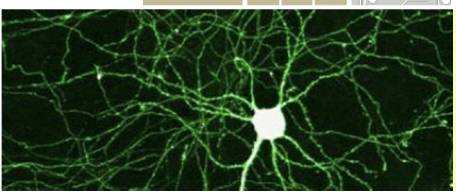


Inteligencia artificial 2

TEMA 1 APRENDIZAJE

Tema 1
Aprendizaje

1. Introducción
2. Aprendizaje no supervisado
3. Aprendizaje supervisado
4. Representación del conocimiento y aprendizaje

1. INTRODUCCIÓN

En los sistemas inteligentes, para que sigan siéndolo, es fundamental el aprendizaje, cuyo objetivo es que el rendimiento del sistema mejore a partir de su experiencia.

El **aprendizaje automático** estudia métodos para conseguir incrementar el rendimiento; y en ellos se utilizan mucho los algoritmos genéticos tanto como métodos de búsqueda como para encontrar la representación adecuada. Se basan en la evolución biológica, evolucionando en paralelo una población de soluciones posibles. Lo que se hace es codificar las posibles soluciones de un problema en forma de cromosomas (secuencia de genes –bits-); después en iteraciones sucesivas se seleccionan los que corresponden a mejores soluciones y se desechan los peores. Con los cromosomas seleccionados se construyen nuevos cromosomas (mediante cruces, mutaciones…).

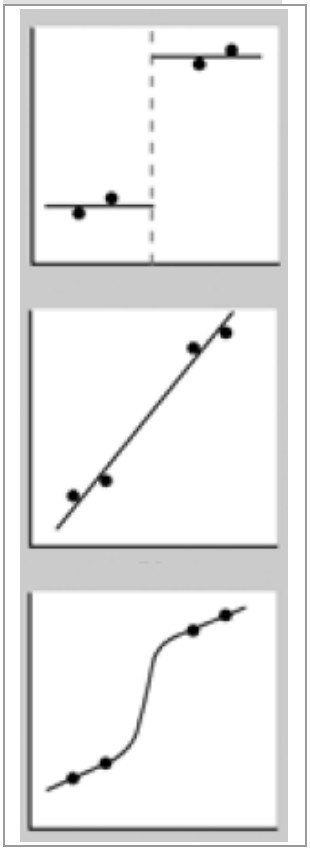
La cuestión de inferir sistemas difusos y resolverlos mediante algoritmos genéticos es complicada. Tenemos que codificar las posibles soluciones del problema (funciones de pertenencia a un conjunto difuso u otro), definir los operadores de manipulación genética y una función de evaluación.

Existen varias formas de **clasificar el aprendizaje**; nosotros nos basaremos en la supervisión, es decir, de qué tipo de información se dispone sobre el resultado que tiene que dar un sistema basado en el conocimiento. Así tenemos:

- **Aprendizaje supervisado**: Hay conocimiento completo sobre cual es la respuesta que se tiene que dar en una determinada situación. Así nuestro objetivo es obtener un sistema basado en el conocimiento que reproduzca la salida cuando estemos en una situación que corresponda a una entrada determinada. En general es un problema de optimización o más bien de aproximación de una función a partir de unos ejemplos (aprendizaje inductivo (se extrae conocimiento a partir de los ejemplos). Dentro de este tipo de aprendizaje encontramos distintos tipos de problemas:
    - **Problemas de regresión**: Cada ejemplo incluye un atributo con la solución y, en este caso, es de tipo numérico. Las redes neuronales como las entradas y salidas son habitualmente numéricas, constituyen un ejemplo de este tipo.
    - **Problemas de clasificación**: El atributo con la solución de cada ejemplo es categórico o bien binario y el objetivo del aprendizaje es inducir un modelo que predice con esmero este atributo. Las redes bayesianas que suelen utilizar principalmente valores categóricos son un ejemplo de ello.
    - **Problemas de búsqueda**: Algoritmos de búsqueda donde la solución se va construyendo a base de aplicar pasos distintos. Cuando el método de aprendizaje sirve para mejorar el conocimiento de forma que sea más sencillo o eficiente de utilizar en la búsqueda se habla de aprendizaje aunque no hay conocimiento nuevo sino reelaboración del existente; es, por el contrario al aprendizaje inductivo, aprendizaje deductivo.

- **Aprendizaje por refuerzo**: No se dispone del valor que corresponde a la salida sino solamente una gratificación o penalización según el resultado que se obtenga del sistema. Por ejemplo si un robot planea una trayectoria y choca con un objeto, recibe una penalización, y si no hay colisión recibe una gratificación, así el robot tendrá en cuenta estos estímulos al planear las trayectorias. Un ejemplo de ello son los algoritmos genéticos para sistemas difusos; dado que no se proporciona la solución correcta, sino sólo una evaluación de la solución planteada.
- **Aprendizaje no supervisado**: Solo se dispone de información sobre las entradas y no sobre las salidas; el aprendizaje tiene que extraer conocimiento útil a partir de la información disponible; son ejemplo los algoritmos de categorización que permiten ordenar la información disponible.

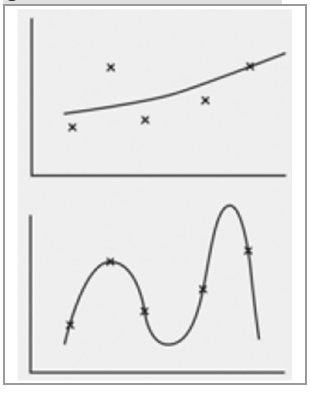
Sesgo y varianza



El **sesgo** y la **varianza** son dos conceptos que afectan mucho la construcción de modelos mediante métodos de aprendizaje inductivo. Debemos recordar que las reglas deben ser capaces de generalizar a partir de unos pocos ejemplos. Si tenemos unos ejemplos de pacientes con cierta enfermedad, el modelo que generemos debe ser válido para estos pacientes ejemplos pero también para todo el resto que no se ha incluido en el modelo dado que no se pueden incluir todos los posibles ejemplos existentes. Así en la figura lateral observamos 3 posibles generalizaciones a partir de cuatro puntos de ejemplo. El primer ejemplo considera rectas horizontales con discontinuidad, el segundo rectas y el tercero polinomios de grado 3. La generalización vemos que plantea dificultades porque hay muchas generalizaciones posibles. Se provoca sesgo cuando se restringe el espacio de búsqueda o la elección del formalismo de representación (así si en el ejemplo solo permitimos rectas, perdemos el último tipo de ejemplo). El sesgo por tanto, permite construir generalizaciones sin que nos desborde el tamaño del espacio de búsqueda, pero a veces, se pueden perder generalizaciones interesantes porque el conocimiento necesario no se incorpore al sistema.

Por otro lado la varianza corresponde a la posible dispersión de los valores. En la construcción de modelos, el sesgo decrece a medida que el modelo se complica, mientras que la varianza aumenta. Esto se explica porque con más parámetros medibles, el modelo se va ajustando más a los ejemplos; la varianza en cambio aumenta porque pequeños cambios en los ejemplos provocan grandes cambios en los modelos.

Sesgo grande y varianza pequeña
Sesgo pequeño y varianza grande



Esta contradicción la podemos estudiar cuando utilizamos polinomios para aproximar un conjunto de puntos. Cuando se aproxima un conjunto de n puntos, a mayor grado del polinomio menor es la diferencia entre el valor estimado y el valor real (el sesgo tiende a cero). Sin embargo, la varianza crece porque a medida que crece el grado del polinomio variaciones muy pequeñas en la posición de los puntos pueden provocar variaciones del modelo muy grandes. Así, si a partir de por ejemplo 12 puntos en un plano aproximamos la posición de una recta, el cambio de un punto no provocará cambios muy grandes en la posición de la recta; por el contrario si la aproximación de un polinomio es de grado 11, el cambio de un solo punto puede provocar que la nueva curva sea completamente distinta de la anterior. Por lo tanto, aunque a priori pudiera parecer que si disponemos de unos ejemplos tenemos que preferir un sesgo pequeño con varianza grande (ejemplo del polinomio de grado elevado en el que el modelo está ajustado en exceso a los datos y no tiene capacidad de generalización) esto no es así, aunque es muy difícil equilibrar la dificultad del modelo para un ajuste adecuado de los datos.

2. APRENDIZAJE NO SUPERVISADO

Este tipo de aprendizaje corresponde al caso en el que disponemos de un conjunto de datos para los que no se conocen las variables de salida; así aplicamos métodos de aprendizaje a los datos para extraer información útil (conocimiento). Estudiaremos distintas formas de llegar a este conocimiento:

Algoritmos de categorización

Es el método de aprendizaje no supervisado más utilizado. A partir de un conjunto de datos se construyen un conjunto de categorías. Las categorías agrupan datos "parecidos" y lo suficientemente "distintos" de los datos de otras categorías. La dificultad proviene de esta categorización y los parámetros que se pueden seguir para llevarla a cabo. Para esta categorización primero debemos conocer son los tipos de atributos que nos podemos encontrar:

- Numérico: El dominio lo determinan un subconjunto de números reales.
- Binarios o boléanos: Los datos satisfacen una característica (valor 1) o no la satisfacen (0).
- Ordinales: Se establece una relación de orden, aunque pueda ser numérico cuantificar este número a veces no tiene interés. Por ejemplo el nivel de estudios (puedo ir de 0 –nada- a 5 –estudios superiores- pero no tiene interés restar niveles).
- Nominales: El dominio lo regentan un conjunto de términos y solo podemos comprobar si son iguales o diferentes, por ejemplo los colores, no podemos establecer diferencias entre ellos.

Se puede pasar de atributos categóricos a binarios de dos formas principalmente:

- Ponemos tantas filas y columnas como atributos y un 1 cuando se satisface en fila y columna y un 0 cuando no.
- Para atributos ordinales podemos además colocar un 1 si está añadido o no, es decir una persona con nivel de estudios superiores (nivel 5) tendría un 1 en los niveles 0, 1, 2, 3, 4 y 5.

Siguiendo con los tipos de atributos:

- Atributos ordinales difusos: Es un atributo ordinal pero donde cada término tiene asociado un conjunto difuso. La semejanza o diferencia entre términos se calcula a partir de los conjuntos difusos asociados.
- Atributos con estructura jerárquica: Hay términos más semejantes a otros pero no podemos cuantificar un nivel numérico. Por ejemplo un pato se parece más a un gorrión que una cebra, dado que los primeros son aves.

Estudiados los tipos de atributos nos corresponde ahora **definir las distancias** y semejanzas para los distintos tipos de atributos. Tenemos 4 coeficientes distintos para ello y seleccionar uno y otro dependerá de las propiedades de los atributos, el tipo de escala y, en especial, de la experiencia:

- **Coeficientes basados en distancias**: Se basan en la definición de una distancia en un cierto espacio. Las distancias son funciones complementarias a los coeficientes, ya partir de una distancia definida en el intervalo [0, 1] y la semejanza siempre será positiva y máxima según se acerque a 1. Un tipo de distancia es la **euclidiana** que crece con el número de atributos a evaluar por lo que se calcula una distancia media dividiendo entre el número de atributos.
- **Coeficientes de asociación**: Aplicados a atributos nominales permiten evaluar la coincidencia o no de valores. Se consideran atributos binarios con un 1 cuando se cumplen y un 0 cuando no, existe el coeficiente de coincidencia simple, el coeficiente de Jaccard (no se incluyen los elementos negativos).
- **Coeficientes de correlación**: Miden la proporcionalidad o independencia entre los vectores que representan los objetos. El más representativo es el de Pearson
- **Coeficientes de semejanza probabilística**: Incluyen estadísticos para medir la homogeneidad de los objetos.

Otros aspectos a tener en cuenta son:

- **Los valores ausentes**: Por errores de codificación, por valores eliminados para proteger la confidencialidad de los datos o porque el atributo no es aplicable a todos los registros estos valores ausentes pueden aparecer y se pueden tratar sustituyéndolos por el valor esperado (la media o la moda) o ignorarlo a la hora de hacer los cálculos.
- **Normalización de los datos**: Para calcular semejanzas y distancias el dominio de los atributos puede convertirse en un problema. Por ejemplo si estamos midiendo igualdades y diferencias en el personal, es evidente que el sueldo de dos personas será mucho más diferente que el número de hijos, el dominio en este último caso está mucho más limitado. Existen dos dormas de igualar dominios:
    - **Ranging**: Todos los valores quedan en el rango [0, 1]. A todos los valores se le resta el menor y se le divide por la diferencia entre el mayor y el menor, con lo que el más pequeño queda como 0, el mayor como 1 y el resto como valores intermedios. Se utiliza en el caso de atributos binarios y ordinales o nominales.
    - **Estandarización**: Se normalizan los valores de forma que el valor medio sea 0 y la desviación 1. Se utiliza cuando los atributos son mayoritariamente numéricos o categóricos.
- **Importancia de los atributos**: Todos los atributos no tienen que tener el mismo peso para todos los estudios y esto hay que definirlo a priori. Hay funciones de semejanza y distancia que incluyen este tipo de pesos.
- **Estructuras posibles**: Ya para las categorías estudiadas encontramos 3 tipos de estructuras posibles:
    - Categorías disjuntas: Cada objeto puede pertenecer solo a una categoría, aunque esto no asegura que todo objeto tenga una categoría.
    - Categorías que se superponen: Los límites entre categorías no son nítidos y, por ende, algunos elementos están solapados en más de una categoría.
    - Categorías organizadas en forma de árbol jerárquico: La relación entre categorías se expresa mediante una estructura de árbol (dendograma). Los conceptos que comparten un nodo padre son más semejantes entre ellos que con los demás.

Particiones y particiones difusas de objetos

Muchos algoritmos de categorización se basan en la idea de optimizar una función objetivo. Los dos métodos más habituales son el K-Means (o C-Means) y el Fuzzy C-Means.

El primero de ellos se basa en encontrar una partición del conjunto de objetos. Sigue un esquema iterativo en el que en cada iteración se busca primero la mejor partición a partir de unos centros y a continuación se vuelven a calcular los centros de la nueva partición construida.

Fuzzy C-Means es una generalización del anterior; es decir, este método construye una partición difusa, a diferencia del anterior que la construía nítida. Se basa en la definición de la constante m, cuanto mayor es, más difusa es la categoría y cuando su valor es 1 es el caso del C-Means.

Cuando un problema está mal condicionado, echamos mano de la **regularización** que consiste generalmente en incluir algún término adicional en la función objetivo, como puede ser la entropía.

En los **mapas autoorganizativos** las neuronas no son entidades aisladas, sino que se organizan en una estructura, y ahora se selecciona la neurona más próxima y el prototipo correspondiente se modifica, pero no sólo la de esa neurona, sino también las vecinas.

Jerarquías de objetos

La alternativa a las particiones de los capítulos anteriores son las estructuras definidas con categorías o estructuras jerárquicas (dendogramas). Hay dos formas principales de llevarlas a cabo:

- **Métodos aglomerativos**: Se construye el árbol de abajo a arriba, juntando en cada paso las categorías más semejantes entre sí por un orden de importancia (es decir, con un peso asignado a cada categoría a comparar). Se llevan a cabo a través de funciones de semejanza, criterio de agregación (hay que determinar qué categorías pasan a formar parte de la nueva categoría) y criterio de clasificación (criterio para recalcular semejanzas). Como el espacio de búsqueda correspondiente a las jerarquías posibles es muy grande, normalmente la construcción de la jerarquía se basa en métodos ávidos.
- **Métodos partitivos**: Se parte de un único conjunto de objetos que corresponde a la categoría más general (conteniendo todos los objetos) y en pasos sucesivos se refinan las categorías que hay y se hacen particiones.

3. APRENDIZAJE SUPERVISADO

Ahora disponemos de un conjunto de ejemplos y para cada ejemplo conocemos el valor de salida (valor que desconocíamos en el aprendizaje no supervisado). Tenemos varios tipos de métodos para este aprendizaje:

Métodos de categorización

El espacio donde se representan los objetos está dividido en diversas categorías; las clases corresponden a los valores que puede tomar gi, mientras que las categorías son construcciones a partir de los datos (por ejemplo mediante los algoritmos del capítulo anterior). Así, una clase y una categoría pueden tener intersección nula, coincidencias parciales o ser exactamente iguales.

- **Vecino más próximo**: Es el método más sencillo y se basa en el almacenamiento de un conjunto de ejemplos, la clase de un objeto se determina a partir del objeto más próximo. Para ello necesitamos la definición de distancia o semejanza vista en el apartado anterior. También podemos no conformarnos con el objeto más próximo sino con los k objetos más próximos.
- **Vector Quantization**: Está basado en el vector quantization: se proporciona un ejemplo, se localiza la neurona más próxima y se modifican los prototipos, se selecciona el prototipo más próximo de entre todos los que hay en todas las categorías y clases. Ya localizado, la modificación depende de si la clase de este prototipo coincide o no con la del objeto que tratamos. Si coincide acercamos su prototipo al objeto, si no coincide, lo alejamos del mismo.
- **C-Means**: Consiste en construir varias categorías para cada clase, después cada categoría permitirá clasificar unos cuantos objetos. El algoritmo C-Means obtiene para cada categoría un centroide o prototipo que corresponde a una descripción de una categoría; así como el resultado son c categorías para cada una de las D clases, tendremos cxD centroides. La clasificación de nuevos objetos sigue aquí el mismo esquema del método basado en el Vector Quantization explicado más arriba.

Máquinas de vectores de soporte

Son herramientas para clasificar objetos descritos mediante datos numéricos (y estadísticos) que consideran sólo la existencia de dos clases, y por tanto se considera la clase a la que pertenece al objeto o la clase a la que no pertenece.

- **Máquinas lineales**: Clasifican los objetos basándose en la construcción de un hiperplano en el espacio de datos, que separa los objetos que pertenecen a una clase (a un lado del hiperplano) de los que pertenecen a la otra clase (al

otro lado del hiperplano). Cuando las clases son linealmente separables, la mejor regla de clasificación corresponderá a encontrar el hiperplano que define un espacio más amplio con los objetos, obligando a que las distancia entre los objetos y el hiperplano siempre sean lo mayor posible. Cuando las clases no son linealmente separables, hay que tener en cuenta el solapamiento entre clases. Se permiten errores de objetos que estén al otro lado del hiperplano que realmente les pertenece, teniendo una nueva variable para cada objeto, siendo su valor distinto de 0 cuando no quedan bien clasificados; pero además tenemos un límite K del número de objetos que pueden no encontrarse en el lado del plano que les corresponda. También hay una constante C definida por el usuario y corresponde a una medida de hasta qué punto se permite tener datos mal clasificados; cuanto mayor es C, más se penalizan los puntos mal clasificados.

- **Máquinas no lineales**: Transformamos ahora el espacio de datos original en un nuevo espacio diferente y mayor.

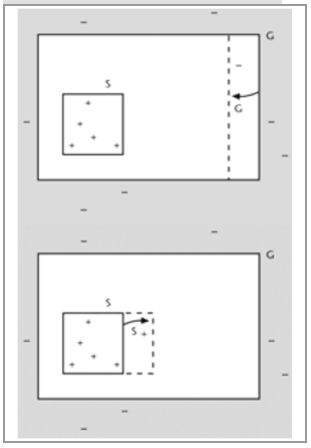
Descripciones lógicas de conceptos

Estos algoritmos parten de un conjunto de ejemplos positivos (ejemplos que corresponden al ejemplo que queremos describir) y de un conjunto de ejemplos negativos (no corresponden al concepto); a partir de ahí se construyen descripciones lógicas de dos tipos:

- **De general a particular**: Se empieza con una descripción general que todo ejemplo satisface, y a partir de los ejemplos negativos se va restringiendo la descripción para que no los englobe; es lo que ocurre en la figura 1 de la tabla lateral.
- **De particular a general**: Se considera una descripción lo más concreta posible, y a partir de aquí, en cada paso, se intenta generalizar una descripción sacando o generalizando una de las condiciones de la descripción. Es lo que ocurre en la figura 2 cuando aumentamos el espacio S más restrictivo.

Esto no siempre puede realizarse, porque podemos tener expresiones en las que G y S no definan realmente un espacio uno dentro de otro o más restrictivo.

Descripciones lógicas de Conceptos



4. REPRESENTACIÓN DEL CONOCIMIENTO Y APRENDIZAJE

Los sistemas de programación lógica inductiva construyen modelos a partir de ejemplos, intentando superar varias limitaciones como son la representación limitada (los métodos inductivos sólo permiten construir modelos con expresiones equivalentes al cálculo proposicional), no permite considerar conocimiento previo sobre el dominio del sistema y existe sesgo de vocabulario (sólo se pueden utilizar aquellos términos que inicialmente ya aparecen en los ejemplos suministrados).

Por tanto, utilizando lógica de primer orden, se intentan superar estas barreras, tanto para representar el conocimiento previo como para representar lo que se aprende. Con un conocimiento previo K, un conjunto de ejemplos positivos (E+) y un conjunto de ejemplos negativos (E- que no se pueden demostrar a partir de K), los sistemas de programación lógica inductiva tienen que encontrar un conjunto de cláusulas en lógica de primer orden que llamamos H que permitan demostrar E+ pero que no permitan demostrar E-.

TEMA 2 AGENTES Y SISTEMAS MULTIAGENTE

Tema 2
Agentes y sistemas multiagente
1. Agentes inteligentes
2. Sistemas multiagente

1. AGENTES INTELIGENTES

Según Woodridge: "Un agente es un sistema computacional situado en un cierto entorno, que es capaz de llevar a cabo acciones de manera autónoma y flexible sobre este entorno para intentar alcanzar unos ciertos objetivos". Un agente tiene distintas características importantes:

- **Autonomía**: Un agente tiene que trabajar sin intervención directa de humanos y mantener el control sobre su estado interno y sus acciones. Un agente realizará una acción determinada en función del estado interno y las circunstancias que le rodean, no se le puede obligar a realizar ninguna acción por orden directa.
- **Reactividad**: Un agente reacciona de manera rápida y apropiada a los cambios del entorno (el entorno es un ambiente real y no una simbolización y entornos de laboratorio, por lo que nunca se quedará desfasado).
- **Capacidad de comunicación**: Debe ser capaz de comunicarse con otros agentes o con humanos; así un conjunto de agentes podrán intercambiar información y coordinar sus acciones para resolver problemas más complejos.
- **Proactividad**: Un agente tiene que ser capaz de tomar la iniciativa y llevar acciones para alcanzar objetivos sin que explícitamente se le haya pedido que haga esa acción determinada.

Otras características menos importantes son:

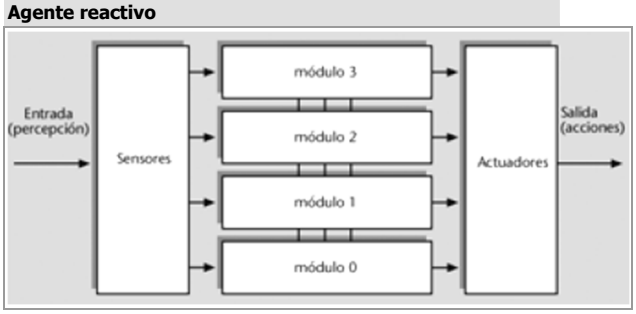
- **Continuidad temporal**: Un agente es un proceso que se ejecuta de manera permanente, no es un programa con un principio y un fin.
- **Movilidad:** Puede pasar físicamente de un ordenador a otro por una red electrónica y continuar su ejecución en el nuevo entorno.
- **Planificación, razonamiento**: Un agente debe razonar sobre el estado del entorno y sus posibles acciones y elaborar planes que le ayuden a satisfacer sus objetivos.
- **Aprendizaje**: Con técnicas de aprendizaje automático, el agente mejora su eficacia y aumenta el rango de problemas que puede resolver.
- **Carácter**: Debe tener un aspecto lo más humano posible, para que los usuarios depositen su confianza en él.

1.1. Agentes reactivos

Las características básicas de un sistema reactivo son:

- El entorno en que se mueven los agentes es real, el mundo es el mejor modelo y no hacen falta modelos computacionales ni representaciones simbólicas.
- Los datos que recibe el agente lo hace a partir de sensores sencillos. Como el mundo es el actual y los datos se recogen en tiempo real, el modelo nunca se quedará desfasado.
- Se basan en un conjunto de reglas simples, que interaccionan con otros agentes y a partir de ahí pueden desarrollar un comportamiento más complejo.

Agente reactivo



- Cada agente está formado por varios módulos que operan de manera autónoma y son responsables de distintas tareas, existiendo una comunicación entre niveles; pero no existiendo ningún modelo global ni ningún módulo planificador ni director de toda la actividad, que se convertiría en un punto débil. Así por ejemplo podemos tener un robot, que en su nivel 0 (figura lateral) se encargaría de evitar el choque de obstáculos. El nivel 1 podría ejecutar acciones para moverse por el entorno, y con ayuda del nivel 0 se desplaza sin chocarse. El nivel 2 podría explorar el entorno y detectar objetos lejanos, yendo hacia ellos con ayuda de los dos niveles inferiores.

Todas las características anteriores confieren a este sistema mucha robustez (tolerancia a fallos debida a la independencia de los módulos), flexibilidad (como están guiados por reglas simples que se aplican sobre los valores de los sensores, pueden adaptarse fácilmente a cambios dinámicos en su entorno) y tienen un tiempo muy rápido de respuesta, es decir solo deben ejecutar la regla para el cambio que los sensores detectan, confían en que la mayor parte de la actividad diaria es rutinaria y repetitiva. Este es también una de las críticas que reciben, ya que no tienen capacidad de planificación a medio plazo ni objetivos determinados. Otras críticas son que el diseñador debe pensar en todas las reglas necesarias para reaccionar ande los distintos datos que pueden recoger los sensores; y en especial se les achaca una aplicabilidad muy limitada.

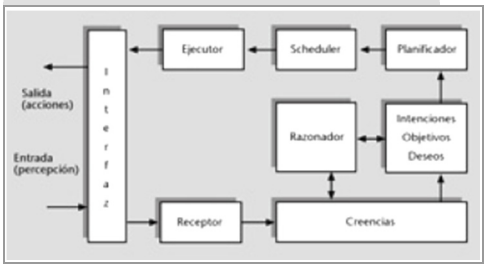
1.2. Agentes deliberativos

Este tipo de agentes son más clásicos en la inteligencia artificial. Tienen una base de conocimientos donde se almacena un modelo simbólico del entorno, un motor de inferencia capaz de hacer procesos de razonamiento lógico sobre la información del dominio y a partir de ahí desarrollan sus acciones. Eso sí, hay que ir actualizando la representación del mundo existente en su base de conocimientos para que no se quede desfasada. EL modelo más utilizado de agente deliberativo es la arquitectura BDI (beliefs-desires-intention, creencias-deseos-intenciones).

- Creencias: Conocimiento que tiene el agente de su entorno (parecido a la base de conocimientos tradicionales).
- Deseos: Situaciones futuras a las que el agente quiere llegar, pueden ser irrealizables e incluso contradictorias.
- Objetivos: Subconjunto de los deseos sobre los cuales el agente puede actuar para cumplirlos.
- Intenciones: Subconjunto de los objetivos priorizados para ser cumplidos.
- Planes: secuencia de acciones que el agente ejecuta para llevar a cabo sus intenciones.

Como podemos ver en el modelo lateral, la interfaz recibe los datos del entorno, el receptor los recoge y actualiza las creencias de la base de datos, se generan deseos, objetivos e intenciones por este orden, que recoge el planificador (intenciones) y junto al planificador de tareas y al ejecutor se ejecutan para llegar a la intención objetivo. El principal inconveniente de todo el sistema es el elevado coste computacional que una intención a medio plazo puede tener, si alguna de estas fases (decisión y planificación) tarda mucho en ejecutarse, puede que haya cambiado ya el entorno de trabajo cuando se quieran llevar a cabo, por ello muchos agentes son híbridos, tienen una parte deliberativa (para cumplir objetivos a medio y largo plazo) y una parte reactiva (para saltar rápidamente frente a situaciones de alarma).

Modelo BDI



1.3. Tipos de agentes

Encontramos tres tipos de agentes distintos: los agentes de interfaz, los de información y los agentes colaborativos. Éste último lo estudiaremos en el apartado siguiente.

Un **agente de interfaz** es como un asistente personal de un usuario humano, que colabora con el usuario en alguna tarea y le ayuda en su entorno de trabajo sin que explícitamente se le haya pedido. Un caso habitual es que dan asistencia preactiva a un usuario que está utilizando o aprendiendo a utilizar una aplicación. El agente observa continuamente las acciones del usuario en la utilización de la aplicación y sugiere maneras más eficientes de hacer alguna tarea o ayudar al usuario cuando no sabe hacer alguna acción. Son agentes con un nivel no muy elevado de razonamiento y planificación y no suelen cooperar con otros

agentes, suelen ser autónomos, con mucha capacidad de aprendizaje y muy preactivos.

Un ejemplo de este tipo de agentes, es el proyecto Letizia, proyecto que pretende crear una ayuda en la búsqueda de información, almacenando y organizando la información que puede resultar útil para el usuario. Así el agente de interfaz monitoriza constantemente las acciones del usuario y saca conclusiones. Por ejemplo si un usuario guarda una dirección de interes en sus enlaces del explorador es porque la página le interesa, y Letizia estudia bien su contenido; si un usuario sigue un enlace es que tiene un interés potencial, pero si vuelve atrás, se rechaza ese contenido. Si el usuario humano permanece "mucho" tiempo en una misma página (en relación a su longitud) es que le interesa; si pincha un enlace de mitad de la página es que le interesa ese enlace, pero rechazamos todos los anteriores, etc.

Un **agente de información** es una entidad computacional de software autónoma, que tiene acceso a fuentes de información múltiples, heterogéneas y geográficamente distribuidas, y que, de manera preactiva, adquiere, mantiene y maneja el acceso a información relevante para el usuario. Así el agente adquiere y utiliza la información, la sintetiza y presenta al usuario y se adapta dinámicamente a las nuevas peticiones del mismo.

2. SISTEMAS MULTIAGENTE

Una de las características más importante de un agente inteligente es su capacidad para comunicarse con otros agentes, así le puede pasar información, preguntar por un dato, pedir ayuda para resolver un problema.

Un sistema multiagente es un conjunto de agentes autónomos inteligentes que se comunican para resolver un problema de manera conjunta, cuando los componentes de este tipo de sistemas quieren cooperar en la resolución del problema, se habla de un conjunto de agentes colaborativos.

Las ventajas de un sistema multiagente son muchas:

- Conocimiento especializado: No hace falta que un agente tenga todo el conocimiento complejo sobre un entorno, cada uno de los agentes puede especializarse e intercambiar información.
- Distribución: Se pueden tratar problemas en el que los datos están geográficamente distribuidos.
- Reutilización: Se pueden aprovechar sistemas multiagente construidos previamente al realizar uno nuevo.
- Fiabilidad y eficiencia: Son robustos ante fallos dado que el trabajo está repartido entre varios agentes y si uno falla, otros toman el relevo o se reparten las tareas, además pueden trabajar todos en paralelo resolviendo el problema a la vez, más rápidamente que si solo lo hiciera un algoritmo centralizado.

Podemos aplicar sistemas multiagente a dominios en los que podemos descomponer el problema final en varios subproblemas que se puedan ir resolviendo de forma relativamente independiente, o dominios donde el conocimiento necesario para su resolución esté distribuido en distintos sistemas.

Como vemos un **sistema de comunicación** adecuado entre agentes es fundamental, existen dos grandes tipos: el sistema de pizarra y el paso de mensajes.

- **Sistema de pizarra**: Es tradicional dentro de la inteligencia artificial distribuida y consiste en una pizarra que proporciona a todos los agentes que forman el sistema un área común de trabajo en la que pueden intercambiar datos, información o conocimiento. Esta pizarra se convierte en el área central del sistema y en un punto débil de robustez. La información habitual que puede

existir en la pizarra suelen ser tareas y/o resultados. Es un sistema simple en el intercambio de información, flexible, coordinado y cooperativo, pero tiene puntos débiles, ya que un agente para llegar a encontrar la información que necesita tiene que estar buscando toda la que se encuentra en la pizarra (a veces se divide en zonas según materias para facilitar este uso); nadie avisa a los agentes cuando hay nueva información disponible, por lo que periódicamente cada agente debe ojear la pizarra, pudiendo formarse un cuello de botella en el sistema en este punto. Para resolver este problema a veces existen dos figuras clave en el sistema de pizarra: el moderador (centraliza la gestión de tareas entre los agentes) y el distribuidor (agente encargado de avisar a los demás agentes registrados en una pizarra de los cambios que se producen en ésta y enviarles la información que les puede resultar útil).

- **Paso de mensajes**: Cuando un agente A se quiere comunicar con otro B, le lanza un mensaje que sólo es legible por el receptor.

La FIPA (Foundation for Intelillent Physical Agents) es una organización sin ánimo de lucro cuya misión es definir estándares que deben cumplir los agentes y los sistemas, para que puedan ser realmente colaborativos.

Arquitectura FIPA de un sistema multiagente

La plataforma de agentes proporciona la estructura básica, el sistema de transporte de mensajes sirve precisamente para eso, transportar mensajes; también encontramos al facilitador de directorio que es un agente que proporciona un servicio de páginas amarillas dentro del sistema, porque conoce los servicios que pueden ofrecer todos los agentes; y el sistema de gestión de agente controla el acceso y uso de la plataforma; almacena las direcciones de los agentes y ofrece un servicio de páginas blancas.

Existe un lenguaje estándar para comunicación entre agentes y las conversaciones que debe establecerse entre ellos: los llamados protocolos de comunicación, que definen la secuencia de mensajes que se tienen que intercambiar un conjunto de agentes para una acción comunicativa concreta.

En este marco son fundamentales las **ontologías**, que proporcionan una conceptualización del dominio de trabajo. Habitualmente incluyen información sobre las clases de objetos que hay en el dominio, sus atributos o las acciones que pueden hacer los agentes del sistema.

Por ejemplo, si un agente quiere realizar una reserva en un restaurante, debe referirse a la ontología restaurantes, que contiene varios objetos:

- Objeto restaurante: Tiene la información de contacto, los precios, la capacidad, atributos de aparcamiento, tarjetas, especialidades, etc.
- Cinfo: información complementaria como el nombre, dirección postal, teléfono, fax y otros.
- Objeto menú: Con el precio del menú y el precio de la carta.
- Objeto carta: dividido en entrantes, primeros, segundos y postres.
- Objeto primeros: dividido en pastas, sopas, legumbres, verduras y arroz. Cada uno de estos objetos contiene distintos platos.
- Objeto platos: nombre, precio, calorías, etc.

Los **sistemas multiagentes cooperativos** son aquellos en los que se utilizan mecanismos de cooperación, que se puede conseguir con un paso de mensajes o sin comunicación entre los agentes. En este último caso, la cooperación se establece de manera indirecta, mediante el entorno; cada agente observa el entorno y ve los efectos de las acciones de otros agentes y adecua su actividad de manera apropiada.

La **planificación global parcial** es un mecanismo flexible que permite que los diferentes componentes de un sistema multiagente coordinen sus actividades para intentar resolver de manera distribuida y lo más eficiente posible un problema común. En esta técnica cada agente es capaz de obtener información sobre el estado actual y los objetivos alcanzados por los otros agentes, y pueden

utilizar esta información para optimizar su trabajo. Se crean planes globales parciales para cada agente que se pueden intercambiar con otros e incluso modificar sobre la marcha, atendiendo a la evolución y el entorno del dominio.

Texto elaborado a partir de:
Inteligencia artificial II
Enric Mor i Pera, Antonio Moreno Ribas, Vicenç Torra i Reventós
Junio 2006