18
- RIPPER: 25
- LMT: 22
- Logistic: 20
- LogitBoost: 26
- MultiBoostAB: 24
- OneR: 29
- SMO: 25
- Stacking: 24

Los algoritmos proporcionan resultados más o menos similares y si en unos casos unos mejoraban los resultados de otros en otros casos de estudio los resultados eran los contrarios por lo que de modo general se podría decir que proporcionaban resultados bastante homogéneos. Para llevar a cabo los análisis, se procedía a realizar la extracción del conocimiento a partir de todos los elementos a excepción de uno que posteriormente se clasificaba. Este proceso se repetía iterativamente con cada uno de los elementos.

5.2.3 Análisis de sistema de planificación

Finalmente, una vez analizado el proceso de análisis usado en el sistema, sólo nos queda por analizar el proceso de planificación automática. El análisis previo es necesario para determinar que los algoritmos creados y seleccionados son adecuados para llevar a cabo el proceso de análisis de datos. En este apartado, se verán los mismos casos de estudio vistos anteriormente.

El proceso llevado a cabo para el análisis de la eficiencia del sistema de planificación es muy sencillo y se limita a llevar a cabo una serie de planificaciones manuales para posteriormente llevar a cabo una planificación automática. A partir del plan generado, se estudia la eficiencia teórica del nuevo plan y se estima si es adecuado o no. Por tanto, el análisis de este apartado se centrará más en el estudio de las diferentes fases del ciclo de razonamiento de un planificador basado en casos y los resultados que se van obteniendo en cada una de ellas.

Según el caso de estudio planteado, la única fase en la que representa cierta relevancia el orden de ejecución de las acciones y las acciones llevadas a cabo es la fase de filtrado presente en el agente de filtrado. Las demás fases, carecen de interés analítico puesto que la selección del servicio más adecuado se limita a seleccionar el de mayor eficiencia en función de la medida de eficiencia definida.

La medida de eficiencia definida para el agente de filtrado está definida por la expresión (18)

$$e(p) = \frac{s}{N} + \frac{i'}{I} \qquad (18)$$

Donde:

- S es el número de sondas
- N el número inicial de sondas
- I’ el número de individuos mal clasificados
- I el número total de individuos mal clasificados en la fase de agrupación o de extracción de conocimiento según se defina.

En base a la medida definida en (18) y el algoritmo establecido en el apartado (4.1.1) se procede a realizar la planificación del sistema a partir de la información de los planes recuperados de la memoria. Los planes recuperados según el procedimiento indicado en el apartado (4.1.1.1) se muestran en la tabla (Tabla 13). La primera columna contiene el identificador del plan, las cuatro siguientes contienen los diferentes servicios disponibles para llevar a cabo el filtrado. Al final, se tiene la medida de eficiencia calculada en base a (18) y finalmente la clase asociada a cada uno de los planes (0 plan no eficiente y 1 plan eficiente). Los números de las tablas representan el orden de ejecución de los servicios, en caso de que la casilla aparezca en blanco quiere decir que el servicio no se ejecuta para ese plan. Así, por ejemplo el plan 1 está formado por la ejecución de los servicios $(s_4 \circ s_3 \circ s_2 \circ s_1)(e_0)$ sobre el estado inicial e_0, el plan 2 estaría formado por la ejecución de los servicios $(s_4 \circ s_2 \circ s_1)(e_0)$.

Plan	Variabilidad (s_1)	Uniforme (s_2)	Correlación (s_3)	Puntos de corte(s_4)	Eficiencia	Clase
p_1	1	2	3	4	0.1301	1
p_2	1	2		3	0.1482	1
p_3	1				0.1972	0
p_4		1		2	0.1462	1
p_5			1		0.2036	0
p_6		1			0.1862	0
p_7				1	0.1932	0
p_8			1	2	0.186	0

Tabla 13. Casos Recuperados

A partir de los datos, el agente de filtrado según la tabla (Tabla 13) selecciona el flujo de ejecución de los servicios que mejor se adapta al caso de estudio. Los resultados obtenidos para las diferentes conexiones de los nodos se muestran en la tabla (Tabla 14). Los subíndices mostrados en la tabla S_{ij} se corresponden con la conexión entre los servicios de modo que si se tienen S_{01} quiere decir que primero se ejecuta el servicio 0 y posteriormente el 1. Los índices se corresponden con los valores mostrados en la cabecera de la tabla (Tabla 13).

Arco	Peso
S_{01}	0,095874882
S_{12}	0,087847732
S_{03}	0
S_{04}	0,022592519
S_{12}	0,138575444
S_{1f}	0,001969253
S_{23}	0,093643439
S_{24}	0,047136312
S_{2f}	0,022649751
S_{34}	0,040345464

S_{3f}	0,077693557
S_{4f}	0,032507173

Tabla 14. Pesos de las conexiones

Siguiendo la información mostrada en la tabla (Tabla 14) se construye el grafo de planes. La información del grafo se usa para generar el nuevo plan construyendo la ruta máxima que une el nodo inicial y final. La figura (Fig 52) muestra el grafo y la ruta final seguida resaltada en negrita. Tal y como se puede ver, el plan seleccionado es el plan S_{01}, S_{12}, S_{23}, S_{3f} este plan no coincidía con ninguno de los planes previos llevados a cabo. Al aplicar dicho plan, la eficiencia final obtenida fue de 0,1277 lo que mejora ligeramente la eficiencia del plan p_1 que era el más eficiente.

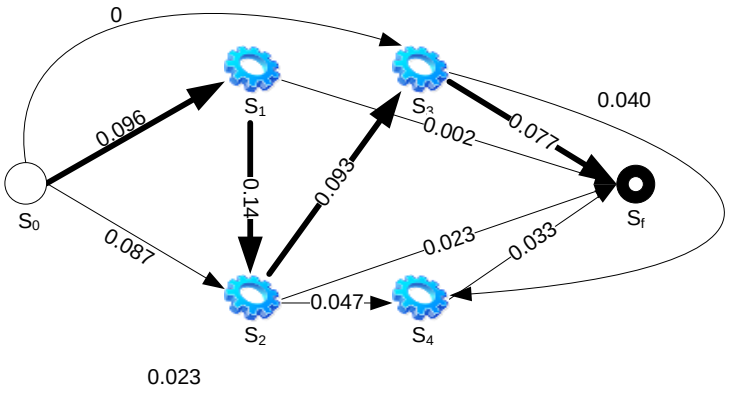

Fig 52. Grafo dirigido de planes

Capítulo VI

Conclusiones

VNiVERSiDAD
D SALAMANCA

En este trabajo se ha presentado un modelo de planificación automática enfocado al análisis de datos de arrays de expresión. El aumento considerable del volumen de información que proporcionan los chips de microarrys con respecto a otros problemas, requiere de procedimientos que faciliten y automaticen tareas de análisis y extracción de información de relevancia.

El modelo de planificación definido, se integra en una arquitectura multiagente genérica orientada al análisis de datos integrada con la arquitectura SOA. La integración del procesamiento mediante servicios web facilita la extensibilidad y adaptabilidad del modelo permitiendo incorporar nuevas técnicas de análisis de datos. El corazón de la arquitectura son los agentes CBR-BDI y agentes con capacidad de planificación basados en casos bajo el modelo BDI, agentes deliberativos que incorporan mecanismos de razonamiento basado en casos y de planificación basada en casos. Estos agentes dotan a la arquitectura de una gran capacidad de adaptación al contexto y facilitan el proceso de toma de decisiones. Se trata de elementos de computación que trabajan de forma distribuida. En este sentido y dada la capacidad de comunicación existente en los sistemas multiagente, es posible hablar de una computación ubicua. Por otro lado, la arquitectura trata de facilitar la comunicación ubicua, incorporando agentes con características especiales de comunicación, que actúan como mediadores entre diferentes plataformas de agentes o arquitecturas orientadas a servicios. Por último, pero no menos importante, la arquitectura dispone de agentes encargados de actuar como interfaces entre usuarios y el sistema. Este hecho facilita la interacción con los usuarios y la personalización en los servicios ofrecidos.

En este trabajo, se han estudiado diferentes alternativas de planificación automática y su integración con diferentes técnicas para realizar análisis de expresión. Se ha realizado un estudio de las alternativas existentes para llevar a cabo análisis de expresión haciendo especial hincapié en las técnicas estadísticas y de minerías de datos existentes para el análisis de microarrays y las existentes que se pueden adaptar para su uso. El conjunto de las posibles técnicas a aplicar es bastante amplio por lo que en este trabajo se presenta un modelo de planificación que permite seleccionar el flujo que mejor se adapta al problema en función de los análisis previos llevados a cabo. Para llevar a cabo dicha tarea, el modelo de planificación integra un sistema de planificación basado en el modelo de planificación basado en casos bajo el modelo BDI. El modelo de planificación basado en casos BDI integra a partir de su representación simbólica del entorno definida en la arquitectura BDI un motor de razonamiento basado en los CBR. EL motor de razonamiento se caracteriza por la definición de un ciclo de razonamiento formado por una serie de etapas que permiten recuperar planes pasados y adaptarlos para resolver problemas actuales. Este modelo híbrido ha permitido crear un modelo de planificación automático que permite a partir de los planes recuperados, definir una nueva planificación estimada como más eficiente en función de métricas de eficiencia definidas.

Para poder llevar a cabo la planificación automática, es fundamental el modelo simbólico BDI que permite definir de modo simbólico los planes llevados a cabo

mediante una base de creencias. Los planes no son más que un conjunto ordenado de acciones que se aplican sobre un estado inicial que viene establecido por un conjunto de variables que define el entorno. En este caso, las acciones son los algoritmos a aplicar y las variables que definen el entorno son el conjunto de datos obtenidos a partir de los microarrays. Mediante esta definición formal de los planes y haciendo uso del motor de razonamiento basado en planes se ha conseguido generar planes de mayor eficiencia mediante la incorporación de técnicas estadísticas, redes bayesianas y teoría de grafos en las diferentes etapas del CBR. Por tanto, la arquitectura propuesta, permite por un lado descomponer el problema en agentes que colaboran entre sí para la obtención de un resultado final, permite separar la funcionalidad de la lógica de negocio mediante la incorporación de servicios web y finalmente permite aprender de las experiencias pasadas mediante la planificación llevada a cabo con el modelo de planificación basado en planes con el modelo BDI. Por tanto, la arquitectura propuesta facilita la incorporación de nuevos agentes que utilicen diferentes técnicas de modelado y estrategias de aprendizaje de forma que se puedan realizar experimentos adicionales con nuevas técnicas y puedan ser comparados con los resultados iniciales se han presentado en este documento. De la misma forma los agentes CBR-BDI y los agentes con capacidad de planificación basados en el modelo BDI facilitan la incorporación de nuevos algoritmos a cualquiera de las etapas del ciclo CBR mediante la inclusión de nuevos servicios web.

Finalmente, a parte de la investigación realizada en los modelos de planificación, se ha realizado un estudio y desarrollo de diferentes técnicas usadas para poder llevar a cabo un análisis de expresión, para ello, se han analizado diferentes técnicas que han permitido seleccionar y crear nuevas para poder llevar a cabo un análisis de expresión de modo eficiente. Además, el estudio de estas técnicas han contribuido a la elaboración del mecanismo de planificación automático puesto que parte de él, se fundamenta en la incorporación de redes bayesianas.

6.1 Contribuciones de la Investigación

Este trabajo hace algunas nuevas contribuciones en el ámbito de la Teoría de Agentes, análisis de microarrays y mecanismos de planificación. Se ha propuesto un modelo de planificación integrado en una arquitectura multiagente que incorpora diversas técnicas para llevar a cabo análisis de datos. A continuación se resumen las aportaciones realizadas:

1. Marco para el análisis de datos de microarrays
   Se ha realizado un estudio de las técnicas existentes para el análisis de datos y se ha contribuido con la aportación de nuevas en los casos en los que se estimaba necesario.

   a. Se han desarrollado técnicas para realizar filtrados de la información mediante la incorporación de test estadísticos que permiten filtrar en gran medida la información no útil para la realización de los análisis.
   b. Se ha desarrollado una técnica de cluster que integra tanto las redes neuronales como los dendogramas para realizar procesos de revisión de cluster.
   c. Se ha elaborado un extenso estudio de alternativas para la extracción del conocimiento ajustando los datos para optimizar el rendimiento y proporcionando representaciones para facilitar la interpretación de los resultados.
2. Marco de la Inteligencia artificial distribuida
   Se ha realizado un estudio de las técnicas existentes para la elaboración de sistemas con capacidad de aprendizaje y colaboración. Posteriormente, estas técnicas se han aplicado para elaborar mecanismos de planificación automática.
3. Marco para la planificación automática
   Se ha desarrollado un sistema de planificación automático que permite estimar la eficiencia de planes para el cálculo ordenado de acciones en planes que permitan mejorar los resultados obtenidos anteriormente.
   Esta planificación automática permite realizar flujos de trabajo de modo automático y extraer relaciones de dependencia en efectividad en base a la secuencia de acciones que son difícilmente extraíbles manualmente. El modelo propuesto incorpora conceptos como el modelo de planificación basado en casos, modelo BDI, las redes bayesianas y teoría de grafos.
4. Marco de adaptación
   Se ha creado un sistema de adaptación de planes para los agentes, normalmente esta fase suele incorporar técnicas de caja negra como las redes neuronales que son bastante difícil de interpretar. En esta investigación se ha realizado una reutilización de los planes mediante la incorporación de las redes bayesianas y teoría de grafos que facilita el proceso de revisión y explicación de los resultados obtenidos.
5. Diseño de agentes inteligentes
   Mediante la incorporación del los mecanismos de planificación automática comentados anteriormente.
6. Propuesta de una arquitectura multiagente aplicable a entornos inteligentes
   Se ha propuesto una arquitectura multiagente que facilita el análisis de datos de arrays de expresión que integra las arquitecturas SOA.
7. Aplicación de la arquitectura
   Se ha aplicado la arquitectura a casos de estudio reales con los que se han validado tantos las técnicas empleadas como el proceso de planificación automática.

Finalmente, es posible concluir que se ha realizado un trabajo de investigación, en el que se ha estudiado el estado del arte de la teoría de agentes, sistemas multiagente, y técnicas aplicadas en los análisis e expresión. Se han estudiado los mecanismos deliberativos aplicables a los agentes, las herramientas para la construcción de sistemas multiagente y de entornos inteligentes, y un problema real referente a la importancia de la creación de entornos inteligentes en el análisis de datos de microarrays. A partir de estos estudios, se ha procedido a construir una arquitectura multiagente con capacidad de planificación automática mediante la incorporación del modelo de planificación basado en casos y el modelo BDI.

6.2 Trabajo futuro

Este trabajo deja abiertas una gran cantidad de puertas para un trabajo futuro. A continuación se muestran algunas de ellas:

1. Comprobar y validar los sistemas multiagente de este trabajo dentro de los entornos estudiados, de tal forma que se puedan evaluar los modelos propuestos en términos de tiempo, facilidad y calidad de análisis y diseño, tiempo de cálculo de respuestas, calidad de las respuestas, etc.
2. Negociación
   Como ya se ha indicado a lo largo de este trabajo, se ha decidido enfocar las interacciones hacia el ámbito dentro del entorno. Sin embargo los distintos sistemas multiagente deben cooperar entre sí para obtener comunicación y computación ubicua. Sería deseable que se produjese una interacción entre sistemas multiagente de diferentes características y, a ser posible, una negociación de parámetros entre ellos.
3. Seguridad
   Se podrían dotar de seguridad las comunicaciones realizadas dentro de nuestro sistema multiagente, ya que los datos intercambiados pueden considerarse importantes y debe garantizarse privacidad.
4. Aumentar la cantidad de algoritmos
   Se considera adecuado aumentar la cantidad de algoritmos disponibles para el desarrollador en cada una de las etapas de los análisis de expresión.
5. Resolución de nuevos problemas prácticos
   Para comprobar la validez de la arquitectura propuesta, sería deseable aplicarla a nuevos problemas prácticos. De esta forma sería posible comprobar si se adapta de forma adecuada a la resolución de problemas de características distintas, o bien si se ha construido una arquitectura muy restringida al problema concreto que se ha estudiado durante la realización de este trabajo.
6. Extender el modelo a nuevos tipos de análisis

Incorporar diferentes tipos de microrrays para analizar los datos, además de los arrays de expresión existen otros tipos de arrays como los CGH que permiten determinar variaciones en regiones cromosómicas. La incorporación de estas características junto con la capacidad de negociación podría dar lugar a un sistema más abierto que el actual.

7. Definir nuevos mecanismos de planificación automática
   En base a la planificación seguida se pueden definir nuevos sistemas estimación de la eficiencia de los algoritmos. Así, se está llevando a cabo estudios de sustituir las redes bayesianas por funciones gain usadas en los árboles de decisión para estimar la influencia de acciones en los planes.

Capítulo VII

Research Overview

VNiVERSiDAD
D SALAMANCA

Introduction

Cancer diagnosis is a field requiring novel automated solutions and tools, able to facilitate the early detection, even prediction, of cancerous patterns. 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 [Quackenbush, 2001]. Currently, there are several kinds of microarrays such as CGH arrays [Shinawi, Cheung, 2008], expression arrays [Affymetrix]. Expression arrays contain information about certain thousands of genes in a patient's samples, the genes are represented by means of a series of probes that are composed of oligonucleotids. The number of oligonucleotids is up to one million per sample. . The large amount of data requiring analysis makes it necessary to use data mining techniques in order to reduce processing time. 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.

Biomedical systems have become very important in recent years, automating tedious tasks and providing novel decision support systems in fields such as cancer diagnosis. Many of these systems are based on the use of artificial intelligence and attempt to imitate human behavior; but in general, they focus on very specific tasks such as providing decision support in medical environments [Chua *et al.*, 2008] [François *et al.*, 1992]. This research presents an entirely new perspective that focuses on the concept of Intelligent Biomedical Organizations (IBO), proposing an architecture capable of modelling biomedical organizations through multi-agent systems. IBO consists of a virtual organization in which each of the roles that are carried out in biomedical organizations is modelled by agents, and the collaboration between the roles is represented as interactions between intelligent agents. Furthermore, IBO is capable of automatic reorganization, and as such is capable of evolving and adapting to social changes that take place within the organization. Initially, IBO was conceived as a dynamic multi-agent architecture and was oriented to KDD (Knowledge Discovery in Databases) as applied to biomedical databases, thus providing an innovative decision support system.

There are different approaches for decision support systems, including myGrid [Stevens *et al.*, 2004] [Vittorini *et al.*, 2008], Gene-CBR [Riverola *et al.*, 2006], that base their functionality on the creation of web services that are implemented according to OGSA (Open Grid Services Architecture) [Foster, 2002]. The main disadvantage, however, is that the user must be the responsible for creating the sequence of actions

that resolve a specific problems. These systems provide methods for solving complex problems in a distributed way through SOA [Erl, 2005] and Grid architectures, but lack adaptation capabilities. However there do exist new research lines focused on reasoning mechanisms with a high capacity for learning and adaptation, notably case-based Reasoning (CBR) systems [Kolodner, 1993], which solve new problems by taking into account knowledge obtained in previous experiences [Kolodner, 1993] [Leake, Kendall-Morwick, 2008] and integrating within agents and multi-agent systems. The disadvantage of CBR-based decision support systems for classification in medical databases is the high dimensionality of the data and its corresponding complexity. Multi-agent systems [Wooldridge, Jennings, 1995] are an emerging alternative that provide distributed entities, called agents, with autonomous reasoning skills that facilitate the development of distributed applications. Moreover, some proposals provide the agents with special capabilities for learning adapting mechanisms [Glez-Bedia, Corchado, 2002]. Case based planner BDI (Belief, Desire, Intention) agents [Glez-Bedia, Corchado, 2002] 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, and to incorporate a novel automatic discovery and planning method based on previous plans.

This work presents IBO, 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 existing plans and previous results. IBO 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. IBO can simulate and analyze laboratory work processes and the behavior of the workers, facilitating any adaptation required when facing anomalous situations and predicting possible risks. In this way, it is possible to analyze large amounts of data from microarrays in a distributed way. The core of IBO is a case based planner agent [Glez-Bedia, Corchado, 2002] specifically designed to act as Web services coordinator, making it possible to reduce the computational load for the agents in the organization and expedite the classification process.

The IBO model was applied to case studies, consisting of the classification of leukemia patients and brain tumors from microarrays, and the multi-agent system developed incorporates novel strategies for data analysis and classification. The process of studying a microarray is called expression analysis [Lander *et al.,* 2001] and consists of a series of phases: data collection, data pre-processing, statistical analysis, and biological interpretation. These analysis phases basically consist of three stages: normalization and filtering; clustering and classification; and extraction of knowledge. In this study, a multi-agent system based on the IBO architecture models the phases of the expression analysis performed by laboratory workers and incorporates innovative algorithms implemented as Web services, as filtering techniques based on statistical analysis,

allowing a notable reduction of the data dimensionality and a classification technique based on a ESOINN [Furao *et al.,* 2007] and SODTNN (Self Organized Dynamic Tree Neural Network) neural network. The core of the system is reasoning agents based on the case based planner BDI [Glez-Bedia, Corchado, 2002] mechanism. Furthermore, the system incorporates other agent types to accomplish complementary tasks required for expression analysis.

The next section, provides the specific problem description of the microarray data analysis. Section 7.2 focuses on the architectural problem that essentially motivated the development of IBO. Section 7.3 describes the main characteristics of the Intelligent Biomedic Organizations and briefly explains its components. Section 7.4 presents a case study consisting of a distributed multi-agent system for cancer detection scenarios developed using IBO. Finally section 7.5 presents the results and conclusions obtained.

7.1 Microarray Data Analysis

Microarray has become an essential tool in genomic research, making it possible to investigate global gene expression in all aspects of human diseases. Microarray technology is based on a database of gene fragments called ESTs (Expressed Sequence Tags), which are used to measure target abundance using the scanned fluorescence intensities from tagged molecules hybridized to ESTs [Lipshutz *et al.,* 1999]. Specifically, the HG U133 plus 2.0 [Affymetrix] are chips used for expression analysis. These chips analyze the expression level of over 47000 transcripts and variants, including 38500 well-characterized human genes. It is comprised of more than 54000 probe sets and 1300000 distinct oligonucleotide features. The HG U133 plus 2.0 provides multiple, independent measurements for each transcript. In addition to this chip, Affymetrix has another series of chips that can perform expression analysis, including the U95Av2 chip, which hast 12600 genes and has been used to study various diseases. The use of Multiple probes provides a complete data set with accurate, reliable, reproducible results from every experiment. Microarray technology is a critical element for genomic analysis and allows an in-depth study of molecular characterization of RNA expression, genomic changes, epigenetic modifications or protein/DNA unions.

Expression arrays [Affymetrix] are a type of microarray that have been used in different approaches to identify the genes that characterize certain diseases [Taniguchi *et al.*, 2008] [Avogadri, Valentini, 2009] [Margalit *et al.*, 2005]. In all cases, the data analysis process is essentially composed of three stages: normalization and filtering; clustering; and classification. The first step is critical to achieve both a good normalization of data and an initial filtering to reduce the dimensionality of the data set with which to work [Armstrong, Van de Wiel, 2004]. Since the problem at hand is

working with high-dimensional arrays, it is important to have a good pre-processing technique that can facilitate automatic decision-making about the variables that will be vital for the classification process. In light of these decisions it will be possible to reduce the original dataset. Moreover, the choice of a clustering technique allows data to be grouped according to certain variables that dominate the behaviour of the group. After organizing into groups it is possible to extract knowledge and classify patients within the group that presents the most similarities.

Case-based reasoning [Riverola *et al.,* 2006] is particularly applicable to this problem domain because it (i) supports a rich and evolvable representation of experiences, problems, solutions and feedback; (ii) provides efficient and flexible ways to retrieve these experiences; and (iii) applies analogical reasoning to solve new problems [Jurisica, Glasgow, 2004]. CBR systems can be used to propose new solutions or evaluate solutions to avoid potential problems. The research in [Aaronson *et al.*, 1993] suggests that analogical reasoning is particularly applicable to the biological domain, in part because biological systems are often homologous (rooted in evolution). Moreover, biologists often use a form of reasoning similar to CBR, where experiments are designed and performed based on the similarity between features of a new system and those of known systems. In [Arshadi, Jurisica, 2005] a mixture of experts for case-based reasoning (MOE4CBR) is proposed. It is a method that combines an ensemble of CBR classifiers with spectral clustering and logistic regression, but does not incorporate extraction of knowledge techniques and does not focus on dimensionality reduction. Some approaches such as [Riverola *et al.,* 2006] provide CBR solutions and knowledge extraction techniques, facilitating the comprehension of the classification process. The following section presents alternatives for distributing services in CBR systems.

7.2 Virtual Organizations and Multi-Agent Architectures

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. These organizations could appear in emergent or dynamic agent societies, such as grid domains, peer-to-peer networks or other contexts where agents dynamically group together to offer compound services. However, although in the theoretical level technology seems to be able to allow the development of this new kind of systems, to investigate in theories, models, mechanisms, methods and tools to develop systems with reorganization capabilities that provide them with the ability to adapt themselves to environmental changes is still necessary.

Multi-system agents are characterized by being autonomous, reactive, proactive, socially skilled, etc., [Wooldridge, Jennings, 1995] for which they are perfectly suited for creating organizational models in which each agent can send and receive messages with an individual, organization or actor in the real society that is being modelled [David *et al.,* 2004]. Additionally, the interaction among agents can correspond to the interactions that exist in the real world [David *et al.,* 2004]. Once the model has been established, a simulation is initiated and the system's behaviour is observed. If the agents possess adequate learning and adapting capabilities, it is possible to obtain knowledge and detect patterns of behaviour. There are many agent-based social simulation models that try to analyze different social phenomena [Epstein, Axtell, 1996] [David *et al.,* 2004]. [Schelling, 1978] carried out the first agent-based social simulation in which each person was represented by an agent and the interaction between the agents represented relevant social processes. Epstein and Axtell [Epstein, Axtell, 1996] developed the first large-scale agent-based model, Sugarscape, which allows entire societies to be modelled and different social processes, such as death, sickness, sex, reproduction, culture, conflict, war, etc., to be observed. Nevertheless, there is still much work to do, especially in the field of automated re-organization of virtual organizations [David *et al.,* 2004]. In the framework of this research, we are specially interested in organizations characterized by the need of classification of massive amounts of data and the integration of multi-agent systems and web services can be highly recommended.

The integration of multi-agent systems with SOA and Web Services approaches has been recently explored [Ardissono, 2004]. 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. [Oliva *et al.*, 2008] 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). [Shafiq *et al.,* 2006] propose a gateway that allows interoperability between Web Services and multi-agent systems. [Liu, 2007] proposes a multi-agent architecture to develop inter-enterprise cooperation systems using SOA and Web Services components and communication protocols. [Walton *et al.,* 2006] presents a technique to build multi-agent systems using Web Services, defining a language to represent the dialogs among agents. There are also frameworks, such as Sun's Jini and IBM's WebSphere, which provide several tools to develop SOA-based systems. Jini uses Java technology to develop distributed and adaptive systems over dynamic environments. [Rigole *et al.*, 2002] have used Jini to create agents on demand into a home automation system, where each agent is defined as a service in the network. WebSphere provides tools for several operating systems and programming languages. However, the systems developed using these frameworks are not open at all because the framework is closed and services and applications must be programmed using a specific programming language that supports their respective proprietary APIs (Application Programming Interface).

These alternatives have been explored as possible solutions for virtual organizations based on data analysis. However, most of them are in early stages of development, so it is not possible to actually know their potential in real scenarios. In this sense, we have used an architecture, which not only provides communication and integration between distributed agents, services and applications; but also proposes a new method to facilitate the development of distributed multi-agent systems by means of modelling the functionalities of the agents and the systems as services. Another feature in this architecture is security, which is managed by the agents. All communications must take place via the agents, so services cannot share their resources unless the agents allow it.

7.3 Intelligent Biomedic Organizations

IBO (Intelligent Biomedic Organizations) 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 IBO is a novel mechanism for the implementation of the stages of CBR mechanisms through Web services that provides a dynamic self-adaptive behaviour to reorganize the environment. Moreover, IBO provides communication mechanisms that facilitate integration with SOA architectures.

IBO was initially designed to model the laboratory environments oriented to the processing of data from expression arrays. To do this, IBO 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 IBO 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 layer constitutes the core of IBO and define a virtual organization for massive data analysis, as can be seen in Figure 1. Figure 1 shows four types of agent layers:

- Organization: The organization agents run on the user devices or on servers. The agents installed on the user devices create a bridge between the devices and the system agents that perform data analysis. The agents installed on servers will be responsible for conducting the analysis of information following the case based planner BDI [Glez-Bedia, Corchado, 2002] 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, it is imperative to establish a previous configuration of the workflow at the analysis layer.

- 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 case based planner BDI [Glez-Bedia, Corchado, 2002] 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 the IBO is applied. Specifically, the microarray case study includes that 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 multi-agent 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 (Foundation for Intelligent Physical Agents-Agent Communication Language) and SOAP (Simple Object Access Protocol) protocols.

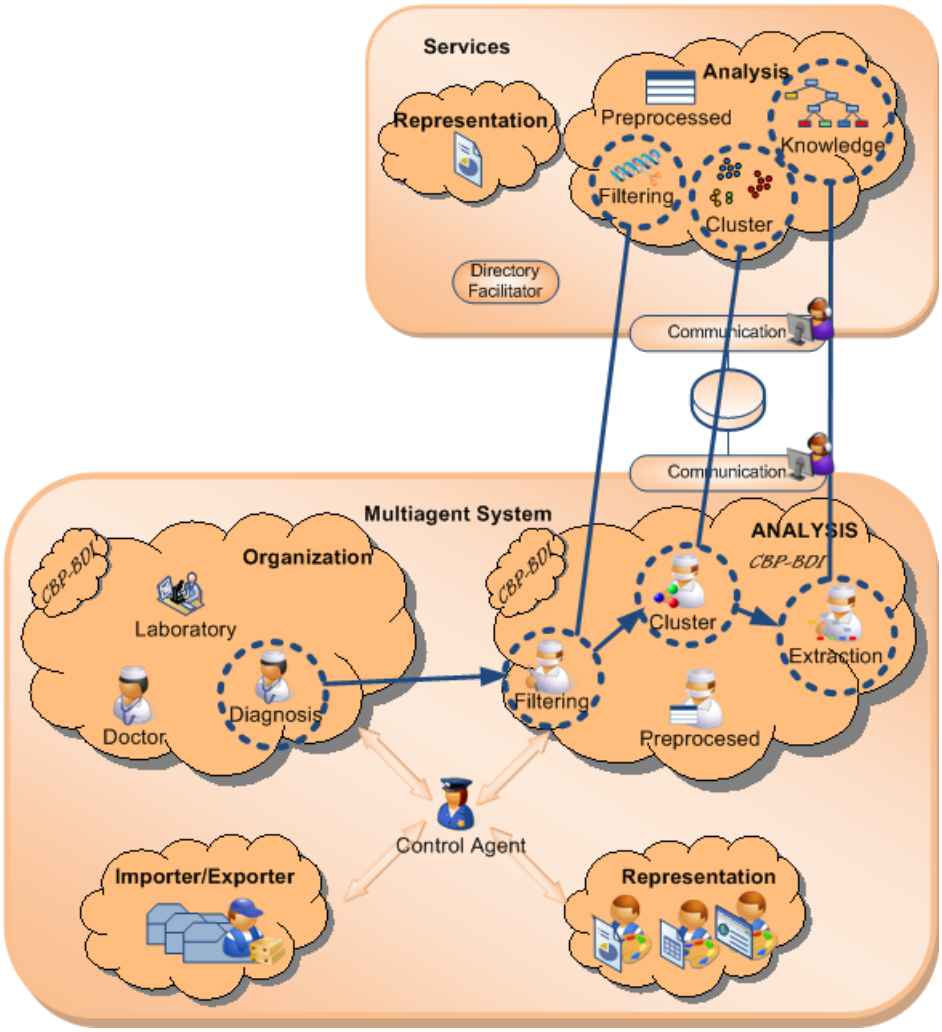

Fig. 1. IBO Architecture

As shown in Figure 1, the agents from the different layers interact to generate the plan for the final analysis of the data. The different system agents are distributed according to the layers and the connections that each type of agent can make with the other types of system agents and services. For example, in order to carry out its task, the Diagnosis agent at the organizational layer uses a specific sequence to select agents from the analysis layer. In turn, the agents at the analysis layer select the services that are necessary to carry out the data study: the filtering agent at the analysis layer selects from the services and workflow that are appropriate for the data.

The agents at the organizational layer are case based planner BDI agents with the ability to generate plans automatically based on previously existing plans in the system. Each of the case based planner BDI agents handles its own memory of cases in which it stores past experiences related to the specific tasks assigned to the agent. As a

result, each agent manages its own memory of cases, which is updated each time a global plan is carried out.

7.3.1 Methodology for Modelling Organizations with IBO

One of the major problems in the development of an architecture based on a multi-agent system is that there are currently no clear standards or well developed methodologies for defining the steps of analysis and design that need to be taken. There are at present a number of methodologies: Gaia [Wooldridge *et al.,* 2000], AUML (Agent UML) [Corchado, Laza, 2003] [Odell *et al.,* 2004], INGENIAS [Pavón *et al.,* 2007a] [Pavón *et al.*, 2007b], TROPOS [Giorginia *et al.,* 2005], MESSAGE [EURESCOM 2001]. The option chosen to define an appropriate analysis and design methodology for the problem to be resolved is one that combines Gaia and AUMLand attempts to take advantage of both (Bajo *et al.,* 2009).

IBO provides a modelling mechanism that uses GAIA and AUML to analyze and design the architecture: with GAIA it can obtain a previous high level analysis and design oriented to organizations; with AUML it can obtain a subsequent design, very similar to the implementation. Following the Gaia [Wooldridge *et al.,* 2000] methodology, in the analysis phase it is possible to obtain the different roles that exist in the system. Figure 2 shows an example of a model for the Planner role. A role can be seen as an abstract description of an entity, which will have a series of responsibilities and permission. The responsibilities represent the functionality of a particular role and can be divided into two categories: liveness and safety. Permission is associated with the available resources and their limited use. Protocols are services that are provided to other agents, in this case InformPlan. Activities are actions that the agent carries out. Liveness represents the responsibilities that each agent has, which are defined by a flow of protocols and activities, in this case the responsibility of the agent is created by the plan CreatePlan.

Name		Create Plan
Description		Genera un plan a partir de la información de los planes existentes
Protocols Activities		InformPlan CalculateGraph CalculateBayesianNetwork CalculateWeight CalculatePlan
Permissions		
Responsibilities	Liveness	CreatePlan: RequestEfficientPlans · RequestInefficientPlans · CalculateGraph · CalculateBayesianNetwork · CalculateWeight · CalculatePlan
	Safety	Neither void plan

Fig. 2. Planner roles model

The information gathered by the model for each role makes it possible to reach the model for the GAIA agents. Figure 3 presents the different roles that the Diagnose agent can perform. The tree leaves represent the different agent roles: CreatePlan, Create WorkFlow, Update Plan.

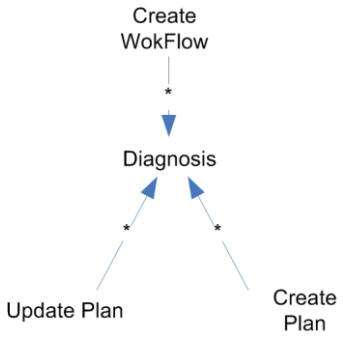

Fig. 3. Model of agents for Diagnosis Agent

The high level analysis and design carried out with the GAIA methodology makes it possible to model the system in terms of organization, but since the level of detail it provides is too low, it becomes necessary to use the AUML methodology, which can provide a more detailed design. In this way, it is possible to obtain the capabilities and services for each type of agent. Figure 4 provides an example of a diagram for the AUML classes for the Diagnosis agent in which it is possible to observe the different services and capabilities that the agent provides. The services correspond to the functionalities that this agent offers to other agents or users, while the capabilities demonstrate the functionalities that the agent can carry out. As shown in Figure 4, the Diagnosis agent is a case based planner BDI agent that is equipped with automatic planning capabilities

that, as a result, require it to have planning and replanning capabilities based on both the information from plans that have been stored in the case base and the information provided by the new plan to be carried out.

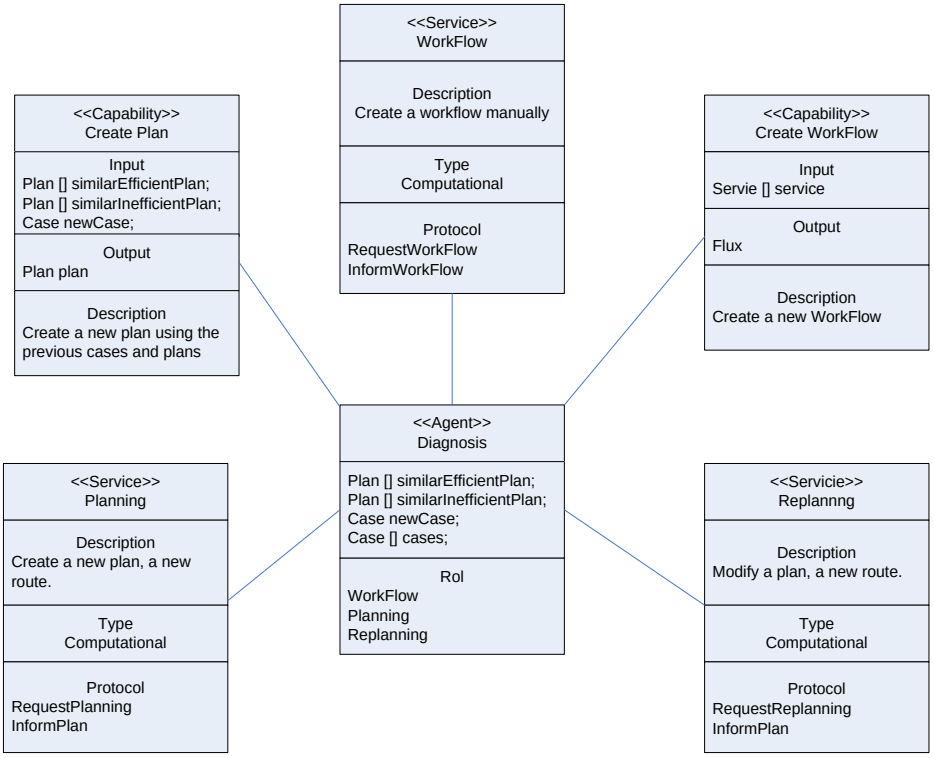

Fig. 4. Diagram of the AUML classes for the Diagnosis agent

The following sections detail the two most innovative elements in IBO: the type of case based planner BDI agent specialized in coordination tasks; and a dynamic distribution mechanism for services.

7.3.2 Coordinator Agent based on case based planner

The agents in the organization layer and the agents in the analysis layer have the capacity to learn from the analysis carried out in previous procedures. To do so, they adopt the model of reasoning, a specialization of case-based reasoning (CBR) [Kolodner, 1993]. The primary concept when working with CBRs is the concept of case. A case can be defined as a past experience, and is composed of three elements: A problem description, which describes the initial problem; a solution, which provides the sequence of actions carried out in order to solve the problem; and the final state, which describes the state achieved once the solution was applied. A CBR manages cases (past

experiences) to solve new problems. The way cases are managed is known as the CBR cycle, and consists of four sequential phases: retrieve, reuse, revise and retain.

Case-based planning is the idea of planning as remembering [Glez-Bedia, Corchado, 2002]. 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. 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.
- 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. That is, relations allowing to establish a link between plan actions.

Case-based Reasoning (CBR) is a type of reasoning based on past experiences [Kolodner, 1993]. CBR systems solve new problems by adapting solutions that have been used to solve similar problems in the past, and learn from each new experience. The primary concept when working with CBR systems is the concept of case, which is described as a past experience composed of three elements: an initial state or problem description that is represented as a belief; a solution, which provides the sequence of actions carried out in order to solve the problem; and a final state, which is represented as a set of goals. CBR manages cases (past experiences) to solve new problems. The way cases are managed is known as the CBR cycle, and consists of four sequential phases: retrieve, reuse, revise and retain. The retrieve phase starts when a new problem description is received. Similarity algorithms are applied so that the cases with the problem description most similar to the current one can be retrieved from the cases memory. Once the most similar cases have been retrieved, the reuse phase begins by adapting the solutions for the retrieved cases in order to obtain the best solution for the current case. The revise phase consists of an expert revision of the proposed solution. Finally, the retain phase allows the system to learn from the experiences obtained in the three previous phases, and consequently updates the cases memory.

Case based planner comes from CBR, but is specially designed to generate plans (sequence of actions) [Corchado *et al.,* 2008b]. In this model, the proposed solution for solving a given problem is a plan. This solution is generated by taking into account the plans applied for solving similar problems in the past. The problems and their corresponding plans are stored in a plans memory. The reasoning mechanism generates plans using past experiences and planning strategies, which is how the concept of Case-Based Planning is obtained [Glez-Bedia, Corchado, 2002]. Case based planner consists of four sequential stages: the retrieve stage, which recovers the past experiences most similar to the current one; the reuse stage, which combines the retrieved solutions in order to obtain a new optimal solution; the revise stage, which evaluates the obtained

solution; and retain stage, which learns from the new experience. Problem description (initial state) and solution (situation when final state is achieved) are represented as beliefs, the final state as a goal (or set of goals), and the sequences of actions as plans. The case based planner cycle is implemented through goals and plans. When the goal corresponding to one of the stages is triggered, different plans (algorithms) can be executed concurrently to achieve the goal or objective. Each plan can trigger new sub-goals and, consequently, cause the execution of new plans. In practice, what is stored is not only a specific problem with a specific solution, but also additional information about how the plans have been derived. As with case-based reasoning, the case representation, the plans memory organization, and the algorithms used in every stage of the case-based planning cycle are essential in defining an efficient planner.

The case based planner BDI agents stem from the BDI model and establish a correspondence between the elements from the BDI model and the case based planner 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 case based planner agents together with the BDI model and generating case based planner BDI agents makes it possible to formalize the available information, the definition of the goals and actions that are available for solving the problem, and the procedure for solving new problems by adopting the case based planner reasoning cycle.

Agents with BDI architecture have their origins in the practical reasoning of the traditional philosophy. These agents are supposed to be able to decide in each moment what action to execute according to their objectives. The practical reasoning undergoes two phases: in the first one the goals are defined and in the second one it is defined how to achieve such goals A representation based on an action requires an agent architecture in which the way to acquire and process the knowledge of the world, at the reasoning stage, is closely related to the way in which plans are constructed and used, in the phase of execution. In this section, we show how such a requirement can be achieved through a BDI agent model [Kolodner, 1983] [Kolodner, 1983b] [Joh, 1997] [Corchado *et al.*, 2008b]. 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

$$M = \{\tau_1, \tau_2, \cdots, \tau_s\} \text{ with } s < \infty \tag{19}$$

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

$$B = \{b_i / b_i = \{\tau_1^i, \tau_2^i, \cdots, \tau_n^i\}, n \le s \quad \forall i \in N\}_{i \in N} \subseteq M \tag{1}$$

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

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

$$e_j^t = \{b_1^{jt}, b_2^{jt}, \cdots b_r^{jt}\}_{r \in N} \subseteq B \quad \forall j,t \tag{2}$$

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

$$d : \underset{e_0}{E} \rightarrow \underset{e^*}{E} \tag{3}$$

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:

$$i : \underset{(b_1, b_2, \cdots, b_n, e_0)}{BxBx^{n)} \cdots xBxE} \rightarrow \underset{e^*}{E} \tag{4}$$

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,

$$a_j : \underset{e_i}{E} \rightarrow \underset{a_j(e_i)=e_j}{E} \tag{5}$$

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.

$$p_n : \underset{e_0}{E} \rightarrow \underset{p_n(e_0)=e_n}{E} \tag{6}$$

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

Based on this representation, the case based planner BDI agents in IBO 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. In IBO 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. In IBO, 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, allowing each plan to be represented by a path in the graph. The presence of an arch that connects to a specific node implies the execution of a service associated with the end node. As a result, a sequence of nodes represents a sequence of actions/services and the order in which they are carried out, so that each plan identified

in (6) can be represented as a route in a graph. Each of the nodes in the graph is associated with a set of variables with corresponding value, thus forming a set of beliefs that describe each of the states of the graph. Additionally it is necessary to indicate that each of the nodes corresponding to the services is also included in a pair of fictitious nodes that correspond to the start and end nodes. The start and end nodes are necessary to establish the initial service of a plan, as well as to be able to establish the end point of a specific plan. Figure 5 provides a graphical representation of a service plan. As shown, the 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 following sequence $(s_7 \circ s_5 \circ s_3 \circ s_1)(e_0)$. e_0 represents the original state that corresponds to Init, which represents the inicial problema description e_0. Final represents the final state of the problem e^*.

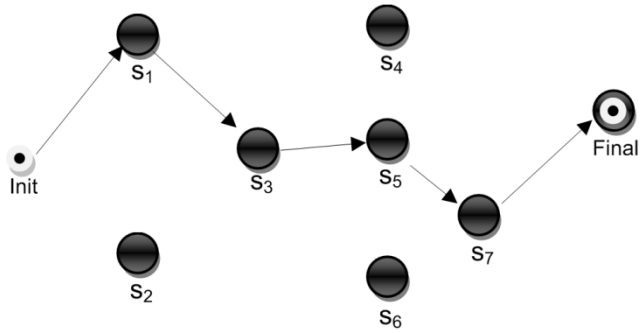

Fig 5. Planning route for disconnected graph.

This way, a case based planner BDI agent works with plans that contain the information associated with the actions that it should carry out, i.e., each agent at the analysis layer defines its own memory of plans with the information it needs. The information required for each of the agents at the analysis layer depends on its functionality. Some of them require executable actions such as service compositions, while other agents only need to select the service that best suits its needs without having to carry out any composition. Table 1 provides an example of the description of the case structure for a filtering agent. As shown, a filtering agent will consider the number of cases, the number of variables, the optimization, quality and description of a problem.

Table 1. Filtering agent case

Variable	Field Type
numberCases	Integer
numberVariables	Integer
optimization	Integer
quality	Real
problem	Integer

Additionally, the information for the plans is defined by the sequence of actions/services applied. Table 2 presents the information contained in the filtering plan, which in this case is a list of services. Finally, it is necessary to define the structure for each service. Table 3 shows the structure of a service that is defined by the number of variables, the name of the services, and a list of parameters used by the service.

Table 2. Services for a filtering agent case

Task Field	Field Type
Service	ArrayList of Services

Table 3. Service

Task Field	Field Type
numberVariablesFinal	Int
service	String
Parameter	ArrayList of parameter

Case based planner 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 (we will focus on one agent in particular within the analysis layer, specifically the filtering agent):

1. Extract the set of cases similar to the current case with the best and worst output.
2. Generate the directed graph with the information from the different plans.
3. Generate a TAN (Tree Augmented Naive Bayes) classifier for the cases with the best and worst output respectively, using the Friedman-Godsmidtz [Friedman *et al.,* 1997] algorithm.
4. Calculate the execution probabilities for each service with respect to the classifier generated in the previous step.
5. Adjust the connections from the original graph according to a metric.
6. Construct the graph.

7.3.2.1 Similar Cases

During the retrieval stage, the plans with the greatest and least efficiency are selected from among those that have been applied. Microarrays are composed of probes that represent variables that mark the level of significance of specific genes. The set of these variables defines the initial state e0 as shown in equation (3).

The retrieval of those cases is performed in one of two ways according to the case study. To retrieve cases, it is important to consider whether there has been a previous analysis of a case study with similar characteristics. If so, the corresponding plans for the same case study are selected.

If there are no plans that correspond to the same case study, or if the number of plans is insufficient, the plans corresponding ot the most similar case study are retrieved. The selection of the most similar case study is performed according to the cosine distance applied to the following set of variables:

- Number of probes
- Number of cases
- Coefficient of the Pearson variation [Kuo *et al.*, 2003] for e_0.

The number of efficient and inefficient cases selected is predetermined so that at the end of this stage the following set of elements is obtained.

7.3.2.2 Constructing a directed graph

The different plans are represented in the graphs as shown in figure 5. The plans represented in graphical form are joined to generate one directed graph that makes it possible to define the new plans based on the minimization of a specific metric. That way, for example, given the graphs shown in figure 6, a new graph is generated that joins the information corresponding to both graphs.

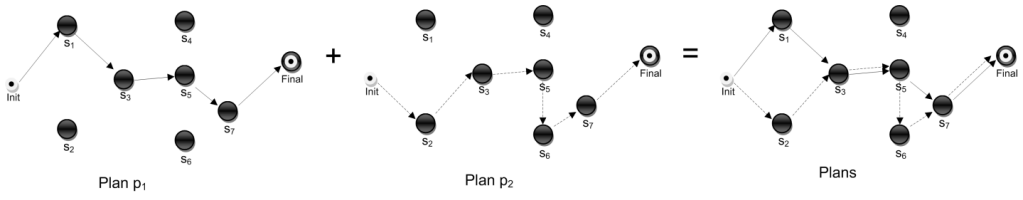

Fig 6. Composition of the graphs

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, which defines the plan obtained from the recovered plans. This value is estimated based on the efficiency of the plans recovered as indicated in section 7.3.2.5

The new plans are generated through the construction of the graph of plans shown in figure 6. The weights are estimated based on the existing plans by applying a TAN classifier and the probabilities of execution of the services. The probabilities that a particular number of services may have been executed for classes for the efficient and not efficient plans obtained with the TAN are combined with the probabilities of execution of the servicies to update the weigths. The TAN classifier provides a tree that takes into account two Bayesian networks. The entry data to the bayesian networks is broken down into the following elements:

- Plans with a high efficiency are assigned to class 1.
- Plans with a low efficiency are assigned to class 0.

A Bayesian network is calculated for each of the classes according to the recovered plans. The Bayesian networks are constructed by following the Friedman-Goldsmidtz [Friedman *et al.,* 1997] algorithm. Having two different classes, two Bayesian networks will be generated, one for each of the classes.

7.3.2.3 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 (the tree). Thus, by applying the Friedman-Goldsmidtz [Friedman *et al.,* 1997] algorithm, the two classes that are considered are efficient and inefficient. The Friedman-Goldsmidtz algorithm makes it possible to calculate a Bayesian network based on the dependent relationships established through a metric. The metric considers the dependent relationships between the variables according to the classifying variable. In this case, the classified variable is efficient and the remaining variables indicate whether a service is or is not available. The metric proposed by Friedman be defined as:

$$I(X,Y \mid Z) = \sum_{x \in X} \sum_{y \in Y} \sum_{z \in Z} P(x,y,z) \cdot \log \left[\frac{P(x,y \mid z)}{P(x \mid z) \cdot P(y \mid z)} \right] \qquad (7)$$

Based on the previous metric, the probabilities are estimated according to the frequencies of the data. The Friedman-Goldsmidtz [Friedman *et al.*, 1997] algorithm is broken down into the following steps:

- Calculate the value of $I(X,Y \mid C)$ for the different variables/attributes X, Y that may be interconnected in the original graph, class C varies between the similar efficient and inefficient cases.
    - Construct a complete nondirected graph
    - Establish the different attributes/variables as nodes.
- Within the connections, establish the values obtained in the first step as weights. For the arcs that do not have connections, set the value as 0.
- Create a maximum tree based on the Kruskal [Campos et al., 2008] algorithm.
- Convert the nondirected tree into a directed tree. The initial connection and the selection of the next node to connect will indicate the direction of the connections.
- Finally, construct the TAN model by adding a node that represents class C and an arc that connects to C for each of the attributes.

7.3.2.4 Probabilities of the services

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 consider the dependences between services and influence the final value of the weights assigned to the arcs in the graph. The probabilities are calculated taking into account the TAN model.

Figure 7 shows a classifying model for a multi-connected network. The calculation of the probabilities for a tree is carried out according to a process similar to the example shown in the figure

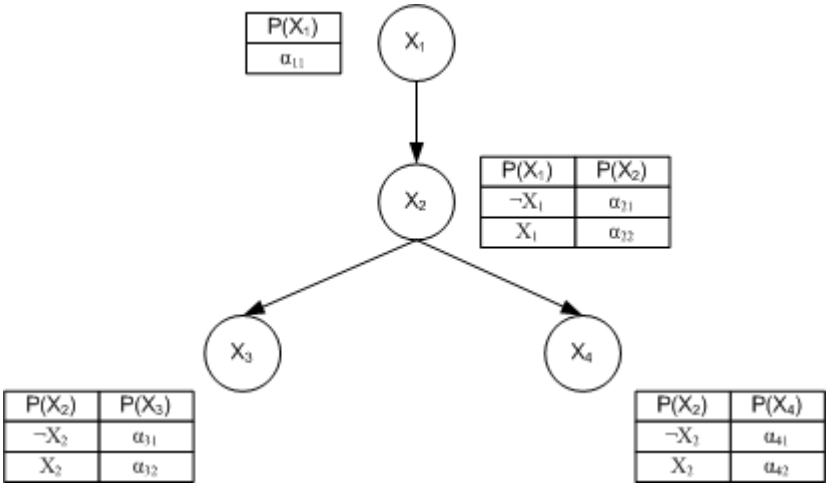

Fig 7. Bayesian network

Assuming that the set of random variables can be defined as follows $U = \{X_1, X_2, ..., X_n\}$, we can assume that the variables are independent. A Bayesian network for U is defined as a tuple formed by two elements $B = < G, T >$ where G represents an acyclic directed graph in which the nodes are variables and the connections between the nodes for T contain the connection probabilities between variables. The probabilities are represented by $P_B(x_i \mid \pi_{x_i})$ 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

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

As shown, for this, it is necessary to order the variables according to the appearance within the acyclic directed graph. This way, in figure7, and give the probability tables indicated, the probability that this will occur $X_4, \neg X_3, X_2, X_1$ is given by

$$P(X_4, \neg X_3, X_2, X_1) =$$
$$P(X_4 \mid \neg X_3, X_2, X_1) \cdot P(\neg X_3, X_2, X_1) =$$
$$P(X_4 \mid \neg X_3, X_2, X_1) \cdot P(\neg X_3 \mid X_2, X_1) \cdot P(X_2, X_1) =$$
$$P(X_4 \mid \neg X_3, X_2, X_1) \cdot P(\neg X_3 \mid X_2, X_1) \cdot P(X_2 \mid X_1) \cdot P(X_1)$$

Keeping in mind the relationships of independence between variables, it is possible to simplify the previous probabilities as follows:

$$P(X_4 \mid \neg X_3, X_2, X_1) = P(X_4 \mid X_2)$$

The probability that X_4 occurs is independent from X_3, X_1 occurring, thus it is possible to reduce to what has occurred in the only case that precedes the tree, which is X_2. This way, the set of parent nodes X_4 is represented by $\Pi_{X_4} = \{X_2\}$. Similarly, simplifying the previous expressions, we have:

$$P(X_4, \neg X_3, X_2, X_1) = P(X_4 \mid X_2) \cdot P(\neg X_3 \mid X_2) \cdot P(X_2 \mid X_1) \cdot P(X_1)$$

Substituting the previous expression with real values, we obtain the following:

$$P(X_4, \neg X_3, X_2, X_1) = \alpha_{42} \cdot (1 - \alpha_{31}) \cdot \alpha_{21} \cdot \alpha_{11}$$

7.3.2.5 Connection considerations

Using the TAN model, we can define the probability that a particular number of services may have been executed for classes 1 and 0 for the efficient and not efficient plans as explained in Section 7.3.2.2. This probability is used, together with the probability of execution, 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 equation (8). The function has been defined in such a way that the plans of high quality are those with values closest to zero.

$$c_{ij} = P(j,1) \cdot I(i,j,1) \cdot t_{ij}^1 - P(j,0) \cdot I(i,j,0) \cdot t_{ij}^0 \quad (8)$$

$$t_{ij}^1 = \frac{\sum_{p \in G_{ij}^1, s \in G^1} (1 - (q(p) - \min(q(s)))) + 0.1}{\#G_{ij}^1} \quad (9)$$

$$t_{ij}^{0} = \frac{\sum_{p \in G_{ij}^{0}, s \in G^{0}} q(p) - \min(q(s)) + 0.1}{\#G_{ij}^{0}} \quad (10)$$

where:

- $I(i, j, 1)$ is the probability that service i for class 1 is executed before of service j
- $P(j, 1)$ the probability that service j for class 1 is executed. The value is obtained based on the Bayesian network defined in the previous step.
- $I(i, j, 0)$ the probability that service i for class 0 is executed before of service j
- $P(j, 0)$ the probability that service j for class 0 is executed. The value is obtained based on the Bayesian network defined in the previous step.
- G_{ij}^{s} 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.
- $\#G_{ij}^{s}$ 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

7.3.2.6 Graph construction

Once the graph for the plans has been constructed, the minimal route is calculated that goes from the start node to the end node. In order to calculate the shortest/longest route, the Dijkstra algorithm is applied since there are implementations for the complexity 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.

IBO can take advantage of the characteristics of the case based planner BDI in order to enhance the services organization. In this way it is possible to dynamically

distribute tasks and make use of services. The next section focuses on the distribution of services in IBO.

7.3.3 Services distribution in IBO

The IBO services are broken down into categories based on the functionality that is carried out. The services are broken down similar to the agents at the analysis layer, so that each agent at the analysis layer is aware of the services that are available to carry out different tasks. Knowing which services are available is useful to carry out the manual selection of the flow of data analysis. For automated planning, it is not necessary to have that information available, since the information is already available and associated with different cases stored in the system. In the case study on microarrays that is presented in the following section, the services are broken down into the categories that are necessary for carrying out an expression analysis: preprocessed, filtered, cluster-classification, knowledge extraction.

7.4 Case Study: Using IBO to Develop a Decision Support for Patients Diagnosis

The IBO multi-agent architecture has been used to develope a decision support system for the classification of leukemia and brain tumors patients. To do so, three case studies were established in which data was taken from a biomedical database corresponding to samples taken with microarrays. The first case study uses data corresponding to patients suffering from leukemia and focuses on the classification of the type of leukemia. The second case study also analyzes the data corresponding to 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 aim of establishing three case studies with such different characteristics is to evaluate the efficiencies of the IBO system, and its level of adaptability for being used in different types of organizations.

The microarrays used in the case studies contain information that corresponds to the patients affected by leukemia and brain tumors. 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) [Corchado *et al.,* 2009]. Additionally, the system was applied to patients with CLL leukemia in order to detect the characteristics of the 3 different subtypes, (B lymphocytes, T lymphocytes, Natural killer

cells) also using the data obtained with the HG U133 plus 2.0 chip. Finally, the system was applied to patients with brain tumors. This case study had already been carried out by Nutt y Horng [Nutt *et al.,* 2003] [Horng *et al.,* 2009], the data was obtained from the Affymetrix U95Av2 GeneChips [Golub *et al.,* 1999]. The data included 4 different types of brain tumors. A classification was carried out for 28 glioblastomas and 22 anaplastic oligodendrogliomas, and further classified according to classic or not classic (Nutt *et al.,* 2003).

The IBO was used to model the organizations corresponding to each case study, and a support system was provided for the decision based on obtaining a classification method and clustering patients, as well as a detection method for the patterns that characterize the different diseases for each patient. The analysis of data from microarrays is made in a distributed manner and certain tedious tasks are automated. For each of the case studies, the process consists of three steps: Initially the laboratory personnel hybridizes the samples, then the data analysis is done and finally a human expert interprets the results obtained. The multi-agent system built from the IBO architecture reproduces this behaviour. The aim of the tests performed is to determine whether the system is able to classify new patients based on the previous cases analyzed and stored. Next, the developed agents and services are explained.

7.4.1 Services Layer

The services implement the algorithms that allow the analysis expression of the microarrays [Lander *et al.,* 2001] [Corchado *et al.,* 2009]. These services are invoked by the agents and present novel analysis techniques.

7.4.1.1 Pre-processing Service

This service implements the RMA algorithm and a novel control and errors technique. The RMA (R*obust Multi-array Average*) [Irizarry, 2003] algorithm is frequently used for pre-processing Affymetrix microarray data. RMA consists of three steps: (i) Background Correction; (ii) Quantile Normalization (the goal of which is to make the distribution of probe intensities the same for arrays); and (iii) Expression Calculation. During the Control and Errors phase, all probes used for testing hybridization are eliminated. Occasionally, some of the measures made during hybridization may be erroneous; although this is not the case with the control variables. In this case, the erroneous probes that were marked during the RMA must be eliminated.

7.4.1.2 Filtering Service

The filtering service eliminates the variables that do not allow classification of patients by reducing the dimensionality of the data. Three services are used for filtering: **Variability:** The variables with low variability have similar values for each of the individuals, and so they are not significant during the classification process. The first stage is to remove the probes that have low variability according to the following steps: Calculate the standard deviation for each of the probes, standardize the above values, discard of probes for which the value of z meet the following condition: $z < \alpha$. **Uniform Distribution:** All remaining variables that follow a uniform distribution are eliminated. The variables that follow a uniform distribution will not allow the separation of individuals. The contrast of assumptions followed is explained below, using the Kolmogorov-Smirnov [Brunelli, 2001] test. H_0: The data follow a uniform distribution; H_1: The analyzed data do not follow a uniform distribution. **Correlations:** The linear correlation index of Pearson is calculated and correlated variables are removed so that only the independent variables remain. **Cufoff points**: Delete those probes which don't have significative changes in the density of individuals.

7.4.1.3 Clustering Service

It addresses both the clustering and the association of a new individual to the most appropriate group. The services included in this layer are: the ESOINN neural network and the NN (Neural Gas) clustering algorithm (Nearest Neighbor). Additional services in this layer for clustering are the Partition around medoids (PAM) [Saitou, Nie, 1987] and demdograms (Kaufman and Rousseeuw 1990).

The ESOINN [Furao *et al.*, 2007] (Enhanced self-organizing incremental neural network) clustering technique is variation of neural network SOINN (self-organizing incremental neural network) [Shen, 2006]. ESOINN consists of a single layer, so it is not necessary to determine the manner in which the training of the first layer changes to the second. With a single layer, ESOINN is able to incorporate both the distribution process along the surface and the separation between low density groups. The initial phase could be understood as a phase of competition, while in a second phase, the network of nodes begins to expand just as with a NG. The classification is carried out bearing in mind the similarity of the new case using the NN cluster. The similarity measure used is as follows:

$$d(n,m) = \sum_{i=1}^{s} f(x_{ni}, x_{mi}) * w_i \qquad (11)$$

Where s is the total number variables, n and m the cases, w_i the value obtained in the uniform test and f the Minkowski [Groenen, Jajuga, 2001] Distance that is given for the following equation.

$$f(x,y) = \sqrt[p]{\sum_i \left| x_i - y_j \right|^p} \text{ with } x_i, y_j \in R^p \quad (12)$$

This dissimilarity measure weights those probes that have a less uniform distribution, since these variables don't allow a separation.

7.4.1.4 Knowledge Extraction Service

The extraction of knowledge technique applied has been CART (Classification and Regression Tree) [Breiman *et al.,* 1984] algorithm. The CART algorithm is a non parametric test that allows extracting rules to explain the classification carried out. There are others techniques to generate the decision trees, as the methods based on ID3 trees [Quinlan, 1979], although the results can be considered very similar to those provided by CART.

7.4.2 Agents Layer

The agents in the analysis layer implement the case based planner 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 of the recovered for each phase of the analysis process.

In the analysis layer, at the stage Preprocessed only a service is available, so that the agent only selects the settings. The efficiency is calculated by the deviation in the microarray once have been preprocessed. 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.

$$e(p) = \frac{s}{N} + \frac{i'}{I} \quad (13)$$

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, CART has only been implemented for this task so that the agent responsible for conducting the selection method does not take into account the plans recovered, as in the previous phase Efficiency is determined by the number of misclassified individuals.

In the organization layer, the diagnosis agent is in charge of choosing the agents for the expression analysis [Lander *et al.*, 2001]. The diagnosis agent establishes the number of plans to recover from the plans memory for each of the agents as well as the agents to select from the analysis layer. In a similar way the laboratory agent and the human expert select the agents from the organization layer that will be used. If the review at all stages is positive, and the human expert feels good result, the plan is stored in the memory of plans for further use.

The case study could be easily generalized to similar environments. Next, the results obtained after applying IBO to a real scenario are presented.

7.5 Results and Conclusions

This work has presented the IBO multi-agent architecture and its application to three real problems. IBO facilitates tasks automation by means of intelligent agents capable of autonomously plan the stages of an expression analysis. The characteristics of this novel architecture facilitate a organizational-oriented approach where the dynamics of a real scenario can be captured and modelled into case based planner BDI agents. In this way, the agents are based on the BDI model and act as controllers and coordinators of the organization. The complex functionalities of the agents, are modelled as distributed services. IBO provides a service oriented approach and facilitates the distribution and management of resources. Moreover, IBO facilitates the distributed execution of complex computational services, reducing the number of crashes in agents, since IBO separates the processing tasks from the agent architecture and, consequently, reduces the posibility of failure due to agents. The multi-agent system developed integrates within web services aimed to reduce the dimensionality of the original data set and a novel method of clustering for classifying patients. The multi-agent perspective allow the system to works in a way similar to how human specialists operate in the laboratory, but is able to work with great amounts of data and make decisions automatically, thus reducing significantly both the time required to make a prediction, and the rate of human error due to confusion. IBO has been applied to three different case studies and a number of tests have been carried out in each of them. The tests were oriented both to evaluate the efficiency and the adaptability of the approach. In the following paragraphs we present the concrete results obtained and extract the consequent conclusions.

The first step will be to analyze the efficiency of the service distribution system proposed in IBO, which is carried out by dynamically generating plans. To do so, we focus on the case study corresponding to the classification of patients affected by different types of leukemia, since it is the case study with largest amount of data and classification groups. After this we analyze the adaptability of the system for different scenarios. With the second experiment we establish a comparison for the efficiencies obtained in each of the case studies. Furthermore, we study the evolution of the system *return* output over the long term, which allows us to obtain a measure of its level of adaptability to the problems being considered. It should be noted that in order to carry out the system evaluation, we must establish the values of the different function parameters for the web services in the different case studies once a particular plan has been selected. The selection of the parameter values is done according to the statistical criteria for the previously defined confidence and interval tests.

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. The first step that was carried out was to incorporate each of the services based on the plans, for which it was necessary to carry out an analysis that only considered the services in question, so that it would be possible to incorporate a new arc in the graph for the plans. If the arc is not present in the graph, the service will never make a selection since it will remain disconnected. This process is carried out in a manner that has been pre-determined for each of the services through a series of steps indicated in the Control Agent each time that a new service registers for a specific analysis. According to the identification of the problem described in table 1, the filtering agent will select the plans with the greatest efficiency, considering the different execution workflows for the services that are in the plans. An execution workflow consists of a sequence of services that are executed in a specific order. Table 4 shows the efficiency obtained for the service workflows obtained using the equation (18) 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.

Table 4. Efficiency of the plans

Variability (z)	Uniform (α)	Correlation (α)	Cutoff	Efficiency	Class
1	2	3	4	0.1301	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

Based on the plans shown in table 4, a new plan is generated following the procedures indicated in section 7.3.2. Once the new plan has been generated, it is necessary to establish the configuration parameters for the services that can define the signification levels for the statistical tests and other parameters that are used to carry out the filtering process. Table 5 shows the possible service configurations for the new plan generated. The bolded text indicates the configuration with optimal efficiency. The data shown in table 4 corresponds to a sample plan generated by the filtering agent in the analysis layer. The filtering agent in the analysis layer selects the 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 equation (18). The table shows the selected configuration in bold text. 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.

Table 5. Plans of the filtering phase and plan of greater efficiency

Variability (z)	Uniform (α)	Correlation (α)	Probes	Errors	Efficiency
-1.0	0.25	0.95	2675	21	0.1485
-1.0	0.15	0.90	1341	23	0.1333
-1.0	0.15	0.95	1373	24	0.1386
-0.5	0.15	0.90	1263	24	0.1365
-0.5	0.15	0.95	1340	23	0.1333
-1.0	**0.1**	**0.95**	**785**	**24**	**0.1277**
-1.0	0.05	0.90	353	32	0.1574
-1.0	0.05	0.95	357	34	0.1669
-0.5	0.05	0.9	332	47	0.2278
-0.5	0.05	0.95	337	53	0.2562
-1.0	0.01	0.95	54	76	0.3594

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 7.3.2.3 and 7.3.2.4. Once the graph plans have been generated, a more efficient plan is generated according to the procedures indicated in section 7.3.2.6, with which we can obtain the plan that best adjusts to the data analysis. Table 6 shows the plans generated by the filtering process that best adjusts to the different case studies in which the system was applied.

Table 6. 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 plans have been generated, the evolution of the system is evaluated for each of the case studies. Figure 8 shows the evolution of the efficiency according to equation (11). This measure of efficiency was selected because it provides a global measure of the results from an expression analysis. As shown, the efficiency has improved as the system has acquired efficiency over time, where the improvement in the case study with the greatest number of individuals is the most significant. This is partly due to the functioning of the actual case based planner BDIs, which improve their return output with the number of cases. The different radii show the number of plans previously carried out, and the great efficiency reached by the same plan in the specific number of plans carried out.

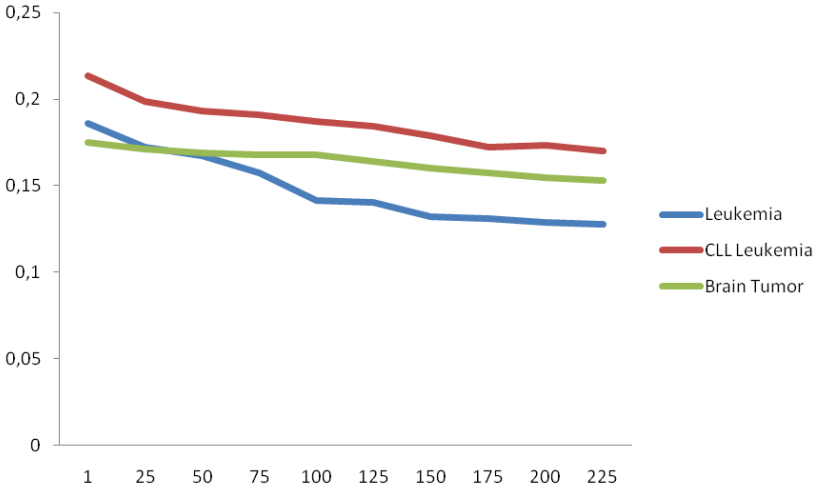

Fig 8. Evolution of the performance based on the filtering metric for the different case studies

In Figure 9 it is possible to observe the performance of IBO for each agent 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 multi-agent system. Each of the agents of the organization layer selects the agents from the analysis layer of the previous plans 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 9, while the agent from the organization layer can be seen at the top. In each chart the efficiency measure used is shown. The circle for the case based planner BDI agent is the highest efficiency according to the definitions applied.

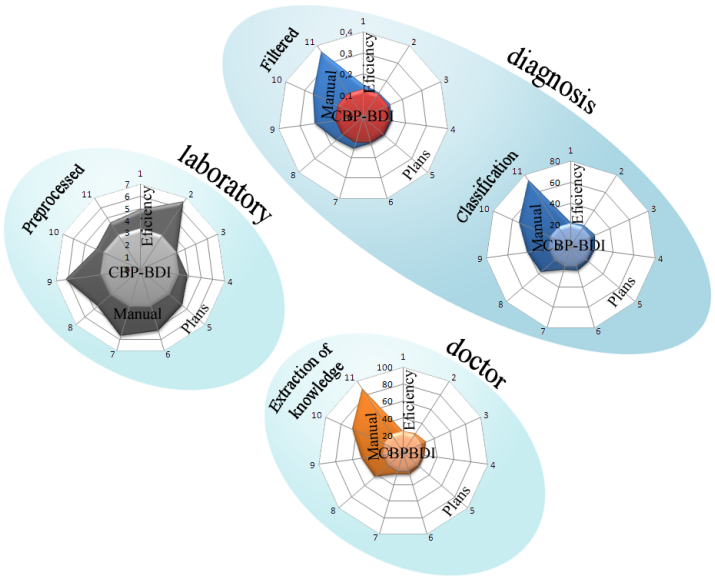

Fig. 9. Comparison of the performance between the manual and the automatic (with IBO) planning

With regards to the classification process, we were able to obtain promising results for each of the case studies. As shown in figures 10a 10b 10c, the probes recovered by the knowledge extraction agent are those that provide the relevant information that makes it possible to classify new individuals. As an example, it is possible to observe the classification carried out by the CLL leukemia patients, for one of its subtypes, and for patients with Nonclassic glioblastoma. The probes were extracted based on the information recovered with CART. In the first image, it is possible to see the 3 probes that best characterize the patients with CLL leukemia. As can be seen, the individuals of the CLL subtype are separated from the rest, which indicate that there are certain probes can be used as indicators to classify this subtype of individuals. In the second image, we can see the decision tree saves that distinguish the patients with the 3 subtypes of leukemia. In this case, the three first probes retrieved allow the separation from the rest of the groups. It can be seen how the elypsoids group at least the 50% of the individuals of each of the types. As shown, the individuals are separated in space. Finally, the last image represents the patients with anaplastic and oligodendroglioma tumors. In this last case, it is easy to observe that the clusters are clearly separable.

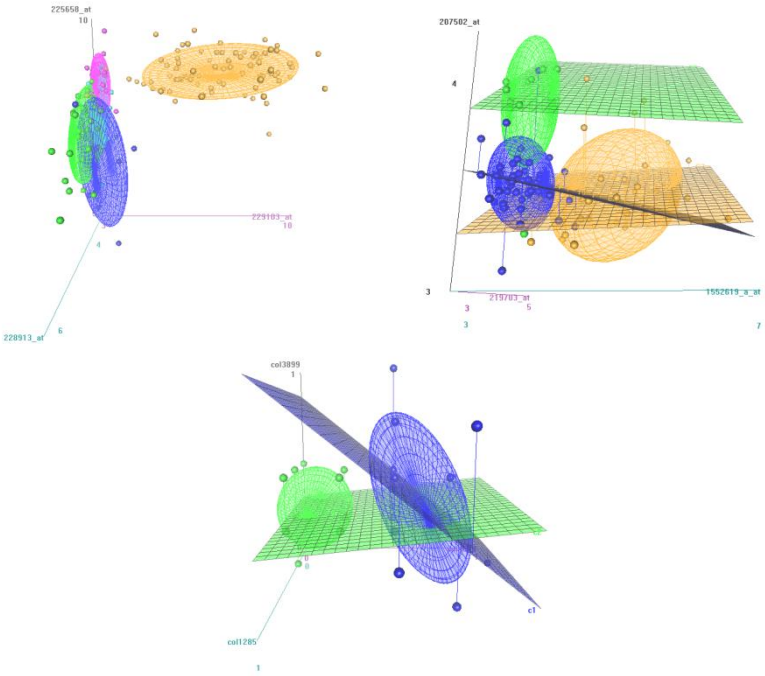

Fig. 10. Patients with CLL leukemia, CLL leukemia subtype and patients with anaplastic and oligodendrogliomas tumors.

The Bayesian networks make it possible to generate the plans that best adjust to the different case studies, allowing the generation of new plans based on existing information without actually needing an existing memory of plans. If we follow the same procedure as the one established for selecting the parameters, it would be necessary to have an extensive memory plans and the definition of a mechanism for carrying out the composition of efficient plans for generating a new and more efficient plan.

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 is capable of learning based on previous experiences and of generating new behaviors and, as a result, creating an application that adapts to the information that characterizes the case studies.

IBO distributes the functionality among Web services, automatically calculates the expression analysis and allows the classification of patients with different pathologies from the data obtained from the microarray data. IBO improves the performance provided by the manual procedure for selecting workflow analyses. The separation of the functionality into services makes it possible to very simply extend the system, thus allowing the incorporation of new services that facilitate the analysis of the data.

Capítulo VIII

Referencias

VNiVERSiDAD
D SALAMANCA

[Aamodt, Plaza, 1994] Aamodt A. Plaza E. (1994) "Case-Based Reasoning: foundational Issues, Methodological Variations, and System Approaches", AICOM. Vol. 7 (1).

[Aaronson *et al.*, 1993] Aaronson J.S., Juergen H., Overton G.C. (1993) "Knowledge Discovery in GENBANK". In Proceedings of the First International Conference on Intelligent Systems for Molecular Biology. 3–11.

[Aha, Kibler, 1991] Aha D., Kibler D. (1991) "WInstance-based learning algorithms", Machine Learning. Vol. 6 37-66

[Affymetrix] Affymetrix. GeneChip® Human Genome U133 Arrays http://www.affymetrix.com/support/technical/datasheets/hgu133arrays_datasheet.pdf último acceso marzo 2010

[Affymetrix b] Affymetrix (2005) Guide to Probe Logarithmic Intensity Error (PLIER) Estimation (2005) http://www.affymetrix.com/support/technical/technotes/plier_technote.pdf último acceso marzo 2010

[Affymetrix c] http://www.affymetrix.com/about_affymetrix/media/image-library.affx

[Akhbardeh *et al.*, 2008] Akhbardeh A., Nikhil, Koskinenb P.E., Yli-Harja O. (2008) "Towards the experimental evaluation of novel supervised fuzzy adaptive resonance theory for pattern classification", Pattern Recognition Letters. Vol. 29 (8) 1082-1093

[Ardanuy, Martín, 2003] Ardanuy R, Martín Q. (2003) "Estadística". Universidad de Salamanca

[Ardissono, 2004] Ardissono L., Petrone G., Segnan M. (2004) "A conversational approach to the interaction with Web Services", Computational Intelligence. Vol. 20 693-709.

[Armstrong, Van de Wiel, 2004] Armstrong N.J., Van de Wiel M.A. (2004) "Microarray data analysis: From hypotheses to conclusions using gene expression data", Cellular Oncology. Vol. 26 (5-6) 279-290.

[Arshadi, Jurisica, 2005] Arshadi N., Jurisica I. (2005) "Data Mining for Case-Based Reasoning in High-Dimensional Biological Domains", IEEE Transactions on Knowledge and Data Engineering. Vol. 17 (8) 1127-1137.

[Avogadri, Valentini, 2009] Avogadri R., Valentini G. (2009) "Fuzzy ensemble clustering based on random projections for DNA microarray data analysis". Artificial Intelligence in Medicine. 45 (2) 173-183

[Bajo *et al.*, 2006] Bajo J., De Paz J., Tapia D., Corchado J., (2006) "Deliberative Agents for Distributed Monitoring and Evaluation of the Air-Sea Interaction", In Proceedings of INADIS 2006. Vol. 4224 23–28

[Bajo *et al.*, 2007d] Bajo J., De Paz J., Tapia D., Corchado J. (2007), "Deliberative Agents for Distributed Monitoring and Evaluation of the Air-Sea Interaction". International Journal of Computer Science (INFOCOMP) 16-25

[Bajo *et al.*, 2007c] Bajo J., Tapia D., de Luis A., De Paz F., Corchado J. (2007) "Hybrid Architecture for a Reasoning Planner Agent" In Proceedings of KES-HAIS 2007. Vol. 4693 461-468

[Bajo *et al.*, 2007a] Bajo J., Corchado J., Tapia D., Rodríguez S., De Paz J., Sánchez J., Saavedra A., (2007) "Arquitectura Multiagente para Entornos Dinámicos: Tecnología e Inteligencia Aplicadas" In Proceedings of CEDI'07 - UCAMI'07. 25-34

[Bajo *et al.*, 2007b] Bajo J., Julián V., Corchado J., Carrascosa C., De Paz Y., Botti V., De Paz J. (2008) "An Execution Time Planner for the ARTIS Agent Architecture" Engineering Applications of Artificial Intelligence. Vol. 21 (5) 769-784

[Bajo *et al.*, 2009] Bajo J., De Paz J.F., De Paz Y., Corchado J.M. (2009) "Integrating Case Planning and RPTW Neuronal Networks to Construct an Intelligent Environment for Health Care." Expert Systems with Applications. Vol. 36 (3) 5844-5858.

[Baladandayuthapani *et al.,* 2007] Baladandayuthapani V., Ray S., Mallick B.K. (2005) "Bayesian Methods for DNA Microarray Data Analysis", Handbook of Statistics. Vol. 25 (1) 713-742

[Bauer, Huget, 2003] Bauer B., Huget M.P. (2003) "FIPA Modeling: Agent Class Diagrams". Working Draft, foundation for Intelligent Physical Agents. www.auml.org.

[Bianchia *et al.,* 2007] Bianchia D., Calogero R., Tirozzi B. (2007) "Kohonen neural networks and genetic classification", Mathematical and Computer Modelling. Vol. 45 (1-2) 34-60

[Bolstada *et al.*, 2007] Bolstada A.K., Van Veen B.D., Nowak R.D., Wakai R.T. (2007) "An expectation-maximization algorithm for space-time sparsity regularization of the MEG inverse problem", In Proceedings of international Congress Series Vol. 1300, 113-116

[Breiman *et al.*, 1984] Breiman L., Fried J.H., Olshen R.A., Stone C.J. (1984) "Classification and regression trees", Wadsworth International Group.

[Brooks, 1990] Brooks R.A. (1990) "'Elephants Don't Play Chess', DESIGNING AUTONOMOUS AGENTS: Theory and Practice From Biology to Engineering and Back", 3-17

[Brooks, 1991] Brooks R.A. (1991) "Intelligence without reason". In Proceedings of the Twelve International Joint Conference on Artificial Intelligence. 569-595

[Brunelli, 2001] Brunelli R. (2001) "Histogram Analysis for Image Retrieval". Pattern Recognition. Vol. 34, 1625-1637

[Bussman, Müller, 1993] Bussman S., Müller H.J. (1993) "A communication achitecture for cooperating agents". Computers and artificial intelligence. Vol. 12(1), 37-54.

[Campos, Ricardo, 2008] Campos R., Ricardo M. (2008) "A fast algorithm for computing minimum routing cost spanning trees" A fast algorithm for computing minimum routing cost spanning trees. Vol. 52 (17) 3229-3247

[Carbó *et al.*, 2005] Carbó J., Molina J.M., Dávila J. (2005) "Fuzzy Referral based Cooperation in Social Networks of Agents", AI Communications. Vol. 18, (1) 1-13

[Carpenter, Grossberg, 1987] Carpenter G.A., Grossberg S., (1987) "The ART of adaptive pattern recognition by a self-organizing neural network." IEEE Trans. Computer, 77–88.

[Cestnik *et al.*, 1987] Cestnik B., Kononenko I., Bratko I. (1987). ASSISTANT 86: "A knowledge elicitation tool for sophisticated users." Progress in machine learning. 31-45

[Chua *et al.*, 2008] Chua, A., Ahna, H., Halwanb, B., Kalminc, B., Artifond, E., Barkune, A., Lagoudakisf, M. and Kumar, A. (2008) "A decision support system to facilitate management of patients with acute gastrointestinal bleeding" Artificial Intelligence in Medicine. Vol. 42 (3) 247-259.

[Cohen, 1995] Cohen W.W. (1995) "Fast effective rule induction" In Proceedings of the 12th International Conference on Machine Learning, Morgan Kaufmann 115-123

[Corchado, Laza, 2003] Corchado, J. M., Laza, R. (2003) "Constructing Deliberative Agents with Case-based Reasoning Technology", International Journal of Intelligent Systems. Vol. 18, (12) 1227-1241.

[Corchado *et al.*, 2005] Corchado J.M., Molina J.M., Pavón J. (2009) "Agentes Software y Sistemas Multi-Agente", Person. 29-63.

[Corchado *et al.*, 2009] Corchado J.M., De Paz J.F., Rogríguez S., Bajo J. (2009) "Model of experts for decision support in the diagnosis of leukemia patients" Artificial Intelligence in Medicine. Vol. 46 (3) 179-200.

[Corchado *et al.*, 2008a] Corchado J.M., Bajo J., De Paz Y. and Tapia D.I. (2008) "Intelligent Environment for Monitoring Alzheimer Patients, Agent Technology for Health Care". Decision Support Systems. Vol. 44 (2) 382-396.

[Corchado *et al.*, 2008b] Corchado J. M., Gonzalez-Bedia M., De Paz Y. ,Bajo J. y De Paz J.F, (2008) "Replanning mechanism for deliberative agents in dynamic changing environments". Computational Intelligence. Vol. 24 (2) 77-107

[Costache *et al.*, 2009] Costache G.N., Corcoran P., Puslecki P. (2009) "Combining PCA-based datasets without retraining of the basis vector set", Pattern Recognition Letter. Vol 30 (16) 1441-1447

[Cox, Ferry, 1993] Cox T.F., Ferry G. (1993) "Discriminant analysis using non-metric multidimensional scaling". Pattern Recognition. Vol. 26 (1) 145-153

[Cvijović *et al.*, 2005] Cvijović. Z., Radenković. G., Maksimović V., Dimčić B. (2005) "Application of ANOVA method to precipitation behaviour studies". Materials Science and Engineering A. Vol. 397 (1-2) 195-203

[Dantzig *et al.*, 1954] Dantzig G.B., Fulkerson D.R., Johnson S.M. (1954) "Solution of a large-scale traveling-salesman problem", Operations Research, 2, 393-410.

[David et al, 2004] David N., Marietto M.B., Sichman J.S., Coelho H. (2004) "The Structure and Logic of Interdisciplinary. Research in Agent-Based Social Simulation". Journal of Artificial Societies and Social Simulation. Vol. 7 (3).

[Decker, 1987] Decker K. S., (1987) "Distributed Problem-Solving Techniques: A Survey", IEEE Transactions on Systems, Man, and Cybernetics. Vol. 17 (5) 729-740

[Dellaert, 2002] Dellaert F. (2002) "The Expectation Maximization Algorithm", Technical Report. Georgia Institute of Technology

[Drogoul *et al.*, 1995] Drogoul A., Corbara B., Lalande S. (1995) "{MANTA}: New Experimental Results on the Emergence of (Artificial) Ant Societies" Artificial Societies: The Computer Simulation of Social Life, 190-211

[Duda *et al.*, 1973] Duda R.O., Hart P. (1973) "Pattern classification and Scene Analysis." New York: John Wisley & Sons.

[EURESCOM, 2001] EURESCOM (2001) MESSAGE: "Methodology for engineering systems of software agents". Technical report P907-TI1, EURESCOM.

[Epstein, Axtell, 1996] Epstein J.M., Axtell, R.L. (1996) "Growing Artificial Societies: Social science from the bottom up". MIT Press. 224.

[Erl, 2005] Erl, T. (2005) "Service-Oriented Architecture (SOA): Concepts, Technology, and Design", Prentice Hall PTR.

[Ferguson, 1992] Ferguson I. A. (1992) "TouringMachines: An Architecture for Dynamic, Rational, Mobile Agents". PhD thesis, clare Hall, University of Cambridge, UK.

[Fikes, Nilsson, 1971] Fikes R.E. y Nilsson N.J. (1971) "STRIPS: A new approach to the application of theorem proving to problem solving". Artificial Intelligence. Vol. 2 189-208.

[FIPA, 2002] FIPA, (2002) Foundation for Intelligent Physical Agents, FIPA ACL Message Structure Specification. http://www.fipa.org/specs/fipa00061/SC00061G.pdf

[Foster *et al.*, 2002] Foster I., Kesselman C., Nick J., Tuecke S. (2002) "The Physiology Of The Grid: An Open Grid Services Architecture For Distributed Systems Integration". Technical Report of the Global Grid Forum.

[François *et al.*, 1992] François P., Cremilleux B., Robert C., Demongeot J. (1992) "MENINGE: A medical consulting system for child's meningitis. Study on a series of consecutive cases". Artificial Intelligence in Medicine. Vol. 32 (2) 281-292.

[Friedman *et al.*, 1996] Friedman N., Geiger D., Goldszmidt M., (1996) "Bayesian Network Classifiers", Cop. Kluwer Academic Publishers.

[Fritzke, 1993] Fritzke B. (1993) "Growing Cell Structures – A Self-organizing Network for Unsupervised and Supervised Learning". International Computer Science Institute.

[Fritzke, 1995] Fritzke B. (1995) "A growing neural gas network learns topologies". G. Advances in Neural Information Processing Systems 7, 625-632

[Fu, Chen, 1993] Fu L., Chen T. (1993) "Sensitivity analysis for input vector in multilayer feedforward networks", In Proceedings of IEEE International Conference on Neural Networks. Vol. 1 215–218

[Furao *et al.*, 2007] Furao S., Ogura T., Hasegawa O. (2007) "An enhanced self-organizing incremental neural network for online unsupervised learning". Neural Networks, Vol. 20 893-903

[García *et al.*, 2005] García J., Berlanga A., Molina J.M., Casar, J.R. (2005) "Methods for Operations Planning in Airport", Decision Support Systems, Applied Intelligence. Vol. 22 (3) 183--206

[Georgeff, Lansky, 1987] Georgeff M.P., Lansky, A.L. (1987). "Reactive reasoning and planning." In *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87)*, 677-682.

[Georgeff, Rao, 1998] Georgeff M.P. y Rao A. (1998) "Rational Software Agents: From Theory to Practice. In Agent Technology: Foundations, applications and markets". Jennings & Wooldridge (Eds). 139-160.

[Giorginia *et al.*, 2005] Giorginia, P., Mylopoulos, J. and Sebastiani, R. (2005) "Goal-oriented requirements analysis and reasoning in the Tropos term methodology", Engineering Applications of Artificial Intelligence. Vol. 18 (2) 159-171

[Glez-Bedia, Corchado, 2002] Glez-Bedia M., Corchado J. (2002) "A planning strategy based on variational calculus for deliberative agents", Computing and Information Systems Journal. Vol. 10 (1) 2-14.

[Golub *et al.*, 1999] Golub T. R., Slonim D. K., Tamayo P., Huard C., Gaasenbeek M., Mesirov J. P., Coller H., Loh M. L., Downing J. R., Caligiuri M. A., Bloomfield C. D., Lander E. S. (1999) "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring" Science. Vol. 286 531-537

[Groenen, Jajuga, 2001] Groenen P.J.F., Jajuga K. (2001) Fuzzy clustering with squared Minkowski distances. Fuzzy Sets and Systems. Vol. 2 (2) 227-237.

[De Haan *et al.*, 2009] De Haan J.R., Bauerschmidt S., van Schaik R.C., Piek E., Buydens L.M.C., Wehrens R., (2009) "Robust ANOVA for microarray data", Chemometrics and Intelligent Laboratory Systems. Vol. 98 (1) 38-44.

[Hartigan, Wong, 1979] Hartigan, J. A. and Wong, M. A. (1979) "A K-means clustering algorithm", Applied Statistics. Vol. 28, 100-108.

[Hodge, Austin, 2001] Hodge V.J. and Austin J. (2001) "Hierarchical Growing Cell Structures: TreeGCS". IEEE Transactions on Knowledge and Data Engineering. Vol. 13 (2) 207-218.

[Holder *et al.*, 2001] Holder, D., Raubertas, R. F., Pikounis, V. B., Svetnik, V., and Soper, K. (2001) "Statistical analysis of high density oligonucleotide arrays: a SAFER approach". In proceedings of the ASA Annual Meeting, Atlanta. 1633-1640

[Holmes *et al.*, 2007] Holmes G., Hall M., Prank E. (2007) "Generating Rule Sets from Model Trees", Advanced Topics in Artificial Intelligence. (1747/1999) 1-12

[Holte, 1993] Holte R.C. (1993). "Very simple classification rules perform well on most commonly used datasets". Machine Learning, Vol. 11 63-91.

[Horng *et al.*, 2003] Horng J.T., Wu L.C., Liu B.J., Kuo J.L. Kuo W.H., Zhange, J.J. (2009) "An expert system to classify microarray gene expression data using gene selection by decision tree". Expert Systems with Applications. Vol. 36 (5) 9072-9081.

[Hubbell, 2002] Hubbell E., Lie W.M., Mei R., (2002) "Robust estimators for expression analysis", Bioinformatics. Vol. 18 (12) 1585-1592

[Hunt *et al.*, 1966] Hunt E.B., Marin J., Stone P.J. (1966). "Experiments in induction" New York: Academic Press. New York.

[Irizarry *et al.*, 2003] Irizarry R.A., Hobbs B., Collin F., Beazer-Barclay Y.D., Antonellis K.J., Scherf U. and T.P. Speed (2003) "Exploration, Normalization, and Summaries of High density Oligonucleotide Array Probe Level Data", Biostatistics. Vol. 4 249-264.

[Ivanciuc, 2007] Ovidiu I. (2007) "Applications of Support Vector Machines in Chemistry", Reviews in Computational Chemistry. Vol. 23 291-400

[Jennings, 1993] Jennings N.R. (1993) "Specification and implementation of a belief-desire-joint-intention architecture for collaborative problem solving". Journal of Intelligent and Cooperative Information Systems. Vol. 2 289-318.

[Jennings *et al.*, 1998] Jennings, N. R., Sycara, K., & Wooldridge, M. (1998). "A Roadmap of Agent Research and Development.", Autonomous Agents and Multi-Agent Systems Journal . Vol. 1 (1) 7-38.

[Joh, 1997] Joh D.Y. (1997) "CBR in a Changing Environment", Case-based Reasoning Research and Development ICCBR-97. Vol. 126.

[Jurečkováa, Picek, 2007] Jurečkováa J., Picek J. (2007) "Shapiro–Wilk type test of normality under nuisance regression and scale", Computational Statistics & Data Analysis. Vol. 51 (10) 5184

[Jurisica, Glasgow, 2004] Jurisica I., Glasgow J. (2004) "Applications of case-based reasoning in molecular biology", Artificial Intelligence Magazine, Special issue on Bioinformatics. Vol. 25 (1) 85-95.

[Kaufman, Rousseeuw, 1990] Kaufman, L., Rousseeuw, P.J. (1990) "Finding Groups in Data: An Introduction to Cluster Analysis". Wiley, New York

[Keith *et al.*, 1997] Keith D., Pannu A., Sycara K. y Williamson M. (1997) "Designing Behaviors for Information Agents", AUTONOMOUS AGENTS '97.

[Kerr *et al.*, 2008] Kerr G., Ruskina H.J., Cranea M. and Doolan P. (2008) "Techniques for clustering gene expression data", Computers in Biology and Medicine. Vol. 38 (3) 283-293.

[Kibriya *et al.*, 2007] Kibriya M.G., Jasmine F., Argos M., Verret W.J., Rakibuz-Zaman M., Ahmed A., Parvez F., Ahsan H. (2007) "Changes in gene expression profiles in response to selenium supplementation among individuals with arsenic-induced pre-malignant skin lesions", Toxicology Letters. Vol 169 (2) 162-176

[Kruskal, Wallis, 1952] Kruskal W.H., Wallis W.A. (1952) "Use of Ranks in One-Criterion Variance Analysis", Journal of the American Statistical Association. Vol. 47 (260) 583-621.

[Kohonen, 1982] Kohonen, T. (1982) "Self-organized formation of topologically correct feature maps". Biological Cybernetics. 59-69

[Kokko, 2006] Kokko A, (2006) "Expression Microarray Technology as a Tool in Cancer Research". TKK Dissertations 40

[Kolodner, 1983a] Kolodner J. (1983a) "Maintaining organization in a dynamic long-term memory", Cognitive Science. Vol. 7. 243-280.

[Kolodner, 1983b] Kolodner J. (1983b) "Reconstructive memory, a computer model:" Cognitive Science. Vol. 7. (4). 281-328.

[Kolodner, 1993] Kolodner J. (1993) "Maintaining organization in a dynamic long-term memory". Morgan Kaufmann.

[Kuo *et al.*, 2003] Kuo, C.D., Chen G.Y., Wang Y.Y., Hung M.J., Yang J.L. (2003) "Characterization and quantification of the return map of RR intervals by Pearson coefficient in patients with acute myocardial infarction", Autonomic Neuroscience. Vol. 105 (2) 145-152

[Labidi, Lejouad, 1993]. Labidi, S. y Lejouad, W. (1993) "De l'Intelligence Artificielle Distribuée aux Systèmes Multi-Agents".

[Lander *et al.,* 2001] Lander E.S., Linton L.M., Birren B., Nusbaum C., Zody M.C., Baldwin J., Devon K., Dewar K., Doyle M., FitzHugh W., Funke R., Gage D., Harris K., Heaford A., Howland J., Kann L., Lehoczky J., LeVine R., McEwan P., McKernan K., Meldrim J., Mesirov J.P., Miranda C., Morris W., Naylor J., Raymond C., Rosetti M., Santos R., Sheridan A., Sougnez C., Stange-Thomann N., Stojanovic N.,

Subramanian A., Wyman D., Rogers J., Sulston J., Ainscough R., Beck S., Bentley D., Burton J., Clee C., Carter N., Coulson A., Deadman R., Deloukas P., Dunham A., Dunham I., Durbin R., French L., Grafham D., Gregory S., Hubbard T., Humphray S., Hunt A., Jones M., Lloyd C., McMurray A., Matthews L., Mercer S., Milne S., Mullikin J.C., Mungall A., Plumb R., Ross M., Shownkeen R., Sims S., Waterston R.H., Wilson R.K., Hillier L.W., McPherson J.D., Marra M.A., Mardis E.R., Fulton L.A., Chinwalla A.T., Pepin K.H., Gish W.R., Chissoe S.L., Wendl M.C., Delehaunty K.D., Miner T.L., Delehaunty A., Kramer J.B., Cook L.L., Fulton R.S., Johnson D.L., Minx P.J., Clifton S.W., Hawkins T., Branscomb E., Predki P., Richardson P., Wenning S., Slezak T., Doggett N., Cheng J.F., Olsen A., Lucas S., Elkin C., Uberbacher E., Frazier M., Gibbs R.A., Muzny D.M., Scherer S.E., Bouck J.B., Sodergren E.J., Worley K.C., Rives C.M., Gorrell J.H., Metzker M.L., Naylor S.L., Kucherlapati R.S., Nelson D.L., Weinstock G.M., Sakaki Y., Fujiyama A., Hattori M., Yada T., Toyoda A., Itoh T., Kawagoe C., Watanabe H., Totoki Y., Taylor T., Weissenbach J., Heilig R., Saurin W., Artiguenave F., Brottier P., Bruls T., Pelletier E., Robert C., Wincker P., Smith D.R., Doucette-Stamm L., Rubenfield M., Weinstock K., Lee H.M., Dubois J., Rosenthal A., Platzer M., Nyakatura G., Taudien S., Rump A., Yang H., Yu J., Wang J., Huang G., Gu J., Hood L., Rowen L., Madan A., Qin S., Davis R.W., Federspiel N.A., Abola A.P., Proctor M.J., Myers R.M., Schmutz J., Dickson M., Grimwood J., Cox D.R., Olson M.V., Kaul R., Raymond C., Shimizu N., Kawasaki K., Minoshima S., Evans G.A., Athanasiou M., Schultz R., Roe BA., Chen F., Pan H., Ramser J., Lehrach H., Reinhardt R., McCombie W.R., de la Bastide M., Dedhia N., Blöcker H., Hornischer K., Nordsiek G., Agarwala R., Aravind L., Bailey J.A., Bateman A., Batzoglou S., Birney E., Bork P., Brown D.G., Burge C.B., Cerutti L., Chen H.C., Church D., Clamp M., Copley R.R., Doerks T., Eddy S.R., Eichler E.E., Furey T.S., Galagan J., Gilbert J.G., Harmon C., Hayashizaki Y., Haussler D., Hermjakob H., Hokamp K., Jang W., Johnson L.S., Jones T.A., Kasif S., Kaspryzk A., Kennedy S., Kent W.J., Kitts P., Koonin E.V., Korf I., Kulp D., Lancet D., Lowe T.M., McLysaght A., Mikkelsen T., Moran J.V., Mulder N., Pollara V.J., Ponting C.P., Schuler G., Schultz J., Slater G., Smit A.F., Stupka E., Szustakowski J., Thierry-Mieg D., Thierry-Mieg J., Wagner L., Wallis J., Wheeler R., Williams A., Wolf Y.I., Wolfe K.H., Yang S.P., Yeh R.F., Collins F., Guyer M.S., Peterson J., Felsenfeld A., Wetterstrand K.A., Patrinos A., Morgan M.J., de Jong P., Catanese J.J., Osoegawa K., Shizuya H., Choi S., Chen Y.J (2001) "Initial sequencing and analysis of the human genome", Nature. Vol. 409: 860-921

[Laza, Corchado, 2001] Laza R., Corchado J.M. (2001) "Creation of deliberative agents using a CBR model ", Computing and Information Systems Journal. Vol. 8 (2) 33-39.

[Leake, Kendall-Morwick, 2008] Leake, D. and Kendall-Morwick, J. (2008) "Towards Case-Based Support for e-Science Workflow Generation by Mining Provenance" Advances in Case Base Resoning. Vol. (5239) 269-283.

[Lee *et al.*, 2005] Lee J.W., Lee J.B., Park M., Heun S., (2005) "Song An extensive comparison of recent classification tools applied to microarray data", Computational Statistics & Data Analysis. Vol. 48 (4-1) 869-885.

[Lessmann, Voß, 2009] Lessmann S., Voß S. (2009) "A reference model for customer-centric data mining with support vector machines", European Journal of Operational Research. Vol. 199 (2) 520-530.

[Li *et al.*, 2009] Li H., Liang Y, Xu Q. (2009) "Support vector machines and its applications in chemistry", Chemometrics and Intelligent Laboratory Systems. Vol. 95 (2) 188-198

[Lipshutz *et al.*, 1999] Lipshutz R.J., Fodor S.P.A., Gingeras T.R., Lockhart D.H. (1999) "High density synthetic oligonucleotide arrays", Nature Genetics. Vol. 21 (1) 20-24.

[Liu, 2007] Liu, X. (2007) "A Multi-Agent-Based Service-Oriented Architecture for Inter-Enterprise Cooperation System", In Proceedings of the Second international Conference on Digital Telecommunications (ICDT'07). IEEE Computer Society.

[Lo, 2008] Lo S. (2008) "Web service quality control based on text mining using support vector machine", Expert Systems with Applications. Vol. 34 (1) 603-610

[López *et al.*, 2002] López M., Mallorquín P., Vega M. (2002) "Microarrays y Biochips de ADN", Genoma España.

[Man, Qin, 2007] Ma J., Qin Z.S. (2007) "Different normalization strategies for microarray gene expression traits affect the heritability estimation". BMC Proceedings.

[Mann, Whitney, 1947] Mann H.B., Whitney D.R. (1947) "On a test o whether one of two random variables is stochastically larger than the other", Annals of Mathematical Statistics. 50-60

[Maes, 1989] Maes P. (1989) "Situated Agents Can Have Goals", Designing Autonomous Agents: Theory and Practice From Biology to Engineering and Back, 49-71, Maes,

[Margalit *et al.*, 2005] Margalit O., Somech R., Amariglio N., Rechav G., (2005) "Microarray based gene expression profiling of hematologic malignancies: basic concepts and clinical applications", Blood Reviews. Vol. 19 (4) 223-234.

[Martinetz, Schulten., 1991] Martinetz T., and Schulten K. (1991) "A neural-gas network learns topologies", Artificial Neural Networks. Vol 1. 397-402

[Müller *et al.*, 1994] Müller J.P., Pischel M., Thiel M. (1994) "A pragmatic approach to modelling autonomous interacting systems", In Pre-proceedings of the 1994 Workshop on Agent Theories, Architectures, and Languages. 226-240.

[Müller, Pischel, 1994] Müller J.P., Pischel M. (1994) "Modelling interacting agents in dynamic environments". In Proceedings of the Eleventh European Conference on Artificial Intelligence (ECAI-94), 709-713.

[Müller, 1996] Müller J.P. (1996) "The Design of Intelligent Agents: A Layered Approach", Lecture Notes in AI Vol. 1177.

[Murthy, 1994] Murthy S.K., Kasif, S., Salzberg, S. (1994) “A system for the induction of obliquedecision trees.” Journal of Articial Intelligence Research. Vol. 2 1-33.

[Nau *et al.*, 1999] Nau D., Cao Y., Lotem A., Muñoz-Avila H. (1999) “SHOP: Simple Hierarchical Ordered Planner”, In Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence. 968-975

[Nau *et al.*, 2003] Nau D., Au T.C., Ilghami O., Kuter U., Murdock J.W., Wu D., Yaman F. (2003) “SHOP2: an HTN planning system”, Journal of Artificial Intelligence Research. Vol. 20 (1) 379-404

[Noriega *et al.*, 2009] Noriega N.C., Kohama S.G. and Urbanski H.F. (2009) “Gene expression profiling in the rhesus macaque: Methodology, annotation and data interpretation” Methods. Vol. 49 (1) 42-49.

[Nutt *et al.*, 2003] 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. (2003) “Gene expression-based classification of malignant gliomas correlates better with survival than histological classification”, Molecular Biology and Genetics. Vol. 63 (7) 1602-1607.

[Nwana, 1996] Nwana H. (1996) "Software Agents: An Overview" Knowledge Engineering Review. Vol. 11, 205-244.

[Odell *et al.*, 2004] Odell J., Levy R., Nodine M. (2004) “FIPA Modeling TC: Agent Class Superstructure Metamodel. FIPA meeting and interim work”.

[Odell, Huget, 2003] Odell J., Huget M.P. (2003) “FIPA Modeling: Interaction Diagrams”.

[Oliva *et al.*, 2008] Oliva, E., Natali, A., Ricci, A., and Viroli, M. (2008) “An Adaptation Logic Framework for {J}ava-based Component Systems. Journal of Universal” Computer Science. Vol. 14 (13) 2158-2181.

[Olson, 2006] Olson N.E. (2006) “The Microarray Data Analysis Process: From Raw Data to Biological Significance”. NeuroRX. Vol. 3 (3) 373-383.

[Pan, 2002] Pan (2002) “A comparative review of statistical methods for discovering differentially expressed genes in replicated microarray experiments”. Bioinformatics. Vol. 48 (4) 546-554

[Pavlidis, 2003] Pavlidis P. (2003) “Using ANOVA for gene selection from microarray studies of the nervous system”. Methods. Vol. 31 (4) 282-289

[Pavón *et al.*, 2007] Pavón J., Gómez J., Fernández A., Valencia J. (2007) “Development of intelligent multi-sensor surveillance systems with agents”, Journal of Robotics and Autonomous Systems, Vol. 55 (12) 892-903

[Pavón *et al.*, 2007] Pavón J., Gómez-Sanza J., Fernández-Caballero A., Valencia-Jiménez, J.J. (2007) "Development of intelligent multisensor surveillance systems with agents Robotics and Autonomous Systems" Vol. 55 (12) 892-903

[Pavón *et al.*, 2008] Pavón, J., Arroyo, M., Hassan, S., Sansores, C. (2008) "Agent-based modelling and simulation for the analysis of social patterns". Pattern Recognition Letters 29 1039-1048

[Pawlak, 1982] Pawlak Z. (1982). "Rough Sets", International Journal of Information & Computer Sciences. Vol. 11, 341-356.

[Pawlak, 1997] Pawlak Z. (1997). "Rough Sets, approach to knowledge-based decision support", European Journal of Operational Research. Vol. 99 48-57.

[Pednault, 1989] Pednault E.P. (1989) "ADL: exploring the middle ground between STRIPS and the situation calculus", In Proceedings of the first international conference on Principles of knowledge representation and reasoning. 324-332

[Penberthy, 1992] Penberthy J.S. and Weld D. (1992) "UCPOP: A Sound, Complete, Partial-Order Planner for ADL", In Proceedings of KR-92. 103-114.

[Platt, 1999] John C. (1999) "Platt Fast training of support vector machines using sequential minimal optimization", Advances in Kernel Methods. 185 - 208

[Quackenbush, 2001] Quackenbush J. (2001) "Computational analysis of microarray data", Nature Review Genetics. Vol. 2 (6) 418-427.

[Quinlan, 1979] Quinlan J.R. (1979) "Discovering rules by induction from large collections of examples", Expert systems in the micro electronic age. 168-201.

[Quinlan, 1993] Quinlan, J. R. (1993) "C4.5: Programs For Machine Learning." Morgan Kaufmann Publishers Inc.

[Quinlan, 2002] Kohavi, R., Ross Quinlan, R. (1999) "Decision Tree Discovery" Handbook of Data Mining and Knowledge Discovery. 267-276.

[R, 2009] http://www.r-project.org/

[Rao, Georgeff, 1991] Rao A.S., Georgeff M.P. (1991) "Modeling Rational Agents within a BDI-Architecture". In proceedings Second International Conference on Principles of Knowledge Representation and Reasoning (KR91). 473-484.

[Rao, Georgeff, 1995] Rao A.S., Georgeff M.P. (1995) "BDI Agents from Theory to Practice". In Proceedings of the First International Conference on Multi-Agents Systems".

[Riesbeck, Schank, 1989] Riesbeck C.K., Schank R.C. (1989) "Inside Case-Based Reasoning". Lawrence Erlbaum Ass. Hillsdale.

[Rigole *et al.*, 2002] Rigole P., Holvoet T., Berbers, Y. (2002) "Using Jini to Integrate Home Automation in a Distributed Software-System" In proceedings 4th

international Workshop on Distributed Communities on the Web. Vol. 2468 291-304.

[Riva *et al.*, 2005] Riva A, Carpentier AS, Torrésani B. and Hénaut A. (2005) "Comments on selected fundamental aspects of microarray analysis Comments on selected fundamental aspects of microarray analysis", Computational Biology and Chemistry. Vol. 29 (5) 319-336.

[Riverola *et al.*, 2006] Riverola, F., Díaz, F., and Corchado, J.M. (2006) "Gene-CBR: a case-based reasoning tool for cancer diagnosis using microarray datasets", Computational Intelligence. Vol. 22 (3-4) 254-268.

[Rung *et al.*, 2009] Rung-Wei Po, Yuh-Yuan Guh, and Miin-Shen Yang, (2009) "A new clustering approach using data envelopment analysis", European Journal of Operational Research. Vol. 199 (1) 276-284

[Saitou, Nie, 1987] Saitou N., Nie M. (1987) "The neighbor-joining method: A new method for reconstructing phylogenetic trees". Mol. Biol. Vol. 4 406-425.

[Salvador, 2003] Salvador M., (2003) "Análisis de Correspondencias", Estadística. http://www.5campus.com/leccion/correspondencias último acceso marzo 2010

[Sawa, Ohno-Machado, 2003] Sawa T., Ohno-Machado L. (2003) "A neural network based similarity index for clustering DNA microarray data", Computers in Biology and Medicine. Vol. 33 (1) 1-15

[Schelling, 1978] Schelling, T. (1978). "Micromotives and macrobehavior". New York: W. W. Norton.

[Shafiq *et al.,* 2006] Shafiq M.O., Ding Y., Fensel D. (2006) "Bridging Multi Agent Systems and Web Services: towards interoperability between Software Agents and Semantic Web Services", In Proceedings of the 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC'06). IEEE Computer Society, Washington, DC. 85-96.

[Shinawi, Cheung, 2008] Shinawi M., Cheung S.W. (2008) "The array CGHnext term and its clinical applications", Drug Discovery Today. Vol. 13 (17-18) 760-770.

[Shen, 2006] Shen F. (2006) "An algorithm for incremental unsupervised learning and topology representation". Ph.D. thesis. Tokyo Institute of Technology

[Simon, 2009] Simon R., (2009) "Analysis of DNA microarray expression data", Best Practice & Research Clinical Haematology. Vol. 22 (2), 271-282

[Sommaruga, 1993] Sommaruga, L., (1993) "Cooperative Heuristics for Autonomous Agents: an Artificial Intelligence Perspective", Tesis Doctoral. Universidad de Nottingham, 1993.

[Sridharan, 1986] Sridharan N., (1986) "Report on the 1986 Workshop on Distributed Artificial Intelligence", The AI Magazine. 75-85.

[Stevens *et al.*, 2004] Stevens R., McEntire R., Goble C., Greenwood M., Zhao J., Wipat A., Li P. (2004) "myGrid and the drug discovery process", Drug Discovery Today. BIOSILICO. Vol. 2 (4) 140-148.

[Sucar, 2009] Redes Bayesianas http://ccc.inaoep.mx/~esucar/Clases-mgp/caprb.pdf. último acceso marzo 2010

[Taniguchi *et al.*, 2008] Taniguchi M., Guan L.L., Basara J.A., Dodson M.V. and Moore S.S., (2008) "Comparative analysis on gene expression profiles in cattle subcutaneous fat tissues", Comparative Biochemistry and Physiology Part D: Genomics and Proteomics 3 (4) 251-256.

[Tapia *et al.*, 2006] Tapia D., Bajo J., De Paz J., Corchado J., (2006) "Hybrid Multiagent System for Alzheimer Health Care". In Proceedings of HAIS 2006.

[Tapia *et al.*, 2008] Tapia D.I., De Paz J.F., Rodríguez S., Bajo J. and Corchado J.M., (2008) "Multi-Agent System for Security Control on Industrial Environments", International Transactions on System Science and Applications Journal. Vol. 4 (3) 222-226.

[Theron, De Paz, 2006] Theron R., De Paz J.F. (2006) "Visual Sensitivity Analysis for Artificial Neural Networks". In Proceedings of IDEAL.Vol. (4224) 191–198.

[Tibshirani *et al.*, 2002] Tibshirani. R., Hastie. T., Narasimhan. B., Chu. C., (2002) "Diagnosis of multiple cancer types by shrunken centroids of gene expression", PNAS. Vol. 99 (10) 6567–6572

[Timofeev, 2004] Timofeev, R. (2004). "Classification and Regression Tress (CART) Theory and Applications" Master Thesis. CASE – Center of Applied Statistics and Economics. Berlin.

[Tusher *et al.*, 2000] Tusher. V.G., Tibshirani. R., Chu. G. (2000) "Significance analysis of microarrays applied to the ionizing radiation response", PNAS. Vol. 98 5116-5121.

[Ungera, 2005] Ungera T., Koradec Z., Lazarovd O., Terranod D., Sisodiad S.S., Mirnics K. (2005) "True and false discovery in DNA microarray experiments: Transcriptome changes in the hippocampus of presenilin 1 mutant mice". Methods. Vol. 37 (3) 261-273

[Van der Laan *et al.*, 2002] van der Laan M.J., Pollard K.S., Bryan J. (2002) "A New Partitioning Around Medoids Algorithm". Journal of Statistical Computation and Simulation. Vol. 73 (8) 575-584.

[Vapnikand, Lerner, 1963] Vapnikand V., Lerner A., (1963) "Automat Remote Control",Pattern Recognition Using Generalized Portrait Method. Vol. 24,774–780.

[Vittorini *et al.*, 2008] Vittorini P., Michettia M., di Orio, F. (2008) "A SOA statistical engine for biomedical data". Computer Methods and Programs in Biomedicine. Vol. 92 (1) 144-153.

[Waldinger, 1977] Waldinger R. (1977) "Achieving several foals simultaneously". In Machine Intelligence 8. Ellis Horwood Limited, Chichester.

[Walton *et al.,* 2006] Walton C. (2006) "Agency and the Semantic Web". Oxford University Press, Inc. 272

[Wang, 2009] Wang F., (2009) "Factor Analysis and Principal-Components Analysis", International Encyclopedia of Human Geography. 1-7

[Watson, Marir, 1994] Watson I. Marir F. (1994) "Case-Based Reasoning: A Review". The knowledge Engineering Review. Vol. 9. Nº3.

[Weld, 1994] Weld D.S. (1994) "An Introduction to Least Commitment Planning", AI Magazine. Vol 15 (4) 27-61.

[Weka, 2009] http://www.cs.waikato.ac.nz/ml/weka/

[Wooldridge, 2002] Wooldridge M. (2002) "An Introduction to MultiAgent Systems", Chichester, England: John Wiley & Sons

[Wooldridge, Jennings, 1995] Wooldridge M., Jennings N. R. (1995) "Intelligent Agents: Theory and Practice", The Knowledge Engineering Review. Vol. 10(2) 115-152.

[Xu, 1997] Xu, L. (1997) Bayesian Y.Y. (1997) "machine, clustering and number of clusters" Pattern Recognition Letters. Vol. 18 (11-13) 1167-1178

[Yang *et al.,* 2008] Yang Q., Li X, and Shi X. (2008) "Cellular automata for simulating land use changes based on support vector machines". Computers & Geosciences. Vol. 34 (6) 592-602.

Capítulo IX

Glosario

VNiVERSiDAD
D SALAMANCA

ACI (Análisis de Componentes Independientes)

ACP (Análisis de Componentes Principales)

ADL [Pednault, 1989] (Action Description Language)

ADN (Ácido DesoxirriboNucleico)

AGNES (Agglomerative Nesting)

ANOVA (ANalysis Of VAriance)

ARN (Ácido RiboNucleico)

ART (Adaptive Resonance Theory)

AUML (Agent UML)

BDI (Belief Desire Intention)

CART (Classication and Regression Trees)

CBR (Case Base Reasoning)

CLARA (Clustering Large Applications)

CLS (Concept Learning System)

DIANA (Divise Anlysis)

EM (Expectation Maximization)

EP (Elemento de proceso)

ESOINN (Enhanced Self-Organizing Incremental Neural Network)

ESTs (Expressed Sequence Tags)

GCS (Growing Cell Structure)

HTN (Hierarchical Task Network)

IBO (Intelligent Biomedical Organizations)

ICA (Independent Component Analysis)

ID3 (Induction Decision Trees)

IREP (Incremental Reduced Error Pruning)

K-NN (K-Nearest Neighbours)

KDD (Knowledge Discovery in Databases)

MAS5.0 (Microarray Affymetrix Suite5.0)

NG (Neural Gas)

NN (K-Nearest Neighbours)

OC1 (Oblique Classier 1)

OGSA (Open Grid Services Architecture)

PAM (Partitioning Around Medoids)

PCA (Principal Component Analisys)

PLIER (Probe Logarithmic Intensity Error)

POP (Partial Order Planner)

REP (reduced error pruning)

RIPPER (Repeated Incremental Pruning Produce Error Reduction)

RMA (Robust Multi-array Average)

RNA (Redes Neuronales Artificiales)

SAM (Significance Analysis of Microarrays)

SHOP (Simple Hierarchical Ordered Planner)

SMA (Sistema Multiagente)

SMO(Sequential Minimal Optimization)

SOA (Servicio Orientado Arquitectura)

SOAP (Simple Object Access Protocol)

SODTNN (Self Organized Dynamic Tree Neural

SOINN (Self-Organizing Incremental Neuronal Network)

SOM (Mapas Autorganizados de Kohonen)

SOM (Self-Organizing Map)

STRIPS (Stanford Research Institute Problem Solver)

SVM (Support Vector Machine)

TAN (Tree Argumented Navie Bayes)

TSP (Travelling salesman problem)

UDDI (Universal Discovery Description and Integration)