SABACO: Extensiones a los Algoritmos de Optimización basados en Colonias de Hormigas para la Toma de Decisiones Influenciada por Emociones y el Aprendizaje de Secuencias Contextuales en Ambientes Inteligentes

Tesis Doctoral

UNIVERSIDAD
POLITECNICA
DE VALENCIA

Jose Antonio Mocholí Agües

jmocholi@dsic.upv.es

Departamento de Sistemas Informáticos y Computación

UNIVERSIDAD POLITÉCNICA DE VALENCIA

Director: **Dr. Francisco Javier Jaén Martínez**

Valencia, 28 de febrero de 2011

Diseño Plantilla Cubierta:
Diseño Cubierta: Jose Antonio Mocholí Agües
Ilustración Portada:

Ilustración Contraportada:

Resumen

En el trabajo que presentamos en esta tesis hacemos inicialmente una revisión de cómo ha ido evolucionando la interacción hombre máquina en el contexto de la computación, desde los primeros y escasos computadores hasta el momento actual, en el que los avances tecnológicos han permitido que, en muchos de los escenarios en los que se desarrolla nuestra vida diaria, estemos rodeados de diversos dispositivos electrónicos con los que interactuamos para hacer uso de alguno de los servicios que ofrecen. Veremos cómo esta difusión tecnológica ha introducido los sistemas de información en ámbitos más allá del contexto del trabajo, como la educación o el hogar, y ha dado lugar a la aparición de los conocidos como ambientes inteligentes, en los que son los sistemas presentes en el entorno los que deben adaptarse al usuario y al contexto en el que se encuentra, adaptación que, dados los nuevos contextos en los tiene lugar la interacción con el usuario, plantea algunos retos.

En particular, en el presente trabajo identificamos dos factores clave que los ambientes inteligentes deben tener en cuenta para tomar las decisiones y llevar a cabo las acciones adecuadas para conseguir una mejor adaptación al usuario y al contexto. Estos factores son la influencia de las emociones en la interacción y la utilización de la información contextual histórica. Por ello hacemos una revisión tanto de las propuestas de sistemas de decisión influenciados por emociones existentes en el área de la computación afectiva, como de las propuestas de sistemas sensibles al contexto. A continuación presentamos la metaheurística de optimización basada en colonias de hormigas como un punto de partida genérico desde el que diseñar sistemas que tengan en cuenta los factores clave identificados. Finalmente, exponemos las dos propuestas algorítmicas diseñadas basadas en las colonias de hormigas, una para la toma de decisiones influenciada por emociones, y otra propuesta para la utilización de la información contextual disponible, incluyendo la histórica, como base para la predicción de qué situaciones contextuales ocurrirán con mayor probabilidad en el futuro más inmediato. Para ambas propuestas se muestra con casos de estudio cómo sería su aplicación a un dominio de problema y los resultados empíricos que demuestran que alcanzan los objetivos marcados.

Resum

Al treball que presentem en esta tesi fem al inici una revisió de com ha anat evolucionant la interacció home màquina al context de la computació, des dels primers i escassos computadors fins al moment actual, en el que els avanços tecnològics han permès que, a molts dels escenaris en els que té lloc la nostra vida quotidiana, estiguem envoltats de diversos dispositius electrònics amb els que interactuem per a fer ús d'algun dels servicis que oferixen. Vorem com esta difusió tecnològica ha introduït els sistemes d'informació a àmbits més enllà context del treball, com la educació o la llar, i ha donat lloc a la aparició dels coneguts com ambients intel·ligents, als que son els sistemes presents a l'entorn els que han d'adaptar-se a l'usuari i al context on es troba, adaptació que, donats els nous contextos als que té lloc la interacció amb el usuari, planteja alguns reptes.

En particular, al present treball hi identifiquem dos factors clau que els ambients intel·ligents cal que tinguen en compte a l'hora de prendre les decisions i dur a terme les accions adequades per tal d'aconseguir una millor adaptació a l'usuari i al context. Estos factors son la influència de les emocions a la interacció i la utilització de la informació contextual històrica. Per aquesta raó fem una revisió tant de les propostes de sistemes de decisió influenciats per emocions que existixen a l'àrea de la computació afectiva, com de les propostes de sistemes sensibles al context. Tot seguit presentem la metaheurística d'optimització basada en colònies de formigues com un punt de partida genèric des del que dissenyar sistemes que tinguen en compte els factors clau identificats. Finalment, exposem les dues propostes algorítmiques dissenyades i que estan basades en colònies de formigues, una per a la presa de decisions influenciada per emocions, i altra proposta per a la utilització de la informació contextual disponible, incloent la històrica, com a base per a la predicció de quines situacions contextuals ocorreran amb major probabilitat al futur més immediat. Per a ambdues propostes es mostra amb casos d'estudi com seria la seua aplicació a un domini de problema i els resultats empírics que demostren que arriben als objectius marcats.

Abstract

In the work we present in this thesis we initially make a review of the evolution of the human computer interaction, from the very first and scarce computers until the present moment, when technological breakthroughs have allowed that, in many of the scenarios in which our everyday life take place, we were surrounded by a wide range of electronic devices with which we interact in order to consume some of their services. We will elaborate on how this technological diffusion has introduced information systems in fields beyond the work context, such as education or our homes, and it has given rise to the appearance of the known as intelligent ambients, in which the systems that are in the environment have to adapt themselves to the user and the context of the environment. This adaptation raises some challenges, since interaction with users takes place in new contexts.

Particularly, this work identifies two key factors that intelligent ambient should take into account to make the appropriate decisions and perform the appropriate actions in order to achieve a better adaptation to users and context. These factors are the influence of emotions on the interaction, on one side, and the use of the historic contextual information. Due to that, we make a review of both the approaches of decision making systems biased by emotions from the field of affective computing, and the approaches of context-aware systems. Then we present the ant colony optimization metaheuristic as a generic starting point from which we can design systems that take into account the key factors identified. Finally, we elaborate on the two algorithmic approaches based on ant colony optimization we have designed: one for decision making biased by emotions, and another for the use of the available contextual information (including the historical one) as a basis for foreseeing which contextual situations will occur with a higher probability in the nearest future. For both approaches we present through case studies how their use on a problem domain would be, and also the experimental results that show they achieve the goals set.

Palabras clave

Palabras clave:

Colonias de hormigas, Ambientes inteligentes, Computación afectiva, Toma de decisiones, Información contextual histórica

Paraules clau:

Colònies de formigues, Ambients intel·ligents, Computació afectiva, Presa de decisions, Informació contextual històrica

Keywords:

Ant Colonies, Ambient intelligence, Affective computing, Decision making, Historical context data

Dedicatoria

Datos de la tesis

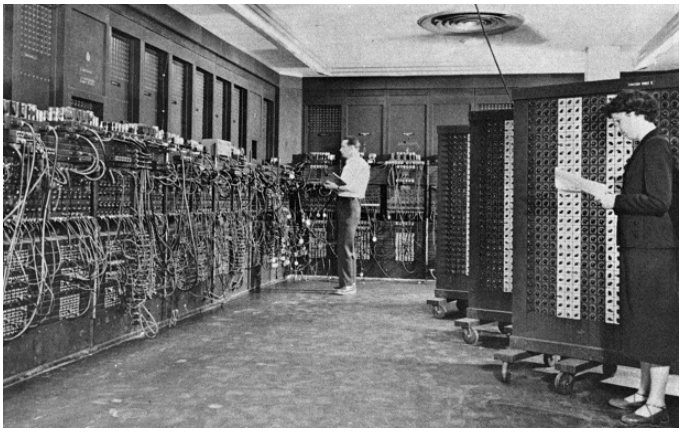
Título de la tesis:	SABACO: Extensiones a los Algoritmos de Optimización basados en Colonias de Hormigas para la Toma de Decisiones Influenciada por Emociones y el Aprendizaje de Secuencias Contextuales en Ambientes Inteligentes
Presentada por:	Jose Antonio Mocholí Agües
	jmocholi@dsic.upv.es
Dirigida por:	Fco. Javier Jaén Martínez
	fjaen@dsic.upv.es
Universidad:	Universidad Politécnica de Valencia
Departamento:	Sistemas Informáticos y Computación
Programa de doctorado:	Programación declarativa e Ingeniería de la Programación
	Memoria para optar al grado de Doctor en Informática
Depósito:	Valencia, a 28 de febrero de 2011
Defensa:	Valencia, a __ de ____ de 2011

Agradecimientos

Índice general

Índice de Figuras

Índice de Tablas



ENIAC, el primer ordenador electrónico multipropósito. Programarlo requería manipular cables de interconexión y accionarhasta 6000 interruptores.

Capítulo 1

Introducción

Desde los días de los primeros computadores, una de las necesidades que los fabricantes y diseñadores de computadores abordaron fue la necesidad de facilitar el trabajo de las personas involucradas en el manejo y aprovechamiento de los mismos, desde los directivos hasta los operarios, pasando por los programadores. Al principio, las mejoras se limitaban, por ejemplo, a facilitar la sustitución de válvulas en la época de los computadores de válvulas de vacío, o a crear mecanismos que facilitasen la creación y carga en el computador de las tarjetas perforadas que contenían los programas y los datos que estos manejaban. A medida que los computadores ganaban en fiabilidad y capacidades, su utilización empezó a salir de los centros de investigación en los que se desarrollaban y devino una herramienta nueva en trabajos científicos y de ingeniería. De esta manera, cada vez más personas con escasos conocimientos en computadores se convertían en usuarios de los mismos, por lo que se acrecentaba la necesidad de mejorar las interfaces de los computadores, ya no sólo desde el punto de vista hardware, sino también desde el punto de vista del software (aunque con escaso protagonismo en los inicios). Así, ligada a la historia de los computadores y del software, nacía el área de *Human Computer Interaction* (HCI) (Pew, 2003), (Grudin, 2008).

En aquellos primeros años, durante las décadas de los 50 y los 60, el funcionamiento de los computadores consistía en cargar un programa junto con los datos necesarios a partir de unas tarjetas perforadas o de una cinta magnética, ejecutar el programa (sin ningún tipo de interacción), devolver el resultado de la ejecución y, a continuación, volver a empezar con el siguiente programa. Este tipo de funcionamiento, conocido como proceso por lotes (*batch processing*), restringía la interacción real con el computador a los operadores, encargados de suministrar el programa y los datos y de recoger el resultado. Por ello no debe resultar extraño que los primeros trabajos publicados en HCI estuvieran relacionados con la mejora del diseño de botones e interruptores, y fuesen realizados por expertos en ergonomía (Grudin, 2008). Estos expertos en ergonomía (*human factors* en USA) solían ser psicólogos que comenzaron realizando tareas de selección y formación de personal para la industria, a principios del siglo XX, y que más tarde, con las guerras mundiales, se involucraron en el diseño de sistemas militares, equipamientos y puestos de trabajo con el objetivo de que pudiesen ser dominados en poco tiempo por las personas encargadas de manejarlos (Pew, 2003), (Grudin, 2008). Su interés se centraba en influir en los diseños para que se acomodasen a las capacidades y limitaciones cognitivas, motrices y de percepción de las personas a interactuar con el producto diseñado. En otras palabras, su objetivo era conseguir un alto nivel de productividad de los usuarios en poco tiempo a través de mejorar la usabilidad. De esta manera es fácil entender su involucración natural en el diseño de hardware, software, y del HCI en general, a medida que esta área iba creciendo (Pew, 2003).

De esas dos décadas cabe destacar como avances en el mundo de la computación por parte de actores del campo de HCI las visiones, y los prototipos desarrollados para ejemplificarlas, como por ejemplo el trabajo de Douglas Englebart: su visión era "aumentar el intelecto humano" a través del uso de los computadores (Englebart, 1963). En el prototipo que desarrolló para dar vida a su visión, Englebart rodeó al usuario con monitores CRT y dispositivos experimentales de entrada-salida como el ratón. En su visión, el usuario no se encargaba simplemente de "alimentar" al computador y esperar el resultado, sino que el usuario interactuaba con el computador y éste le daba el soporte necesario para realizar sus tareas. El prototipo que desarrolló incluía un sistema jerárquico de gestión de la información que era capaz de trabajar con texto, diagramas, imágenes, videoconferencias (de manera limitada) y de manipularlos y visualizarlos en "ventanas" que se podían mostrar en cualquiera de los monitores. En una época en la que la interacción con los computadores era mayoritariamente indirecta y realizada a través de formularios de texto, la interacción unipersonal con un computador equipado con varios monitores en los que se mostraba una interfaz gráfica con texto, imágenes y video suponía

toda una revolución. Su prototipo no llegó nunca a desarrollarse comercialmente aunque, muchos años después, aportaciones como el ratón, el sistema de hipertexto, metáforas gráficas como las "ventanas" y su visión del uso unipersonal del computador dieron lugar y forman parte de lo que hoy en día conocemos como ordenador personal.

Las siguientes dos décadas trajeron avances en el hardware de los ordenadores que fueron incrementando su rendimiento y capacidades a la vez que disminuían su precio. Algunos de los modelos producidos en serie que aparecieron contaban con capacidades gráficas limitadas que, aunque impedían o limitaban en gran medida su utilización para la investigación realizada en el área de gráficos por computador, provocaron que parte de los investigadores de esa área se centrasen en el estudio, diseño y creación de sistemas gráficos interactivos que buscasen una relación simbiótica entre hombre y máquina (Pew, 2003). De todos estos trabajos y prototipos iniciales surgirían a principios de la década de los 80 los primeros sistemas con interfaz gráfica, o en los que se ejecutaban aplicaciones con interfaz gráfica, que fueron ampliamente comercializados como el Xerox Star o el Apple Lisa (Grudin, 2008). Las interfaces gráficas de usuario (IGU) posibilitaron que un público más amplio y sin conocimientos técnicos utilizase ordenadores para llevar a cabo su trabajo, y que la interfaz de usuario de los sistemas fuese más intuitiva. Sin embargo, también incrementaban la complejidad y, como resultado, aumentaba la posibilidad de confusión por parte del usuario. De igual forma, las nuevas capacidades de representación y la desaparición de algunas restricciones de diseño convertían el diseño de IGUs en un reto. Los primeros test de usabilidad de software empezaron a realizarse en esos años (Grudin, 2008) y, justificando su necesidad a través de la idea de conseguir evitar incurrir en costes innecesarios, la comunidad HCI abogó por la inclusión de requisitos de usabilidad como parte integral del diseño de nuevos sistemas y sus interfaces.

1.1 Planteamiento del problema

Aunque en términos generales el área de HCI sigue, desde un punto de vista global, centrada en mejorar la interacción (la usabilidad) eliminando todo aquello que genera frustración, dejando que la interfaz haga todo el trabajo y reduciéndolo a secuencias de pasos triviales (Lazzaro, 2008), la visión de cómo se ha transformado nuestra relación con los ordenadores que se presenta en (Sellen, 2009), por un lado, y la evolución que se puede apreciar en (Norman, 1982) (Norman, 1983) (Norman, 1988) (Norman, 2004), por otro, nos muestran nuevas áreas y tendencias que se debe explorar.

La difusión tecnológica en la que vivimos desde hace más de una década, marcada especialmente por la presencia de sistemas de información en cada vez más ámbitos de nuestra vida, ha ido abriendo nuevas áreas a explorar en HCI más allá de los puestos de trabajo, como la educación o el hogar (Sellen, 2009). Esta apertura a nuevos horizontes ha provocado que los objetivos de los diseñadores ya no se limiten a aumentar la funcionalidad o la utilidad, y que ahora indaguen sobre la capacidad para provocar, atraer, perturbar o encantar a los usuarios. Este cambio ha sido debido, según los autores de (Sellen, 2009), a cinco transformaciones o cambios que han modificado y siguen redefiniendo cómo nos relacionamos con la tecnología en general y los ordenadores en particular, a saber:

- Fin de la continuidad en las interfaces. Con la presencia de sistemas informáticos más allá del escritorio (por ejemplo, en coches, aviones, ropa, juguetes,…), es necesario plantearse cuestiones como si es necesaria una interfaz, cómo ha de ser, dónde ha de estar o qué podrá hacer el usuario con ella.
- Aumento de la dependencia tecnológica. La utilización (directa o indirecta) de sistemas informáticos está cada vez más presente en nuestras vidas: cuando hacemos la compra, cuando viajamos, cuando nos divertimos, y, por supuesto, cuando trabajamos.
- Expansión de la conectividad. La forma de estar en contacto con los demás, cómo nos comunicamos, es uno de los aspectos de nuestras vidas que más ha cambiado con la tecnología. Todos usamos dispositivos y medios que nos permiten estar en contacto con los demás, a cualquier hora, en cualquier lugar.
- Fin de lo efímero. La tecnología actual y su accesibilidad permite capturar y guardar cada vez más y más cosas que queremos recordar de nuestras vidas. Y a medida que avanza la tecnología, también cambia el conjunto de cosas que podemos capturar, así como también cambia cómo gestionamos esos "recuerdos" y cómo los compartimos con los demás.
- Desarrollo de la creatividad. Con cada avance tecnológico los sistemas informáticos nos permiten hacer más cosas, o las mismas pero de una manera distinta o en un entorno diferente. Y cada vez más y más gente de distinta índole tiene acceso a estas nuevas herramientas, y las utilizan tanto para trabajar, como para entretenerse o expresarse, lo que, en este último caso, permite dar rienda suelta a la creatividad de cada uno de nosotros.

A partir de estas transformaciones, los autores enumeran una serie de cuestiones a las que HCI debe responder y que deben marcar los nuevos objetivos a perseguir. En este sentido, en (Sellen, 2009) destacan como cada

vez más investigadores y profesionales tienen en cuenta la variedad de contextos y la diversidad de usuarios para "*explicar la naturaleza del uso como una cuestión de experiencia*" de usuario.

En este mismo sentido otros autores como Weiser (Weiser, 1991) reclaman un cambio profundo en el modo en el que los usuarios han de relacionarse con la tecnología como consecuencia de las transformaciones descritas anteriormente. En sus propias palabras:

> *"The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it. (…) Most important, ubiquitous computers will help overcome the problem of information overload. There is more information available at our fingertips during a walk in the woods than in any computer system, yet people find a walk among trees relaxing and computers frustrating. Machines that fit the human environment, instead of forcing humans to enter theirs, will make using a computer as refreshing as taking a walk in the woods."*

Weiser reclama una nueva generación de sistemas de información denominados por él mismo como "ubicuos" en los que son estos los que se integran de forma transparente en la vida diaria de los usuarios, adaptándose en cada momento al contexto en el que éste se encuentra y sin requerir una intervención explícita por su parte. Este tipo de sistemas, también conocidos como *context-aware*, tienen por tanto como objetivo incrementar la usabilidad y la efectividad en el uso de las tecnologías teniendo en cuenta las características de la situación en la que se encuentra el usuario. Si bien diversos autores han intentado dar múltiples definiciones a la noción de contexto, tal y como se describe en (Baldauf, 2007), quizás la que más éxito ha tenido es la proporcionada por Dey y Abowd (Dey, 2000) en la que se refieren al contexto como:

> *"…any information that can be used to characterize the situation of entities (i.e., whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves."*

La percepción de la importancia del contexto como elemento fundamental para el desarrollo de una nueva generación de sistemas de información bautizada bajo varios términos como "ubicuos", "pervasivos", o "ambientes inteligentes", ha resultado en múltiples propuestas de modelos y arquitecturas para dar soporte a esta nueva forma de computación (Baldauf, 2007).

El concepto de "inteligencia ambiental" (*Ambient Intelligence*, AmI) (Marzano, 2003) está estrechamente relacionado con la computación ubicua que proponía Weiser, aunque se encuentra más enfocado a la provisión de un comportamiento inteligente por parte del entorno. Ese comportamiento inteligente del entorno se refleja tanto en el uso de interfaces de usuario adaptadas al contexto, como en la toma de decisiones y el control compartido

de los recursos por parte del sistema. Idealmente, la inteligencia ambiental actuaría como catalizador para desarrollar nuevos usos y aplicaciones de los entornos de computación ubicua. Desde ese punto de vista, el término computación ubicua estaría relacionado con la infraestructura física presente en un entorno, mientras que la inteligencia ambiental vendría a referirse al comportamiento o programación que hace uso del entorno. En un entorno de inteligencia ambiental las personas tendrán a su disposición interfaces naturales integradas en el entorno, conformando junto con los objetos cotidianos un medioambiente electrónico e interconectado que responderá a la presencia de los individuos inmersos en él de una forma inteligente, adaptativa y no intrusiva.

Entre las dimensiones de contexto a considerar en el desarrollo de un ambiente inteligente tiene especial relevancia aquella que tiene relación con las sensaciones que perciben los usuarios en su relación con el ambiente que les rodea. Así, se han empezado a utilizar conceptos como placer, estética, diversión, aburrimiento o enfado para describir la naturaleza multifacética de lo sentido/experimentado durante la interacción con un sistema. Asimismo, en (Sellen, 2009) los autores también remarcan cómo palabras como magia, embrujo, asombro, excitación y sorpresa han empezado a ser utilizadas al discutir la influencia de la tecnología en el aspecto emocional de los usuarios. Llegados a este punto, y observando cómo se ha ido haciendo más evidente la relevancia de las emociones experimentadas por el usuario durante la interacción, creemos que se debería añadir una transformación más a la lista descrita anteriormente: las emociones como un factor clave de la interacción hombre-máquina.

En esa misma línea podemos apreciar en (Norman, 1982) (Norman, 1983) (Norman, 1988) (Norman, 2004) una evolución de la idea central que nos deja patente cómo el factor emocional de los usuarios, el estado emocional[1] de la persona, ya es tenido en cuenta y es una pieza sustancial en los trabajos de interacción hombre-máquina. Así, si en (Norman, 1982) (Norman, 1983) el autor definía unas métricas para medir la satisfacción del usuario respecto a un cierto diseño basadas en la velocidad de interacción,

[1]El término "emoción" se puede utilizar en un sentido amplío o en uno más restringido. En el sentido restringido se refiere a lo que podríamos denominar emociones verdaderas o básicas, dónde la emoción es de manera temporal la característica dominante de la mente: reemplaza el proceso de deliberación usual y dirige con fuerza la manera de actuar. El sentido más amplío se refiere a la emoción subyacente, aquella que influye en los pensamientos y acciones de una persona, en mayor o menor medida, pero sin tomar el control. Y el estado emocional describiría cualquier estado mental en el que la emoción, tanto en el sentido amplío como en el restringido, (se puede considerar que) juega un papel importante (Cowie, 2001).

facilidad de aprendizaje, conocimientos requeridos y número de errores, y en (Norman, 1988) se centraba en la usabilidad más pragmática, en (Norman, 2004) el autor define tres niveles de diseño y su estrecha relación con las emociones que desencadenan para explicar que convierte en "bueno" un diseño a ojos de los usuarios. Y es que el único objetivo emocional de mejorar la usabilidad de las interfaces es reducir la frustración de los usuarios y, aunque importante, no ofrece ninguna recompensa emocional sustancial por la consecución de tareas complejas que podrían inspirarles a explorar y aprender las capacidades de un producto, lo cual les conduciría a ser más eficientes en su trabajo (Lazzaro, 2008). Simplificar en exceso una tarea la convierte en demasiado predecible, repetitiva, y muy simple de realizar lo cual incrementa el aburrimiento y apatía del usuario, lo que a su vez reduce su concentración y satisfacción con el uso del sistema (Lazzaro, 2008), tal y como también se desprende de la teoría de *flow* o de las experiencias óptimas. La mayor parte de la vida cotidiana consiste en una sucesión de eventos y experiencias, aparentemente irrelevantes, sobre las que interviene un proceso psicológico de selección (Csikszentmihalyi, 1985), ya que no podemos prestar atención a todos y cada uno de los estímulos de nuestro entorno. Y el factor más determinante que rige ese proceso psicológico de selección es la "calidad de la experiencia", es decir, preferimos dirigir nuestra atención a eventos de nuestro entorno que asociamos con estados de conciencia positivos y gratificantes. En este sentido nos encontramos con lo que en (Csikszentmihalyi, 1988) los autores denominan *flow* o experiencia óptima, que se caracteriza por: la percepción de retos atrayentes, unas habilidades personales apropiadas, altos niveles tanto de concentración como de gozo y compromiso, control de la situación, pérdida de la conciencia de uno mismo, atención sostenida, reacciones positivas, claridad en los objetivos de la actividad y por una motivación intrínseca en realizarla (Deci, 1985).

Sin embargo, y a pesar de que los ambientes inteligentes han de tomar decisiones para adaptarse al entorno en el que se encuentran, las decisiones que toman los ambientes inteligentes desarrollados hasta la fecha sólo tienen en cuenta un modelo de contexto que se basa en el estado del entorno y la situación emocional del usuario para el instante de tiempo actual. Por el contrario, nosotros pensamos que las nuevas generaciones de ambientes inteligentes habrán de basar las decisiones que tomen no sólo en la situación contextual actual, sino también en aquéllas que con mayor probabilidad se darán en el futuro, marcando como un objetivo a alcanzar con las decisiones tomadas el llevar a los usuarios a estados de *flow* en su relación con el ambiente. Un ejemplo de escenario en el que nos encontramos con un ambiente inteligente que actúa de la manera que hemos descrito lo encontramos a continuación.

Ocasionalmente Gonzalo utiliza su vehículo para ir con la familia a visitar a los abuelos, que viven a unos 200 km. Esta ruta de largo recorrido ha sido aprendida en base a experiencias previas. El ambiente inteligente en el coche de Gonzalo detecta que está moviéndose por una zona fuera del área urbana de su lugar de residencia. A partir de la información contextual que describe el área en el que se encuentra el vehículo (por ejemplo, las coordenadas de la celda que está prestando el servicio de telefonía al dispositivo móvil de Gonzalo o el hecho que se esté desplazando en un vehículo a motor), el ambiente hace uso de la infraestructura de aprendizaje de rutas para ver si hay rutas seguidas en el pasado por Gonzalo compatibles con esa descripción contextual. Efectivamente la hay (es la ruta que sigue Gonzalo cuando va a ver a sus padres) y el algoritmo de aprendizaje devuelve la información de la misma (una secuencia de descriptores contextuales de las áreas que atraviesa Gonzalo con su vehículo). Al analizar la ruta, se detecta que es una ruta de largo recorrido y que es posible que Gonzalo tenga que repostar. Puesto que el servicio de provisión de información del vehículo indica que el nivel de combustible en el depósito es de un 25% y que el combustible utilizado es gasoil, el ambiente decide buscar de forma proactiva estaciones de servicio en la ruta obtenida por el algoritmo de aprendizaje de rutas. Consulta los servicios de dichas estaciones y obtiene el precio por litro de gasoil en cada una de ellas y su localización. Con estos datos encuentra una estación en el trayecto que tiene el precio del combustible más barato y decide informar verbalmente a Gonzalo con el siguiente mensaje: "Existe una estación de servicio a 95 km en la localidad de Tarazona con un precio de 0.85 euros el litro de gasoil. Es un 15% más barato que la estación de servicio utilizada habitualmente".

En el ejemplo anterior el usuario Gonzalo se encuentra en un ambiente inteligente constituido por todos los sensores y dispositivos que forman parte o se hallan en su coche. A partir de los datos recogidos por los sensores y teniendo en cuenta información relativa a experiencias previas relacionadas con esos mismos datos o similares, el ambiente infiere que Gonzalo está realizando un desplazamiento de largo recorrido y que va a necesitar repostar. Por lo que, antes de que llegue si quiera a mostrarse el testigo de reserva de combustible, el ambiente sugiere el repostaje en una estación de servicio que se encuentra de camino, consiguiendo de esta manera que el usuario se mantenga en un estado cercano a *flow*, ya que se evita que se preocupe por buscar una gasolinera en la que repostar antes de quedarse sin combustible y además se le consigue un beneficio en forma de ahorro económico.

Teniendo en cuenta el impacto tan positivo que tendría en los ambientes inteligentes el hecho de que añadieran la situación emocional del usuario a la información contextual y que consideraran la información contextual histórica en su funcionamiento, en el trabajo presentado en esta tesis nos planteamos la

definición de técnicas automatizadas de toma de decisiones basadas en estados afectivos y con la capacidad de predecir situaciones contextuales futuras para su aplicación en el desarrollo de ambientes inteligentes.

1.2 Metodología

1.2.1 Metodología investigadora

La metodología investigadora a seguir en la realización de la presente tesis es Investigación-Acción (*Action Research*, AR) (Avison, 1999) (Davison, 2004). La Investigación-Acción no se refiere a un método de investigación concreto, sino a una clase de métodos que tienen en común las siguientes características (Baskerville, 1999):

- Orientación a la acción y al cambio.
- Focalización en un problema.
- Un modelo de proceso "orgánico" que engloba etapas sistemáticas y algunas veces iterativas.
- Colaboración entre los participantes.

Esta metodología investigadora tiene una doble finalidad: generar un beneficio al "cliente" de la investigación y, al mismo tiempo, generar "conocimiento de investigación" relevante (Kock, 2001). Por tanto, AR es una forma de investigar de carácter colaborativo que busca unir teoría y práctica entre investigadores y profesionales ("practicantes") mediante un proceso de naturaleza cíclica. En un análisis más formal de los participantes en AR, (Wadsworth, 1998) identifica los siguientes cuatro tipos de roles en este método (en algunas ocasiones la misma persona o equipo puede desempeñar más de un rol):

- El investigador, el individuo o grupo que lleva a cabo de forma activa el proceso investigador.
- El objeto investigado, es decir, el problema a resolver.
- El grupo crítico de referencia, aquél para quien se investiga en el sentido de que tiene un problema que necesita ser resuelto y que también participa en el proceso de investigación (aunque menos activamente que el investigador). En él hay tanto personas que saben que están participando en la investigación, como otras que participan sin saberlo.
- El beneficiario (*stakeholder*), aquél para quien se investiga en el sentido de que puede beneficiarse del resultado de la investigación, aunque no participa directamente en el proceso. Puede ser el receptor de documentos,

informes, etc. En este grupo, por ejemplo, caben tanto las empresas que se benefician de un nuevo método para resolver problemas en tecnologías de la información, como los técnicos que aplican dicha metodología.

Un proceso de investigación que emplea AR está compuesto de grupos de actividades organizadas formando un ciclo característico. En (Padak, 1994) los autores identifican los siguientes pasos, que deben seguirse en las investigaciones que utilicen este método:

1. Planificación: Identificar las cuestiones relevantes, que guiarán la investigación, que deben estar directamente relacionadas con el objeto que se está investigando y ser susceptibles de encontrarles respuesta. En esta actividad se buscan caminos alternativos, líneas a seguir o reforzar algo existente. El resultado es que se definen claramente otros problemas o situaciones a tratar.
2. Acción: Variación de la práctica, cuidadosa, deliberada y controlada. Se efectúa una simulación o prueba de la solución. Es cuando el investigador interviene sobre la realidad.
3. Observación: Recoger información, tomar datos, documentar lo que ocurre. Esta información puede proceder prácticamente de cualquier sitio (bibliografía, medidas, resultados de pruebas, observaciones, entrevistas, documentos, etc.). También se conoce como "evaluación".
4. Reflexión: Compartir y analizar los resultados con el resto de interesados, de tal manera que se invite al planteamiento de nuevas cuestiones relevantes y, como añade (Wadsworth, 1998), "a profundizar en la materia que se está investigando para proporcionar conocimientos nuevos que puedan mejorar las prácticas, modificando éstas como parte del propio proceso investigador, para luego volver a investigar sobre estas prácticas una vez modificadas". También se conoce como "especificación del aprendizaje". En algunas variantes de AR no es una etapa realmente, sino un proceso continuo que ocurre durante todo el tiempo.

Con estas características, podemos decir que el proceso definido por AR es iterativo, de forma que se va avanzando en soluciones cada vez más refinadas mediante la compleción de ciclos, en cada uno de los cuales se ponen en marcha nuevas ideas, que son puestas en práctica y comprobadas en el ciclo siguiente. Este ciclo caracteriza a AR como un proceso reflexivo de aprendizaje y búsqueda de soluciones. El carácter cíclico supone volver a reevaluar o replantear las acciones o caminos a seguir, ponderando diagnóstico y reflexión.

1.2.2 Metodología de desarrollo

El proyecto seguirá una metodología de desarrollo en espiral e incremental basada en el proceso de desarrollo RUP (*Rational Unified Process*) (Kruchten,

2000). Siguiendo esta metodología abordaremos las actividades de análisis, diseño, implementación y evaluación de las distintas componentes tecnológicas de forma iterativa e incremental abordando en cada iteración un subconjunto de los requisitos identificados inicialmente para cada una de los objetivos específicos descritos en la sección anterior. Esta metodología permite, por un lado, validar con los usuarios de forma temprana los requisitos que se acometan en cada iteración y, por otro, manejar adecuadamente la complejidad del desarrollo. Esta metodología se ha convertido en un estándar *de facto* para proyectos de desarrollo software y permite estructurar el proyecto en torno a las fases de Concepción, Elaboración, Construcción y Transición (Figura 1) en cada una de las cuales se desarrollan de forma iterativa y con distinto esfuerzo las distintas disciplinas o actividades de desarrollo.

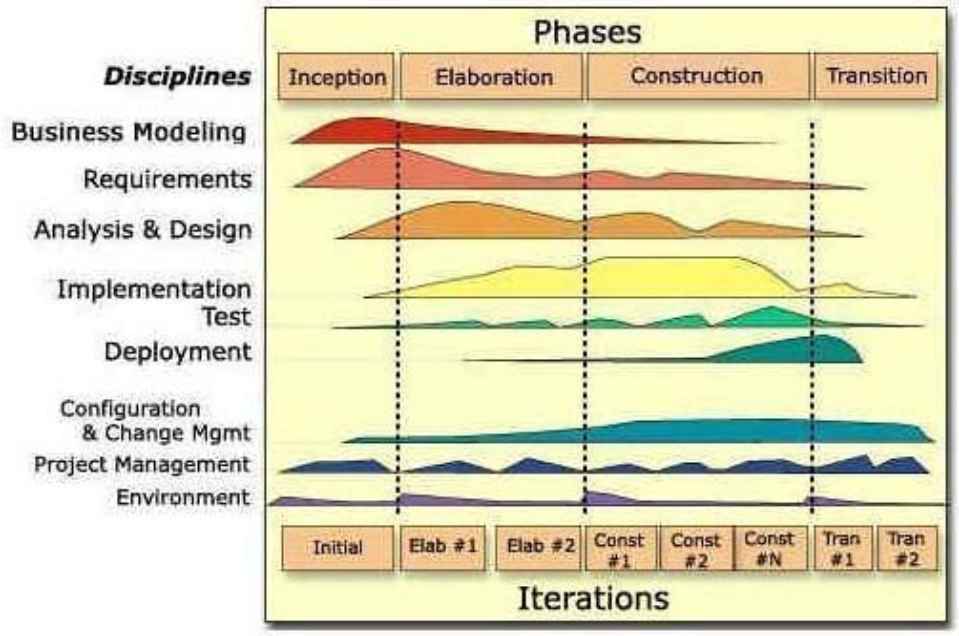


Figura 1. Fases de la Metodología RUP.

En esta tesis aplicaremos una versión modificada de la metodología RUP en la que las fases de elaboración, construcción y transición se aplicarán siguiendo un modelo en espiral incremental y no secuencial en cascada, como muestra la Figura 1. De esta forma podremos abordar la elaboración construcción y transición de prototipos intermedios que puedan constituirse como entregables susceptibles de ser evaluados y validados por los usuarios. La utilización de esta versión modificada de RUP está en sintonía con el carácter iterativo e incremental de la metodología de investigación, por lo que las metodologías

propuestas son compatibles: cada iteración de la metodología investigadora puede alimentar a la iteración de la metodología de desarrollo.

1.3 Resumen de aportaciones

A continuación resaltamos las principales aportaciones de esta tesis:

- Describe el estado del arte en las técnicas utilizadas para la creación de sistemas de toma de decisiones con capacidades afectivas, así como en sistemas sensibles al contexto (véase el capítulo 2).
- Asimismo, presenta los métodos metaheurísticos como una aproximación capaz de abordar simultáneamente ambos aspectos (véase el capítulo 2).
- Describe la metaheurística de optimización basada en colonias de hormigas, presentando sus características, las propuestas más destacadas, los dominios de aplicación y las tendencias de investigación actuales en esta técnica (véase capítulo 3).
- Define una propuesta algorítmica para la toma de decisiones basada en estados afectivos que está basada en la optimización con colonias de hormigas, exponiendo los conceptos necesarios y las modificaciones llevadas a cabo a esta metaheurística (véase capítulo 4).
- Define una propuesta algorítmica, basada en la metaheurística de optimización con colonias de hormigas, que es capaz de utilizar información semántica que describe un contexto para obtener las situaciones contextuales futuras más probables, teniendo en cuenta para ello situaciones contextuales pasadas (véase capítulo 5).

1.4 Estructura de la tesis

En este primer punto se ha hecho una introducción al área de HCI y su excesiva concentración en mejorar la interacción entre los usuarios y el software de su entorno con el que interactúan mediante la reducción de la frustración mediante la simplificación al extremo de cualquier tarea que debieran realizar. Esto, según la teoría de *flow*, termina generando aburrimiento y apatía y también otros estados emocionales que alejan al usuario de cualquier interés intrínseco por utilizar el software. Además, hemos destacado como la creciente abundancia de dispositivos electrónicos (de muy diversas características y capacidades) a nuestro alrededor ha traído consigo la aparición de una nueva generación de sistemas de información, conocidos como ambientes inteligentes, que aglutinan los dispositivos y objetos del entorno en

un medioambiente electrónico, que se adaptan al contexto del entorno en el que se encuentran automáticamente y están ideados para responder de una manera inteligente y no intrusiva a la presencia de las personas en su entorno. Sin embargo, también hemos destacado como las propuestas de ambientes inteligentes desarrolladas hasta el momento no tienen la información histórica de contexto recopilada para mejorar su adaptación, con lo que pierden una oportunidad valiosa de anticiparse a situaciones del entorno que volverán a tener lugar en un futuro cercano con una alta probabilidad. Así, finalmente, planteamos la necesidad de inclusión de modelos afectivos en los ambientes inteligentes para que puedan "tomar decisiones" para adaptarse al estado emocional del usuario, así como la necesidad de recopilar y utilizar información contextual del entorno a lo largo del tiempo con el fin de conocer qué situaciones contextuales se darán con mayor probabilidad en el futuro más próximo y poder así tomar las decisiones más adecuadas. Por todo ello, el resto del presente documento está organizado de la siguiente manera: en el capítulo 2 introduciremos el área de la *computación afectiva* y de los sistemas de decisión, haciendo una revisión tanto de las propuestas de sistemas de decisión influenciados por emociones como de las propuestas de sistemas sensibles al contexto, además de presentar los métodos metaheurísticos como un punto de partida genérico desde el que diseñar una propuesta con las características deseadas; en el capítulo 3 presentaremos la técnica metaheurística elegida, exponiendo sus orígenes, cómo funciona la técnica de manera genérica, las características diferenciadoras de las variantes más relevantes, los dominios de problema en los que se ha aplicado con éxito, así como las líneas de investigación más destacadas que permanecen abiertas. En el capítulo 4 presentamos nuestra propuesta de algoritmo de la metaheurística elegida para la toma de decisiones influenciada por emociones, para lo que primero presentaremos de manera más extensa y formal la propuesta en la que nos basaremos, luego presentaremos el caso de estudio elegido y mostraremos cómo hemos aplicado nuestra propuesta al dominio de aplicación seleccionado dentro del caso de estudio. En el capítulo 5 describiremos en profundidad nuestra propuesta algorítmica para utilizar la información contextual disponible, incluyendo la histórica, como base para predecir qué situaciones contextuales ocurrirán con mayor probabilidad en el futuro más inmediato, y entonces expondremos los dos dominios de aplicación en los que hemos probado la propuesta realizada y los resultados de los experimentos llevados a cabo. Por último, en el capítulo 6, resumimos las conclusiones y las aportaciones del presente trabajo, así como presentamos una serie de posibles extensiones al trabajo expuesto en esta tesis.

1.4.1 Convenciones tipográficas

En la redacción de esta tesis se han empleado una serie de convenciones tipográficas, habituales en los textos impresos, encaminadas a facilitar la lectura de este documento. Las descripciones de las convenciones utilizadas son las siguientes:

- Los términos o vocablos provenientes de otros idiomas pero que son comúnmente aceptados y utilizados, en el lenguaje del campo científico al que correspondan, aparecen resaltados en cursiva, por ejemplo, el *flow*.
- Las citas procedentes de terceras fuentes aparecen en cursiva y convenientemente delimitadas, por ejemplo, "*operibus credite et non verbis*".
- El código fuente proveniente de un lenguaje de especificación en pseudo-código o de un lenguaje de programación aparece en bloques sombreados con tipografía no proporcional.
- Las referencias bibliográficas aparecen entre paréntesis siguiendo el convenio estándar indicando un acrónimo y un año de producción. En caso de tener dicho autor varias publicaciones en ese mismo año utilizamos letras del alfabeto para distinguir cada una de ellas, por ejemplo, (Mocholí, 2007a). La lista completa de referencias bibliográficas se encuentra ordenada alfabéticamente en la sección de Bibliografía.
- La existencia de una nota al pie [2] viene indicada por un superíndice numérico, único para cada capítulo, que se corresponde con la nota situada al final de la página y etiquetada con el mismo número.

[2] Ejemplo de nota al pie.

HAL 9000, en "2001: Una odisea del espacio" (1968), era capaz de razonar y de expresar e interpretar emociones.

Capítulo 2

Estado del arte

En el capítulo anterior hemos puesto de relieve la importancia para el futuro desarrollo de los ambientes inteligentes de dos aspectos clave en el éxito de estos sistemas: el contexto, que describe la situación en la que se desarrolla una determinada actividad, y la componente afectiva de los usuarios participantes. Un ambiente que tenga que ser considerado como inteligente debe hacer uso del contexto para ser capaz de exhibir un comportamiento en el que la toma de decisiones ha de adaptarse a la especificidad de la situación. En esa misma línea, consideramos que el estado emocional de los usuarios presentes en el entorno del que se captura el contexto es un factor clave en la toma de decisiones por parte del ambiente inteligente. Por esta razón es necesario realizar un estudio detallado de la literatura actual en relación a los sistemas de toma de decisiones con capacidades afectivas en particular y, de forma más general, con capacidades de sensibilidad al contexto. A partir del estudio de dichos sistemas podremos estar en situación de discutir los aspectos a abordar en el presente trabajo de cara a poder definir ambientes inteligentes que sean capaces de realizar el proceso de toma de decisiones de forma contextual y afectiva.

2.1 Toma de decisiones en sistemas con capacidades afectivas

En este punto haremos una introducción al área de la computación afectiva, que aborda la creación, introducción y utilización de modelos afectivos en el contexto de la computación, y al área de la toma de decisiones (*decision making*), prestando especial atención a los sistemas para la toma de decisiones que tengan en cuenta las emociones en el proceso de selección de la decisión a tomar.

2.1.1 Computación afectiva

Los ordenadores siempre han sido vistos como paradigmas de la lógica, la racionalidad y la predictibilidad. Estos paradigmas son, para muchos investigadores, las bases de la inteligencia y por ello han sido el centro de atención de aquellos investigadores que trabajan para conseguir una máquina inteligente. Sin embargo, nunca han conseguido construir una máquina que pudiese razonar de un modo inteligente en pro de resolver problemas complicados, o que pudiese interactuar (casi) de igual a igual con nosotros.

Hace ya cuatro décadas, en un trabajo sobre los fundamentos de la cognición humana (Simon, 1967), el autor hacía hincapié en que cualquier teoría general sobre el pensamiento humano y la resolución de problemas debe contar con las influencias generadas por las emociones. De esta manera, algunos investigadores han constatado que nuestra cognición no es un proceso tan lógico como se creía y que las emociones no son siempre ilógicas (LeDoux, 1996). Así tenemos que algunas de las teorías recientes que modelan el pensamiento humano y cómo procesa la información ya tienen en cuenta las emociones (Clore, 1994), (Isen, 2000), (Schwarz, 1990), (Schwarz, 1996). Los investigadores teóricos de las emociones también han apoyado el papel de la emoción como un potente motivador que influye en la opinión, la cognición y la creatividad de una manera importante. Otros resultados que han surgido de la neurología, de la ciencia cognoscitiva y de la psicología indican un papel fundamental de la emoción en la atención, el planeamiento, el razonamiento, el aprendizaje, la memoria, y la toma de decisiones (Picard, 1997).

En los últimos años y de forma cada vez más intensa, los conceptos de emoción y experiencia del usuario han recibido una especial atención dentro de la comunidad de HCI al ser una manera de agregar valor a los productos en la fase de diseño. Ya no es suficiente que un producto sea usable o bonito, necesita evocar respuestas emocionales positivas (Fredrickson, 2001) (Greene, 1988), por lo que se han estudiado nuevos mecanismos para desarrollar interfaces que puedan detectar y responder a emociones (Herbon,

2006), (Money, 2008). La recopilación de la información emocional de los usuarios puede contribuir a crear un contexto emocional en las interfaces de las aplicaciones. Rosalind Picard defiende en (Picard, 2002) que los sistemas que no hacen caso de la componente emocional de la vida humana son inevitablemente inferiores e incompletos, y añade que los sistemas que proporcionan una interacción social y emocional apropiada y útil no son ciencia ficción sino un hecho científico.

En el proyecto HUMAINE[3] definen una interfaz de usuario emocional como una interfaz que mantiene la atención del usuario a través de su capacidad de percibir la emoción del usuario, adaptarse a ella, reaccionar a ella y también su capacidad para iniciarla (Humaine, 2007). Donald Norman indica en (Norman, 2004) que los sistemas interactivos deben ser lo bastante atrayentes para ser agradables y que el estado de ánimo positivo hace a la gente más receptiva a nuevas ideas e incluso a interrupciones, despertando la curiosidad y la atracción, mientras que las emociones negativas hacen que los problemas parezcan más grandes de lo que son y que las personas se concentren más. Ya es una opinión extendida y compartida que las emociones influyen en la interacción hombre-máquina, porque el gusto por lo bonito, la atracción y la diversión son regulados por el sistema emocional humano y son tan importantes como la utilidad o la funcionalidad en el diseño de una interfaz (Norman, 2002) (Shneiderman, 2004).

No es necesario que todas las máquinas presten atención a las emociones, o que tengan capacidades emocionales. Algunas máquinas son útiles como herramientas rígidas y no hay ningún problema en que sigan siéndolo. Sin embargo, hay situaciones donde la interacción hombre-máquina se podría mejorar teniendo máquinas que se adaptasen naturalmente a sus usuarios, y donde saber cuándo, dónde, cómo, y cómo de importante es adaptarse implica disponer de información emocional, posiblemente incluyendo expresiones de la frustración, confusión, teniendo aversión, interés, etc. Y es que, como se indica en (Picard, 1999), el uso adecuado de habilidades emocionales, aunque sean limitadas, puede conllevar una mejora en la calidad de la experiencia de los usuarios, incluso en interacciones de largo plazo. De esta forma, con la finalidad de expandir la interacción hombre máquina mediante la incorporación de la comunicación emocional y los medios adecuados para el manejo de la información emocional, surge el área de la *computación afectiva* (*affective computing,* AC), entendida como toda aquella computación relacionada con, que surja de, o que de forma deliberada influye sobre las emociones (Picard, 1997). AC abarca también la implementación de emociones,

[3] HUMAINE project: http://emotion-research.net/

y por lo tanto puede ayudar al desarrollo de modelos computacionales para dotar, entre otras cosas, a un ordenador de la capacidad de reconocer y de expresar las emociones, desarrollando su capacidad de responder inteligentemente a emociones humanas y permitiéndole regular y utilizar sus emociones (Picard, 1997).

Una máquina puede expresar emociones sin tenerlas, al igual que los seres humanos podemos expresar emociones que no estamos sintiendo. El requisito básico para que una máquina tenga la capacidad de expresar emociones es que tenga canales de comunicación, como son la voz o las imágenes, y una capacidad de comunicar información afectiva sobre esos canales (Picard, 1997). Por ejemplo, una máquina o un agente software que exhibe una cara podría utilizar una expresión tal como una sonrisa, o un carácter animado que interactúa con un niño podrá personificarse y ser muy expresivo emocionalmente, y podría ser considerado un éxito en un contexto de entretenimiento si el niño lo encontrase más agradable y quisiese pasar más tiempo con él (Picard, 1997).

Por esa razón se utilizan personajes digitales en entornos de entretenimiento y juegos, y es que prometen ser un compañero casi humano en actividades como la enseñanza, dar consejos y la distribución de información, entre otras (Graf, 2005). Los humanos somos capaces de reconocer varias señales en el habla que nos permiten inferir fácilmente el estado emocional de nuestro interlocutor y adaptarnos a él. De igual forma, un personaje digital, o cualquier otro tipo de interfaz hombre máquina, debería ser capaz de hacer lo mismo para facilitar la interacción. Por ejemplo, en una situación en la que un usuario se está sintiendo muy frustrado porque la interfaz de manera 'estúpida' muestra la misma 'torpe' reacción que en los 10 intentos anteriores, una interfaz de usuario sensible a la emociones sería muy beneficiosa (Graf, 2005).

Un ejemplo de impacto positivo del uso de un personaje digital que expresa emociones en un entorno de realidad aumentada se describe en (Graf, 2003). La figura virtual era capaz de establecer una relación personal entre el usuario visitante y el contenido a mostrar. Los autores asociaron este hecho al comportamiento humano del personaje (que incluía la expresión de algunas emociones) y a su incorporación en un espacio real.

Y para que los personajes digitales puedan generar ese comportamiento emocional, elegir qué expresiones emocionales van a mostrar, y lograr el objetivo de establecer un vínculo con el usuario o influir en su estado emocional, es necesario dotarlos de modelos afectivos que cuenten con *sistemas de toma de decisiones* capaces de tener en cuenta las emociones en su funcionamiento. En el siguiente punto hablaremos de estos sistemas.

2.1.2 Toma de Decisiones

Los humanos estamos constantemente tomando decisiones. Todo lo que hacemos, consciente o inconscientemente, es el resultado de alguna decisión que hemos tomado. La información que vamos recopilando nos ayuda a comprender lo que nos rodea y lo que ocurre en nuestro entorno, de manera que podamos razonar adecuadamente a la hora de tomar decisiones sobre lo que sucede a nuestro alrededor.

El área de investigación relacionada con la toma de decisiones estudia la identificación y selección de alternativas basadas en los valores y preferencias de quien ha de tomar la decisión. Tomar una decisión implica la existencia de una variedad de opciones entre las que elegir, donde no sólo queremos conocer el mayor número posible de posibilidades sino que también queremos elegir aquella que mejor se ajusta a nuestros objetivos, deseos, preferencias, etc. (Harris, 1998). Gracias a este campo de estudio se desarrollaron métodos (a los que más tarde se daría soporte computacional) para ayudar a la toma decisiones en múltiples situaciones, aunque con especial énfasis en la toma de decisiones de gestión empresarial o en áreas de la industria (ver (Fülöp, 1998) para una amplia revisión de estos métodos). El desarrollo y difusión de los ordenadores también condujo a la creación de modelos computacionales que automatizaban determinados mecanismos para la selección de la mejor decisión; en (Tyrrell, 1993) el autor realiza un detallado estudio de un buen número de estos modelos computacionales. Finalmente, es necesario destacar la tremenda aportación que los investigadores del campo de la inteligencia artificial han realizado a los sistemas para la toma de decisiones, ya sea directamente (ver (Pomerol, 1997) para un listado de sistemas para la toma de decisiones) o indirectamente (p. ej., las redes neuronales o los algoritmos genéticos no fueron concebidos para la toma de decisiones, pero empezaron a ser utilizados por su buen comportamiento en la resolución de problemas que cuentan con múltiples variables).

Avances en otras disciplinas de la ciencia como la neuropsicología (Adolphs, 1996) han destacado la influencia de la información biológica del sujeto (salud, estado emocional, sensaciones físicas) para dirigir el proceso de toma de decisiones hacia resultados que le fuesen beneficiosos, basándose en experiencias pasadas en las que se disponía de información similar. La hipótesis tras esta influencia se conoce como la *hipótesis del marcador somático*. Esta hipótesis afirma que las decisiones que son tomadas en circunstancias similares a experiencias previas, y cuyo resultado podría ser potencialmente dañino o beneficioso, inducen una respuesta somática que "marca" los posibles resultados futuros como importantes y como peligrosos o beneficiosos (Damasio, 1994). De esta manera, cuando un marcador negativo

se asocia con un determinado resultado, nos sirve como alarma para evitar las acciones/decisiones que conducen a ese resultado.

En el siguiente punto hablaremos sobre los sistemas de toma de decisiones más utilizados para simular comportamientos naturales en personajes digitales, centrándonos en aquellos sistemas de los que existen ejemplos que sí tienen en cuenta las emociones.

2.1.3 Estado del arte

Entre la variedad de propuestas e implementaciones de sistemas y modelos computacionales para la toma de decisiones utilizados en la generación sintética de comportamiento, hemos elegido los sistemas que detallamos en este punto por ser los más utilizados y relevantes dentro del campo de la computación afectiva.

2.1.3.1 Sistemas Multiagente (MAS)

El término agente (Wooldridge, 1995) se suele utilizar de un modo general para denotar a los sistemas basados en software que tienen las siguientes propiedades:

- autonomía: los agentes funcionan sin ninguna intervención directa por parte de humanos u otros agentes y tienen el control (de algún tipo) tanto sobre sus propias acciones como sus estado interno (Castelfranchi, 1995);
- habilidades sociales: los agentes son capaces de interactuar con otros agentes (y puede que incluso con humanos) mediante algún tipo de lenguaje de comunicación (Genesereth, 1994);
- reactividad: los agentes perciben su entorno (que puede ser tanto el mundo real, un usuario vía una interfaz de usuario, un grupo de agentes, o puede que una combinación de todos), y son capaces de responder de manera oportuna a los cambios que puedan ocurrir;
- proactividad: los agentes no sólo reaccionan a cambios en el entorno, son capaces de tomar la iniciativa y mostrar un comportamiento con un objetivo determinado.

Desde la última década muchos investigadores han utilizado los sistemas multiagente para diseñar y construir modelos en los que implementar *agentes emocionales*, ya que las características descritas hacen de los agentes una herramienta muy útil para representar seres vivos con un sistema software (Wooldridge, 1995). De entre las propuestas existentes destacaremos aquellas que proporcionan un modelo más avanzado y complejo de afectividad.

2.1.3.1.1 Cathexis

La propuesta que hacen los autores de Cathexis (Velasquez, 1998a) (Velasquez, 1998b) representa una aproximación muy interesante al estudio de los mecanismos emocionales, ya que incluye una arquitectura de agentes simple, modular y extensible que da soporte a emociones de tipo primario y secundario (definidas en (Damasio, 1994)). Cathexis describe un modelo computacional distribuido para la generación de emociones (Velasquez, 1996) (Velasquez, 1997). Las emociones generadas son incorporadas a los agentes para influir en su comportamiento.

En su propuesta, una red de nodos conectados entre sí modela las emociones. Cada nodo se conoce como un "protoespecialista" y representa a una familia de emociones. Hay definidos 6 tipos de protoespecialista, es decir, seis familias de emociones: cólera, miedo, tristeza, felicidad, repugnancia y sorpresa. Cada nodo está asociado a unos sensores que detectan los estímulos tanto internos como externos. Esos estímulos detectados pueden activar las emociones de los protoespecialistas o bien cambiar la intensidad de la emoción que ya esté activada. La intensidad de una determinada emoción depende de los valores percibidos por los sensores y varía con el tiempo, esto es, una vez calculada la intensidad para una emoción, su nivel se decrementa con el tiempo a menos que los sensores detecten nuevos estímulos que la activen.

Las emociones activadas junto con sus intensidades afectan el comportamiento del agente, es decir, las acciones que realiza. A continuación resumimos los pasos del algoritmo que propone Velasquez (Velasquez, 1997) para generar el comportamiento del agente emocional:

1. Se recogen tanto las variables internas al agente como los estímulos del entorno.
2. Los valores tanto de impulsos como de emociones son actualizados.
3. Se actualizan los valores de todos los comportamientos tomando para ello los valores de los estímulos externos y de las variables internas.
4. Se elige como comportamiento a activar aquél con mayor valor después de la actualización.

Como se puede ver se trata de un sistema emocional en el que no se han explorado aspectos más cognitivos de las emociones como la previsión, la evaluación de la situación y de las posibles consecuencias de la respuesta emocional, el control de las emociones y el aprendizaje relacionado con las emociones.

2.1.3.1.2 Modelo emocional basado en niveles hormonales

En (Cañamero, 1997) la autora propone una arquitectura de agentes que utiliza motivaciones y emociones para realizar la selección de comportamiento. En esa arquitectura propuesta las emociones son activadas bien por eventos

relevantes o bien por patrones de simulación, y son identificadas por patrones definidos de parámetros fisiológicos (por ejemplo, niveles hormonales) específicos a cada emoción. Las emociones se activan en paralelo a la monitorización continua que realiza un sistema de control de la motivación que busca eventos significativos (externos o internos). La conexión de la emociones con el comportamiento es a través de su influencia sobre la motivación. Además de esa influencia, las emociones también afectan a cómo el agente percibe tanto el mundo que le rodea como su propio "cuerpo", lo que a su vez puede afectar al comportamiento.

Al igual que en la propuesta anterior podemos concluir que los autores no han tenido en cuenta que aspectos como la previsión de estados futuros según qué decisión se tome puede hacer que el comportamiento del agente parezca más natural e inteligente.

2.1.3.1.3 FLAME

La propuesta que se conoce como FLAME (*Fuzzy Logic Adaptive Model of Emotions*) un sistema capaz de generar emociones en agentes. Utiliza lógica difusa para representar los eventos, los objetivos y las emociones (El-Nasr, 1999). Estos valores de la lógica difusa son usados junto con un sistema de reglas para seleccionar el comportamiento del agente.

La arquitectura de agentes de FLAME está compuesta de 3 componentes principales (El-Nasr, 2000): la componente emocional, la componente de toma de decisiones y la componente de aprendizaje. El funcionamiento del algoritmo de selección del comportamiento del agente emocional se resume en los siguientes pasos:

1. El agente percibe eventos de su entorno.
2. Los eventos percibidos pasan a través de la componente emocional y de la de aprendizaje. La componente emocional evalúa los eventos y calcula los niveles de las emociones del agente. La valoración de un evento se basa en los objetivos del agente y las experiencias aprendidas en el pasado. La componente de aprendizaje mantiene las asociaciones y los patrones que enlazan entre los eventos utilizados en el proceso emocional.
3. Los niveles para todas las emociones son filtrados para producir un único estado emocional para el agente y, entonces, se selecciona el comportamiento emocional relacionado con el estado emocional obtenido.
4. La componente de toma de decisiones recibe el comportamiento emocional que influye en la acción específica a tomar por el agente.

El modelo de evaluación de las emociones en FLAME está basado en el modelo presentado en (Ortony, 1988), que consiste en un conjunto de reglas que dan lugar a emociones según las expectativas y los niveles de deseo de ocurrencia de un agente. Así, en esta propuesta las reglas de lógica difusa son utilizadas para medir el deseo de ocurrencia de un evento de la siguiente manera: si un objetivo O se ve afectado por un evento E en un grado A (donde A es un descriptor difuso cuyos valores son: alto, medio y bajo), y la importancia del objetivo O es B (donde B es otro descriptor difuso cuyos valores son: alta, media y baja), entonces el deseo de ocurrencia del evento E será D.

Esta propuesta adolece de un grave inconveniente por el hecho de utilizar un sistema basado en reglas, a saber, una vez definidas las reglas el sistema no se actualiza y la ocurrencia de eventos no contemplados en las reglas definidas pasará inadvertida para el agente, en el mejor de los casos, o puede que el agente se comporte de forma extraña e irracional a ojos de los usuarios.

2.1.3.2 Redes neuronales

Una red neuronal (artificial) es un modelo computacional inspirado en la biología que está formado por unos elementos de procesamiento (llamados neuronas) y una serie de conexiones entre esos elementos con unos coeficientes asociados (pesos), lo que constituye la estructura neuronal, además de unos algoritmos utilizados para "entrenar" la red (fijar los valores de los pesos) y otros para utilizar la red con datos reales (Anderson, 1995).

La estructura de una neurona artificial está definida por: una serie de entradas, que tienen unos "pesos" asociados; una función de entrada, cuyo cometido es agregar los valores recibidos de todas las entradas de que disponga la neurona; y una función de activación, que calcula el nivel de activación de una neurona como una función del valor agregado de entrada y de su estado previo (Davalo, 1991). Finalmente, la neurona envía por sus conexiones de salida (axones) una señal de salida equivalente al valor de activación calculado.

Las redes neuronales son también conocidas como modelos conexionistas debido al papel que juegan en ellas las conexiones entre las neuronas. Los pesos asociados a las conexiones son el resultado del proceso de entrenamiento y representan la memoria del modelo. Las principales características de las redes neuronales son (Anderson, 1995):

- aprendizaje: una red puede empezar sin ningún tipo de conocimiento sobre el problema que va a tratar y ser entrenada mediante un conjunto de ejemplos del problema, es decir, pares de valores de entrada suministrados y valores de salida esperados (lo que se conoce como entrenamiento supervisado), o bien empezar sólo con datos de entrada (entrenamiento no

supervisado); durante el entrenamiento los pesos de las conexiones van modificándose de tal manera que la red neuronal "aprende" a producir los valores de salida esperados a partir de los valores de entrada suministrados;

- generalización: ante un vector de valores de entrada diferente a los ejemplos utilizados en el entrenamiento, la red neuronal produce la mejor salida de acuerdo a los ejemplos utilizados en el entrenamiento. La correspondencia parcial es la opción más adecuada en muchos casos puesto que los datos que ya conocemos no coinciden de manera exacta con los nuevos;
- alto grado (potencial) de paralelismo: durante el procesamiento de los datos de entrada, muchas neuronas pueden activarse simultáneamente;
- robustez: aunque algunas neuronas empiezan a funcionar de manera errática, el sistema en su conjunto puede seguir funcionando (bastante) bien.

Un ejemplo de utilización de una red neuronal como sistema de toma de decisiones influenciado por emociones lo podemos encontrar en (Inoue, 1996), donde los autores presentan un sistema que simula un hábitat virtual donde coexisten animales herbívoros y carnívoros. Los herbívoros seleccionan la siguiente acción a realizar según su propio sistema de toma de decisiones, mientras los carnívoros actúan al azar. Dentro del entorno virtual la hierba crece con el paso del tiempo, sólo los herbívoros copulan y pueden reproducirse, y el tiempo de vida restante lo marca un nivel de energía que se incrementa al comer y se reduce al realizar acciones y con el tiempo. Si el nivel de energía llega a cero, el animal muere. El sistema de toma de decisiones de los herbívoros lo simula una red neuronal de 3 capas, que cuenta con una capa de entrada con 171 unidades de información del entorno. Esa información se mide con señales binarias. Todos los animales pueden ver en todas direcciones hasta dos posiciones más allá de su posición actual. El área de visión se comunica a la red neuronal con un valor de 0 si no hay ningún animal en esa posición, o 1 en caso contrario. Entonces la red produce como salida la acción a realizar. El sistema propuesto cuenta además con un algoritmo genético utilizado para evolucionar el sistema de toma de decisiones (Inoue, 1996).

Aunque el sistema realmente no modela ninguna emoción de partida, con el paso de las generaciones, los autores observaron que los animales se comportaban como si sintiesen algunas emociones. Este hecho constata uno de los puntos débiles de las redes neuronales en general, a saber, están concebidas como unas cajas negras, lo que hace que extraer información de ellas sea complicado. Otros problemas son la dificultad para entrenarlas si no se seleccionan los valores iniciales adecuados, y que no existen criterios

prácticos que se puedan utilizar para seleccionar la topología (número de nodos y capas), únicamente criterios muy conservadores (Poggio, 1990).

2.1.3.3 Hierarchical network of decisions

Entre los autores de esta propuesta se encuentra Rosalind Picard, la persona que acuño el término *affective computing* aglutinando conceptos, teorías y modelos sobre emociones y cognición de diversos campos de la ciencia (Picard, 1997).

En (Ahn, 2005) se presenta la propuesta en la que se usa una red jerárquica de decisiones para dotar de comportamiento emocional a entidades. En esta propuesta cada nodo/decisión (los autores utilizan el término *affective-cognitive decision*, ACD) de la red tiene un proceso interno que puede seleccionar la decisión de nivel inferior más adecuada (probabilísticamente) conocidos los estados cognitivo y emocional actuales. Este proceso mantiene varios modelos probabilísticos para simular ACDs de niveles inferiores junto con sus valores cognitivos y emocionales. En otras palabras, para medir cómo de atractiva es una cierta decisión en términos cognitivos y emocionales. Una vez que un ACD de un nivel inferior con un valor interno positivo ha sido localizado, la entidad procede a ejecutarlo y obtiene la "recompensa" externa. Entonces, todos los modelos probabilísticos son actualizados para aprender qué ACDs son mejores en cada situación.

Aunque este algoritmo utiliza *reinforcement learning* para hacer que las entidades aprendan cuáles son las decisiones más apropiadas para cada par de estados cognitivo y emocional, y a pesar que realiza una simulación para realizar la selección del ACD, pierde su efectividad al hacer que el proceso de búsqueda termine en el primer ACD con valor positivo que se encuentre (ni siquiera el máximo de los encontrados, por ejemplo). Además, no tiene en cuenta secuencias pasadas de ocurrencias de estados cognitivos y emocionales para, por ejemplo, sacar provecho del conocimiento de las entidades para simular secuencias de decisiones que podrían mejorar la recompensa obtenida a medio plazo.

2.2 Toma de Decisiones en Sistemas Sensibles al Contexto

Tal y como vimos en el capítulo de introducción de esta tesis, la toma de decisiones basadas en la sensibilidad al contexto tiene una relevancia muy significativa en los ambientes inteligentes puesto que el comportamiento que calificamos como inteligente, y que esperamos que los ambientes exhiban, no puede existir sin la adecuada consideración del contexto del entorno. Es por ello por lo que se hace imprescindible que la información contextual a tener en

cuenta no sólo sea "capturada", sino también modelada para que los distintos dispositivos electrónicos presentes en el ambiente inteligente puedan procesar la información de contexto de manera eficiente. Un buen modelado de la información contextual les permitirá compartir información de manera sencilla, permitiendo tanto su procesamiento como la inserción de nuevo conocimiento (información derivada o inferida de los datos de contexto iniciales) por parte de sistemas heterogéneos. No obstante, un buen modelado de la información contextual debe también contemplar al ser humano y ofrecer mecanismos ágiles para la gestión y manipulación de la información contextual.

El tratamiento semántico de la información contextual reúne ambas características, al presentar una estructura apta para la manipulación tanto por seres humanos como por máquinas, y añade un valor semántico a los datos, de manera que hace posible el razonar sobre ellos e inferir nueva información o conocimiento del entorno. Así, la utilización de ontologías para el modelado y tratamiento de la información contextual en un sistema de computación sensible al contexto implica ciertas ventajas con respecto a otras técnicas. Si analizamos las principales aplicaciones de las ontologías, según (Gruninger, 2002), pueden identificarse tres grandes tipos de uso para las mismas que representan a su vez áreas en las que las tecnologías basadas en ontologías cuentan con importantes ventajas:

- Comunicar y compartir información: por sus facilidades para ser incluidas en un proceso computacional, así como para poder ser entendidas por humanos, las ontologías pueden servir como vocabulario de comunicación para diversos agentes, sean éstos máquinas, aplicaciones o seres humanos. Este tipo de aplicación presenta las ontologías como nexos de unión entre agentes y seres humanos, superando otras técnicas exclusivas de agentes software.

- Inferencia lógica y razonamiento: las ontologías pueden utilizarse para inferir conocimiento implícito partiendo de un conjunto de conocimiento explicito y unas reglas o procesos lógicos de inferencia. Esta característica permite la creación de sistemas ubicuos o de inteligencia ambiental, disponiendo de múltiples herramientas para el desarrollo de comportamiento inteligente que generalmente no se encuentran disponibles para las restantes técnicas de modelado contextual analizadas.

- Reutilización del conocimiento: el uso de ontologías y el establecimiento de relaciones jerárquicas entre ellas permite mediante la importación ontológica reutilizar ontologías generales en la definición de ontologías para un dominio particular. Esto representa un avance respecto a otras técnicas de modelado dado que permite el desarrollo comunitario de

ontologías más o menos estándares capaces de agilizar los procesos de desarrollo e implementación de sistemas inteligentes.

Sin embargo, hasta la fecha, la mayor parte de sistemas que dan soporte a la filosofía en la que se basan los ambientes inteligentes entienden el contexto como un elemento relacionado únicamente con la situación actual en la que se encuentra el entorno o el usuario, sin tener en cuenta las relaciones temporales que puedan darse entre diferentes situaciones contextuales. En este sentido nos encontramos con situaciones paradójicas en las que sistemas *context-aware* como Context Toolkit (Salber, 1999), CoBrA (Chen, 2004), CASS (Fahy, 2004), SOCAM (Gu, 2004) y CORTEX (Biegel, 2004) sí almacenan de manera persistente la información de contexto capturada, por lo que sería posible aprender y generar comportamiento para ciertas situaciones contextuales que se dieron en el pasado y que pudieran ser relevantes, aunque ninguno de ellos utiliza algoritmo de aprendizaje alguno que les permita generar una respuesta más proactiva, más inteligente, en definitiva, más adecuada. En (Baldauf, 2007) se puede encontrar una revisión más pormenorizada de las características de estos y otros sistemas.

Sin embargo, el comportamiento secuencial es uno de los elementos fundamentales presentes en todo tipo de seres dotados de un nivel básico de inteligencia. En particular, en las actividades humanas podemos encontrar diversos ejemplos en los que el comportamiento secuencial es crítico, desde el razonamiento para la resolución de problemas hasta la articulación del lenguaje. En el campo de las ciencias de la computación el aprendizaje secuencial es una componente importante en múltiples dominios: procesado del lenguaje natural, planificación, robótica, reconocimiento del habla, predicción de series temporales, etc.

En el ámbito de los ambientes inteligentes, uno de los aspectos más relevantes que suele caracterizar a los usuarios es la movilidad. Como consecuencia de la misma, trazan rutas a diario en el espacio físico en el que se mueven y en las que desarrollan distintos tipos de actividades. Dicho de otro modo, podemos afirmar que dichos usuarios siguen "secuencias contextuales" no sólo caracterizadas por la información relativa a la posición física en la que se encuentran sino también por el contexto (situación, actividad,…) en el que dicha presencia en el espacio físico cobra su verdadero sentido. Una secuencia contextual es una sucesión en el espacio y en el tiempo de situaciones de interés contextual. Una situación contextual es un concepto genérico definido como una colección de propiedades contextuales con sus correspondientes valores. Por ejemplo, una situación contextual donde las propiedades definidas fueran simplemente coordenadas de un espacio bidimensional permitiría la definición de secuencias contextuales que representarían simplemente rutas

geográficas; mientras que si las situaciones de interés contextual fueran descripciones de elementos turísticos (museos, monumentos, edificios históricos,…) a los que accede un usuario, entonces tendríamos secuencias contextuales de naturaleza turística.

El aprendizaje de secuencias contextuales de cualquier tipo o nivel de abstracción es fundamental para poder:

- Obtener información sobre situaciones contextuales que es probable que se den en el futuro para poder tomar decisiones adecuadas por parte del usuario con anticipación.
- Realizar con efectividad una provisión anticipada de información o de servicios que serán necesarios en el futuro permitiendo iniciar tareas complejas de búsqueda, filtrado y personalización de información y servicios con antelación.
- Obtener información desconocida para un usuario derivada a partir del análisis de las secuencias conocidas de otros usuarios con los que se comparte una situación contextual.

2.2.1 Toma de Decisiones Basadas en Secuencias Contextuales

Es evidente que el aprendizaje de secuencias es una tarea no trivial y que ha sido analizada desde diferentes puntos de vista entre los que destacan los aspectos relacionados con la predicción de secuencias y los que abordan el reconocimiento de las mismas. Hasta la fecha se han realizado propuestas algorítmicas para el aprendizaje secuencial entre los que podemos encontrar redes neuronales recurrentes, modelos de Markov ocultos o algoritmos inmunes.

2.2.1.1 Redes Neuronales

El problema del aprendizaje de secuencias ha sido tradicionalmente formalizado en la teoría de redes neuronales con lo que se ha denominado "aprendizaje de secuencias temporales". El estudio de dichas secuencias ha sido objeto de estudio en diferentes dominios, desde la planificación de trayectorias en robótica hasta en visión artificial y reconocimiento del habla. En la mayor parte de las ocasiones las redes neuronales propuestas proveen dos mecanismos distintos: uno para el aprendizaje de la información espacial y otro para la temporal. El mecanismo principal almacena la secuencia de eventos independientemente de las dependencias temporales entre ellos. Posteriormente, el segundo mecanismo, también denominado memoria a corto plazo, extrae y aprende las relaciones temporales entre las secuencias. Un ejemplo claro de este tipo de aprendizaje es el trabajo de Moga (Moga, 2002)

en el que se define un modelo neuronal en el que se utiliza neuronas predictivas para aprender a predecir los intervalos de tiempo que tienen lugar entre dos eventos. El intervalo comienza con el disparo de neuronas de derivación y finaliza con el disparo de neuronas de entrada. Las neuronas predictivas aprenden dichos intervalos gracias a la existencia de neuronas granulares. Este trabajo demuestra que el modelo de red neuronal propuesto permite a un robot aprender distintos tipos de bailes (entendidos como secuencias de eventos que han de ejecutarse).

Por otro lado, también existen estrategias que utilizan memorias asociativas con mecanismos de aprendizaje de Hebbian para el aprendizaje de secuencias, como son las redes de Hopfield. En este sentido trabajos como los de (Wang, 2003), (Miyoshi, 2004) o (Maurer, 2005) demuestran el uso de este tipo de redes para el aprendizaje de series temporales, pero no permiten el aprendizaje efectivo cuando dichas series presentan solapamientos. Este problema queda resuelto en (Berthouze, 2006), en el que se presenta un modelo de red neuronal compuesto por un modulo central consistente en un vector de columnas independientes de neuronas con aprendizaje débil (*Leaky learning*); un módulo de entrada y un módulo de contexto. Los patrones de entrada alimentan directamente al módulo central mientras que los patrones de contexto alimentan al módulo de contexto que está conectado de forma completa con el módulo central. La regla de aprendizaje tiene como objetivo minimizar el error entre la salida actual y el próximo input. Una copia del contexto inferido es realimentada al módulo de contexto mediante conexiones uno a uno. A su vez, el módulo de salida se conecta con cada columna del módulo central. El diseño de las neuronas en el módulo central está orientado a controlar el tiempo que transcurre hasta la primera respuesta, la duración de la respuesta y el tiempo hasta que la neurona vuelve a estar en reposo, de forma que la red neuronal pueda codificar la información temporal asociada a una secuencia de entrada. El proceso de aprendizaje está dirigido por un ajuste dinámico de la velocidad de aprendizaje. Esto permite que la red aprenda únicamente cuando es necesario y pase a modo de consolidación en otro caso. De esta forma la red muestra características de plasticidad necesarias cuando una nueva información secuencial sobrescribe de forma parcial o completa la memoria de secuencias aprendidas con anterioridad. Por otro lado, la integración de información de contexto permite evitar las interferencias que tendrían lugar entre secuencias con elementos en común. La información contextual proporciona a la red regularidades con poca frecuencia de cambio que permite distinguir entre diferentes secuencias basándose en cuándo y dónde aparecen las mismas.

Este tipo de modelo permite integrar de forma transparente las fases de aprendizaje y de rememoración de forma que la red neuronal pasa gradualmente de aprender a predecir los elementos de las secuencias. Además

de evitar las interferencias entre secuencias con segmentos comunes, este mecanismo permite completar secuencias en patrones incompletos o contaminados con ruido y proporciona un mecanismo para explorar de forma selectiva el espacio de secuencias aprendidas durante la fase de *free-recall*.

Si bien queda demostrado en (Berthouze, 2006) la efectividad para la memorización de secuencias en contextos, los resultados empíricos se basan en secuencias de hasta tamaño 6 y orientados a la memorización de letras que aparecen en cadenas, esto es, la escalabilidad del modelo y la capacidad para memorizar secuencias en las que la naturaleza de los datos sea compleja no son aspectos abordados en ese trabajo.

Alternativamente, en (Ichishita, 2007) se demuestra el uso de redes neuronales compuestas por neuronas de pulsos con un mecanismo de propagación hacia delante y que no presenta los problemas de interferencia u olvido que se presentan en las redes neuronales recurrentes. Con este tipo de red se han realizado experimentos para el aprendizaje de secuencias de composiciones musicales en las que se dan secuencias de notas que han de ser memorizadas por la red neuronal. Si bien el mecanismo es capaz de aprender secuencias temporales de longitud 800, la codificación de la información (*pitch* de la nota y duración) determina en gran medida la naturaleza de la red neuronal por lo que el modelo propuesto no es aplicable en general al aprendizaje de secuencias de índole más genérica.

2.2.1.2 Modelos de Markov ocultos

Una cadena de Markov $q = \{q_t\}, t \in \mathbb{N}$ es un proceso estocástico de Markov discreto. Un proceso estocástico se llama de Markov si conocido el presente, el futuro no depende del pasado.

Formalmente una cadena de Markov se define como $(\mathcal{Q}, \mathcal{A})$, donde $\mathcal{Q} = \{1,2,\dots,\mathbb{N}\}$ son los posibles estados de la cadena y $\mathcal{A} = (a_{ij})_{n\times n}$ es una matriz de transición de estados en el modelo. La condición principal para que sea una cadena de Markov establece que las probabilidades de transición y emisión dependen únicamente del estado actual y no de los pasados. En términos probabilísticos: $P[q_t = j \mid q_{t-1} = i, q_{t-2} = k, \dots] = P[q_t = j \mid q_{t-1} = i] = a_{ij}(t)$.

Un modelo oculto de Markov (*Hidden Markov Model, HMM*) es un proceso estocástico que consta de un proceso de Markov no observado (oculto) y un proceso observado cuyos estados son dependientes estocásticamente de los estados ocultos. En estos sistemas se evoluciona con el paso del tiempo de forma aleatoria transitando entre estados y emitiendo al azar un símbolo de un alfabeto Σ. Solamente los símbolos del alfabeto emitidos por el proceso son

observables, pero no la secuencia de estados *q*, y por ello se denomina "oculto" de Markov, puesto que el proceso de Markov es no observado.

Desde otro punto de vista, un HMM se puede modelar como un autómata de estados finitos dado que son como estos pero con transiciones entre estados anotados con probabilidades y con la única restricción de que las probabilidades de todas las transiciones salientes de un estado han de sumar la unidad.

Los HMMs se pueden considerar como sistemas generativos estocásticos, a menudo empleados en el modelado de series de tiempo. Tradicionalmente los HMMs han sido de gran utilidad en áreas de investigación como el reconocimiento del habla dado que se pueden capturar mediante estos modelos las dependencias temporales existentes en una serie de sonidos lo cual es de gran utilidad para identificar fonemas. En este sentido el problema se plantea como un proceso de ajuste del modelo que es el inverso al proceso de generación de una secuencia de elementos observables, esto es, el problema es encontrar al mejor autómata que describa una secuencia de observaciones. Este procedimiento puede aplicarse a cualquier secuencia de observaciones categóricas. Especialmente interesante es el trabajo presentado en (Kishner, 2005), en el que se presenta una forma de aplicación de HMMs para capturar las dependencias temporales y las dependencias multivariable que aparecen en series temporales multivariable. En dicho trabajo se modulariza el proceso de construcción de dichos modelos mediante la separación de ambos tipos de dependencias combinando el uso de HMMs con árboles y bosques condicionales de Chow-Liu (Chow, 1968).

Si bien los HMMs han demostrado su efectividad para modelar variables de tipo discreto en series temporales todavía existen diversos problemas que impiden su uso de forma generalizada para el aprendizaje de secuencias temporales en dominios o aplicaciones reales multivariable. Por un lado, los HMMs son muy sensibles a la presentación de secuencias incompletas puesto que éstas afectan seriamente a la calidad de los modelos. Por otro lado, dada la naturaleza de representación de los estados del problema los HMMs presentan dificultades a la hora de incorporar a los mismos el conocimiento proporcionado por expertos. Además, esto se une a la problemática de identificar las variables que tienen un impacto significativo en las variables observadas dado que la inclusión de variables innecesarias degrada las prestaciones en términos computacionales de dichos modelos.

2.2.1.3 Algoritmos inmunes

Estos algoritmos se basan en la propia naturaleza del sistema inmune. Los sistemas inmunes naturales protegen a los animales de los patógenos externos

peligrosos, como bacterias, virus, parásitos y toxinas. Para ello el problema a resolver es el siguiente: diferenciar lo "propio" de lo "ajeno", es decir, diferenciar a "uno mismo" de los "otros" peligrosos y eliminar esos "otros". Evidentemente, el sistema inmune no tiene una lista completa de qué es lo que puede ser ajeno y qué no lo es; menos aún, es posible que esta lista esté codificada en el código genético de los seres humanos y el resto de mamíferos superiores (que son los que tienen un sistema inmune). Pero lo cierto es que lo consiguen, y lo hacen a través de una serie de mecanismos:

- Diversidad: manteniendo "bibliotecas" de posibles componentes de antígenos (donde un antígeno es una sustancia que poseen los virus, las bacterias, etc. que hace que el sistema inmune reaccione), y que se usan para reconocerlos;
- Especificidad: pueden crear un anticuerpo (que son los elementos que utiliza el sistema inmunológico para detectar y neutralizar los elementos extraños) específico para cada antígeno;
- Memoria: pueden guardar las respuestas inmunes llevadas a cabo anteriormente; y
- Tolerancia: no atacan al propio organismo, sólo a aquello que se considera "extraño".

Los sistemas inmunes son sistemas adaptables en los que el aprendizaje debe ser un mecanismo que pueda evolucionar de forma similar a la evolución biológica. Su función principal es la de proporcionar un mecanismo de defensa para el cuerpo que pueda identificar y eliminar aquel material que sea extraño y peligroso. Algunos modelos de sistemas inmunes se basan en un universo en el que los antígenos (material extraño) y los linfocitos (células que hacen el reconocimiento del material extraño) están presentes mediante cadenas de diferentes símbolos. Esta definición de modelo se puede emplear para, por ejemplo, estudiar varios aspectos diferentes de los sistemas inmunes, como puedan ser la detección de patrones comunes (esquemas) en ambientes ruidosos, su habilidad para descubrir y mantener la cobertura de varias clases de patrones, y su habilidad para aprender eficientemente, incluso cuando no se han definido todos los anticuerpos o antígenos presentes en el sistema.

Aunque la cantidad de modelos y aplicaciones de los sistemas inmunes va en ascenso, sólo existe un esquema muy general de cuáles son los elementos esenciales que debe poseer un sistema inmune artificial (SIA):

- Representación de los componentes del sistema.
- Un conjunto de mecanismos para evaluar las interacciones.

- Un proceso de adaptación que gobierne las dinámicas del sistema.

A continuación se presentan un par de modelos de sistema inmune artificial, cada uno de ellos basado en un proceso determinado llevado a cabo en los sistemas inmunes naturales. Pero antes de exponerlos es conveniente definir una serie de conceptos:

- Antígeno: sustancia química que poseen ciertos elementos (tales como virus y bacterias). Esta sustancia química hace reaccionar al sistema inmune para neutralizar los elementos que poseen dicha sustancia.
- Anticuerpo: elemento que utiliza el sistema inmunológico para detectar y neutralizar los elementos extraños.
- Linfocito: tipo de glóbulo blanco encargado de reconocer antígenos específicos, producir los anticuerpos necesarios y eliminar las células extrañas.

Desde el punto de vista del aprendizaje de secuencias contextuales, los conceptos anteriores tendrían la siguiente correspondencia: el antígeno representaría una secuencia de situaciones contextuales frente a la que el sistema ha de reaccionar; el linfocito sería la parte del sistema encargada de reconocer la ocurrencia de la secuencia contextual a partir de la sucesión de situaciones contextuales y de generar el anticuerpo adecuado, es decir, elegir el comportamiento que el sistema usará como respuesta a la secuencia contextual.

2.2.1.3.1 Selección Negativa

El algoritmo de Selección Negativa está inspirado en el mecanismo principal del timo (órgano linfático), que es el órgano que produce un conjunto de células T (un tipo de linfocitos) capaces de envolver a los antígenos ajenos. La cuestión biológica reside en que en el timo se producen varios tipos de células T, algunas de las cuales reaccionan frente a antígenos propios, lo cual no resulta interesante. Por ello estas células T son eliminadas, y a todo este proceso se le denomina selección negativa.

El primer algoritmo de selección negativa fue propuesto por Forrest (Forrest, 1994) para detectar la manipulación de datos llevada a cabo por un virus informático. El punto de partida de este algoritmo es un conjunto de cadenas propias $\mathcal{S}$ que definen el estado normal del sistema. La tarea a realizar después es generar un conjunto de detectores $\mathcal{D}$ que sólo reconozcan y envuelvan a lo que sea complementario a $\mathcal{S}$. Una vez se obtengan, estos detectores se pueden utilizar para monitorear el estado del sistema, $\mathcal{S}$, y por tanto detectar si ha sido modificado o no.

2.2.1.3.2 Selección Clonal

La teoría de la selección clonal ha sido utilizada como principio para el desarrollo de un SIA que lleve a cabo tareas de optimización y reconocimiento de patrones. En particular, se fundamenta en el proceso de maduración de las células B (un tipo de linfocitos). Este proceso se comienza cuando una célula B reacciona ante un antígeno. En este momento se comienzan a clonar (madurar) células B del mismo tipo que la que ha detectado el antígeno; así se evita que proliferen células B para otros antígenos diferentes. Estos SIA también suelen utilizar la idea de células de memoria para mantener las buenas soluciones al problema que se está solucionando. Castro y Timmis en (Castro, 2002b) subrayan dos características importantes que se pueden utilizar del mencionado proceso de maduración/clonación. La primera es que la proliferación de las células B es proporcional a la afinidad con el antígeno que envuelve, de tal forma que cuanta mayor afinidad, más células B de ese tipo se producen. La segunda está relacionada con las mutaciones sufridas por los anticuerpos de una célula B. Estas mutaciones son inversamente proporcionales a la afinidad con el antígeno que envuelven. Usando estas dos características, Castro y Von Zuben (Castro, 2002a) han desarrollado uno de los SIA basados en selección clonal más utilizados: CLONALG. Éste es utilizado principalmente para tareas de reconocimiento de patrones y optimización de funciones multimodales. Cuando se aplica en el reconocimiento de patrones, un conjunto de patrones $\mathcal{S}$ a detectar se consideran como antígenos. Así que la tarea de CLONALG es producir un conjunto de anticuerpos $\mathcal{M}$ que detecten los miembros de $\mathcal{S}$.

2.3 Toma de Decisiones Basada en Métodos Metaheurísticos: Una Aproximación Integradora

A menudo el tamaño del espacio de un problema puede verse incrementado exponencialmente a medida que también se incrementa la complejidad del mismo. Por lo tanto, para abordar instancias de problema de gran tamaño, es habitual utilizar métodos aproximados capaces de encontrar soluciones subóptimas en un tiempo relativamente pequeño. Estos algoritmos, generalmente conocidos como heurísticas, se basan en el conocimiento que se tiene del problema para construir soluciones o mejorarlas, tal y como lo haría un experto en ese tipo de problemas. Sin embargo, esta característica hace que normalmente sólo sean aplicables a un tipo específico de problemas. Por esta razón el foco de atención se ha fijado en una nueva clase de algoritmos, conocida como metaheurística. En (Dorigo, 2004) se define una metaheurística como un conjunto de conceptos algorítmicos que se pueden utilizar para definir métodos heurísticos aplicables a un amplio espectro de problemas; y en (Glover, 1997) los autores ofrecen una explicación que clarifica lo que es una

metaheurística: una metaheurística alude a una estrategia maestra que guía y modifica otras heurísticas con el fin de producir soluciones más allá de aquéllas que generalmente se obtienen en la búsqueda de óptimos locales.

Los dos dominios que hemos considerado como claves en el avance y el futuro de los ambientes inteligentes, la toma de decisiones basada en estados afectivos y en la predicción de situaciones contextuales, pueden ser vistos como problemas que comparten el inconveniente descrito de escalabilidad: la complejidad puede aumentar exponencialmente con el tamaño del problema. Por esta razón consideramos oportuno tener en cuenta las propuestas metaheurísticas como posible solución o estrategia a tomar como base. Sin embargo, esta similitud no bastaría para justificar una aproximación unificada a ambos dominios. Lo que realmente justifica explorar la aproximación metaheurística para abordar ambas problemáticas es el hecho de que comparten una característica más fundamental: poder ser representados como problemas de optimización. Esto es así porque lo que se pretende en ambos es maximizar alguna función de una o múltiples variables, por un lado el beneficio afectivo total obtenido como consecuencia de la toma de una serie consecutiva de decisiones y, por otro, el grado de similitud o de compatibilidad semántica entre una situación contextual de referencia y los estados contextuales que previsiblemente se darán en el futuro, en los que nos basaremos para poder tomar de forma anticipada decisiones compatibles con dicha predicción. En ambos casos, las posibles soluciones pueden verse como conjuntos ordenados de estados del problema. Quizás esta visión no sea tan evidente en el caso de la toma de decisiones influenciada por emociones, por lo que pasamos a detallar este punto.

El campo de la toma de decisiones está íntimamente ligado al campo de resolución de problemas (*problem solving*), incluso podríamos considerarlo un caso particular de éste ya que utilizan básicamente la misma definición (Newell, 1972) para hablar del espacio del problema que abordan (*problem solving* de manera general y *decision making* busca la mejor acción a realizar). El espacio del problema está formado por:

- un estado inicial;
- un estado objetivo que se quiere alcanzar;
- una serie de operadores para transformar el problema del estado inicial al estado objetivo en una secuencia de pasos;
- un conjunto de restricciones que deben ser satisfechas, incluyendo restricciones en la aplicación de los operadores.

De esta manera, el proceso de resolución del problema puede verse como una búsqueda del camino que conecta el estado inicial con el final.

Hecha la oportuna aclaración, a continuación presentamos dos de los métodos metaheurísticos más relevantes y empleados en multitud de dominios dada su versatilidad: los algoritmos genéticos y las colonias de hormigas.

2.3.1 Algoritmos genéticos

Los algoritmos genéticos son métodos adaptativos que pueden usarse para resolver problemas de búsqueda y optimización. Están basados en el proceso genético de los organismos vivos. A lo largo de las generaciones, las poblaciones evolucionan en la naturaleza de acorde con los principios de la selección natural y la supervivencia de los más fuertes, postulados por Darwin (Darwin, 1859). Por imitación de este proceso, los algoritmos genéticos son capaces de ir creando soluciones para problemas del mundo real. La evolución de dichas soluciones hacia valores óptimos del problema depende en buena medida de una adecuada codificación de las mismas.

Los principios básicos de los algoritmos genéticos fueron establecidos por (Holland, 1975), y se encuentran bien descritos en varios textos (Goldberg, 1989), (Davis, 1991), (Michaliewicz, 1992), (Reeves, 1993).

En la naturaleza los individuos de una población compiten entre sí en la búsqueda de recursos tales como comida, agua y refugio. Incluso los miembros de una misma especie compiten a menudo en la búsqueda de un compañero. Aquellos individuos que tienen más éxito en sobrevivir y en atraer compañeros tienen mayor probabilidad de generar un gran número de descendientes. Por el contrario individuos poco dotados producirán un menor número de descendientes. Esto significa que los genes de los individuos mejor adaptados se propagarán en sucesivas generaciones hacia un número de individuos creciente. La combinación de buenas características provenientes de diferentes ancestros, puede a veces producir descendientes “superindividuos", cuya adaptación es mucho mayor que la de cualquiera de sus ancestros. De esta manera, las especies evolucionan logrando unas características cada vez mejor adaptadas al entorno en el que viven.

Los algoritmos genéticos usan una analogía directa con el comportamiento natural. Trabajan con una población de individuos, cada uno de los cuales representa una solución factible a un problema dado (el conjunto de soluciones de un problema recibe el nombre de fenotipo) y en el que la información de la solución se codifica en una cadena, generalmente binaria, llamada cromosoma. Los símbolos que forman la cadena son llamados los genes. Cuando la representación de los cromosomas se hace con cadenas de dígitos binarios se le conoce como genotipo. Los cromosomas evolucionan a través de iteraciones,

llamadas generaciones. En cada generación, los cromosomas son evaluados usando alguna medida de aptitud. A cada individuo se le asigna un valor o puntuación, relacionado con la bondad de dicha solución. En la naturaleza esto equivaldría al grado de efectividad de un organismo para competir por unos determinados recursos. Cuanto mayor sea la adaptación de un individuo al problema, mayor será la probabilidad de que el mismo sea seleccionado para reproducirse, cruzando su material genético con otro individuo seleccionado de igual forma. Este cruce producirá una nueva generación de individuos, descendientes de los anteriores, que compartirán algunas de las características de sus padres. Cuanto menor sea la adaptación de un individuo, menor será la probabilidad de que dicho individuo sea seleccionado para la reproducción, y por tanto de que su material genético se propague en sucesivas generaciones.

De esta manera se produce una nueva población de posibles soluciones, la cual reemplaza a la anterior y verifica la interesante propiedad de que contiene una mayor proporción de buenas características en comparación con la población anterior. Así, a lo largo de las generaciones, las buenas características se propagan a través de la población. Favoreciendo el cruce de los individuos mejor adaptados, van siendo exploradas las áreas más prometedoras del espacio de búsqueda. Si el algoritmo genético ha sido bien diseñado, la población convergerá hacia una solución óptima del problema.

Un esquema básico para algoritmos genéticos se presenta a continuación:

```
BEGIN
   Obtener la población inicial al azar.
   WHILE NOT criterio_parada DO
   BEGIN
      Evaluar los individuos de la población.
      Seleccionar padres de la población.
      Producir hijos a partir de los padres seleccionados.
      Mutar los individuos hijos.
      Extender la población añadiendo los hijos.
      Reducir la población extendida.
   END
END
```

El funcionamiento del esquema algorítmico presentado puede agruparse y explicarse en los siguientes pasos:

- Inicialización: Se genera aleatoriamente la población inicial de cromosomas, los cuales representan las posibles soluciones del problema. En caso de no hacerlo aleatoriamente, es importante garantizar que dentro de la población inicial, se tenga la diversidad estructural de estas soluciones para tener una representación de la mayor parte de la población posible o al menos evitar la convergencia prematura.
- Evaluación: A cada uno de los cromosomas de esta población se le aplicará la función de aptitud (o función de *fitness*) para saber cuán "buena" es la solución que representan. Los cromosomas más aptos serán los elegidos para reproducirse y ser los padres de la nueva generación de individuos.
- Criterio de parada: El AG se deberá detener cuando se alcance la solución óptima, pero ésta generalmente se desconoce, por lo que se deben utilizar otros criterios de detención. Normalmente se usan dos criterios: correr el AG un número máximo de iteraciones (generaciones) o detenerlo cuando no haya cambios en la población. Mientras no se cumpla esta condición, se dan los siguientes pasos:
    - Selección: Después de saber la aptitud de cada cromosoma se procede a elegir los cromosomas que serán cruzados en la siguiente generación. Los cromosomas con mejor aptitud tienen mayor probabilidad de ser seleccionados de acuerdo con criterios previamente seleccionados como puede ser elitista, proporcional, ruleta, torneo, etc.
    - Cruzamiento: El cruzamiento (o *crossover*) es el principal operador genético, representa la reproducción sexual, opera sobre dos cromosomas a la vez para generar dos descendientes donde se combinan las características de ambos cromosomas padres.
    - Mutación: Modifica al azar parte del cromosoma de los individuos, y permite alcanzar zonas del espacio de búsqueda que no estaban cubiertas por los individuos de la población actual.
    - Reemplazo generacional: Una vez aplicados los operadores genéticos, se seleccionan los mejores individuos de la población resultante, en base a diferentes criterios, para conformar la población de la generación siguiente.

Para una formalización matemática adecuada de los algoritmos genéticos así como una explicación de los distintos operadores que utilizan (mutación, selección, reproducción,...) ver (Reeves, 1993).

2.3.2 Colonias de hormigas

Las colonias de hormigas, al igual que otros tipos de sistemas biológicos, son sistemas que presentan una organización social enormemente estructurada y en las que existen mecanismos de coordinación que permiten a las mismas realizar conjuntamente tareas que serían difícilmente abordables de manera individual. En muchas especies de hormigas casi todos sus mecanismos de comunicación se basan en el intercambio de sustancias químicas conocidas como feromonas. En particular, en la vida social de las hormigas son de especial importancia las feromonas de rastreo (*trail pheromones*) que utilizan para marcar de manera química caminos en el terreno que les conduzcan desde el hormiguero hasta los lugares donde han encontrado alimentos. Si una hormiga no encuentra ningún rastro de feromona se mueve de forma aleatoria, pero si detecta dicha sustancia tiende a seguir con mayor probabilidad el rastro. Existen numerosos experimentos (Pasteels, 1987) (Goss, 1989) que han demostrado que las hormigas escogen aquellos caminos que tienen más concentración de feromonas y que en la práctica cuando varios caminos se bifurcan las hormigas eligen el camino por el que continuar su marcha basándose en la intensidad de feromonas. De esta forma, como las hormigas depositan feromonas en el camino que han elegido, este mecanismo supone un proceso de refuerzo que concluye con la formación de caminos intensamente marcados. Este proceso tiene una razón de ser dado que como han demostrado los experimentos del puente doble de Deneubourg (Goss, 1989) el mecanismo de autorefuerzo conduce a la selección de aquellos caminos más cortos.

Basándose en este comportamiento natural de las hormigas Dorigo y sus colaboradores (Dorigo, 1992) (Dorigo, 1996) (Dorigo, 1999) introducen por primera vez la optimización basada en colonias de hormigas (OCH) para resolver problemas de optimización combinatoria. Para entender de una forma sencilla cómo trabajan los algoritmos de OCH hemos de ver el proceso de resolución como un ejercicio constructivo en el que en cada iteración cada hormiga construye una solución recorriendo el grafo subyacente al problema. Este proceso constructivo puede verse como un proceso de decisión en el que cada hormiga debe decidir desde el último nodo del grafo ya visitado a qué nuevo nodo dirigirse. Para poder llevar a cabo esta decisión el algoritmo mantiene dos tipos de información, heurística y de rastros de feromonas respectivamente. La información heurística mide la atracción (o grado de preferencia) de moverse desde un nodo origen a un nodo destino y dicha atracción no se modifica durante la ejecución del algoritmo. En cambio, la información de feromonas artificiales mide la atracción aprendida de moverse entre dos nodos y actúa como una memoria a largo plazo. Esta atracción aprendida imita a las feromonas naturales y por tanto dicha información se modifica a lo largo de la ejecución del algoritmo.

El sistema de colonias de hormigas es una evolución de algoritmos OCH previos que presenta 3 diferencias significativas (Liang, 2003):

- se modifica la regla de selección probabilista en la que se hace uso de un valor aleatorio de forma que el siguiente nodo a visitar por la hormiga se elige según si ese valor es mayor o menor que una constante; en caso de ser menor o igual se explota la información disponible tanto de naturaleza heurística como de feromonas tomando como transición aquella que maximiza la atracción. En caso contrario, se aplica una exploración aleatoria.
- únicamente la hormiga reina actualiza los rastros de feromonas cuando todas las hormigas han obtenido una solución. A esta actualización se le denomina *offline* y la solución que se tiene en cuenta para esta única actualización es la mejor solución global. Además, en algunas ocasiones se pueden aplicar previamente técnicas de búsqueda local para mejorar la solución obtenida.
- se modifica la actualización de feromonas por cada hormiga cada vez que se ha seleccionado una arista de manera que aquellas aristas visitadas por muchas hormigas tendrán un valor de feromona menor de manera que su atracción para otras hormigas será cada vez menor y por lo tanto se facilitará que no todas las hormigas sigan el mismo camino. A este mecanismo de actualización se le denomina actualización *online*.

Al igual que los algoritmos genéticos y otras metaheurísticas, es necesario que los datos del problema se puedan adaptar a la forma de representación del espacio de búsqueda utilizado por las colonias de hormigas. En este sentido, dado que una de las aplicaciones naturales de los sistemas OCH es la resolución de caminos y que, como hemos comentado, el proceso de *problem solving* puede entenderse como la búsqueda del camino que conecta el estado inicial con el objetivo, la representación del espacio de búsqueda para problemas de *decision making* en colonias de hormigas es directa. También es destacable la capacidad de adaptación de estos sistemas a cambios en el entorno ya que, por ejemplo, ante un cambio como que una decisión en una determinada situación ya no produce el estado emocional esperado, el proceso de actualización de feromonas se encargaría de reflejar ese cambio para todas las hormigas. Además, el hecho de que la solución encontrada sea básicamente un camino, en otras palabras, una secuencia, permite de manera natural obtener predicciones teniendo en cuenta situaciones probables en un futuro cercano.

2.4 Conclusiones

En el punto 2.1.3 se han presentado diversos mecanismos que se están utilizando, o que podrían ser utilizados para crear sistemas de toma de decisiones influenciados por emociones. Como hemos podido comprobar, los sistemas multiagente son frecuentemente utilizados para dotar de comportamiento emocional a personajes o entidades virtuales. Desafortunadamente, y a pesar de la variedad de propuestas de arquitecturas de agentes emocionales, estos sistemas serán tan buenos generando comportamiento emocional como lo sea la técnica, algoritmo o método empleado realmente para seleccionar las acciones a realizar. Por ejemplo, los sistemas de reglas no facilitan la adaptación del sistema a cambios en las características entorno.

Los métodos más tradicionales de *decision making* que podemos encontrar descritos en (Fülöp, 1998), (Pomerol, 1997), (Tyrrell, 1993) no los tenemos en cuenta puesto que: muchos de ellos son específicos a determinados dominios, otros no cuentan siquiera con un modelo computacional (no pasan de ser simples especificaciones de ideas o teorías), y en la mayoría sería un verdadero problema intentar modificarlos para que contemplasen la influencia de estados emocionales.

Otros mecanismos como el descrito por (Ahn, 2005), basado ampliamente en probabilidades y cálculos estadísticos, presentan los inconvenientes de no utilizar el conocimiento adquirido con el *reinforcement learning* para mejorar la recompensa obtenida a medio plazo, o de elegir de forma inadecuada la mejor decisión con el único objetivo de reducir el tiempo de decisión. Este último inconveniente se podría abordar, por ejemplo, paralelizando el sistema pero el propio diseño de esta propuesta (y de muchas otras) hace casi inviable o poco provechoso el intento.

Respecto a las secuencias contextuales, en el punto 2.2.1 hemos realizado un análisis de las aproximaciones existentes para el aprendizaje de secuencias contextuales que indica que existen una gran variedad de posibles soluciones algorítmicas al problema desde un punto de vista teórico. Uno de los primeros aspectos a tener en cuenta es que existe una gran variedad de tareas relacionadas con el problema del aprendizaje de secuencias como son: la clasificación de las mismas, la predicción de series temporales, la estimación de distribuciones probabilistas sobre dominios de secuencias y la transducción de secuencias. Dada la complejidad y variedad de problemas en el ámbito del aprendizaje de secuencias sería ingenuo pensar que existe una única aproximación que domina el campo de investigación. Según la naturaleza de los problemas a resolver se han estudiado diferentes aproximaciones que varían

desde las redes neuronales recurrentes, los modelos de Markov ocultos, el aprendizaje con refuerzo, la computación evolutiva, los sistemas basados en reglas y los sistemas difusos, entre otros. Aunque recientemente parece haber un sentimiento en la comunidad investigadora sobre la necesidad de aproximaciones híbridas para resolver problemas reales, no existe por desgracia una base sistemática que describa de forma rigurosa como proceder para combinar las distintas aproximaciones existentes.

En el ámbito de las redes neuronales, si bien existen dos tipos de aproximaciones según el tiempo esté representado de forma explícita o implícita, lo que da lugar a redes unidireccionales (*feedforward*) o recurrentes respectivamente, existen diversos problemas que dificultan su aplicación. Por un lado, el entrenamiento de una red neuronal no se puede realizar aplicando el algoritmo de aprendizaje sobre cualquier configuración de la arquitectura de la red y con cualquier tipo de parámetros de aprendizaje. Decisiones como el tamaño de las variables de estado, el retardo de las líneas para capturar las dependencias temporales de forma correcta, el grado de conectividad y la elección de las heurísticas adecuadas para completar la información incompleta son críticas en el correcto entrenamiento de la red. Además, se ha demostrado que el problema de la carga, determinar un conjunto de pesos consistente con los datos de entrenamiento, no es resoluble. Por otro lado las redes recurrentes, aunque tengan la capacidad de almacenar comportamientos complejos no lineales, presentan el problema de las dependencias de largo plazo por el que los algoritmos de gradiente descendente comúnmente utilizados son incapaces de almacenar información de errores referidos a entradas que están alejadas en el tiempo de las entradas presentes. Por otro lado, no hemos de olvidar que, desde un punto de vista computacional, algunas redes, como las recurrentes de un único nivel, no son capaces de representar todos los autómatas finitos de estados por lo que tienen una capacidad computacional limitada. Tampoco hay que olvidar que la forma de procesamiento de una red neuronal es difícilmente distribuible dado que la activación de un determinado nivel depende de los niveles anteriores y por tanto es necesario un procesamiento secuencial de la transmisión de la señal que exigiría frecuentes mecanismos de sincronización Si existen, sin embargo, versiones paralelas mediante multiprocesadores con memoria compartida, pero dichos elementos hardware tienen todavía en la actualidad un alto coste y no se pueden obtener de proveedores no especializados.

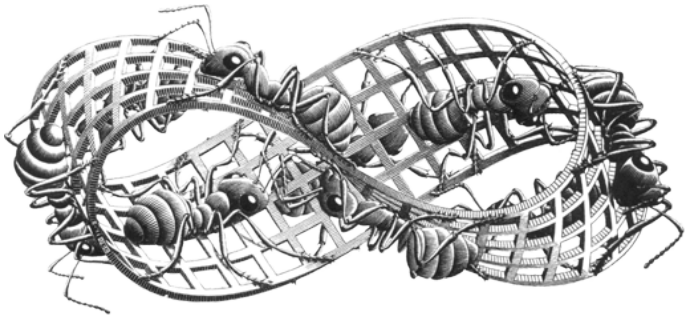
En el ámbito de los modelos de Markov ocultos y en general de las aproximaciones bayesianas, se han producido interesantes avances para abordar el problema de la clusterización. Las aproximaciones más exitosas son las que obtienen un modelo de Markov a partir de una secuencia y a partir de dicho modelo. Considerando cada serie temporal como generada por una

variable estocástica se construye una matriz de probabilidad para cada serie para representar las probabilidades de transición entre estados desde el instante de tiempo actual al instante siguiente. Puesto que cada serie temporal produce una matriz de transiciones es posible aplicar mecanismos de *clustering* bayesianos para obtener los cúmulos que mejor representan a cada una de las series presentadas. Estos modelos sirven para clasificar futuras series e incluso para estimar futuros estados de series que se proporcionan de forma incompleta. Sin embargo, el problema de estas aproximaciones radica en la generalización de los mecanismos propuestos al caso de problemas multivariable por la complejidad computacional asociada al cálculo de las matrices de transición para dichos casos. Sólo en el caso de existir independencia probabilista entre dichas variables podemos obtener modelos computacionalmente abordables. Por otro lado, en el caso de los modelos de Markov ocultos, la identificación de las variables ocultas puede imponer serias limitaciones en la estructura del modelo de regresión (Settimi, 1998). Además del problema de la obtención del modelo en estas aproximaciones, hemos de considerar también la efectividad de las posibles implementaciones paralelas de dichos algoritmos. Si bien existen algoritmos distribuidos que implementan estos modelos de computación éstos se ven penalizados por la gran necesidad de comunicación/sincronización. Estos modelos, al igual que las redes neuronales, pueden ser implementados de forma efectiva en sistemas multiprocesador con memoria compartida.

En último lugar, en el punto 2.3 hemos hablado de los algoritmos genéticos y de las colonias de hormigas como ejemplos de métodos metaheurísticos. Este tipo de métodos son generalistas y pueden ser aplicados a infinidad de dominios, aunque presentan un problema y es que las soluciones que proporcionan no son óptimas ya que, de forma nativa, estos métodos no buscan en todo el espacio del problema, razón por la cual son utilizados para resolver problemas con espacios de búsqueda enormes. De entre estos dos métodos, las colonias de hormigas presentan una mayor adecuación a las características de los dos dominios abordados. En el caso de la toma de decisiones, hemos de tener en cuenta que una de sus aplicaciones más habituales es en la resolución de caminos en grafos, por lo que tanto la representación del espacio del problema como la obtención de la mejor secuencia de decisiones para un futuro cercano son inherentes. Además, su sistema de aprendizaje basado en el depósito y actualización de las feromonas en aquellos caminos (secuencia de decisiones) más beneficiosos hace que sea capaz de adaptarse a cambios en las condiciones de su entorno. En cuanto al ámbito de las secuencias contextuales, algoritmos de refuerzo como los de colonias de hormigas presentan la ventaja de construir el modelo de forma progresiva a medida que se van obteniendo secuencias del fenómeno a analizar.

La propia estructura en forma de grafo de representación del problema en dichos algoritmos permite representar de forma natural las secuencias temporales. Además, al tratarse de algoritmos de refuerzo, y utilizando métricas de similitud entre secuencias, es posible reforzar mediante algún tipo de medida escalar (feromonas en el caso de las colonias de hormigas) aquellos caminos (secuencias) que se presentan con mayor frecuencia. Dado que los nodos pueden representar situaciones contextuales de cualquier complejidad en cuanto al número de variables presentes en el problema, es posible transformar los problemas de clusterización de secuencias (totales o parciales) y de estimación de secuencias futuras como problemas de optimización sobre dichos espacios anotados mediante mecanismos de refuerzo. Además, es conveniente destacar que la propia naturaleza de los algoritmos de colonias de hormigas permite el diseño de forma natural de algoritmos de naturaleza no secuencial, ya que hay muchos agentes artificiales (hormigas) obteniendo soluciones de forma colectiva, y en la literatura ya existen implementaciones efectivas, tanto de carácter distribuido como paralelo (véase (Mocholi, 2005) y (Catala, 2007) respectivamente), para dichos algoritmos.

Por lo tanto, vistas las ventajas que presenta respecto a la capacidad de tratar problemas de talla grande y obtener buenas soluciones en un tiempo razonable, y las facilidades que aporta para realizar algunas modificaciones y mejoras, decidimos utilizar las colonias de hormigas como punto de partida para la creación de una estrategia capaz de abordar problemas de ambos dominios. Los retos a afrontar estriban en poder definir mecanismos abstractos de computación basados en algoritmos OCH que permitan obtener soluciones para problemas cuyo modelo (variables involucradas, métricas de similitud y funciones a optimizar) pueda ser proporcionado de forma específica para cada problema concreto a resolver sin tener que recodificar la estrategia algorítmica.



"Moebius Strip II", por Maurits Cornelis Escher (1963)

Capítulo 3

OCH: variantes y aplicaciones

3.1 Introducción

La idea del funcionamiento de la resolución de problemas de optimización mediante el uso de colonias de hormigas fue fruto de la observación del comportamiento natural de ciertas especies de insectos en general, y de las hormigas en particular. Las primeras investigaciones sobre el comportamiento de insectos que viven en sociedades concluían que los insectos reaccionaban ante ciertas señales desencadenando una reacción codificada en sus genes (Grasse, 1944). Se descubrió que los efectos de esas reacciones podían actuar como nuevas señales (Grasse, 1959), pudiendo desencadenar nuevas reacciones tanto en el espécimen que las produjo como en otros miembros de la colonia. Esta forma tan particular de comunicación indirecta recibió el nombre de *stigmergy* y se diferencia de otras formas de comunicación por dos aspectos principalmente: la naturaleza física de la información producida por los insectos que intervienen como emisores en la comunicación; y la restricción espacial de la información producida, ya que esta información sólo está disponible si se visita el mismo lugar (o en sus proximidades) en el que fue emitida.

La *stigmergy* también ha sido observada en colonias de hormigas permitiéndoles, al igual que a otras especies de insectos sociales, la colaboración entre individuos, siendo de esta manera capaces de mostrar comportamientos

complejos y realizar tareas difíciles, teniendo en cuenta las capacidades de una única hormiga.

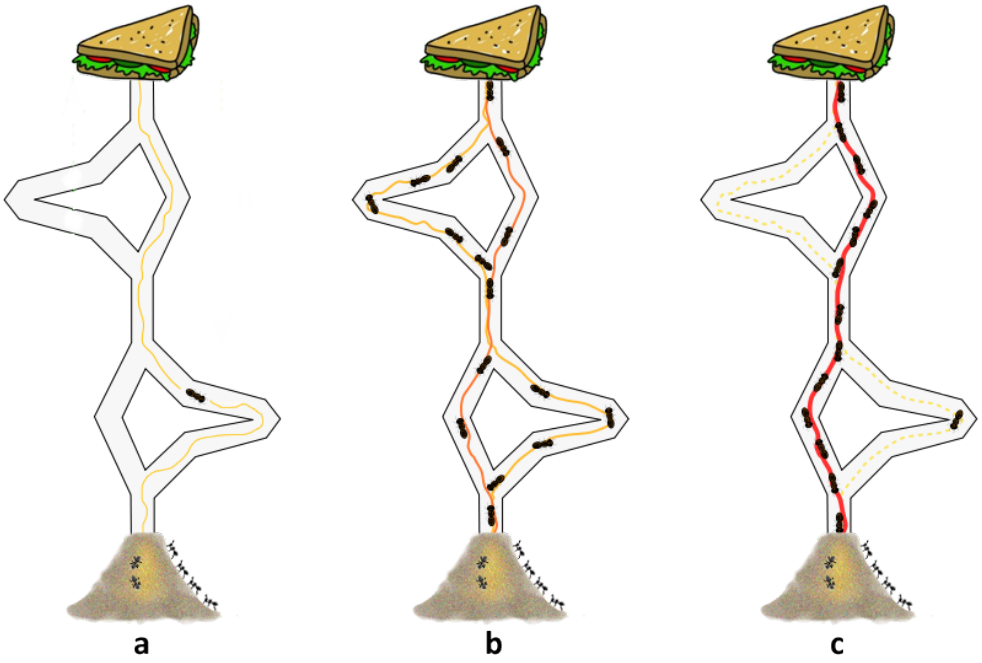


Figura 2. Desde un estado inicial (a) se observa que las hormigas marcan con feromonas la ruta que siguen hasta la comida. Cuantas más hormigas utilizan una cierta ruta, más feromonas depositan en ella (b), llegando a un punto en que el camino más corto es el más utilizado por ser el más marcado con feromonas (c).

Un aspecto interesante del comportamiento de muchas especies de hormigas es su habilidad para encontrar los caminos más cortos entre su hormiguero y las fuentes de alimento. Mientras se mueven entre el hormiguero y la fuente de alimento, algunas especies de hormigas depositan feromonas de rastreo, una sustancia química que pueden detectar y que utilizan para marcar de manera química aquellos caminos de su entorno que conducen desde su hormiguero hasta las fuentes de alimento. Si no se encuentra ningún rastro de feromona, las hormigas se mueven de manera básicamente aleatoria, como si explorasen el entorno, pero cuando detectan la presencia de feromona, tienden a seguir el rastro con mayor probabilidad. Trabajos como los presentados en (Goss, 1989) (Pasteels, 1987) han demostrado que las hormigas prefieren de manera probabilística los caminos marcados con una concentración superior de feromona de rastreo. Cuando deben elegir el camino a elegir de entre varios posibles, las hormigas toman una decisión basándose en la cantidad de feromona depositada en cada camino, esto es, el camino con mayor cantidad depositada de feromona de rastreo es el elegido. Puesto que las hormigas

depositan feromona en el camino que siguen, este comportamiento lleva a un proceso de autorefuerzo que concluye con la creación de caminos marcados con una concentración elevada de feromona de rastreo. A continuación explicamos este mecanismo de descubrimiento del camino más corto haciendo uso para ello de la Figura 2, en la que se representa una variante del puente doble de Deneubourg (Goss, 1989).

El puente doble de Deneubourg consiste en dos puentes que conectan un hormiguero y una fuente de comida. En la configuración representada en la Figura 2 se han utilizado dos piezas idénticas de puente doble en las que los puentes tenían diferente longitud y las piezas estaban rotadas 180 grados una respecto de la otra. Ante este entorno, y partiendo de un estado en el que no hay ningún rastro de feromona en los caminos, los investigadores observaron que las hormigas seguían un comportamiento aleatorio cuando debían elegir entre los dos caminos posibles (situación a en la figura). A medida que más y más hormigas cruzan los puentes la cantidad de feromona depositada en los caminos aumenta, aunque no por igual: los caminos más cortos reciben una cantidad mayor (situación b). La razón detrás de este hecho es la siguiente: las hormigas que eligen el camino más corto tardan menos en alcanzar la fuente de alimento y volver al hormiguero, por lo que los caminos más cortos reciben más feromonas por unidad de tiempo. Una vez que la diferencia en la cantidad de feromona depositada en los diferentes caminos se ha vuelto significativa, las hormigas tienden a seguir el camino con mayor cantidad de feromonas con una mayor probabilidad, lo cual provoca que se deposite más feromona en ese camino. Este hecho, junto con la evaporación natural de la feromona depositada, conlleva que los caminos más largos vayan siendo menos transitados y que el camino más corto entre el hormiguero y la fuente de alimento sea el más utilizado por todas las hormigas (situación c en la Figura 2)

A partir de estos resultados, se desarrolló el siguiente modelo (Goss, 1989) para el comportamiento que había sido observado en los experimentos. Transcurrido un tiempo desde el inicio del experimento, y habiendo elegido seguir m_1 hormigas el primero de los puentes y m_2 el segundo, la probabilidad de que la siguiente hormiga elija el primero de los puentes puede expresarse como

$$P_1(m+1) = \frac{(m_1 + k)^h}{(m_1 + k)^h + (m_2 + k)^h}$$

donde k y h son parámetros utilizados para ajustar el modelo a los datos experimentales.

Tomando como inspiración este modelo básico del comportamiento de las hormigas cuando buscan alimento, es posible diseñar unas "hormigas

artificiales" (agentes software) capaces de resolver problemas de optimización. Estas hormigas artificiales simulan el depósito de feromonas mediante la modificación de variables numéricas con los niveles de feromona. Estas variables están asociadas a los diferentes estados del problema que las hormigas artificiales van visitando mientras construyen soluciones al problema de optimización. Además, tal y como hemos expuesto al explicar el concepto de *stigmergy*, las hormigas artificiales tan sólo podrán consultar las variables con los niveles de feromona si visitan el estado del problema al que están asociadas.

En el siguiente punto describimos con mayor detalle el proceso que sigue una hormiga artificial para construir una solución para un problema de optimización.

3.2 La metaheurística OCH

A continuación describimos de manera general las características y el modo de funcionamiento tanto de las hormigas artificiales, que realizan la búsqueda de caminos, como de los algoritmos OCH, que gestionan la información que comparten las hormigas y la soluciones que éstas van construyendo para facilitar la convergencia a la mejor solución.

3.2.1.1 Hormigas Artificiales

Las hormigas artificiales son la herramienta con la que OCH es capaz de encontrar soluciones a los problemas a los que se enfrenta. Los problemas que estos agentes software son capaces de tratar pertenecen a la categoría de problemas de camino mínimo, que están caracterizados por los siguientes aspectos, tal y como se describe en (Dorigo, 2003):

- Un conjunto de restricciones, posiblemente dependientes del tiempo $\Omega(t)$.
- Un conjunto finito de componentes $C = \{c_1, c_2, \dots, c_{N_c}\}$.
- Los estados del problema se definen en términos de secuencias $x = \langle c_1, c_j, \dots, c_k, \dots \rangle$ de longitud finita sobre los elementos de C. El conjunto de todos los posibles estados se denota como χ.
- Un conjunto de soluciones candidatas $S \subseteq \chi$.
- Un conjunto de estados posibles $\bar{\chi} \subseteq \chi$, definidos mediante un test que es dependiente del problema y que verifica que no es imposible completar una secuencia $\chi \in \bar{\chi}$ en una solución que satisface las restricciones Ω.
- Un conjunto no vacío de soluciones óptimas $S^* \subseteq \bar{\chi}$ y $S^* \subseteq S$.

- Una función de coste $g(s,t)$ asociada a cada solución candidata $s \in S$.
- Una función de coste, o estimación del coste, $J(x,t)$ asociado a los estados que no son soluciones candidatas.

Teniendo en cuenta estas características, podemos decir que una hormiga artificial construye soluciones para los problemas que aborda realizando caminos de forma probabilista en el grafo $G_C = (V, A)$ (grafo de construcción), donde los vértices de V son las componentes de C y las aristas de A conectan componentes de C. Tanto los nodos como las aristas pueden tener asociados dos tipos de información: un valor heurístico de atracción (η_i, η_{ij}, respectivamente) que evalúa el grado de preferencia de moverse de un nodo origen a un nodo destino, y un nivel de feromona (τ_i,τ_{ij}, respectivamente) que indica la atracción aprendida de moverse entre dos nodos y actúa como una memoria a largo plazo. Al igual que en el comportamiento natural de las hormigas, la información sobre el nivel de feromonas depositadas varía durante la ejecución del algoritmo, a diferencia de la información heurística que permanece inalterada.

Una definición más formal del modelo de hormiga artificial es la siguiente: podemos ver una cierta hormiga artificial *k* como un proceso estocástico constructivo que presenta las siguientes propiedades (Dorigo, 2003):

- Hace uso del grafo de construcción $G_C = (V, A)$ en la búsqueda de soluciones óptimas $s^* \in S^*$.
- Tiene una memoria $\mathfrak{M}^k$ para almacenar información sobre el camino seguido hasta un instante dado. Dicha memoria puede ser utilizada para:
    - Construir soluciones factibles.
    - Calcular valores heurísticos η_i, η_{ij}.
    - Evaluar la solución encontrada.
    - Recorrer el camino seguido hacia atrás.
- Tiene un estado de partida x_s^k y una o más condiciones de terminación e^k. Normalmente el estado inicial se expresa como una secuencia vacía o como una secuencia con una única componente.
- Estando en un estado $x_r = \langle x_{r-1}, i \rangle$ y si no se satisface ninguna condición de terminación, se mueve a un nodo j en su vecindad $\mathfrak{A}^k(x_r)$, esto es, al estado $\langle x_r, j \rangle \in \chi$. Si al menos alguna de las condiciones de terminación es cierta entonces la hormiga se detiene.

- Selecciona un movimiento aplicando una regla de decisión probabilística. Dicha regla es función de:
    - Los valores locales heurísticos η_i, η_{ij} y de feromona τ_i, τ_{ij}.
    - La memoria privada de la hormiga en la que se almacena el estado actual.
    - Las restricciones del problema.
- Cuando se añade una componente c_j al estado actual se puede actualizar el nivel de feromona asociado a dicha componente o a la arista seleccionada. Este proceso se conoce como *online step-by-step pheromone trails update* (actualización en línea paso a paso de los rastros de feromonas).
- Una vez se ha construido una solución puede volver hacia atrás el camino realizado y actualizar los niveles de feromonas de las componentes seleccionadas. A este proceso se le da el nombre de *online delayed pheromone trails update* (actualización en línea diferida de los rastros de feromona).

Es importante señalar que las hormigas actúan de forma concurrente e independiente y que, aunque cada hormiga puede encontrar una solución (probablemente pobre), las soluciones de mayor calidad emergen a partir de la interacción colectiva de varias hormigas, y esto es cierto tanto para las hormigas naturales como para las artificiales. Esta interacción colectiva se materializa, en el caso de las hormigas artificiales, mediante la comunicación indirecta que se da a través de las variables que almacenan niveles de feromonas y que las hormigas leen/escriben. Es pues, una forma de aprendizaje colectivo en el que cada hormiga modifica de forma adaptativa la manera en la que el resto de hormigas perciben el problema a resolver.

3.2.1.2 Funcionamiento de un algoritmo OCH genérico

Aparte de las tareas realizadas por las hormigas, un algoritmo OCH suele conllevar la ejecución de otros dos procedimientos: la evaporación de feromonas de rastreo y las *daemon actions*. La evaporación de feromonas es el proceso a través del cual la cantidad depositada de feromonas de rastreo va disminuyendo con el tiempo. La realización de este procedimiento es necesaria ya que ayuda a evitar una convergencia demasiado rápida del algoritmo a una solución subóptima. En otras palabras, se trata de una manera muy conveniente de "olvidar", lo cual favorece que se exploren áreas diferentes del espacio de búsqueda del problema.

Las tareas que se ejecutan en las *daemon actions* son completamente artificiales, es decir, no tienen versión en el mundo natural, y suelen ser acciones centralizadas y globales que no pueden ser realizadas por una hormiga artificial.

Un par de ejemplos posibles de esas tareas serían: aplicar un procedimiento de búsqueda local a las soluciones generadas por las hormigas antes de actualizar los rastros de feromona, o observar la calidad de todas las soluciones construidas por las hormigas y depositar una cantidad de feromona adicional sólo en las transiciones/componentes asociadas a algunas soluciones. En los dos ejemplos reseñados, las *daemon actions* reemplazarían la actualización en línea diferida de los rastros de feromona y el proceso pasaría a llamarse *offline pheromone trails updates* (actualización diferida de rastros de feromonas).

A continuación se describe un algoritmo de OCH genérico en pseudocódigo (Dorigo, 1999):

```
Procedimiento Metaheurística_OCH()
   Inicialización_de_parámetros
   Mientras (criterio_de_terminación_no_satisfecho)
      Programación_de_actividades
         Generación_de_hormigas_y_actividad()
         Evaporación_de_Feromona()
         Daemon_Actions()
      Fin Programación_de_actividades
   Fin mientras
Fin Procedimiento

Procedimiento Generación_de_hormigas_y_actividad()
   Repetir en paralelo desde k=1 hasta m //número_hormigas
      Nueva_hormiga(k)
   Fin repetir en paralelo
Fin Procedimiento

Procedimiento Nueva_hormiga(id_hormiga)
   Inicializa_hormiga(id_hormiga)
   L = Actualiza_memoria_hormiga()
   Mientras (estado_actual ≠ estado_objetivo)
      P = Calcular_probabilidades_de_transición(F, L, Ω)
      siguiente_estado = Aplicar_política_decisión(P, Ω)
      Mover_al_siguiente_estado(siguiente_estado)
```

```
        Si (actualizacion_feromona_en_linea_paso_a_paso)
            Depositar_feromona_en_el_arco_vistado()
        Fin si
        L = Actualizar_estado_interno()
    Fin mientras
    Si (actualizacion_feromona_en_linea_a_posteriori)
        Para cada arco visitado
            Depositar_feromona_en_el_arco_visitado()
        Fin para
    Fin si
    Liberar_recursos_hormiga(id_hormiga)
Fin Procedimiento
```

El primer paso implica la inicialización del los parámetros que se utilizan en el algoritmo. Se deben establecer, entre otros, el rastro inicial de feromona asociado a cada transición, τ_0 (que es un valor positivo pequeño), el número de hormigas en la colonia, *m*, y los pesos que definen la proporción en la que afectarán la información memorística y heurística en la regla de transición probabilística, habitualmente denotados por α y β, respectivamente.

Acto seguido, y mediante la estructura de control representada por Programación_de_actividades, se gestiona la ejecución de 3 componentes: la creación y puesta en marcha de las hormigas, la evaporación de las feromonas de rastreo depositadas, y las *daemon actions*. Cabe destacar que mediante la citada estructura de control no se especifica cómo se ejecutarán las componentes, es decir, es decisión de los diseñadores del algoritmo elegir si las componentes se ejecutarán en paralelo, de forma independiente, o si utilizarán algún mecanismo de sincronización.

Algunas partes del algoritmo descrito son opcionales, como las *daemon actions*, y otras dependen completamente de la implementación, como por ejemplo cómo y cuándo se depositan las feromonas de rastreo. Generalmente las actualizaciones en línea paso a paso y diferida son mutuamente excluyentes, y no es habitual que falten las dos; en ese caso, las *daemon actions* llevarían a cabo el depósito de feromonas (actualización fuera de línea de rastros de feromonas).

Como conclusión de la explicación del algoritmo OCH genérico incidir en que los métodos Calcular_probabilidades_de_transición y Aplicar_política_decisión hacen uso de: los niveles de feromona detectables (F) en el nodo actual, el

estado actual de esa hormiga artificial (L), y las restricciones marcadas en el problema (Ω).

3.3 Algoritmos OCH más relevantes

Una vez descrita de forma genérica la metaheurística OCH hay que destacar que existen en la literatura diversos algoritmos que hacen uso de ella para problemas de naturaleza combinatoria. A continuación describimos de forma breve el Sistema de Hormigas (Dorigo, 1996) por tener un interés de naturaleza histórica, el Sistema de Hormigas MAX-MIN (Stützle, 2000), el Sistema de Hormigas Rank-based (Bullnheimer, 1999a) y el Sistema de Colonia de Hormigas (Dorigo, 1997a), el cual tomaremos como referencia en nuestro trabajo. Además de los indicados, existen otros algoritmos similares, como el Sistema de la Mejor-Peor Hormiga (Cordón, 2000), que alcanzan resultados satisfactorios.

El Sistema de Hormigas

El Sistema de Hormigas (SH) fue el primer algoritmo de OCH propuesto por Dorigo y sus colaboradores en 1991. Los autores proponen tres variantes denominadas SH-ciclo, SH-cantidad y SH-densidad. En el SH-ciclo las hormigas depositan feromonas una vez han completado una solución (*offline update*). En cambio, SH-cantidad y SH-densidad realizan el depósito de feromona a cada paso de construcción de una solución (*online update*). En SH-densidad la cantidad de feromona depositada es constante mientras que en SH-cantidad depende de la atracción heurística η_{ij} de la transición seleccionada.

Los estudios empíricos han demostrado que SH-ciclo es superior al resto y por esta razón y dado que su mecanismo de actualización ha servido de referencia para posteriores algoritmos lo detallaremos a continuación.

Como hemos comentado la feromona se deposita una vez que todas las hormigas han completado una solución. En primer lugar los rastros de feromona de cada transición se evaporan mediante una tasa de evaporación de forma que los rastros se reducen en un factor constante.

$$\tau_{ij}^{new} \leftarrow (1-\rho)\cdot\tau_{ij}^{old}: \rho \in (0,1]$$

Una vez evaporadas las feromonas cada hormiga k recorre su memoria local $\mathfrak{M}^k$ y deposita una cantidad de feromona $\Delta\tau_{ij}^k = f\big(C(S_k)\big)$ en cada arco transitado del grafo de construcción donde la cantidad de feromona depositada depende de la calidad $C(S_k)$ de la solución S_k obtenida por la hormiga k.

$$\tau_{ij}^{new} \leftarrow \tau_{ij}^{old} + \Delta\tau_{ij}^k, \forall a_{ij} \in S_k$$

Por último, el proceso de construcción de la solución sigue el mecanismo descrito en el modelo de hormiga artificial donde la regla de decisión probabilista de dicho modelo se formula como sigue:

$$p_{ij}^{k} = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{u \in \mathfrak{A}_i^k} \cdot [\tau_{iu}]^{\alpha} \cdot [\eta_{iu}]^{\beta}}, & \text{si } j \in \mathfrak{A}_i^k \\ 0, & \text{en otro caso} \end{cases}$$

donde $\mathfrak{A}_i^k$ es el vecindario alcanzable $\mathfrak{A}^k(x_r)$ estando en el estado $\langle x_r^k, i\rangle \in \chi$, η_{ij} representa la información heurística local disponible, y $\alpha,\beta \in \mathfrak{N}$ son dos factores que ponderan la importancia de la información heurística y de los rastros de feromona. Si $\alpha = 0$, aquellos nodos con mejor atracción heurística son los seleccionados y nos encontraríamos ante un algoritmo probabilista de naturaleza voraz. En cambio, si $\beta = 0$, sólo las feromonas dirigen el proceso de construcción de soluciones, de manera que se llega a un rápido estancamiento en el que las hormigas construyen siempre las mismas soluciones que suelen ser óptimos locales. El valor para la heurística local se suele definir como $\eta_{ij} = \frac{s_j}{c_{ij}}$, lo que significa que los nodos con un mejor ratio coste-beneficio resultan más atractivos desde el punto de vista heurístico.

Es importante resaltar que los autores de este trabajo propusieron una mejora final a este procedimiento denominada SH-elitista en la que una vez realizado el proceso anterior la hormiga reina[4] deposita un nivel de feromona adicional en aquellas aristas que pertenecen a la mejor solución encontrada hasta el momento en el proceso de búsqueda. Este incremento de feromona se produce en un factor *E* que indica el número de hormigas elitistas que se consideran:

$$\tau_{\mathrm{ij}}^{\mathrm{new}} \leftarrow \tau_{\mathrm{ij}}^{\mathrm{old}} + E \cdot f\left(C\left(S_{mejor}\right)\right), \forall a_{\mathrm{ij}} \in \mathrm{S}_{mejor}$$

El Sistema de Hormigas MAX-MIN

El Sistema de Hormigas MAX-MIN (SHMM) (Stützle, 2000) fue desarrollado como una mejora de la idea original del SH. Los cambios más importantes son:

[4] La hormiga reina artificial es un tipo especial de hormiga que, como su equivalente natural, no realiza ninguna de las tareas que sí realizan sus congéneres. En la mayor parte de algoritmos OCH propuestos en la literatura, las hormigas reinas se encargan de evaluar las soluciones obtenidas, compararlas, elegir aquélla que se considere mejor según los criterios que se hayan establecido, en algunos casos aplicar métodos de búsqueda local para mejorar la solución elegida, y realizar la actualización *offline* de feromonas de rastreo.

- Sólo la mejor hormiga puede actualizar los rastros de feromonas.
- El valor mínimo y máximo de feromona depositada está limitado.

La evaporación y actualización de los niveles de feromona queda ahora expresada con la siguiente fórmula:

$$\tau_{ij}^{new} \leftarrow (1-\rho)\cdot\tau_{ij}^{old} + \Delta\tau_{ij}^{best}$$

donde $\Delta\tau_{ij}^{best}$ representa el valor que se usará para actualizar el nivel de feromona, y se define de la siguiente manera:

$$\Delta\tau_{ij}^{best} = \begin{cases} \frac{1}{L_{best}} & \text{si la mejor hormiga visitó la arista } (i,j) \text{ en su recorrido} \\ 0 & \text{en otro caso} \end{cases}$$

L_{best} es la longitud del tour de la mejor hormiga, aunque es decisión de quién diseñe un algoritmo basado en este sistema escoger como la mejor hormiga bien a aquella que ha devuelto la mejor solución en la última iteración ($L_{iterbest}$), bien a la que ha encontrado la mejor solución hasta el momento ($L_{bestSol}$), o bien una combinación de ambas.

Una vez actualizados los niveles de feromonas, el proceso de actualización de las mismas finaliza con la comprobación de que todos los niveles de feromonas se encuentran en el rango $[\tau_{min}, \tau_{max}]$, los valores mínimo y máximo permitidos respectivamente.

$$\tau_{ij} = \begin{cases} \tau_{min} & \text{if } \tau_{ij} < \tau_{min} \\ \tau_{ij} & \text{if } \tau_{min} \leq \tau_{ij} \leq \tau_{max} \\ \tau_{max} & \text{if } \tau_{ij} > \tau_{max} \end{cases}$$

En cuanto a cómo establecer los valores de feromona mínimo y máximo permitidos, los autores del SHMM recomiendan que se escojan según el problema a resolver y, aunque en (Stützle, 2000) muestran cómo obtenerlos analíticamente, también indican que pueden obtenerse experimentalmente.

El Sistema de Hormigas Rank-based

El Sistema de Hormigas Rank-based (Bullnheimer, 1999a) es una variante del SH que se caracteriza por ordenar las hormigas, al terminar cada iteración, según la adecuación de la solución que han devuelto a la función a optimizar. Es decir, si lo que buscamos es el camino más corto entre una serie de puntos, este sistema ordenaría las hormigas según la longitud de los caminos que han devuelto en la última iteración llevada a cabo.

La actualización de feromonas se realiza de la siguiente manera: tan sólo las $(w-1)$ mejores soluciones de esta iteración y la mejor solución hasta el

momento reciben feromonas. El peso con el que cada hormiga contribuye a la actualización de los niveles de feromonas viene dado por $max\{0, w-r\}$, donde r representa el orden que ocupa una hormiga, mientras que la mejor solución contribuye con un peso de w. De esta manera, la fórmula de actualización de feromonas queda como sigue:

$$\tau_{\mathrm{ij}}^{\mathrm{new}} \leftarrow (1-\rho)\cdot\tau_{\mathrm{ij}}^{\mathrm{old}} + \sum_{r=1}^{\mathrm{w}-1}(w-r)\cdot\Delta\tau_{ij}^{r} + w\cdot\Delta\tau_{\mathrm{ij}}^{best}$$

donde $\Delta\tau_{\mathrm{ij}}^{\mathrm{r}} = \frac{1}{\mathrm{L}^{\mathrm{r}}}$ y $\Delta\tau_{\mathrm{ij}}^{\mathrm{best}} = \frac{1}{\mathrm{L}^{\mathrm{best}}}$. Los casos de empate se resuelven eligiendo un orden aleatorio para las soluciones con la misma longitud de camino.

El Sistema de Colonias de Hormigas

El Sistema de Colonias de Hormigas (Dorigo, 1997a) es una evolución del SH en el que se proponen diversas modificaciones. En primer lugar se añade una regla de selección probabilista distinta denominada regla proporcional pseudoaleatoria, en la que se hace uso de un valor aleatorio $q_0 \in [0,1]$ de forma que el siguiente nodo a visitar por la hormiga *k* se elige según la siguiente distribución de probabilidad:

Si $q \leq q_0$, entonces

$$p_{ij}^{k} = \begin{cases} 1, & si\ j = \underset{u\in\mathfrak{U}_i^k}{\mathrm{argmax}}\left\{[\tau_{iu}]^{\alpha}\cdot[\eta_{iu}]^{\beta}\right\} \\ 0, & \text{en otro caso} \end{cases}$$

Si no, entonces

$$p_{ij}^{k} = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha}\cdot[\eta_{ij}]^{\beta}}{\sum_{u\in\mathfrak{U}_i^k}[\tau_{iu}]^{\alpha}\cdot[\eta_{iu}]^{\beta}}, & \text{si } j\in\mathfrak{U}_i^k \\ 0, & \text{en otro caso} \end{cases}$$

Como podemos observar, en el primer caso (cuando $q \leq q_0$) se explota la información disponible tanto de naturaleza heurística como de feromonas tomando como transición aquella que maximiza la atracción. Sin embargo en el segundo caso se aplica una exploración aleatoria siguiendo la distribución de probabilidad descrita en el algoritmo SH tradicional.

La segunda modificación de esta versión del algoritmo con respecto al algoritmo original estriba en el hecho de que únicamente la hormiga reina actualiza los rastros de feromonas cuando todas las hormigas han obtenido una solución. A esta actualización se le denomina *offline* y la solución que se tiene en cuenta para esta única actualización es la mejor solución global evaporando previamente los valores de feromonas de todas las aristas que participan en la

mejor solución global. Además, en algunas ocasiones se pueden aplicar técnicas de búsqueda local para mejorar la solución obtenida antes de actualizar las feromonas.

Por último, la tercera variación con respecto al algoritmo SH original deriva de la actualización paso a paso de las feromonas (*online*) realizada por cada hormiga cada vez que se ha seleccionado una arista a_{ij}, y es como sigue:

$$\tau_{ij}^{new} \leftarrow (1-\varphi)\cdot\tau_{ij}^{old} + \varphi\cdot\tau_0 : \varphi \in (0,1]$$

donde φ es un parámetro más de decremento de feromona. La regla anterior aplica simultáneamente una evaporación y una aplicación de τ_0 unidades de feromona donde τ_0 es, como hemos explicado con anterioridad, un valor positivo muy pequeño que sirve de límite inferior para los valores de feromona. Al aplicar esta regla aquellas aristas visitadas por muchas hormigas tendrán un valor de feromona menor de manera que su atracción para otras hormigas será cada vez menor y por lo tanto se facilitará que no todas las hormigas sigan el mismo camino.

En el caso del algoritmo OCH propuesto en (Liang, 2003) se sigue la estrategia planteada por esta aproximación y que, de forma pseudoalgorítmica, podría resumirse como sigue:

```
Inicialización de todos los parámetros
Repetir
   Repetir (para cada hormiga k)
      Construir solución según la regla de decisión
      Aplicar la modificación de feromonas online
   Hasta que todas las hormigas han generado una solución
   Aplicar la búsqueda local  //opcional
   Evaluar soluciones y almacenar la mejor global
   Aplicar la modificación de feromonas offline
Hasta alcanzar el criterio de parada
```

En esta propuesta se permite que las hormigas construyan soluciones no factibles porque, como se sugiere en (Ramalhinho, 1998), puede que éstas se encuentren cercanas a la frontera de las soluciones factibles dado que en estas regiones es probable que se encuentren óptimos globales.

3.4 Dominios de aplicación OCH

La metaheurística OCH ha suscitado un gran interés en la comunidad científica, lo que ha dado lugar al diseño de diversos algoritmos OCH y su aplicación con éxito a un variado grupo de problemas de optimización. Buena parte de esos algoritmos atacan problemas $\mathcal{NP}$-duros y problemas de camino mínimo dinámicos.

Los problemas $\mathcal{NP}$-duros son aquellos para los que los mejores algoritmos conocidos capaces de garantizar que hallarán la solución óptima tienen un coste temporal exponencial, para ese problema y en el peor caso posible. Por esta razón el uso de estos algoritmos se vuelve inviable en muchos casos, mientras que los algoritmos OCH son capaces de encontrar soluciones subóptimas (soluciones de alta calidad pero no la solución óptima) con rapidez. Ante este tipo de problemas los algoritmos OCH suelen añadir a los pasos a realizar algún mecanismo extra, como es el caso de la búsqueda local en el Sistema de Colonias de Hormigas.

En los problemas de camino mínimo dinámico las propiedades del grafo que representa el espacio del problema van cambiando a lo largo del tiempo mientras tiene lugar el proceso de optimización, el cual debe adaptarse a los cambios que van produciéndose. Un ejemplo de este tipo de problema lo constituyen las redes de telecomunicaciones. La topografía del grafo no suele cambiar, sin embargo, sí pueden variar con el tiempo propiedades como el coste de los componentes o el de las conexiones. Ante esa situación, en (Dorigo, 2003) exponen que la adecuación de utilizar algoritmos OCH es proporcional al incremento de la frecuencia de cambio de los costes y al aumento del desconocimiento sobre el mecanismo detrás de esos cambios.

Aparte de esos dos tipos de problemas, la utilidad de la metaheurística OCH ha sido evaluada mediante su aplicación en diferentes problemas y la posterior comparación de su rendimiento con respecto a las técnicas que ya se estaban usando con esos problemas. Esta es la forma de proceder habitual con las nuevas metaheurísticas. Inicialmente, OCH fue evaluada utilizando el conocido problema $\mathcal{NP}$-duro del viajante de comercio (*traveling salesman problem*, TSP). Desde entonces, OCH ha visto comprobado su rendimiento contra más de un centenar de problemas $\mathcal{NP}$-duros de diferente índole (Dorigo, 2003). En la Tabla 1 se recoge una lista de algoritmos OCH propuestos y sus variantes clasificados por tipo de problema y problema específico.

Tabla 1. Lista[5] de aplicaciones de algoritmos OCH clasificados por tipo de problema.

Tipo problema	Problema	Algoritmo	Referencia
Routing	TSP	AS	(Dorigo, 1991), (Dorigo, 1992), (Dorigo, 1996)
		Ant-Q	(Gambardella, 1995)
		ACS	(Gambardella, 1996), (Dorigo, 1997a), (Dorigo, 1997b)
		MMAS	(Stützle, 1997), (Stützle, 2000)
		AS_{rank}	(Bullnheimer, 1999a)
		BWAS	(Cordón, 2000)
	Orienteering problem	ACO-OP	(Liang, 2003)
		Grid-ACO-OP	(Mocholi, 2005)
		GPU-ACO-OP	(Catala, 2007)
	Vehicle routing	HAS-VRP	(Bullnheimer, 1999b)
		MACS-VRPTW	(Gambardella, 1999a)
		D-ants	(Reimann, 2004)
	Sequential ordering	HAS-SOP	(Gambardella, 1997)
	Telecommunications networks	AntNet	(Di Caro, 1998)
		ACO-VWP	(Navarro, 1999)
Assignment	Quadratic assignment	AS-QAP	(Maniezzo, 1994)
		HAS-QAP	(Gambardella, 1999b)

[5] La lista de dominios de aplicación, con los algoritmos OCH que los abordan, es una combinación de las listas que aparecen en (Dorigo, 2003) y (Dorigo, 2006) a la que se han realizado algunas correcciones e introducido datos que faltaban para completar los campos presentados. La columna "Algoritmo" contiene el nombre del algoritmo referenciado o, en su defecto, el nombre del algoritmo en que está basado.

Tipo problema	Problema	Algoritmo	Referencia
		ANTS-QAP	(Maniezzo, 1999)
		MMAS-QAP	(Stützle, 2000)
	Course timetabling	MMAS	(Socha, 2002)
	Graph coloring	ANTCOL	(Costa, 1997)
Scheduling	Project scheduling	ACO-RCPS	(Merkle, 2002)
	Total Tardiness	ACS-SMTTP	(Bauer, 1999), (Merkle, 2003)
	Total weighted tardiness	ACS-SMTWTP	(Besten, 2000), (Merkle, 2003)
	Open shop	Beam-ACO-OSS	(Blum, 2005a)
	Flow shop	MMAS-FSP, SACO	(Stützle, 1998a), (T'kindt, 2002)
	Job shop	AS-JSP	(Colorni, 1994)
Subset	Set covering	Varios[6]	(Lessing, 2004)
	l-cardinality tres	ACO-KCT	(Blum, 2005b)
	Multiple knapsack	AS-MKP	(Leguizamón, 1999)
	Maximum clique	Ant-Clique	(Fenet, 2003)
Otros	Constraint satisfaction	Ant-P-solver	(Solnon, 2000)
	Classification rules	Ant-Miner	(Parpinelli, 2002)
		AntMiner+	(Martens, 2006)
	Bayesian networks	ACO-B	(Campos, 2002)
	Protein folding	ACO-HPPFP-3	(Shmygelska, 2005)
	Docking	PLANTS	(Korb, 2006)

[6] En el trabajo referenciado los autores utilizan los algoritmos MMAS, ACS, un híbrido MMAS-ACS y el *Approximate Nondeterministic Tree-Search* (ANTS), y evalúan su rendimiento habiéndolos modificado para que compartan la misma estrategia de búsqueda local y los mismos tipos de información heurística.

3.5 Líneas abiertas de investigación en OCH

En este punto describimos líneas abiertas de investigación de la metaheurística OCH, tanto en el diseño de aproximaciones con mejor rendimiento y escalabilidad, como en la aplicación a dominios de problema con características especiales.

3.5.1 Paralelización

La obtención de algoritmos paralelos de OCH ha sido sujeto de intensas labores de investigación. El objetivo básico a conseguir es la aceleración del proceso de construcción de soluciones sin comprometer la calidad de las soluciones finales obtenidas. Los algoritmos propuestos en este campo de investigación se clasifican en dos categorías según el paralelismo sea individual (a nivel de hormiga) o global (a nivel de colonia). En el primer caso, los algoritmos propuestos ubican a cada hormiga en un procesador dedicado obteniéndose de esta forma un paralelismo de granularidad fina. En el segundo caso, en cambio, varias hormigas e incluso una colonia entera comparten un mismo procesador o nodo de computación, lo que se conoce como paralelismo de grano grueso. Como ejemplo de paralelismo de grano fino tenemos el trabajo propuesto en (Bolondi, 1993); sin embargo, esta propuesta no consigue tener buenas propiedades de escalabilidad debido al sobre coste asociado a las comunicaciones que son necesarias.

Por otro lado, como representante de paralelismo de grano grueso podemos encontrar el trabajo de Bullnheimer (Bullnheimer, 1998), en el que se propone un mecanismo mediante el cual el intercambio de información entre colonias tiene lugar cada k generaciones de soluciones. En su propuesta se concluye que a medida que se incrementa el valor k se tarda menos tiempo en obtener una solución. Sin embargo, en el trabajo no se discute cómo evoluciona la calidad de las soluciones obtenidas a medida que varía el factor k. Otro trabajo dentro de esta categoría lo hallamos en (Stützle, 1998b), donde se estudia la calidad de las soluciones que se obtienen al realizar varias ejecuciones independientes del algoritmo con un pequeño número de iteraciones, y comparando esa solución con la solución obtenida mediante una única ejecución del algoritmo en la que el número total de iteraciones es la suma total de las iteraciones realizadas por las ejecuciones independientes anteriores. En este trabajo se concluye que, en algunas ocasiones, las ejecuciones independientes de menor número de iteraciones consiguen obtener soluciones de mejor calidad que la ejecución única de mayor duración.

Otras dos propuestas de esta categoría las encontramos en (Michel, 1998) y en (Talbi, 1999). En el primer trabajo se propone un modelo de islas; en este caso

cada isla, o nodo de computación, contiene una colonia de hormigas y las colonias intercambian información sobre la mejor solución que han obtenido tras un número de iteraciones fijo. Si la solución que se recibe de una isla vecina es mejor que la mejor solución encontrada hasta ese momento, la colonia receptora adopta la solución vecina como mejor solución propia. A resaltar la propuesta realizada por uno de los autores en un trabajo posterior (Middendorf, 2000) en la que se investigan diferentes tipos de intercambio de información en algoritmos de hormigas multicolonia, y en la que se demuestra que el intercambio de una pequeña fracción de información heurística puede ser ventajoso a la hora de reducir los tiempos de ejecución del algoritmo. En el segundo trabajo (Talbi, 1999) se utiliza una configuración maestro-esclavo en la que cada esclavo contiene una única hormiga que obtiene una única solución. Cada esclavo envía su solución al nodo maestro (o nido de la colonia), quién calculará una nueva matriz de feromonas a enviar a los nodos esclavos.

Todas las propuestas vistas, tanto las que son de paralelismo de grano fino como las de grano grueso, adolecen de dos problemas que limitan su rendimiento:

- Todas las hormigas o colonias trabajan sobre la totalidad del espacio de búsqueda y esto limita seriamente el número de nodos que se pueden considerar en el grafo de construcción.
- Tanto las hormigas como las colonias han de compartir información sobre feromonas, por lo que es necesario que se comuniquen esta información. Las propuestas que han de comunicar la matriz de feromonas al completo no alcanzan buenas prestaciones y, aún en el caso de que sólo se comparta la información de la mejor solución global, tal y como se describe en (Krüger, 1998), es necesario establecer puntos de sincronización entre las colonias para que dicha comunicación tenga lugar, lo cual afecta negativamente a las prestaciones.

Sin embargo, existen dos propuestas de grano grueso y multicolonia en las que las colonias trabajan como islas aisladas e independientes, procesando únicamente una región parcial del espacio de búsqueda completo, lo que solventa directamente los dos problemas mencionados. En ambas propuestas los espacios parciales se obtienen mediante técnicas de clusterización, de manera que el algoritmo final es capaz de escalar de una forma simple con el tamaño del grafo de construcción.

En (Mocholi, 2005) los autores construyen su propuesta inspirándose en la idea de la computación basada en Grids, en la que se plantea una infraestructura de computación formada por un conjunto de nodos distribuidos que trabajan de

forma desacoplada y que son usados para resolver problemas de gran complejidad, bien por el volumen de datos a tratar o por la computación requerida. La propuesta, llamada GRID-OCH-OP, puede ser vista como una infraestructura maestro-esclavo como la que propone en (Middendorf, 2002), pero en la que no existe propagación de la matriz de feromonas dado que los nodos esclavos trabajan en instancias independientes y parciales con respecto al problema original.

En contraposición al uso de un grid, en (Catala, 2007) se presentan dos propuestas de algoritmo de OCH diseñadas e implementadas para ejecutarse sobre procesadores gráficos modernos, ya que estos presentan unas capacidades de computación superiores a los procesadores de propósito general debido al grado de paralelización de su arquitectura, por lo que con una sola tarjeta gráfica sería posible tener múltiples hormigas artificiales construyendo caminos en paralelo. En ambas propuestas la figura de la hormiga reina (en GRID-OCH-OP equivaldría al nodo maestro) se encarga de enviar los espacios de búsqueda parciales a las hormigas y de unir la soluciones parciales que éstas devuelven. En el proceso de creación de la solución global a partir de las soluciones parciales, ambas propuestas aplican una versión "relajada" (sólo algunos métodos son utilizados) de la búsqueda local propuesta en la metaheurística de Búsqueda por Vecindad Variable (Hansen, 1999), aunque su uso es descartado en (Catala, 2007) por ofrecer una mejora en las soluciones poco significativa con respecto al tiempo empleado.

3.5.2 Problemas de optimización dinámicos

La característica común en los problemas dinámicos es que el espacio de búsqueda del problema varía con el tiempo, por lo que tanto las condiciones de la búsqueda que se está realizando, la propia definición de la instancia del problema a resolver, como la propia calidad de las soluciones ya encontradas cambian o pueden cambiar durante la resolución del problema. Ante este tipo de problemas la capacidad de actualizar las características de la búsqueda que se está llevando a cabo se convierte en un aspecto clave para el éxito de un algoritmo que pretenda abordar este tipo de problemas.

Un tipo de problemas que en parte pertenecen al conjunto de problemas de optimización dinámicos es el enrutamiento en redes de telecomunicaciones. En este tipo de problemas las propiedades del sistema, como por ejemplo el coste de uso de un enlace de comunicaciones o la disponibilidad de los nodos, no son estáticas y varían con el uso que se hace del mismo. Como ya hemos visto en la Tabla 1, los algoritmos de colonias de hormigas ya se han aplicado a estos problemas llegando a convertirse en la aproximación que mejor resultado ofrece, como es el caso de (Di Caro, 1998). A partir de este éxito se han desarrollado otros algoritmos de enrutamiento que han sido capaces de

mejorar el rendimiento obtenido hasta el momento; de entre ellos cabe destacar el algoritmo propuesto en (Ducatelle, 2005) y diseñado para abordar el enrutamiento en redes ad hoc de telefonía móvil, para las que ha demostrado ser igual de competitivo que los algoritmos más utilizados aunque con mostrando una mejor escalabilidad.

3.5.3 Problemas de optimización multiobjetivo

La resolución de problemas de optimización multiobjetivo implica solucionar simultáneamente varios problemas de optimización, cada uno con sus propios objetivos y entre los que posiblemente haya conflictos. Una manera habitual de comenzar a trabajar con múltiples objetivos es ordenarlos (como en (Gambardella, 1999a) o (T'kindt, 2002)) o asignarles un peso según su importancia dentro del dominio del problema (como en (Doerner, 2003) donde los resultados de las funciones de evaluación se combinan en una suma ponderada por el peso asignado al objetivo). Además, para evaluar la calidad de las soluciones encontradas se utiliza una función de evaluación diferente por cada objetivo a optimizar. Si no hay un ordenamiento preestablecido o no se puede asignar con anterioridad un peso a los objetivos, los algoritmos de optimización multiobjetivo intentan encontrar el conjunto de Pareto, es decir, el conjunto de soluciones que son óptimas globalmente a todo el problema ya que ninguna de ellas es peor que las demás teniendo en cuenta su resultado para todas las funciones de evaluación. En (Doerner, 2006) los autores proponen un algoritmo OCH para la búsqueda del conjunto de Pareto para el problema de optimización de una cartera de inversión.

3.6 Conclusiones

En este capítulo hemos descrito con mayor detalle la técnica metaheurística de optimización basada en colonias de hormigas, OCH, explicando en primer lugar la capacidad natural de las hormigas para encontrar el camino más corto entre su nido y una fuente de comida, además de los experimentos que confirman este comportamiento. Luego hemos expuesto las características matemáticas tanto de los problemas que se pueden resolver empleando esta técnica, como del modelo de hormiga artificial utilizado para simular el comportamiento de las hormigas naturales en algoritmos OCH. Más tarde hemos explicado el funcionamiento de un algoritmo OCH genérico y presentado cuatro de los diversos algoritmos existentes en la literatura, uno de los cuales (el sistema de colonias de hormigas) tomamos como base para nuestra propuesta. En el punto 3.4 hemos presentado los dominios de problema más habituales en los que se aplica la metaheurística OCH, así como una lista no exhaustiva de propuestas concretas de algoritmos OCH diseñadas

para abordar tipos de problema específicos. A continuación hemos presentado algunas de las líneas abiertas de investigación que más interés suscitan, explicando las características de los problemas a los que van dirigidas o que características de la metaheurística pretenden mejorar, y nombrado algunas propuestas realizadas y que demuestran la potencia y versatilidad de OCH, ya que han conseguido en muchos casos convertirse en las aproximaciones de referencia en sus dominios, tanto por la calidad de sus soluciones como por el rendimiento demostrado (en términos de tiempo empleado y/o tamaño del problema abordado). Sin embargo, ninguna de las propuestas existentes cubre las características que en el punto 1.1 planteábamos como clave para las futuras generaciones de ambientes inteligentes; a saber, utilizar la información semántica contextual recopilada para prever situaciones contextuales en el futuro cercano, y añadir el estado emocional del usuario del ambiente a la información semántica contextual recopilada y tenerlo en cuenta en el proceso de búsqueda y selección de acciones a desencadenar por parte del ambiente inteligente. Con estas características, lo que pretendemos es que el ambiente sea capaz de adelantarse a situaciones futuras que vayan a darse con alta probabilidad y tome las decisiones más adecuadas para conseguir mantener al usuario en un estado cercano al estado de *flow*.

En los siguientes capítulos de esta tesis presentamos nuestras propuestas algorítmicas basadas en OCH que han sido diseñadas para trabajar con información semántica con datos heterogéneos, de entre los cuáles prestará especial atención al estado emocional para que la decisión sobre qué acción realizar esté influida por el objetivo de acercar al usuario al estado de *flow* o experiencia óptima.

"Es muy probable que las mejores decisiones no sean fruto de una reflexión del cerebro, sino del resultado de una emoción.",

Eduard Punset (2006).

Capítulo 4

Una Propuesta de Algoritmo OCH para la Toma de Decisiones Influenciada por Emociones

4.1 Hierarchical network of affective-cognitive decisions

La gran versatilidad de las técnicas metaheurísticas, que permite su aplicación a una gran variedad de dominios, implica la necesidad de realizar ciertas modificaciones para adaptarlas a las características específicas de cada problema. En el caso que nos ocupa, necesitamos dotar a los sistemas de colonias de hormigas con las estructuras y modelos adecuados que les permitan simular las emociones que deben influir en el proceso de búsqueda de la mejor decisión. Entre las posibles opciones revisadas, nos hemos inclinado por utilizar como base la propuesta *hierarchical network of affective-cognitive decisions*, ya que creemos que es la más adecuada para adaptar un algoritmo OCH puesto que su funcionamiento ya tiene en cuenta en cierta medida la exploración/explotación del espacio de búsqueda, así como que el proceso de actualización de sus modelos es similar a la actualización de feromonas llevada a cabo por las hormigas.

Esta propuesta de sistema de toma de decisiones influenciada por emociones fue introducida por sus autores en (Ahn, 2005), quiénes tomaron como punto de partida de su trabajo las conclusiones alcanzadas por investigadores de áreas como la psicología o la neurociencia afectiva de las que se desprendía que

> *"… los sentimientos humanos y la experiencia emocional tienen un papel significativo y útil en procesos como el aprendizaje o la toma de decisiones. En individuos normales, las influencias relacionadas con experiencias emocionales previas, en situaciones similares, ayudan al proceso de razonamiento y facilitan el procesamiento eficiente del conocimiento y la lógica necesaria en la toma consciente de decisiones".*

Inspirados en los hallazgos de esas áreas de investigación y en el propio proceso de aprendizaje humano, los autores desarrollaron un *framework* computacional afectivo-cognitivo para el aprendizaje y la toma de decisiones. La idea fundamental en su propuesta asume que un agente afectivo (una entidad virtual) cuenta tanto con capacidades cognitivas como emocionales, y que se puede modelar su motivación en base a un sistema de *rewards* en el que el "deseo" por el *reward* interno, surgido de cognición y emoción, más el del *reward* externo, recibido del entorno, podría explicar su motivación en los procesos de aprendizaje y toma de decisiones que realiza.

Para dar soporte a esa idea los autores diseñaron un algoritmo de aprendizaje por refuerzo (*reinforcement learning*) que utiliza la información de los *rewards* obtenidos hasta el momento, tanto internos como externos, para determinar cuál es la mejor decisión a tomar en el estado cognitivo y emocional actual, teniendo en cuenta que existen unos estados cognitivos y emocionales que se debe intentar alcanzar ya que han sido marcados como "deseables", es decir, tienen asociados unos *rewards* positivos, y que, por otro lado, existen también otros estados marcados como "poco deseables" puesto que sus *rewards* asociados son negativos. En su propuesta, el encargado de evaluar cuál es la mejor decisión a tomar es un bucle interno de simulación que computa el *reward* interno obtenido de realizar cada una de las decisiones posibles dados un estado cognitivo y uno emocional. En el siguiente subpunto se expone con mayor detalle la propuesta algorítmica de los autores.

Otra idea clave de esta propuesta la basan los autores en la posibilidad de que cada una de las posibles decisiones a realizar puede estar compuesta por decisiones de menor nivel (Ahn, 2005). De ahí que propongan la necesidad de que los agentes afectivos cuenten con una red jerárquica de decisiones (*hierarchical network of decisions*) en la que cada decisión de nivel superior cuente con un proceso afectivo-cognitivo de toma de decisiones capaz de seleccionar la decisión adecuada de nivel inferior para los estados cognitivo y emocional en los que se encuentra el agente en ese momento. En la Figura 3 se puede observar una representación de un ejemplo de red jerárquica de decisiones. Los autores prefieren utilizar la expresión decisión afectivo-cognitiva (*affective-cognitive decision*, ACD) para referirse a los nodos/decisiones. Un ACD está compuesto bien por ACDs temporalmente secuenciales de menor nivel, o por habilidades primitivas de menor nivel. Además, un ACD cuenta con un objetivo determinado y tiene la capacidad de activar o desactivar otros ACDs

para conseguir ese objetivo. Cuando se alcanza un estado cognitivo asociado al objetivo del ACD, los autores explican que se ha completado un "episodio" y ese ACD se desactiva y devuelve el control al ACD que lo activó.

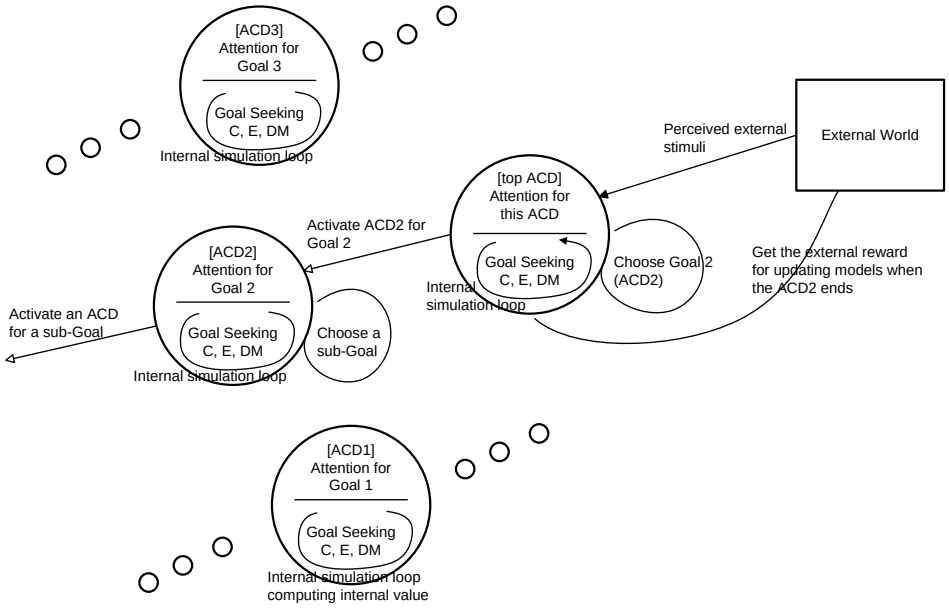


Figura 3. Representación de la red jerárquica de decisiones afectivo-cognitivas para el aprendizaje y la toma de decisiones de manera afectivo-cognitiva. En este ejemplo, en el ACD raíz se selecciona el *Goal* 2 por lo que se activa el ACD2 (Ahn, 2005).

4.1.1 Términos y modelos probabilistas utilizados

Esta propuesta se basa en el uso y mantenimiento de una serie de modelos probabilísticos que explicaremos después de introducir los términos utilizados en su definición y en el algoritmo, y que además aparecerán en lo que resta de capítulo:

- c representa el estado cognitivo actual;
- e representa el estado emocional actual;
- C y E son los conjuntos de todos los estados cognitivos y emocionales, respectivamente ($c \in C = (1,...,|C|)$ y $e \in E = (1,...,|E|)$);
- $\hat{e}$ denota la distribución de probabilidad para el estado emocional actual e ($\hat{e}=(\hat{e}_1,\ldots,\hat{e}_{|E|})$), donde $\hat{e}_j$ representa la probabilidad de que el estado emocional actual sea j;
- d representa la decisión seleccionada a realizar;

- *c'* y *e'* representan el estado cognitivo y el estado emocional resultado de tomar una decisión *d* cuando estábamos en un estado cognitivo *c* y un estado emocional *e*;
- β es la inversa de la temperatura en la función de selección de Boltzmann[7];
- *a* es un factor de descuento que pondera el peso de ciertos operandos en algunas de las fórmulas utilizadas. Toma valores entre 0 y 1;
- γ es un ratio de aprendizaje que controla la velocidad de convergencia de algunos de los modelos probabilistas. También toma valores entre 0 y 1;
- $\lambda_C(d)$ y $\lambda_E(d)$ son, respectivamente, el factor de voluntad (*willpower factor*) y el factor sentimental (*feeling factor*) que controlan el peso con el que cognición y emoción influyen en la selección de un determinado ACD *d*. Toman un valor inicial igual a 1 y, cuando se actualizan (se explica más adelante), se les aplica un factor de descuento, ε_C y ε_E respectivamente, con valor entre 0 y 1.
- r_{CR} es el *reward* cognitivo que se obtiene tras alcanzar un nuevo estado cognitivo. Viene prefijado para el entorno que se desea modelar y toma los valores 1 o -1 dependiendo si el estado cognitivo asociado es deseado o no;
- r_{ext} es el *reward* externo que se recibe del entorno/usuario una vez llevada a cabo la decisión seleccionada. Viene prefijado para el entorno que se desea modelar y toma los valores 1 o -1 dependiendo si el resultado de realizar la acción elegida es deseado o no.

Como hemos comentado, cada ACD mantiene una serie de modelos probabilísticos que son utilizados y mantenidos por el algoritmo y que, con su uso combinado, permiten generar en el agente el deseo, la motivación por tomar ciertas decisiones ante ciertas situaciones que en el pasado proporcionaron mayores *rewards*. Estos son los modelos:

- El *attention model* se utiliza para extraer aquellos datos útiles, para el objetivo del ACD que lo utiliza, a partir de los estímulos externos multimodales percibidos (como por ejemplo el estado en que se encuentra el usuario con él que se interacciona). Este modelo es el encargado de determinar el estado cognitivo *c* para ese ACD. En su propuesta los autores asumen que,

---

[7] En la función de selección de Boltzmann se usa un valor alto de temperatura al principio, lo cual hace que la presión de selección sea baja. Con el paso de las generaciones, la temperatura disminuye, lo que aumenta la presión de selección. De esta manera se incita a un comportamiento exploratorio en las primeras generaciones y se acota a uno más explotatorio hacia el final del proceso evolutivo (Coello, 2008).

para un intervalo de tiempo determinado, un ACD sólo tiene un estado cognitivo.

- *Pr(c'|c,d)* es el modelo cognitivo (*cognition model*) y se trata de un modelo probabilista de transición entre estados cognitivos, es decir, este modelo devuelve la probabilidad de transitar del estado cognitivo *c* al *c'* si tomamos la decisión *d*. Por esta razón también lo denominan *world simulation model*. Para el cálculo de este modelo se hace uso de $Q_C(c,d,c_{new})$, un contador de transiciones que almacena las veces que hemos llegado a un estado cognitivo c_{new} desde un estado cognitivo *c* después de tomar la decisión *d*. Así, tenemos que el cálculo de este modelo se hace de la siguiente manera:

$$Q_C(c,d,c_{new}) = Q_C(c,d,c_{new}) + 1$$

$$\Pr(c' = i \mid c,d) = \frac{Q_C(c,d,c' = i)}{\sum_{k=1}^{|C|} Q_C(c,d,c' = k)}, \quad \forall i \in C$$

- $R_{CR}(c')$ es el modelo de *reward* cognitivo interno (*model of internal cognitive reward*) y permite a los autores modelar qué estados cognitivos son los deseados, es decir, cómo de importante cognitivamente es *c'* (el siguiente estado cognitivo) para que un determinado ACD pueda conseguir su objetivo. Puesto que se calcula a partir de r_{CR}, podríamos decir que actúa como una especie de memoria del *reward* cognitivo real recibido en el pasado para idénticas situaciones. Para los autores, este modelo junto con el modelo cognitivo *Pr(c'|c,d)* les permite representar una cierta "automotivación cognitiva". Este modelo se actualiza con la siguiente fórmula:

$$R_{CR}(c_{new}) = (1-\gamma)\cdot R_{CR}(c_{new}) + \gamma \cdot r_{CR,new}$$

- Aunque en esta propuesta un estado emocional ha sido modelado como un estado discreto de la forma $e \in E = (1,...,|E|)$, los autores comentan que *e* no tiene porque ser conocido explícitamente, basta con disponer de la distribución probabilista $\hat{e}$ del estado emocional actual. Esta distribución se obtiene a partir del modelo emocional *Pr(e'|c',e)* (*emotion cognitive appraisal model*), definido a priori para el entorno que se modela, que representa la probabilidad de que se dé un determinado estado emocional *e'* en función del estado emocional *e* previo, y de haber alcanzado un estado cognitivo *c'*. En la siguiente fórmula podemos observar cómo se calcula $\hat{e}$:

$$\hat{e}_{new,j} = \sum_{k=1}^{|E|} \hat{e}_{old,k} \Pr(e' = j \mid c', e_{old} = k), \quad \forall j \in E$$

- El modelo de transición emocional *Pr(e'| ê, c, d)* (*emotion transiton model*) permite conocer la probabilidad de alcanzar un estado emocional *e'* a partir de la distribución de probabilidad del estado emocional actual *ê*, el estado cognitivo actual *c* y la decisión *d* seleccionada (ACD *d* de menor nivel seleccionado). Como podemos ver a continuación, este modelo se calcula a partir del modelo emocional, el modelo cognitivo y *ê*.

$$\Pr(e' \mid \hat{e}, c, d) = \sum_{i=1}^{|C|} \sum_{k=1}^{|E|} \hat{e}_k \Pr(e' \mid c' = i, e = k) \Pr(c' = i \mid c, d)$$

- $R_{ER}(e')$ es el modelo de incentivo emocional interno (*model of internal emotional reward*) y se utiliza para modelar cuán deseables son los estados emocionales existentes, en otras palabras, modela cómo de emocionalmente deseable es el nuevo estado emocional *e'* alcanzado. Al igual que con el modelo emocional, este modelo también es definido a priori. Según los autores, este modelo junto con el modelo de transición emocional *Pr(e'| ê, c, d)* les permite representar una cierta "automotivación emocional" para la selección de un cierto ACD *d* de nivel inferior.

- El modelo de toma de decisiones (*decision-making model*), *Pr(d | ê, c)*, puede verse como un modelo de búsqueda de subobjetivos o ACD de menor nivel. Este modelo se calcula a partir de la fórmula de selección Boltzmann usando como función de valoración el sumatorio $\sum_{j=1}^{|E|} \hat{e}_j Q_{DM}(e = j, c, k))$.

$$\Pr(d = k \mid \hat{e}, c) = \frac{\exp(\beta \sum_{j=1}^{|E|} \hat{e}_j Q_{DM}(e = j, c, k))}{\sum_{l=1}^{|D|} \exp(\beta \sum_{j=1}^{|E|} \hat{e}_j Q_{DM}(e = j, c, l))}$$

- En la fórmula anterior, uno de los operandos es $Q_{DM}(\hat{e},c,d)$ que representa cómo de "valioso" o "deseable" (desde el punto de vista de los *reward* que se pueden obtener) es tomar la decisión *d* cuando nos encontramos en el estado cognitivo *c* y en el emocional *j*. Para calcular esa "valía" se utilizan: el valor del modelo de *reward* externo R_{ext} para *c* y *d*, el valor del modelo de *reward* cognitivo R_{CR} para todos los estados cognitivos, el valor del modelo de *reward* emocional R_{ER} para todos los estados emocionales, y también se calcula el valor máximo Q_{DM} de las tripletas compuestas por estado cognitivo, emocional y decisión alcanzables desde (*j*,*c*,*d*). Todos estos valores (excepto R_{ext}) son ponderados por la probabilidad de que los estados a los que hacen referencia sean alcanzados, y al valor máximo Q_{DM} encontrado se le aplica un factor *a* que controla/reduce su influencia sobre

el valor Q_{DM} que estamos calculando. En otras palabras, la "valía" o el "deseo" de una cierta tripleta de estado cognitivo, emocional y decisión es mayor si desde ella se pueden alcanzar otras con valores altos de "valía".

$$Q_{DM}(j,c,d) = R_{ext}(c,d) + \sum_{k=1}^{|C|} R_{CR}(c'=k)\Pr(c'=k \mid c,d)$$
$$+ \sum_{k=1}^{|C|}\sum_{l=1}^{|E|} R_{ER}(e'=l)\Pr(e'=l \mid c'=k, e=j)\Pr(c'=k \mid c,d)$$
$$+ \alpha \sum_{k=1}^{|C|}\sum_{l=1}^{|E|} \max_{d'} Q_{DM}(l,k,d')\Pr(e'=l \mid c'=k, e=j)\Pr(c'=k \mid c,d),$$
$$\forall j \in E$$

- $Q_{CR}(c,d)$ es el modelo de *reward* cognitivo (*cognitive reward model*) y se utiliza para calcular el valor cognitivo interno (*internal cognitive value*) $v_{int,C}$.

$$Q_{CR}(c,d) = \sum_{j=1}^{|C|} \left\{ R_{CR}(c'=j) + \alpha \max_{d'} Q_{CR}(j,d') \right\} \Pr(c'=j \mid c,d)$$

- $Q_{ER}(\hat{e},c,d)$ es el modelo de *reward* emocional (*emotion reward model*) y se utiliza para calcular el valor emocional interno (*internal affective value*) $v_{int,E}$.

$$Q_{ER}(j,c,d) = \sum_{k=1}^{|C|}\sum_{l=1}^{|E|} \left\{ R_{ER}(e'=l) + \alpha \max_{d'} Q_{ER}(l,k,d') \right\}$$
$$\cdot \Pr(e'=l \mid c'=k, e=j)\Pr(c'=k \mid c,d), \quad \forall j \in E$$

$$v_{\text{int},E} = Q_{ER}(\hat{e},c,d) = \sum_{j=1}^{|E|} \hat{e}_j Q_{ER}(j,c,d)$$

- $R_{ext}(c,d)$ es el modelo de *reward* externo (*external reward model*) y permite modelar la cantidad de *reward* externo que se recibe al tomar una cierta decisión *d* en un estado cognitivo *c*. Actúa como una especie de memoria del *reward* externo real recibido en el pasado para las mismas situaciones. La convergencia entre el modelo y el incentivo externo real recibido, en una misma situación, se controla a través de un factor de aprendizaje.

$$R_{ext}(c,d) = (1-\gamma)\cdot R_{ext}(c,d) + \gamma \cdot r_{ext}$$

- El valor interno total v_{int} (*total internal value*) se calcula a partir de los valores emocional y cognitivo internos, $v_{int,E}$ y $v_{int,C}$, aplicándoles a cada uno un factor de ponderación dependiente del ACD *d* que se esté evaluando ($\lambda_E(d)$ y $\lambda_C(d)$ respectivamente). Los autores lo utilizan como criterio de parada en

la búsqueda de un ACD *d* de menor nivel adecuado para la situación actual (se deja de buscar si $v_{int}>0$).

$$v_{\text{int}} = \lambda_C(d) \cdot v_{\text{int},C} + \lambda_E(d) \cdot v_{\text{int},E}$$

4.1.2 Explicación de la propuesta algorítmica

El algoritmo descrito en (Ahn, 2005) es ejecutado por todos los ACD y consta de dos fases principales: la fase actora, en la que se busca y selecciona el ACD de menor nivel, y la fase crítica, en la cual se actualizan los modelos una vez finaliza el ACD de menor nivel activado previamente. Ambas fases se encuentran dentro de un bucle sin fin en el que las primeras tareas que se realizan son una serie de comprobaciones, como por ejemplo detectar si se ha recibido la señal de activación/desactivación, o comprobar si se ha completado un episodio. Si el ACD se activa después de esas comprobaciones, obtiene el estado cognitivo *c* actual a través del modelo de atención y actualiza su distribución de probabilidad para el estado emocional $\hat{e}$ a partir del $\hat{e}$ de su ACD de nivel superior. Cabe destacar que el ACD de mayor nivel, el ACD raíz, tiene un comportamiento especial puesto que siempre está activado.

En caso de que el ACD esté activo, antes de ejecutar la fase actora se comprueba si el último ACD de menor nivel elegido ya no está activo o si el estado cognitivo ha cambiado; si se cumple cualquiera de las dos condiciones, el primer paso de esta fase es la inicialización de factores como α, β, γ o λ, además de fijar el número máximo de iteraciones del bucle de búsqueda en 100, convirtiéndose de esta manera en el segundo criterio de parada tras $v_{int}>0$. Dentro del bucle, llamado por los autores bucle de simulación interno o mental (*mental or internal simulation loop*), se procede a calcular el modelo de toma de decisiones $Pr(d \mid \hat{e}, c)$ y a partir de la distribución probabilista obtenida se selecciona un cierto ACD *d*. Después de esto se obtienen los valores internos cognitivo y emocional y se calcula el valor interno total v_{int}. Si los valores internos cognitivo y emocional entran en conflicto (sus valores tienen signos diferentes), se actualizan los factores de voluntad y sentimental aplicándoles los factores de descuento, ε_C y ε_E respectivamente. Entonces se actualiza el contador de iteraciones y se comprueba la guarda del bucle antes de iniciar una nueva iteración. Una vez fuera del bucle se comprueba si el estado cognitivo actual es diferente del previo; de ser así, si el ACD seleccionado es diferente de aquél que esté en ejecución en ese momento, se le envía la señal de desactivación. Para finalizar, y como último paso de esta fase, se envía la señal de activación al ACD de menor nivel seleccionado.

La fase llamada crítica por los autores se ejecuta si se ha recibido la señal de finalización del ACD *d* de menor nivel activado previamente. En caso afirmativo, se relee la distribución de probabilidad $\hat{e}$ para el estado emocional

actual, se percibe el nuevo estado cognitivo c_{new}, y se recuperan el *reward* cognitivo $r_{CR,new}$ y el externo r_{ext}. A partir de estos datos, y utilizando las respectivas fórmulas ya presentadas, se actualiza el modelo de *reward* cognitivo $R_{CR}(c')$, el modelo de *reward* externo $R_{ext}(c,d)$, el modelo cognitivo $Pr(c'|c,d)$, los modelos de *reward* cognitivo $Q_{CR}(c,d)$ y emocional $Q_{ER}(\hat{e},c,d)$, así como la componente $Q_{DM}(\hat{e},c,d)$ del modelo de toma de decisiones $Pr(d \mid \hat{e}, c)$. Una vez realizadas las actualizaciones, termina la fase crítica y comienza una nueva iteración del algoritmo.

4.2 Nuestra propuesta: algoritmo OCH para la toma de decisiones influenciada por emociones

La propuesta de algoritmo que presentamos a continuación tiene como objetivo proveer a entidades emocionales con un mecanismo para la toma de decisiones que, teniendo en cuenta la experiencia (el conocimiento previo) de la entidad y las características propias del entorno/mundo en el que se encuentra, sea capaz de seleccionar una decisión con el objetivo de que la entidad se mantenga en, o se dirija hacia, una serie de estados cognitivos y emocionales en los que "desea" estar (la idea del "deseo" la tomamos de (Ahn, 2005). Así, dependiendo de cómo de deseados por la entidad sean los estados cognitivos y emocionales a los que llegue, éstos le reportarán un mayor o menor premio (*reward*), o satisfacción cognitiva y emocional. Además, dependiendo de la decisión tomada y del estado cognitivo alcanzado, la entidad puede recibir un *reward* externo, por ejemplo, del usuario con el que interactúe.

Para realizar la búsqueda de la decisión a tomar el algoritmo simula la toma, y los efectos que ésta pueda desencadenar, de varias decisiones consecutivas utilizando hormigas artificiales. El número de decisiones consecutivas que se simularán representa el número de pasos en el futuro que queremos que las hormigas prevean, de manera que se seleccione la decisión que se espera reporte a la entidad los mayores *rewards* cognitivos, emocionales y externos en el futuro cercano. El siguiente pseudocódigo resume el funcionamiento del algoritmo que proponemos y que explicaremos en los siguiente subpuntos:

```
Inicialización de los parámetros del algoritmo
Repetir
    Repetir
        Cada hormiga de la colonia simula una secuencia de decisiones
        Elegir la mejor secuencia de decisiones obtenida
    Hasta alcanzar el criterio de parada
```

Tomar la primera decisión de la mejor secuencia de decisiones

Actualizar los parámetros del algoritmo

Mientras la entidad viva

El espacio de búsqueda del problema que las hormigas utilizarán se puede ver como un grafo en el que los nodos son todos los posibles estados cognitivos de la entidad emocional, y en el que las aristas que unen los nodos son las decisiones que la entidad puede tomar.

A continuación introducimos unos términos utilizados en nuestro algoritmo, que no aparecían en la propuesta de (Ahn, 2005), y que aparecerán en lo que resta de capítulo:

- S es el conjunto ordenado de tuplas de la forma $(c_i, \hat{e}_i, d_i)$ que una hormiga prevé, dónde las d son las decisiones que la hormiga ha calculado que maximizan los *rewards* que obtendrá en el futuro cercano la entidad para los c y $\hat{e}$ previstos, siendo $S_0=(c_0, \hat{e}_0, d_0)$ la tupla con el estado cognitivo y emocional a partir de los cuales se comienza la simulación y d_0 la decisión que se debe tomar;
- $M = |S|$, es decir, M es igual al número de pasos en el futuro que queremos que simulen las hormigas y sirve como criterio de parada de la simulación;
- G es un valor calculado a partir de los resultados devueltos por las hormigas (cómo se obtiene G se explica más adelante) y que permite comparar las soluciones devueltas para seleccionar la mejor.

El funcionamiento del algoritmo que proponemos se puede dividir en 3 fases, a saber, inicialización de parámetros, simulación de toma de decisiones y actualización de parámetros, que pasaremos a exponer a continuación.

4.2.1 Inicialización de parámetros

La siguiente lista enumera todos los pasos de inicialización que se siguen para dotar de valores iniciales a los parámetros utilizados en el algoritmo:

- Se fijan los valores del r_{CR} (*reward* cognitivo), 1 para los estados cognitivos deseados y -1 para los no deseados.
- Se inicializan los valores del $R_{CR}(c)$ (modelo de *reward* cognitivo) teniendo en cuenta los valores dados a r_{CR} ya que es a esos valores a los que debe converger finalmente $R_{CR}(c)$.
- Se fijan los valores del $R_{ER}(e)$ (modelo de reward emocional), 1 para los estados emocionales deseados y -1 para los no deseados.

- Se fijan los valores de $Pr(e'|c', e)$ (modelo emocional) con valores entre 0 y 1 dependiendo de si c' y e son estados deseados o no.
- Se fija el r_{ext} (*reward* externo) que se recibirá para cada estado cognitivo alcanzado.
- Se inicializa $R_{ext}(c,d)$ (modelo de reward externo).
- Se inicializa el modelo cognitivo $Pr(c'|c,d)$ y $Q_C(c,d,c_{new})$.
- Se fija M, el número de pasos en el futuro que las hormigas simularán.
- Se fija el número máximo de iteraciones que se usarán las hormigas y la diferencia mínima ε para seguir con las iteraciones que queden.
- Se fija el ratio de aprendizaje γ entre 0 y 1.
- Se recoge el vector de estímulos del entorno.
- Se evalúa en qué estado cognitivo inicial se encuentra la entidad.

4.2.2 Simulación de toma de decisiones

Una vez dotados de valores iniciales los parámetros, la siguiente fase del algoritmo es que cada hormiga simule tomar secuencias de decisiones, y que lo haga varias veces con el objetivo de mejorar las soluciones que construya. Por esta razón, el proceso de búsqueda y selección de la decisión a tomar para el instante actual está formado por dos bucles anidados: el interno llevado a cabo en paralelo por cada hormiga artificial, y el bucle externo en el que las soluciones devueltas por las hormigas son comparadas hasta encontrar aquélla que maximiza la siguiente fórmula:

$$G \equiv Max \sum_{c \in S} R_{CR}(c) + \sum_{k=1, \hat{e} \in S}^{|S|} R_{ER_k}(\hat{e}) + \sum_{c,d \in S} \tilde{R}_{ext}(c,d)$$

donde:

- $R_{CR}(c)$ representa el valor del *reward* cognitivo del estado c.
- $R_{ER_k}(\hat{e}) = \sum_{j=1, e \in E}^{|E|} R_{ER}(e_j) \cdot \hat{e}_j$ es el valor del *reward* emocional para cada paso k de la solución calculada por las hormigas. Cada hormiga mantiene el suyo propio ya que, durante la simulación, cada hormiga crea una copia local del vector de probabilidades ê que irá actualizando a medida que simule tomar decisiones y alcanzar estados cognitivos.

- $\tilde{R}_{ext}(c,d)$ guarda el *reward* externo medio obtenido con esa decisión *d* para ese estado cognitivo *c*. Se actualiza una vez obtenido el r_{ext} resultante de la realización de la decisión elegida. Se utiliza durante la simulación para generar valores de r_{ext} lo más parecidos posibles a la realidad.

Para realizar su trabajo las hormigas artificiales han de compartir cierta información, por lo que podríamos denominarla conocimiento global. En el problema que nos ocupa esta es la información compartida:

- todas las hormigas conocen cuál es el estado cognitivo *c* actual y cuál es el vector de probabilidades $\hat{e}$ para el estado emocional actual;
- las hormigas comparten una matriz tridimensional que, para cada *c*, *e* y *d* (también puede verse como una matriz bidimensional por cada estado cognitivo), tiene un valor que representa cómo de *atractiva* (valiosa) es esa decisión si, estando en ese estado cognitivo y en ese estado emocional, tomamos esa decisión. Esta matriz también puede verse como la unificación de las matrices de feromonas y de atracción que usan las hormigas en una implementación más tradicional de un algoritmo OCH (Liang, 2003). Esta matriz de atracción en nuestra propuesta de algoritmo OCH es equivalente al modelo de toma de decisiones $Q_{DM}(e,c,d)$ de (Ahn, 2005).

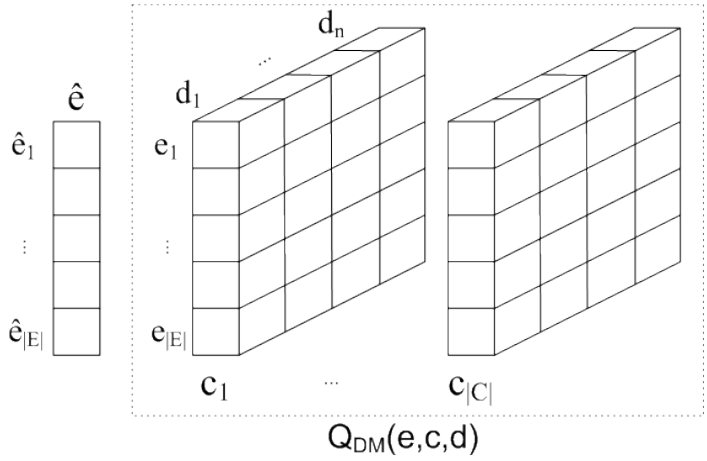


Figura 4. Información compartida por todas las hormigas.

El funcionamiento de la fase de simulación se detalla en forma de pseudocódigo a continuación:

```
Mientras (iteraciones < max_iteraciones) Y (|G_OLD - G_NEW| > ε)
Mientras |S| < M  //bucle realizado por cada hormiga en paralelo
```

Si hacemos explotación, obtener probabilísticamente una decisión para el estado cognitivo y emocional actual según

$$\Pr(d = k \mid \hat{e}, c) = \frac{\exp(\beta \sum_{j=1}^{|E|} \hat{e}_j Q_{DM}(e = j, c, k))}{\sum_{l=1}^{|D|} \exp(\beta \sum_{j=1}^{|E|} \hat{e}_j Q_{DM}(e = j, c, l))}$$

Si hacemos exploración, obtener aleatoriamente una decisión para el estado cognitivo y emocional actual

Simular el estado cognitivo al que llega la entidad, con $\Pr(c_{new} \mid c, d)$, y su *reward* cognitivo $r_{CR,new}$ asociado, con $R_{CR}(c_{new})$.

Actualizar copia local del vector de probabilidades $\hat{e}_{new}$ mediante:

$$\hat{e}_{j,new} = \sum_{k=1}^{|E|} \hat{e}_{old,k} \Pr(e = j \mid c_{new}, e_{old} = k), \quad \forall j \in E$$

Actualizar $R_{ER_k}(\hat{e}_{new})$ para este paso *k* de la simulación mediante

$$R_{ER_k}(\hat{e}_{new}) = \sum_{j=1, e \in E}^{|E|} R_{ER}(e_j) \cdot \hat{e}_{j,new}$$

Simular el reward externo r_{ext} obtenido con $\tilde{R}_{ext}(c_{new}, d)$

Actualizar modelo del *reward* cognitivo:

$$R_{CR}(c_{new}) = (1-\gamma) \cdot R_{CR}(c_{new}) + \gamma \cdot r_{CR,new}$$

Actualizar modelo del *reward* externo:

$$R_{ext}(c_{new}, d) = (1-\gamma) \cdot R_{ext}(c_{new}, d) + \gamma \cdot \tilde{R}_{ext}(c_{new}, d)$$

Actualizar modelo cognitivo:

$$Q_C(c, d, c_{new}) = Q_C(c, d, c_{new}) + 1$$

$$\Pr(c' = i \mid c, d) = \frac{Q_C(c, d, c' = i)}{\sum_{k=1}^{|C|} Q_C(c, d, c' = k)}, \quad \forall i \in C$$

Sincronizar las hormigas para garantizar que no se modifican valores del modelo de toma de decisiones mientras haya hormigas calculando probabilidades para elegir una decisión

Actualizar modelo de toma de decisiones:

$$Q_{DM}(j,c,d) = R_{ext}(c,d) + \sum_{k=1}^{|C|} R_{CR}(c'=k)\Pr(c'=k \mid c,d)$$
$$+ \sum_{k=1}^{|C|}\sum_{l=1}^{|E|} R_{ER}(e'=l)\Pr(e'=l \mid c'=k, e=j)\Pr(c'=k \mid c,d),$$
$$\forall j \in E$$

Evaporar de feromonas

}//Fin Mientras |S| < M

Comparar soluciones y quedarse con la que maximice la fórmula:

$$G \equiv Max \sum_{c \in S} R_{CR}(c) + \sum_{k=1, \hat{e} \in S}^{|S|} R_{ER_k}(\hat{e}) + \sum_{c,d \in S} \tilde{R}_{ext}(c,d)$$

}//Fin Mientras iteraciones

Como habíamos mencionado anteriormente la fase de simulación se lleva a cabo en dos bucles anidados. El bucle interior contiene las tareas realizadas por las hormigas en paralelo con el fin de hallar la mejor secuencia de decisiones. La primera de esas tareas es la selección, mediante el modelo de toma de decisiones $\Pr(d \mid \hat{e}, \hat{c})$, de la decisión con una probabilidad mayor de conducir a mejores *rewards* dados un par de estados cognitivo y emocional. Entonces las hormigas simulan la toma de esa decisión y cuál sería el estado cognitivo alcanzado con el modelo cognitivo $\Pr(c_{new} \mid \hat{c}, d)$. Una vez obtenido el nuevo estado cognitivo, simulan recibir el *reward* cognitivo asociado, $r_{CR,new}$. Entonces las hormigas actualizan su copia local de $\hat{e}$, simulando así los efectos emocionales de alcanzar el nuevo estado cognitivo habiendo tomado la decisión d. También actualizan el modelo de *reward* emocional $R_{ER_k}(\hat{e})$ y hacen uso del *reward* externo medio $\tilde{R}_{ext}(c,d)$ para simular el *reward* externo que la entidad podría haber recibido en la realidad. Luego, las hormigas actualizan los modelos de *reward* cognitivo y externo, $R_{CR}(c_{new})$ y $R_{ext}(c_{new},d)$ respectivamente, haciendo uso del *reward* externo medio en el último caso; y actualizan también el modelo de transición de estados cognitivos $\Pr(c' \mid \hat{c}, d)$ y el modelo de toma de decisiones $Q_{DM}(e,c,d)$.

Por su parte, el bucle externo utiliza la fórmula presentada al inicio de este subpunto para comparar las soluciones devueltas por las hormigas y seleccionar aquélla que la maximice. Además, el bucle externo se encarga de comprobar los criterios de parada, que son: que el número de iteraciones en las que se han usado las hormigas para buscar soluciones no supere un máximo

establecido, y que la diferencia entre la solución obtenida en la iteración previa y la solución de la actual no sea superior a un cierto ε. Una vez alcanzado cualquiera de los dos criterios de parada, se deja de simular y se devuelve la mejor solución encontrada por las hormigas hasta ese momento.

4.2.3 Actualización de parámetros

Una vez que las hormigas han terminado de simular y han devuelto una solución, la entidad procede a: tomar la primera decisión de la secuencia de decisiones devuelta, evaluar su estado cognitivo actual, y a actualizar los parámetros del algoritmo (con valores reales, no aproximados con modelos). El siguiente pseudocódigo describe la actualización de parámetros, en esencia idéntica a la actualización que se realiza durante la simulación pero con diferencias que detallaremos después.

Tomar la decisión *d* del elemento S_0 de la solución *S* elegida

Evaluar en qué estado cognitivo se encuentra la entidad, c_{new}, y su *reward* cognitivo $r_{CR,new}$ asociado

Actualizar el vector de probabilidades $\hat{e}$ compartido para el estado emocional actual, teniendo en cuenta $\hat{e}_{old}$, el vector de probabilidades del estado emocional anterior

$$\hat{e}_{j,new} = \sum_{k=1}^{|E|} \hat{e}_{old,k} \Pr(e = j \mid c_{new}, e_{old} = k), \quad \forall j \in E$$

Recoger el vector de estímulos del entorno y el *reward* externo r_{ext}

Actualizar el modelo del *reward* cognitivo:

$$R_{CR}(c_{new}) = (1-\gamma)\cdot R_{CR}(c_{new}) + \gamma \cdot r_{CR,new}$$

Actualizar el modelo del *reward* externo:

$$R_{ext}(c,d) = (1-\gamma)\cdot R_{ext}(c,d) + \gamma \cdot r_{ext}$$

Actualizar el *reward* externo medio: $\tilde{R}_{ext}(c,d)$

Actualizar el modelo cognitivo:

$$Q_C(c,d,c_{new}) = Q_C(c,d,c_{new}) + 1$$

$$\Pr(c' = i \mid c,d) = \frac{Q_C(c,d,c' = i)}{\sum_{k=1}^{|C|} Q_C(c,d,c' = k)}, \quad \forall i \in C$$

Actualizar el modelo de toma de decisiones:

$$Q_{DM}(j,c,d) = R_{ext}(c,d) + \sum_{k=1}^{|C|} R_{CR}(c'=k)\Pr(c'=k \mid c,d)$$
$$+ \sum_{k=1}^{|C|}\sum_{l=1}^{|E|} R_{ER}(e'=l)\Pr(e'=l \mid c'=k, e=j)\Pr(c'=k \mid c,d),\ \forall j \in E$$

Las diferencias más destacables con respecto a la actualización que llevan a cabo las hormigas cuando simulan son las siguientes:

- No se utilizan modelos para obtener los valores de los parámetros que se necesitan. Por ejemplo, no se usa $\Pr(c_{new} \mid c,d)$ o $\tilde{R}_{ext}(c_{new},d)$ para conocer el estado cognitivo actual o el *reward* externo r_{ext} recibido, respectivamente.
- No se actualiza el modelo de *reward* emocional $R_{ER_k}(\hat{e}_{new})$, ya que es un modelo que mantienen las hormigas y que actualizan en cada paso de la simulación.
- Se actualiza el *reward* externo medio $\tilde{R}_{ext}(c,d)$ con el r_{ext} recibido.
- No se evaporan feromonas.

Con respecto a la propuesta de (Ahn, 2005), que hemos tomado como referente en nuestra aproximación, debemos reseñar la eliminación del operando $\alpha \sum_{k=1}^{|C|}\sum_{l=1}^{|E|} \max_{d'} Q_{DM}(l,k,d')\Pr(e'=l \mid c'=k, e=j)\Pr(c'=k \mid c,d)$ de la fórmula de actualización del modelo de toma de decisiones $Q_{DM}(j,c,d)$. El motivo de esa eliminación se puede entender con la explicación del significado del operando: se encarga de calcular el valor máximo de atracción de cada estado alcanzable desde el par de estados (c, e) para poder añadírselos al valor de atracción de (c, e); estos valores máximos son ponderados por la probabilidad de ser alcanzados y, al valor global de atracción se le aplica un factor a que controla/reduce su influencia sobre el valor de atracción de (c, e). En otras palabras, el valor de atracción de un estado es mayor si desde él se pueden alcanzar estados con valores altos de atracción. De esta explicación se desprende que los autores usan ese operando para "marcar" secuencias de estados que maximizan los *rewards*, uno de los resultados que nosotros obtenemos al utilizar a las hormigas artificiales y por lo que prescindimos de él en la fórmula. Otra razón para no incluirlo es su impacto sobre la eficiencia temporal del algoritmo, ya que implica la realización de recorridos a toda la matriz para calcularlo.

4.3 Caso de estudio

Para comprobar el funcionamiento de nuestras ideas para la toma de decisiones influenciada por emociones necesitábamos un sistema en el que la exhibición de emociones o de un comportamiento emocional (o la percepción de la existencia de éste) tuviese una relevancia importante. El sistema elegido fue eCoology (*an Electronic COOL way of Learning about Ecology*, o una forma divertida de aprender ecología). El objetivo principal de eCoology (Acosta, 2006) es enseñar a niños entre 8 y 12 años de edad a prevenir problemas de salud derivados de una alimentación desequilibrada, de la mala conservación del medio ambiente y de la mala calidad de las relaciones sociales, actuando para ello sobre la educación de los niños de una forma divertida y motivadora usando un "ecosistema híbrido aumentado". Estos sistemas son "híbridos" porque en ellos coexisten entidades tanto naturales como artificiales; son aumentados porque hacen uso de la "Realidad Aumentada" (Azuma, 1997) (Billinghurst, 2002), consistente en añadir gráficos virtuales en tiempo real al campo de visión real de una persona (véase Figura 5), para representar un ecosistema híbrido; y son tratados como "ecosistemas" debido a que ambos tipos de entidades compiten, colaboran y entablan diferentes tipos de relaciones para cumplir sus respectivos objetivos.

Figura 5. Un ecosistema híbrido aumentado de eCoology.

Disponer en eCoology de entidades con un comportamiento emocional es una característica clave para que los niños perciban el ecosistema aumentado como

un entorno de aprendizaje interactivo y atrayente. La expresión visual de comportamientos emocionales puede facilitar los procesos de aprendizaje, a la vez que genera una respuesta positiva en los usuarios que les motiva a mantener la interacción con el sistema. Uno de los ejemplos más claros de comportamiento emocional expresado visualmente tiene lugar cuando las emociones de una entidad influyen en la manera en cómo se mueve en el ecosistema. Por ejemplo, cuando el estado emocional de una entidad hace que quiera acercarse a uno de los niños porque éste alimenta a los animales correctamente, o cuando decide alejarse de determinadas áreas, entidades o niños que no se comportan según sus estados cognitivos y emocionales preferidos.

Para llevar a cabo la tarea de encontrar el camino a seguir por una entidad de un punto a otro del ecosistema eCoology cuenta con un mecanismo de resolución de caminos (*pathfinding*), que explicaremos en el punto que se encuentra a continuación. Este mecanismo devuelve los diferentes caminos alternativos encontrados y se los pasa a una componente de la entidad para que elija uno teniendo en cuenta su estado emocional, tal y como se describe en (Mocholi, 2006). Pero esto implica que las emociones realmente no juegan un papel activo en la búsqueda de un camino. En este punto es donde hemos decidido utilizar nuestra propuesta, y para evaluar su funcionamiento hemos implementado una aplicación que presentaremos en el punto 4.3.2.

4.3.1 Pathfinding en eCoology

El componente encargado de encontrar el camino a seguir por una entidad, y por lo tanto del movimiento que ésta realizará por el ecosistema, está formado por un sistema que mezcla un sistema de partículas con ideas de los algoritmos de resolución de grafos.

Un sistema de partículas puede verse como un sencillo simulador de física en el que a cada uno de los elementos simulados, en este caso las partículas, se le aplica una serie de leyes físicas (Burg, 2000). Las partículas tienen asociadas unas propiedades físicas, como posición, velocidad, masa, etc., y generalmente son tratadas como entes puntuales sin dimensiones espaciales. A partir de las partículas existentes, normalmente generadas por un "emisor" de partículas, el simulador de física se encarga de mantener actualizadas sus propiedades físicas, según las leyes físicas que se haya decidido simular y mientras dure el ciclo de vida de la partícula.

Para que las partículas interaccionen con el entorno que se simula es necesario introducir otros elementos en la simulación, como por ejemplo los deflectores, que son objetos que las partículas no pueden atravesar y contra los que chocan. Se clasifican según su forma (un plano, una esfera, …) y, aunque pueden tener

otras propiedades definidas, tienen asociado un valor que define su coeficiente de rozamiento y que determina cómo se comportan las partículas cuando chocan con su superficie. Otro tipo de elemento a incluir son los atractores, que atraen hacia sí mismos a todas las partículas cercanas. En la Figura 6 podemos ver un ejemplo de sistema de partículas con todos los elementos descritos, a saber, emisor, partículas, deflector y atractor.

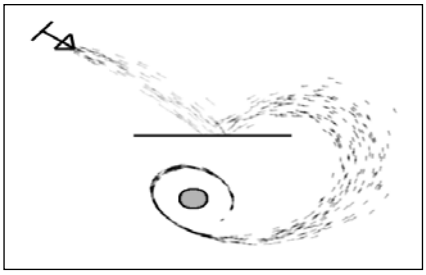


Figura 6. Ejemplo de sistema de partículas.

Con los elementos presentados es posible crear un mecanismo de resolución de caminos bastante simple. Las entidades se modelan como. El punto de destino a alcanzar por una entidad se define como un atractor, lo que provocará que la partícula de la entidad se dirija hacia él. El resto del entorno se modela usando deflectores, lo que permite que las partículas, y por lo tanto las entidades, no atraviesen las paredes. En la Figura 7 podemos ver una representación de cómo funcionaría esta resolución de caminos simplista.

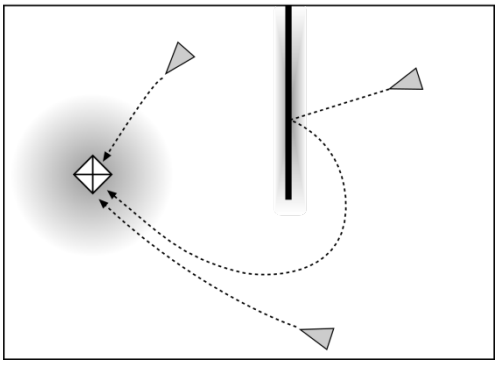


Figura 7. Resolución de caminos basada en un sistema de partículas.

Debido a su simplicidad, este mecanismo provoca que las entidades colisionen y reboten contra las paredes mientras se dirigen a su destino, lo que hace que su comportamiento no sea el deseable para su integración en el espacio aumentado de eCoology. Para resolver este problema se introdujeron varios refinamientos al sistema. En primer lugar se dotó a las entidades con propiedades usadas en la física de vehículos (Eberly, 2003), como son la masa,

la aceleración y la velocidad, lo que resultó en movimientos menos abruptos. En segundo lugar, asignamos dos roles más a los deflectores; por un lado actuarán como repulsores, forzando a las partículas tanto a evitar colisionar con ellos como a esquivarlos en la medida que el resto de fuerzas presentes lo permita, por lo que las trayectorias de las entidades se suavizaron y las entidades dejaron de parecer ciegas ya que ya no tocaban las paredes antes de esquivarlas. Por otro lado, los deflectores son opacos, por lo que las entidades no pueden ver a través. Esto permite generar unas trayectorias más realistas ya que una partícula no se ve directamente atraída o repelida por objetos que no puede ver.

Sin embargo, el objetivo es que las entidades de un ecosistema en eCoology sean capaces de alcanzar su destino, si el escenario que lo define lo permite. En este punto cobra su importancia el otro componente de la mezcla de la que hablamos inicialmente: la resolución de grafos. Para alcanzar el objetivo, al mapa que define el escenario que habitan las entidades se le aplica un grafo, asegurándose que todas las áreas disponen de al menos un nodo en el grafo. Estos nodos actúan como "boyas" y no se utilizan para resolver el camino a recorrer pero facilitan que el sistema de partículas lo haga. Cada boya está unida con una arista en el grafo a todas la boyas que son "visibles" desde su posición en el mapa, es decir, que no están ocultas por un elemento opaco. De esta manera, cuando una entidad busca su destino (una partícula se dirige a su atractor) puede hacer uso de la red de boyas en caso que éste no sea directamente visible. A continuación tenemos explicado en pseudocódigo el funcionamiento descrito dado un destino D para una cierta entidad.

```
Mientras no se alcance D
    Si tiene asignado un atractor que es visible
        Realizar simulación física
    Si no
        Comprobar visibilidad de D
        Si D es visible directamente
            Asignar un atractor ubicado en D
        Si no
            Obtener una boya que dirija a D
        Fin si
    Fin si
Fin mientras
```

En la Figura 8 podemos observar un ejemplo de funcionamiento. En la figura de la izquierda vemos dos partículas representadas por triángulos que comparten el mismo atractor. La que está más la izquierda tiene visión directa del objetivo, por lo que puede realizar la simulación e ir hacia su destino. La otra partícula tiene dos deflectores que dificultan su camino, por lo que deberá hacer uso de las boyas, simbolizadas con la letra "b", para resolver su camino hacia el destino.

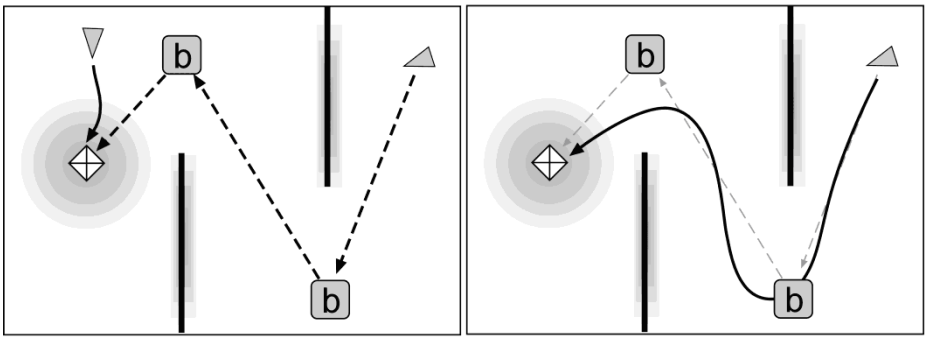


Figura 8. Ejemplo de resolución de un camino.

4.3.2 Pathfinding influenciado por emociones

Para probar la aplicación de nuestra propuesta a este dominio, poder realizar experimentos sin tener que contar con todas las componentes de eCoology y poder visualizar fácilmente tanto el escenario como los caminos realizados por la entidad, desarrollamos una aplicación de escritorio que es una representación 2D de la simulación llevada a cabo por las hormigas y que más tarde exhibirían las entidades de eCoology en el entorno 3D. La aplicación, cuya interfaz de usuario se puede ver en la Figura 9, permite construir interactivamente el escenario (representado como un grid), situar las entidades en él, fijar el número de hormigas a utilizar, y seleccionar si realizar una simulación automática o una interactiva en la que podemos controlar el *reward* externo que la entidad recibe después de cada movimiento. La posición inicial de la entidad queda marcada con una *S* en la celda correspondiente; el destino emocionalmente atractivo se marca con una *F*; y las celdas marcadas con una *M* representan usuarios u otras entidades del ecosistema que hicieron que nuestra entidad alcanzase estados emocionales no deseados en el pasado. En esta implementación el conjunto de decisiones está formado por todos los movimientos que una entidad puede realizar, el grid es la visualización del grafo que representa el mapa del escenario. Los nodos del grafo representan posiciones en el mapa y las aristas los movimientos que llevan de una posición a otra.

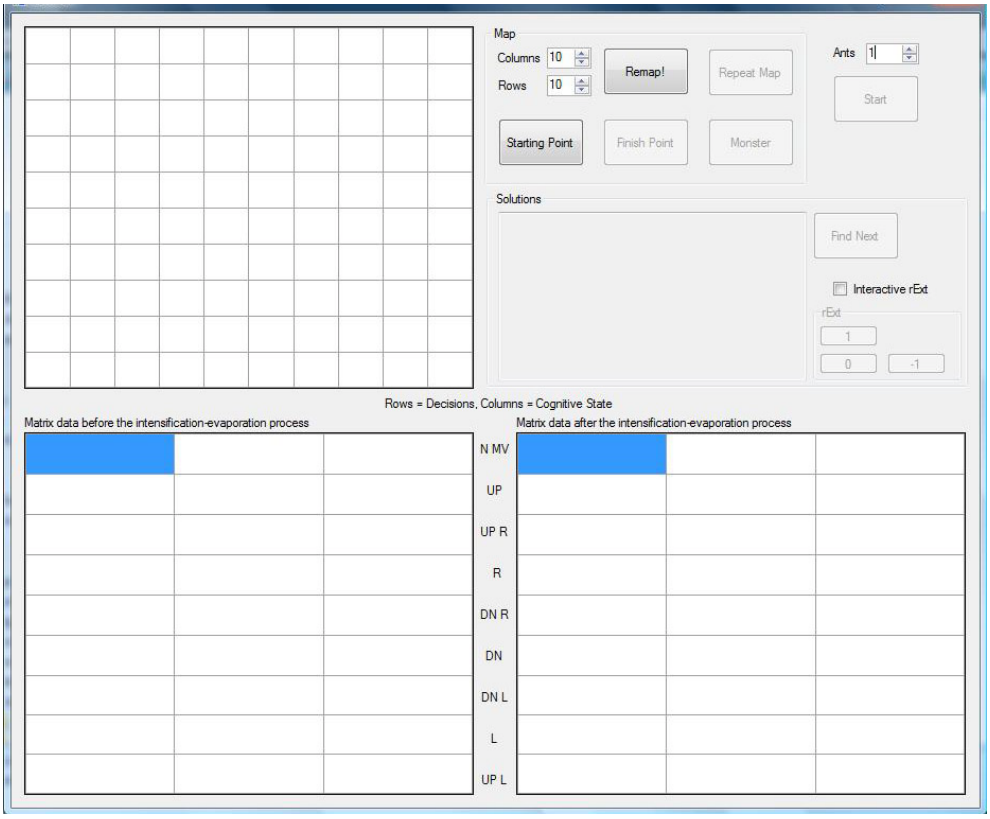


Figura 9. Interfaz de la aplicación de test.

Una vez que hemos construido el mapa, se inicia la simulación y los parámetros del algoritmo de toma de decisiones se inicializan. El valor del *reward* cognitivo r_{CR} para estados cognitivos deseados será 1, y -1 si no son deseados. De igual manera asignamos los valores para $R_{ER}(e)$. Los valores iniciales del modelo de *reward* cognitivo $R_{CR}(c)$ se fijan según r_{CR}, ya que $R_{CR}(c)$ acaba convergiendo a r_{CR}. Si un estado cognitivo c' y un estado emocional e son estados deseados para una entidad, el valor del modelo emocional para esos estados, $\Pr(e' \mid c', e)$, tendrá un valor positivo entre 0 y 1 que representará cuán atractivo es el estado emocional alcanzado. Durante este proceso de inicialización también se asignan los siguientes valores: el *reward* externo r_{ext} que una entidad recibe al alcanzar cada estado cognitivo; M, el número de movimientos en el futuro cercano que las hormigas simularán; el número máximo de iteraciones a realizar; la mínima mejora ε a conseguir, que actuará como criterio de parada, y el ratio de aprendizaje γ, que tomará valores entre 0 y 1. Finalmente, una vez realizada la inicialización de parámetros, la entidad ya

puede recoger sus primeros datos del entorno, como su ubicación o entidades cercanas, que se usarán para establecer el estado cognitivo inicial.

En la siguiente figura podemos observar el resultado final de una ejecución de nuestra propuesta algorítmica aplicada a este dominio tal y como hemos descrito.

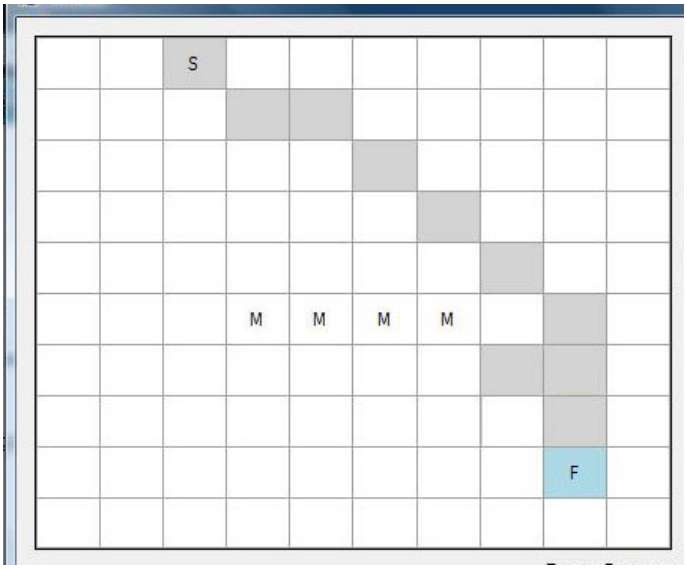


Figura 10. Ejemplo del camino realizado por una entidad con nuestra propuesta.

4.4 Conclusiones

Los algoritmos metaheurísticos son lo bastante genéricos como para ser utilizados en multitud de dominios, lo que por otro lado comporta que no cuenten inicialmente con los mecanismos necesarios para simular emociones que puedan influir en su proceso de búsqueda. Por esta razón hemos expuesto en este capítulo la propuesta *hierarchical network of affective-cognitive decisions* (Ahn, 2005), ya que es la aproximación de la que tomamos los modelos probabilistas a partir de los cuales simular y aprender el efecto cognitivo y emocional de las decisiones que se toman, y que más tarde influirán en el proceso de selección de nuevas decisiones.

También hemos presentado nuestra propuesta de sistema para la toma de decisiones influenciada por emociones utilizando como base algorítmica un sistema de colonias de hormigas, y como mecanismo de generación de la influencia cognitiva y emocional hemos partido de la propuesta de (Ahn, 2005) y hemos realizado una serie de modificaciones para adaptarla a las

características de un sistema de colonias de hormigas (por ejemplo, hemos añadido algunos modelos y una fórmula que es el objetivo a maximizar por las hormigas, y utilizamos la componente $Q_{DM}(\hat{e},c,d)$ como la matriz de atracción y feromonas de las hormigas). Las principales ventajas de nuestra propuesta incluyen la capacidad de abordar espacios de búsqueda grandes (conjunto de decisiones y estados cognitivos y emocionales) gracias al uso de una metaheurística, la mayor eficiencia que proporciona la paralelización intrínseca del uso de hormigas artificiales, y la previsión de secuencias de decisiones que puedan reportar el mayor beneficio en el futuro cercano más probable.

Al final del capítulo hemos presentado eCoology, un juego educativo para niños que utiliza realidad aumentada para mostrar entidades virtuales que tienen emociones en un entorno real, el caso de estudio en el que hemos probado nuestra propuesta. El dominio de aplicación elegido ha sido la resolución de caminos que las entidades deben realizar para desplazarse por el escenario desde su ubicación actual a un destino y teniendo en cuenta sus emociones. Después de presentar qué es eCoology, hemos explicado cómo funciona el mecanismo de *pathfinding* implementado en eCoology basado en un sistema de partículas y algunas ideas de algoritmos de grafos, y por último hemos presentado la implementación de nuestra propuesta para este dominio y la aplicación con la que hemos comprobado su correcto funcionamiento.

"The best of prophets of the future is the past."

Lord Byron. Diario personal. 28 de enero de 1821.

Capítulo 5

Una Propuesta de Algoritmo OCH para la Toma de Decisiones en base a Información Semántica

5.1 Introducción

Al inicio de este documento hemos visto lo necesario que resulta para los ambientes inteligentes tener en cuenta el contexto del entorno para adaptarse adecuadamente y exhibir un comportamiento que se pueda considerar inteligente. Pero para que los distintos dispositivos y sistemas presentes en el ambiente inteligente puedan procesar la información relativa al entorno en el que se encuentran, es necesario que esa información sea modelada adecuadamente para que pueda ser compartida y procesada de manera sencilla, así como aumentada con cualquier nueva información derivada por cualquier sistema o dispositivo del ambiente a partir de la información inicial.

Tal y como exponíamos en el punto 2.2, el tratamiento semántico de la información que se pueda capturar o derivar del entorno facilita que las máquinas puedan procesar e inferir nuevo conocimiento del entorno, además de mantener una estructura apta para que los seres humanos sigan pudiendo manipular la información del contexto del entorno. La manera de conseguir que sistemas heterogéneos sean capaces de compartir entre sí información contextual y que además puedan procesarla para generar nueva información se basa en el uso de metadatos para describir la situación contextual del entorno en un momento dado y de ontologías para dotar de uniformidad, jerarquía y semántica a los metadatos disponibles para que los diferentes componentes del ambiente describan y comprendan el contexto del entorno en el que se encuentran. No obstante, en esta tesis consideramos que el hecho de considerar sólo la información contextual relativa al momento actual limita las posibilidades de adaptación de los ambientes inteligentes y, por lo tanto, de exhibir un comportamiento inteligente. Es por ello que, también en el punto 2.2, planteábamos el uso por parte de los ambientes inteligentes de la

información contextual histórica, de manera que fuesen capaces de aprender y reconocer secuencias de situaciones contextuales para así poder tomar las acciones oportunas bien para evitar alcanzar una situación de contexto similar, o bien para facilitar alcanzarla.

La metaheurística OCH es lo suficientemente genérica como para ser aplicable a infinidad de problemas, como ya habíamos comentado, pero ello implica que debe realizarse una esfuerzo de adaptación para diseñar un algoritmo OCH que aborde cada dominio de problema. Por esta razón, y puesto que en la literatura no existen propuestas de algoritmos OCH similares, a continuación presentamos nuestra propuesta de algoritmo OCH para la predicción de situaciones contextuales a partir de información semántica histórica contextual.

5.2 Algoritmo OCH semántico para la predicción de situaciones contextuales

En este punto presentamos una propuesta algorítmica basada en OCH que se diferencia de otras como el algoritmo ACO-OP (Liang, 2003) en que es capaz de utilizar descripciones semánticas de conceptos para asignar dinámicamente una puntuación o beneficio (*score)* a los nodos del espacio del problema, que representan entidades del dominio del problema por medio de valores de los términos de la ontología utilizada; en el caso de los ambientes inteligentes, un nodo representaría una cierta situación contextual y las aristas entre nodos representarían una relación de secuencialidad entre dos situaciones contextuales. Este proceso de asignación semántica de *score* se realiza cada vez que se quiere resolver una instancia de problema teniendo en cuenta unas determinadas condiciones. Estas condiciones se expresan como un conjunto de valores para algunos términos de la ontología utilizada y que pueden ser vistos como los términos de una consulta. Cada término de la consulta tiene asociado un valor que se utiliza para ajustar la importancia, la relevancia, de un término sobre el resto de términos de la ontología. Entonces, a partir del espacio de búsqueda completo del problema y teniendo en cuenta los términos de la consulta, creamos un espacio de búsqueda más reducido y restringido a los términos de la consulta, a cuyos nodos habremos realizado la asignación semántica de beneficio. Y es sobre este espacio de búsqueda sobre el lanzaremos las colonias de hormigas para que nos devuelvan las secuencias contextuales que contengan las situaciones contextuales futuras más probables para los valores contextuales del entorno representados por los términos de la consulta. A continuación plasmamos en pseudocódigo esta breve explicación del funcionamiento de nuestra propuesta para explicarlo en mayor detalle en el resto de capítulo.

Cargar ontología

Leer consulta

Crear el espacio de búsqueda a partir de los términos de la consulta

Asignar *score* a los nodos del espacio creado

Aplicar algoritmo ACO-OP

Devolver conjunto ordenado de secuencias contextuales

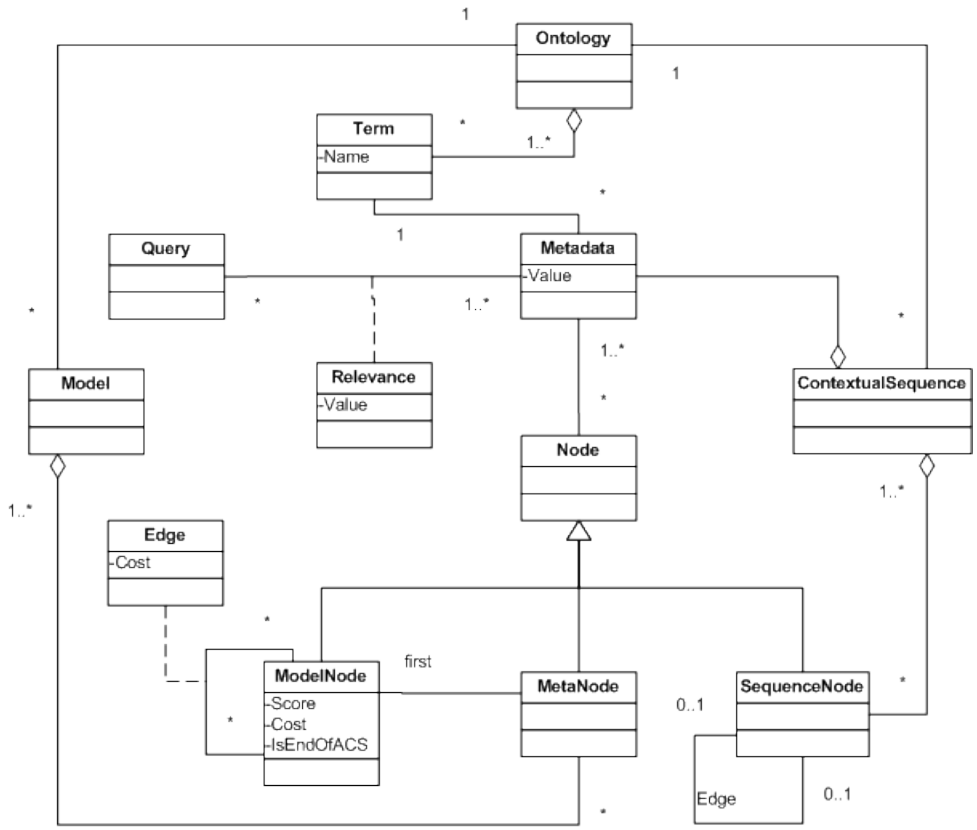


Figura 11. Modelo conceptual de la propuesta.

El modelo conceptual de nuestra propuesta puede verse en la Figura 11 y que pasamos a explicar a continuación. Los términos disponibles para describir el contexto, y que están definidos en la ontología que se esté utilizando, son instancias de la clase Term. Cada uno de los posibles valores de un término es una instancia de la clase Metadata. Una consulta, una instancia de la clase Query, se construye a partir de uno o más metadatos, y cada uno de ellos llevará asociado un valor de Relevance. El algoritmo de colonias de hormigas

trabaja con instancias de la clase Node, que utiliza metadatos para representar su contenido, sus atributos. En el diagrama podemos ver que hay tres especializaciones de la clase Node: la primera es la clase SequenceNode y la utilizamos para representar un conjunto de datos capturados (por ejemplo hora, fecha, actividad, temperatura, localización, etc.) para una situación contextual (el estado del contexto del entorno para un instante de tiempo determinado). Ya que la captura de información contextual se realiza a un cierto ritmo, cada nueva instancia de SequenceNode se enlaza con la inmediatamente anterior para ir dando forma a un objeto de la clase ContextualSequence. Un ejemplo de instancia de esta clase, tomando como dominio del problema el escenario planteado hacia el final del punto 1.1, podría verse como un desplazamiento realizado por Gonzalo con su vehículo, junto a su familia, para visitar a sus abuelos y el resto de información contextual relacionada. Una vez que se ha capturado una secuencia contextual, el sistema la añade a un repositorio que representa el modelo histórico del contexto del entorno específico en el que se encuentra el ambiente inteligente y que está representado en la Figura 11 por la clase Model. Una instancia de Model guarda todas las secuencias contextuales "aprendidas" a lo largo del tiempo de actividad del ambiente. En el escenario de ejemplo planteado, puesto que Gonzalo generalmente repetirá las mismas rutas con mínimas variaciones en cuanto a la información contextual asociada, resulta necesario poder determinar si una secuencia contextual que va a ser añadida al modelo estaría ya representada por las secuencias contextuales previamente almacenadas. En este punto entra en juego la clase MetaNode. Un objeto MetaNode contiene todos los metadatos compartidos por todos los nodos de una secuencia contextual.

El proceso de añadir una nueva secuencia contextual al modelo empieza por el hecho de determinar si alguno de los objetos MetaNode existentes representa ya a esa secuencia. De no encontrarse ningún MetaNode que la represente, se considera que se trata de una nueva secuencia contextual que debe "aprenderse" por lo que el proceso de añadirla empieza con la creación de un nuevo MetaNode y, siguiendo el orden, cada SequenceNode es convertido a una instancia de ModelNode, la tercera especialización de Node. A medida que los ModelNode son creados y se enlazan con los ya existentes, el coste asociado al enlace que los une se calcula a partir de la distancia semántica que haya entre la información contextual de ambos nodos.

Por otro lado, si se halla un MetaNode cuyos metadatos representen una información contextual similar a la de la nueva secuencia contextual, entonces el proceso de adición se encargará de combinarla con el modelo. El proceso de combinación o fusión de la nueva secuencia analiza cada uno de sus nodos para tratar de hallar un ModelNode de la secuencia ya aprendida que tenga unos metadatos similares (dentro de un cierto margen). Si el proceso encuentra

un ModelNode con esas características, actualiza sus metadatos siguiendo una aproximación estadística de manera que el nodo actualizado puede verse como el centroide de un cluster de objetos SequenceNode que tienen metadatos parecidos. En caso de que un nodo de la secuencia contextual fuese muy diferente del ModelNode que le correspondiese (lo cual podría deberse a una desviación de la ruta habitual, por ejemplo), se crearía un nuevo ModelNode para representarlo. Por consiguiente, un ModelNode puede estar enlazado con más de un ModelNode anterior y/o posterior. Con este mecanismo permitimos que se puedan representar ligeras variaciones de una misma secuencia contextual. Este proceso de adición de una nueva secuencia contextual puede describirse en pseudocódigo como sigue:

```
Obtener los metadatos comunes a todos los nodos de la secuencia
Crear un MetaNode con los metadatos comunes
Buscar un MetaNode similar en el modelo histórico
Si lo hay
  Repetir
    Buscar un ModelNode similar al SequenceNode
    Si lo hay
      Actualizar el ModelNode con los metadatos del SequenceNode
      Actualizar el coste asociado a los enlaces
    Si no lo hay
      Convertir el SequenceNode en un ModelNode
      Enlazar el nuevo ModelNode con el anterior
      Calcular el coste del enlace
  Hasta procesar todos los SequenceNode
Si no lo hay
  Repetir
      Convertir el SequenceNode en un ModelNode
      Enlazar el nuevo ModelNode con el anterior
      Calcular el coste del enlace
  Hasta procesar todos los SequenceNode
  Enlazar el primer ModelNode con el MetaNode
  Añadir el MetaNode al modelo
```

Puesto que ya hemos explicado con anterioridad el funcionamiento del algoritmo ACO-OP, y los pasos más importantes y complejos de la propuesta son la creación del espacio de búsqueda y la asignación de *score* a los nodos de ese espacio, a continuación explicaremos cómo se realizan ambos pasos.

5.2.1 Creación del espacio de búsqueda

El espacio de búsqueda global del problema lo conforman todas las secuencias contextuales almacenadas en el modelo histórico del contexto. La cantidad de metadatos del contexto capturados y asociados a cada nodo de una secuencia puede ser igual al número de términos de la ontología. También puede suceder que haya secuencias que no tengan ningún metadato que se corresponda con los términos respecto a los cuales se quiere realizar una búsqueda. Es por esta razón que nuestra propuesta plantea la creación de un espacio de búsqueda a partir de los términos de la consulta. De esta manera se consigue tanto que todas las soluciones tengan una alta correspondencia con la consulta, como mantener un buen rendimiento al reducir el espacio de búsqueda ya que usamos la consulta para "filtrar" qué nodos se convertirán en nodos del grafo.

Una consulta Q puede representarse como un conjunto de tuplas de tres elementos: un término de la ontología, un valor para ese término y un valor que representa el factor de relevancia. El papel del factor de relevancia es fijar la importancia de unos términos de la ontología sobre el resto.

$$Q = \{(term_j, value_j, relevance_j) \mid term_j \in T,\ value_j \in Domain(term_j),\ relevance_j \in [0,100]\}$$

A partir del conjunto de términos de la consulta, $\{term_1, term_2, \ldots, term_n\}$, se seleccionan para formar parte del grafo sólo aquellos nodos que tengan al menos uno de los términos de la consulta entre sus metadatos. Para cada uno de estos nodos se creará un nodo en el grafo que tendrá como metadatos todos los pares `(term, value)` en los que el término aparezca en la consulta. De esta manera todos los nodos no relevantes son omitidos y el espacio de búsqueda se reduce.

5.2.2 Asignación dinámica de *score*

Una vez creado el grafo con los nodos que representará el espacio de búsqueda para la consulta facilitada, se realiza el proceso para asignar un beneficio o *score* a cada nodo que necesita de los siguientes elementos:

- Todos los nodos del modelo pueden definirse como una tupla (ID_i, O_i) en la que ID_i es un identificador único y O_i es un conjunto de pares $\{(term_k, value_k) \mid term_k \in T,\ value_k \in Domain(term_k)\}$ que representan valores asociados a términos de T de una cierta ontología.

- Una consulta Q definida como un conjunto donde

  $Q=\{(term_i, value_i, relevance_i) | term_i \in T, value_i \in Domain(term_i), relevance_i \in [0,100]\}$

- El conjunto M con los pares $(term_i, value_i)$ que coinciden con la consulta

  $M=\{(term_i, value_i) \in Q \mid \exists (term_i, value_i) \in O_i\}$

- Una métrica para la distancia semántica sobre un término definida como una función

  $d_{termk} : Domain(term_k) \times Domain(term_k) \rightarrow [1,100]$

Dada una consulta Q, una colección de métricas de distancia semántica sobre los términos de T, y dado el conjunto M, el beneficio S_i asociado al nodo del grafo que representa la tupla (ID_i, O_i) se calcula con la siguiente fórmula:

$$S_i = \sum_{i=1}^{|M|} \frac{d_{term_i}(value_i, value_i) \cdot relevance_i}{|M|}$$

Reseñar que el valor del beneficio S_i es proporcional a la distancia semántica existente entre los términos que forman parte de la consulta y los metadatos que forman O_i. El conjunto de funciones para evaluar las distancias semánticas para términos se obtiene a partir de la intervención de un experto del dominio, quien es capaz tanto de determinar un valor de distancia fijo entre pares de valores de un término de la ontología (generalmente para términos de naturaleza no cuantitativa), como de crear una función de evaluación que calcule la distancia semántica entre los valores.

5.3 Casos de estudio

En este punto presentaremos 2 dominios de aplicación distintos en los que hemos aplicado la propuesta algorítmica descrita en este capítulo, así como los resultados de los experimentos realizados en cada caso.

5.3.1 Generación automática de listas de reproducción musicales

Una de las características más importantes de la propuesta de algoritmo OCH presentada en este capítulo es su capacidad para utilizar datos heterogéneos pertenecientes a una ontología, que describe semánticamente el contexto del problema a abordar, en su funcionamiento. Por esta razón, y porque la utilización de información contextual histórica se sustenta en la característica mencionada, el experimento que exponemos en este punto fue el primero de

los realizados para comprobar las ideas planteadas y se restringe a la característica mencionada, es decir, a la utilización de información semántica por un algoritmo OCH.

El problema elegido fue la creación de listas de reproducción que se ciñesen, en la medida de lo posible, a las especificaciones o preferencias dadas por un usuario, añadiendo siempre como criterio de parada adicional una duración máxima en minutos para la lista creada. El siguiente es un escenario posible de uso, planteado en (Pohle, 2005), en el que se muestra la utilidad de un sistema con estas características.

Una persona sale de su casa temprano para hacer su carrera matutina diaria. Siempre hace deporte con música, pero no le gusta escuchar la misma música una y otra vez. Así que tiene una gran colección de música en su reproductor portátil. Elige la música que se ajusta a su estado de ánimo actual, sin tener que retrasar el ejercicio, navega por la colección mientras está corriendo. Después de oír la canción que ha elegido, el reproductor seguirá con temas similares. Al final de la ruta, elige de improviso algunos sonidos relajantes mientras trota hacia casa y el reproductor añade más temas similares a la reproducción.

La idea que expone este escenario es el interés que entraña para un usuario el disponer de un mecanismo que le libere de tener que ir seleccionando los álbumes que quiere escuchar cuando sabe que la música que le apetece escuchar se ciñe a ciertos criterios.

Otro posible escenario de uso sería la creación de una lista de reproducción por parte de un usuario de cara a utilizarla en una fiesta o reunión con una temática determinada. Así, por ejemplo, si un usuario quisiese preparar la música para una fiesta cuyo tema fuese, por ejemplo, "los años 60" y que tiene pensado hacer durar ocho horas, tan sólo tendría que realizar una consulta con estos datos y podría por tanto despreocuparse de la música de esa reunión, sabiendo que en todo momento ésta se ceñirá, en la medida de lo posible, a la temática correspondiente.

El repositorio sobre el que trabajamos tiene como unidad esencial álbumes musicales. Al contrario que en otros trabajos se ha utilizado a los álbumes como objeto sobre el que tratar en lugar de a las canciones puesto que el objetivo esencial de las listas de reproducción que obtendremos como resultado sería el segundo que ha sido explicado en el punto anterior. Ya que se pretende que estas listas de reproducción tengan una duración de varias horas, y en un afán de tener cierta homogeneidad entre las distintas canciones, hemos seleccionado los álbumes para ser la unidad de los elementos del repositorio, que en este experimento contiene 71 títulos. Este es sólo el enfoque que hemos

dado para este caso de estudio, ya que nuestra infraestructura permite el uso de cualquier tipo de datos. Para cada uno de los álbumes hemos seleccionado tres tipos de metadatos distintos: el año de lanzamiento, el género y el artista.

- El primer metadato hace referencia al año de publicación del disco y el rango de valores que toma en nuestro caso va desde 1966 hasta 2007.
- Un género musical es una categoría que reúne temas musicales que comparten ciertos criterios. Estos criterios pueden ser musicales, como por ejemplo ritmo, instrumentación, o basarse en características tales como la región de origen, período histórico, o aspectos sociales y culturales. Los géneros pueden dividirse en subgéneros, pero para nuestro estudio haremos uso del término género tanto cuando hablemos de géneros como cuando estemos tratando subgéneros. Además, si bien podría darse la situación de que un álbum determinado pudiese ser interpretado como perteneciente a varios géneros, en nuestro caso hemos simplificado la labor dando un solo género a cada álbum.
- El artista es sencillamente el/los intérpretes del disco. Es el encargado de dar su nombre al álbum, ya sea por haber compuesto las canciones que en él se interpretan, por haberlas interpretado también, o por haber realizado la mezcla de la sesión que contiene el disco (sería el caso de algunos de los discos cuyo artista es un DJ).

Como ya hemos comentado con anterioridad, nuestra aproximación hace uso de funciones de evaluación para calcular la distancia semántica entre dos valores de un mismo término de la ontología. Concretamente, las funciones de evaluación que utilizamos devuelven un valor numérico entre 0 y 100 que representa cuán diferentes son dos valores de un mismo término. A partir de los valores devueltos por las funciones de evaluación se calcula tanto el *score* de los nodos que forman parte del espacio de búsqueda, como el coste de la arista que une dos nodos y que representa el grado de similitud que hay entre ellos: a mayor similitud, menor es la distancia. De esta manera, cuando se está calculando el *score* de un nodo para una determinada consulta, cuantos más metadatos del nodo sean similares a los términos de la consulta y en un mayor grado, mayor será el *score* de ese nodo. Así que estamos primando la proximidad semántica entre los elementos del repositorio (los álbumes) y los elementos de la consulta (preferencias musicales) especificados por el usuario.

La función de evaluación para el metadato del año de publicación es bastante simple: teniendo en cuenta el número de años cubierto por la colección musical del usuario, se calcula la distancia aritmética entre dos años y se aplica una regla de tres tomando como valor que represente 100 el de la máxima

diferencia entre elementos de la colección. Para la definición de las funciones de evaluación de la distancia semántica para los términos género y artista se consultó a expertos musicales en aras de simplificar el proceso, aunque otra forma viable de definirlas hubiese sido utilizar algún método estadístico, como por ejemplo analizar el comportamiento de los usuarios al comprar o escuchar música.

5.3.1.1 Resultados experimentales

Para evaluar nuestro sistema realizamos varias consultas y estudiar el comportamiento de dos variables: frecuencia y saturación. La frecuencia es el porcentaje de la solución que se encuentra dentro del rango delimitado en lo que a distancia con el criterio de consulta se refiere. Así, por ejemplo, si tenemos que la solución tiene 10 álbumes y 5 de ellos tienen una distancia igual a cero con respecto al criterio de la consulta mostrado, la frecuencia para este rango (rango de distancia cero) será de 0.5 (un 50% de los álbumes de la solución tienen distancia cero con respecto al criterio de la consulta).

La saturación es en cambio una media en relación a los álbumes de cada rango. Así la saturación es el porcentaje de los discos totales de un rango dado con respecto al criterio de la consulta mostrado, que aparecen en la solución. Así si en el rango de distancia de 50 a 59 encontramos para un criterio dado 20 discos, y 5 de ellos aparecen en la solución, la saturación total será 0.25 (un 25% de los discos de distancias entre 50 y 59 con respecto al criterio de la consulta seleccionado aparecen en la solución).

A todas las consultas realizadas se les asignó un valor común de tiempo máximo de 1200 minutos. La idea tras este valor era el tener siempre soluciones que no pudiesen ser presentadas con valores que se ciñesen a los criterios de la consulta de forma exclusiva por el hecho de tener la totalidad de éstos una duración conjunta inferior a estas veinte horas. Dicho de otra forma, queríamos evitar tener soluciones en las que tan solo los criterios seleccionados hiciesen aparición.

En nuestro estudio hemos realizado cinco consultas distintas. Todas ellas se han ejecutado un mínimo de cinco veces para facilitar en las gráficas un promedio de resultados obtenidos, que se pueda ajustar de la mejor forma posible a los resultados reales que se obtendrían a través del sistema que hemos creado. A continuación veremos cada una de las consultas realizadas con los gráficos correspondientes a frecuencia y saturación. Además de mostrar los datos citados, en las gráficas también aparecerán los promedios de duración de los álbumes separados por rango puesto que, como comentaremos más

adelante, la duración resulta un factor relevante a la hora de estudiar las soluciones devueltas.

5.3.1.1.1 Consulta: Folk = 100

La primera consulta que realizamos consta de un solo factor, género = folk y tiene una relevancia de 100. Cabe destacar que la relevancia sólo es significativa cuando hay más de un factor en la consulta y por tanto los resultados son de idéntica índole sea cual sea la relevancia señalada en una consulta de un solo factor.

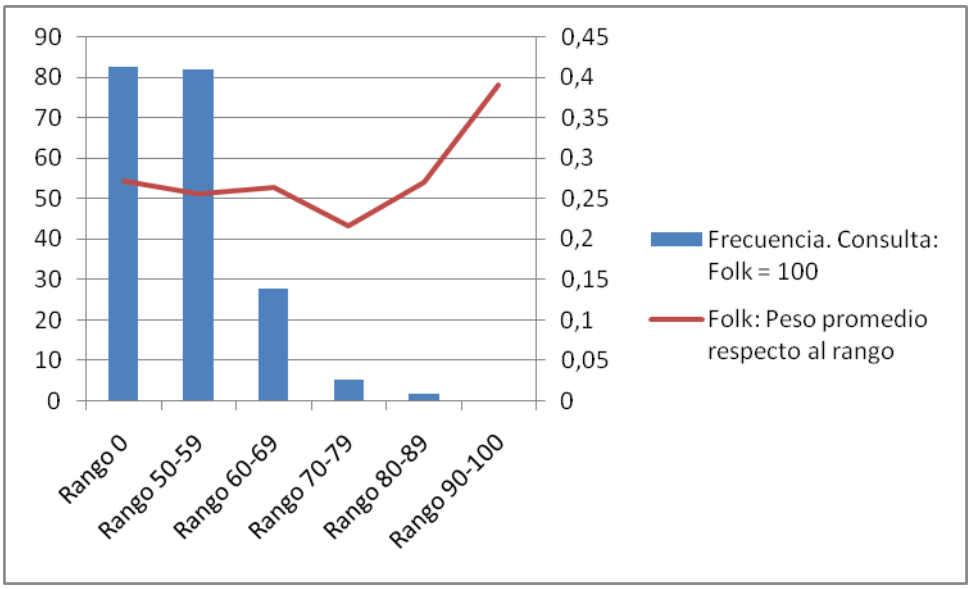


Figura 12. Frecuencia. Consulta: Folk = 100

Como comprobamos en la Figura 12, la frecuencia es muy similar tanto en el rango 0 como en el rango 50-59 y es en ambos casos ligeramente superior al 40%. El hecho de que la frecuencia no sea mayor para aquellos álbumes que tienen un rango 0 es debido a que la cantidad de álbumes de rango 0 (nueve en este caso) es muy inferior al número promedio de álbumes devueltos por las soluciones (más de 20). Con lo cual es imposible que la frecuencia para esta consulta del rango 0 supere el 45% que resulta de dividir nueve entre veinte.

Sí es cierto que puesto que las 17 ocurrencias de discos con rango 50-59 sumadas a las nueve de rango 0 hacen un total de álbumes superior al total devuelto por las soluciones, la frecuencia con la que los álbumes devueltos

pertenecen a alguno de estos dos rangos, que supera el 80% no es todo lo grande que ésta podía ser.

Por último, de la gráfica se desprende también que, como era de esperar, a medida que aumenta el rango, la frecuencia con que álbumes de éste aparece será mucho menor, llegando a un 0% para el rango 90-100.

La línea que marca el peso promedio en este rango no parece relevante por el hecho de que las grandes diferencias se encuentran en rangos muy alejados del criterio mostrado, aunque sí que podría haber esta misma situación ayudado a que los rangos finales desaparezcan de la solución.

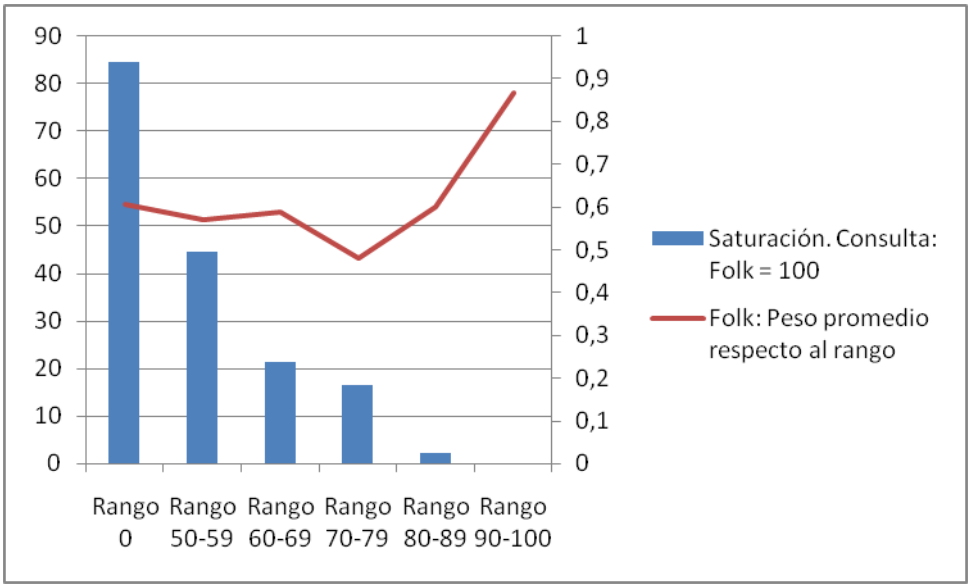


Figura 13. Saturación. Consulta: Folk = 100.

En lo que a la saturación respecta, varias conclusiones pueden sacarse viendo la Figura 13 y una de ellas es bastante importante ya que la veremos reflejada en posteriores resultados.

Como vemos la saturación para el `Rango 0` sobrepasa el 90%, pero no llega al 100% a pesar de haber sólo 9 discos de este rango y obtener soluciones con el doble de álbumes. Esto se debe a que uno de los álbumes del género `folk` tiene una duración muy superior a la del resto de álbumes del mismo género.

El problema con la duración, que viene a ser el peso para el algoritmo de hormigas, es que a la hora de calcular el itinerario (lista de reproducción en

nuestro caso) de mejor puntuación un disco que doble en duración a otros dos sólo tendrá preferencia con respecto a estos otros dos si su puntuación es mayor que la suma de las puntuaciones de los dos discos con la mitad de duración.

Así si tenemos un disco con duración 120 y con una puntuación, pongamos de 5000, otro disco con una duración de 60 y una puntuación de 2500 y un tercero de duración igual a 60 también y de puntuación 2600, aunque el primer disco supera en prácticamente el doble la puntuación de los otros dos, el algoritmo tomaría a los dos discos de 60 minutos a la hora de dar la solución por encima del de 120 ya que con igual duración conjunta, los discos de una hora tienen una puntuación combinada de 5100, superior a los 5000 del disco de dos horas.

Una vez esto está claro, no es de extrañar que en algunas soluciones el disco de mayor duración de `folk`, que supera el promedio de duración de los discos en casi un 90%, no esté presente, dando lugar a una saturación cercana al 100%, pero no exactamente del 100%.

El resto de la gráfica vuelve a demostrar que los resultados que podrían ser intuidos han sido obtenidos, dando así una saturación muy inferior, cercana al 50% a los discos de rango 50-59, y menor aún en el resto de los casos.

5.3.1.1.2 Consulta: Trance = 100

La segunda consulta sobre la que hemos trabajado consta también de un solo factor, género = Trance con relevancia 100. Hemos hecho dos consultas tan similares con el fin de demostrar empíricamente la relevancia que tiene el peso (duración) de los álbumes en el cálculo de las listas de reproducción.

Como se puede comprobar en la Figura 14, los álbumes pertenecientes al rango 0 tienen una duración promedio cercana a los 120 minutos, muy por encima del resto de rangos que fluctúan entre los 40 y los 60 y pocos minutos. Esto significa que en promedio un álbum del rango 0 dura el doble que uno del resto de rangos, y por lo tanto el score debe ser el doble para que a efectos de interés para el algoritmo sea mejor que dos álbumes de los otros rangos.

El número de álbumes de trance es de ocho, y el promedio de álbumes de las soluciones es de 17, con lo cual la relación entre número de álbumes del género y el total de álbumes de las soluciones es prácticamente idéntica al de la consulta folk = 100, por esto podemos intuir que el motivo por el que la

frecuencia baje más de un 5% será seguramente debido al aumento del peso de los álbumes para el rango 0.

En cualquier caso el comportamiento continúa siendo similar al comportamiento en la consulta anterior, si bien el rango 70-79 que apenas participaba de las soluciones para folk, sí que tiene más de un 10% de representación en las soluciones de trance = 100, seguramente debido al hecho de que sea el rango con menor peso de todos.

Un total del 70% de los álbumes de la solución pertenece a los dos primeros rangos, un 12% menos que para la consulta anterior, pero aún un porcentaje bastante alto de la solución. Con el estudio de estas dos consultas podemos por tanto confirmar la relevancia de la duración, puesto que la media para trance es 118 y para folk de tan solo 55.

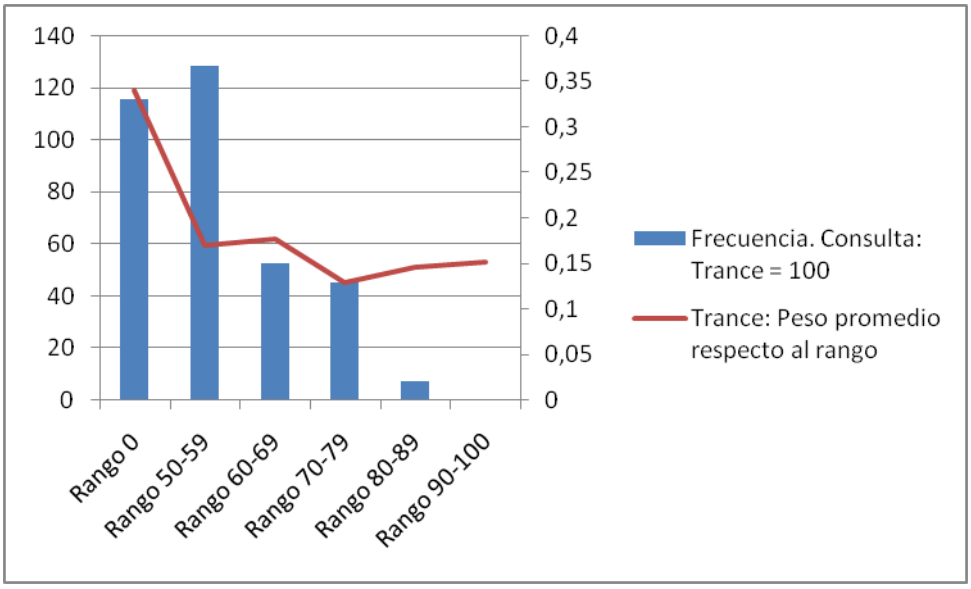


Figura 14. Frecuencia. Consulta: Trance = 100

En lo que a la saturación se refiere, viendo la Figura 15, comprobamos que la relevancia del peso se muestra con más intensidad si cabe. Mientras que la saturación para el rango 0 de la consulta de `folk` era de un 94%, para esta consulta el porcentaje baja drásticamente hasta un 72%, 22 puntos. De los ocho discos del género, en las consultas con mejores resultados sólo aparecen seis, siendo en los peores casos sólo el 50% de ellos los que forman parte de la solución. Además son los discos con mayor duración los que más se resisten a la hora de aparecer en las soluciones.

Es tanto el descenso de la saturación, que para el rango siguiente, rango 50-59, el índice de saturación es sólo un punto menor, 71%, y para el resto de rangos la saturación siempre es mayor a la saturación de la consulta anterior.

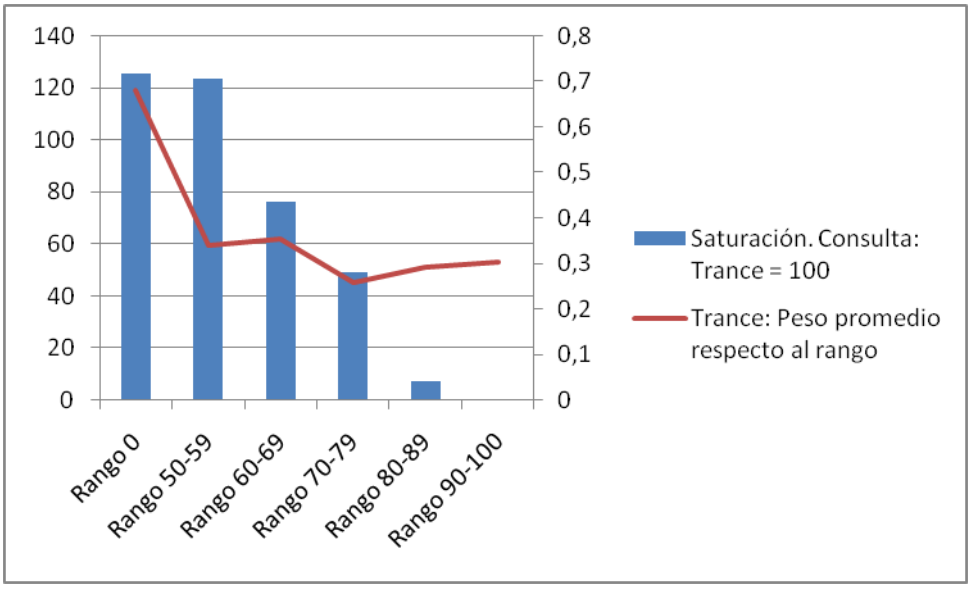


Figura 15. Saturación. Consulta: Trance = 100

5.3.1.1.3 Consulta: Folk = 50; Trance = 50

En esta primera consulta con dos términos hemos tratado de comprobar qué sucede cuando elegimos dos factores sobre el mismo criterio para realizarla, y además le damos la misma relevancia a ambos. En este caso los factores han sido elegidos sobre el tipo de criterio género y sus valores son los dos valores utilizados en las consultas anteriores, folk y trance. Ambos con una relevancia de 50.

Estos dos géneros han sido elegidos por varios motivos. El primero de ellos es que son los géneros más representados en nuestra base de datos con 9 ocurrencias para folk y 8 ocurrencias para trance. El segundo es su disimilitud. La distancia entre ambos es de 95, por lo tanto los álbumes de un género estarán en el rango 90-100 de la gráfica del otro género. El tercer motivo es ver, si cabe una vez más, cómo el hecho de que trance tenga una duración promedio que dobla la de folk hace que su frecuencia en las soluciones, así como su saturación se vea afectada negativamente respecto a la del otro género.

Para estudiar los resultados mostraremos las mismas gráficas que en las consultas anteriores, pero haciendo una gráfica de frecuencia y otra de saturación distinta para cada uno de los factores de las consultas. Así veremos primero los resultados desde el punto de vista del género folk, para posteriormente contrastarlos con los resultados obtenidos desde el punto de vista del género trance.

Si observamos la frecuencia desde el punto de vista del género folk en la Figura 16, vemos resultados similares a los obtenidos cuando en la consulta tan solo teníamos al género folk, con una única, pero importante excepción. Ha habido una disminución importante en la frecuencia de los rangos más cercanos al rango 0. Empezando con este mismo rango, que ha sufrido una disminución en frecuencia de un 0.5%, el rango 50-59 ha bajado un 6,3% y el rango 60-69 un 12,9%. Un total de un 19,7% de disminución en frecuencia que ha ido prácticamente en su totalidad (un 19,4%) a parar al rango 90-100, que es dónde se sitúa el género trance con respecto al género folk.

Así podemos concluir que los resultados son satisfactorios en la medida que el añadir un segundo factor del mismo tipo de criterio sólo influye en soluciones obtenidas anteriormente de forma relevante en los rangos en los que se sitúa el nuevo criterio de búsqueda.

Seguramente el hecho de que el rango en el que se encuentra el segundo factor sea, con diferencia, aquél que más peso tiene en promedio, ha hecho que la frecuencia del rango finalmente sea de menos de un 20%, pudiendo haber sido mayor si los promedios de pesos fuesen similares.

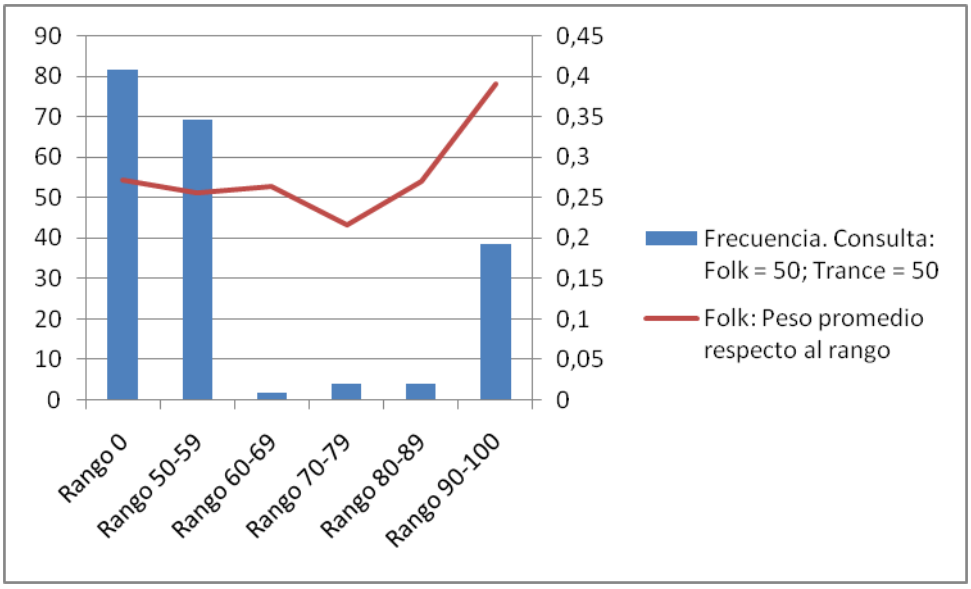


Figura 16. Frecuencia. Consulta: Folk = 50; Trance = 50

Cuando tomamos la saturación vista desde el criterio folk, comprobamos para empezar, como muestra la Figura 17, que hay grandes variaciones con respecto a los resultados obtenidos para la consulta folk = 100, variaciones mucho más significativas que aquellas observadas al estudiar la frecuencia.

Es cierto que folk sigue teniendo una saturación alta, sólo cinco puntos por debajo de la saturación del rango 0 en la consulta folk = 100, pero los rangos cercanos sufren una gran caída debido a la aparición de tantos discos pertenecientes al rango 90-100. Así el rango 50-59 tiene una caída del 9.4%, el rango 60-69 del 21.1% y el rango 70-79 del 5%. El hecho de que el rango 70-79 tenga mayor representación tiene que ver exclusivamente con el bajo número de álbumes que tiene este rango, tan sólo 3, con lo que su saturación es alta aún teniendo en promedio menos álbumes en las soluciones que el rango 60-69.

En el lado positivo de la balanza se sitúan los rangos más lejanos, que se ven influidos por la situación del segundo factor de la consulta (como ya hemos mencionado en el rango 90-100). El último rango pasa de tener un 0% de saturación a tener un 17.3%, pese a su alto peso promedio. Además el rango 80-89 sube hasta el 5%, justo el doble.

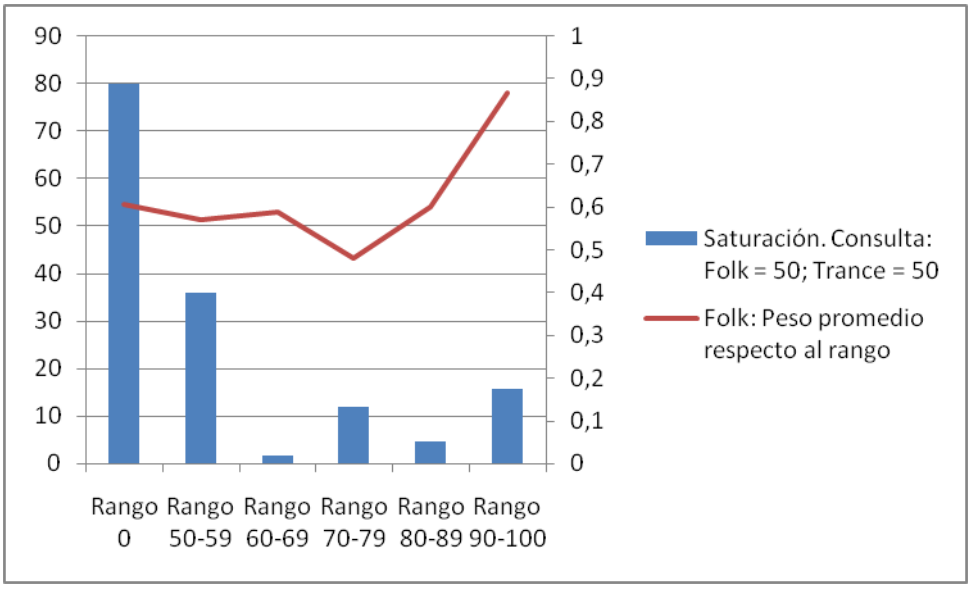


Figura 17. Saturación. Consulta: Folk = 50; Trance = 50

Cuando cambiamos la perspectiva y nos centramos en el género trance en la Figura 18, nos encontramos con la confirmación de todo aquello que habíamos visto desde el género folk.

El hecho de que el peso promedio para el rango 0 sea prácticamente el doble que el de cualquier otro rango ha hecho que la frecuencia se quede por debajo del 15%. Menos de uno de cada cinco discos de la solución era de trance, mientras que casi la mitad lo eran de folk. La solución promedio tenía alrededor de 20 discos, de los cuales, el máximo de discos del rango 0 fue de cuatro.

Completamente opuesta es la situación para el rango más grande, rango 90 – 100 en el que se encuentra el género folk y que representa entre 9 y 10 discos de la solución en todas nuestras pruebas.

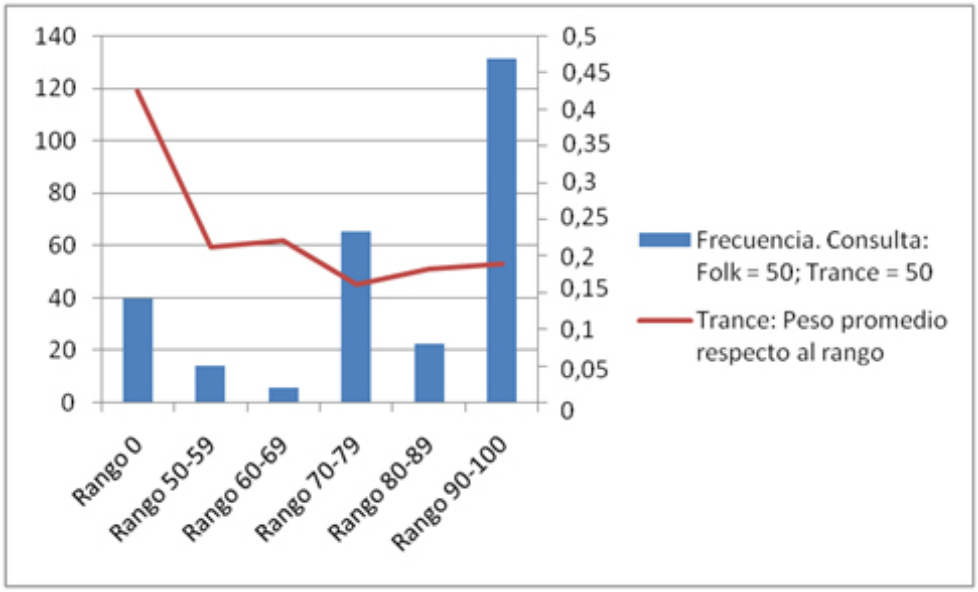


Figura 18. Frecuencia. Consulta: Folk = 50; Trance = 50

Pasamos a la saturación de la Figura 19 para ver mejor si cabe cómo ha afectado el peso del rango 0 en la solución.

Observamos que el único rango que tiene una saturación superior al 50% es el rango 70 – 79. Aquí es donde el problema del peso del `rango 0` ha hecho desvirtuarse más si cabe la solución. Y es que estos datos pueden parecer bastante malos si no somos conscientes de que en realidad, simplemente la balanza se inclina más hacia los rangos más lejanos por la influencia de la distancia con el folk.

Al tener el rango 90 – 100 treinta y un discos, es imposible ver en la saturación el volumen de este, que ya vimos en la frecuencia, y nos tenemos que quedar con una gráfica claramente desequilibrada hacia el centro.

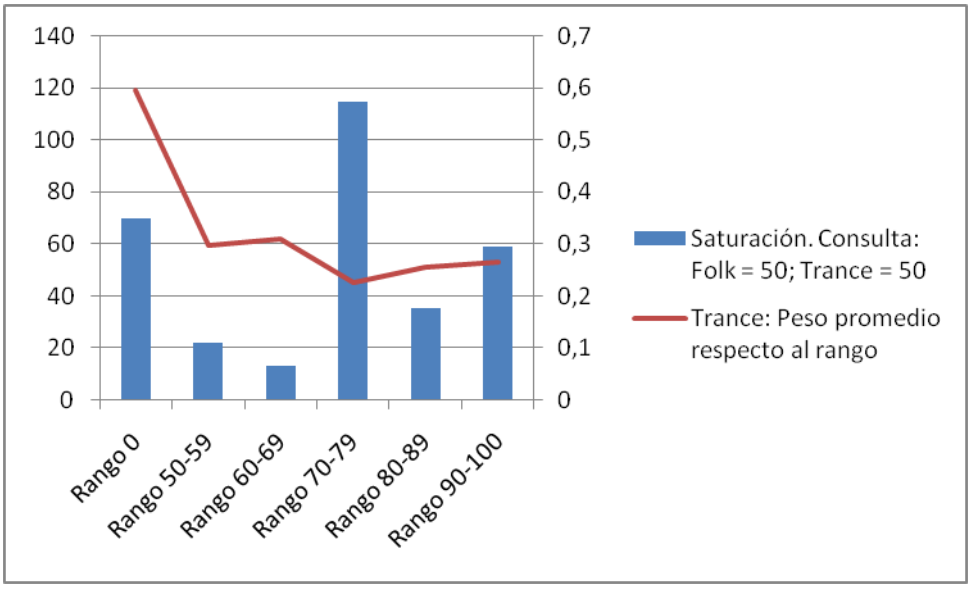


Figura 19. Saturación. Consulta: Folk = 50; Trance = 50

5.3.1.1.4 Consulta: Folk = 80; Trance = 20

En la siguiente consulta realizada decidimos, partiendo de la consulta anterior, comprobar el verdadero peso de la relevancia en los resultados obtenidos. Si en la anterior consulta tanto Folk como Trance tenían una relevancia de 50, en esta ocasión hemos decidido dar al primer género una relevancia cuatro veces mayor a la del segundo término (80 frente a 20) con la esperanza de que esto decante más si cabe la balanza del lado del folk que, recordemos, ya se beneficiaba de su menor peso promedio.

Al tomar la frecuencia vista desde el término folk nos damos cuenta de que efectivamente ésta ha subido considerablemente para los rangos más cercanos al género con mayor relevancia.

La frecuencia del rango 0 era del 40,8% cuando ambos criterios compartían relevancia, pero ahora, el rango que determina los discos de género Folk sube hasta el 46.2%, casi seis puntos que le dejan muy cerca de la mitad de los discos de la solución.

El siguiente rango, el más cercano a Folk pero sin serlo sube también una cantidad similar, un 4.0% hasta el 38.7% mientras que el gran perjudicado es sin duda el rango 90-100, rango en el que se encuentra el género Trance, cuya disminución en relevancia le hace pasar de una frecuencia del 19.4% a una

frecuencia del 4.3%, más de 15 puntos perdidos al pasar de 50 a 20 en relevancia.

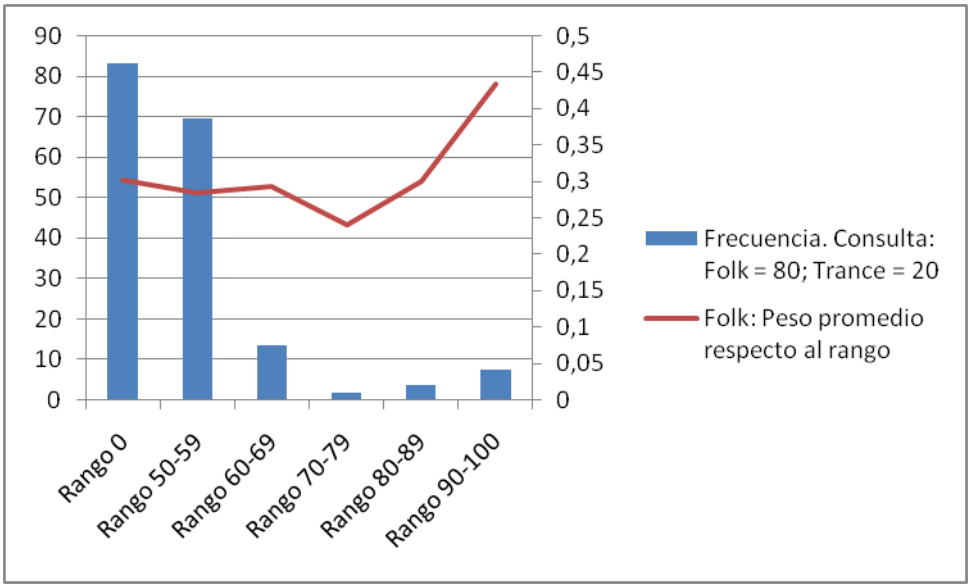


Figura 20. Frecuencia. Consulta: Folk = 80; Trance = 20

Similares son los resultados del rango de Trance en lo que a saturación se refiere. Si antes se incluía en la solución casi uno de cada cinco discos del rango máximo de separación con respecto a Folk (un 17,2%), ahora esta cifra cae en picado hasta el 3,6%, ligeramente menos de 15 puntos perdidos.

La cara de la moneda es folk, el género que ya contaba con un 88,9% de saturación la lleva ya hasta el 95,6%. De los nueve discos del género, en más de la mitad de las ocasiones no queda ninguno fuera de la solución final.

Aumentar la relevancia a folk ha hecho que el género haya aumentado de una forma muy notable, casi diríamos que drástica, su presencia en la solución en detrimento del género que baja su relevancia hasta una cuarta parte de la de folk, el Trance.

Vamos ahora a ver ambas gráficas desde el punto de vista del género que ha sufrido las pérdidas, para ver si efectivamente son tan marcadas como el rango 90-100 sobre Folk hace intuir.

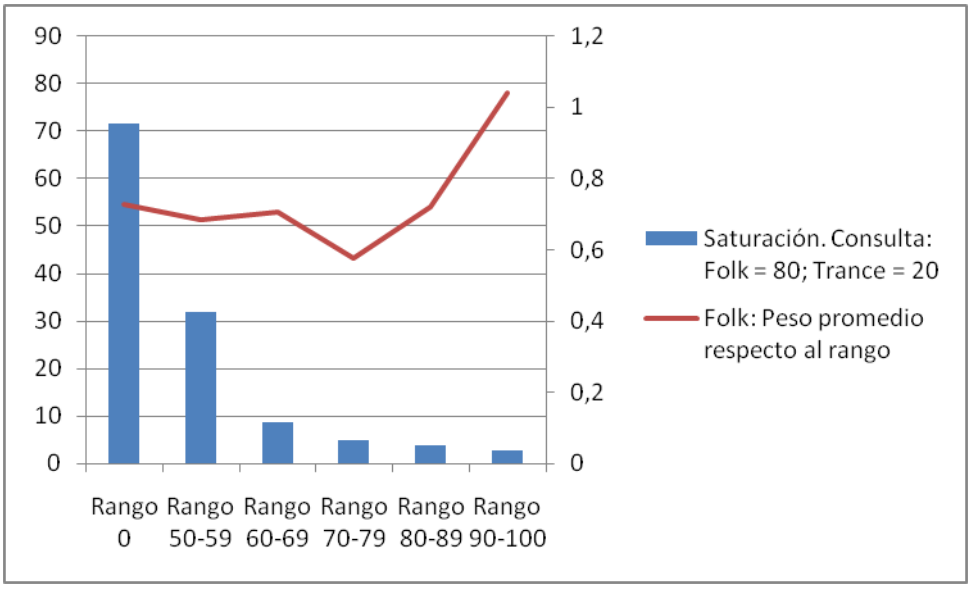


Figura 21. Saturación. Consulta: Folk = 80; Trance = 20

Efectivamente, como podemos comprobar en la gráfica de la frecuencia de la consulta vista desde el género Trance, Figura 22, prácticamente no hay presencia alguna de los rangos más cercanos al género.

Si ya nos parecía poca la frecuencia para el rango 0 en la consulta que tenía una relevancia de 50 para el género Trance, la frecuencia en esta ocasión baja casi diez puntos más para quedarse en un 4,3%. Además el rango más cercano a Trance sin serlo baja hasta quedarse en el 0,0%. Ni un solo disco de este rango ha aparecido en ninguna de las soluciones obtenidas. Y en esta ocasión no es porque el rango tuviese una cantidad irrelevante de discos, ya que un total de nueve se encontraban en él.

Prácticamente son las modificaciones en la solución son simétricas ya que los 9,98 puntos que ha perdido el rango 0 y los 5,10 que ha perdido el rango 50-59 los han básicamente ganado los dos últimos rangos. El rango 80-89 ha subido un 5,82% mientras que el rango 90-100 se lanza hasta el 10,05% más, quedándose con más del 50% de los discos de la solución.

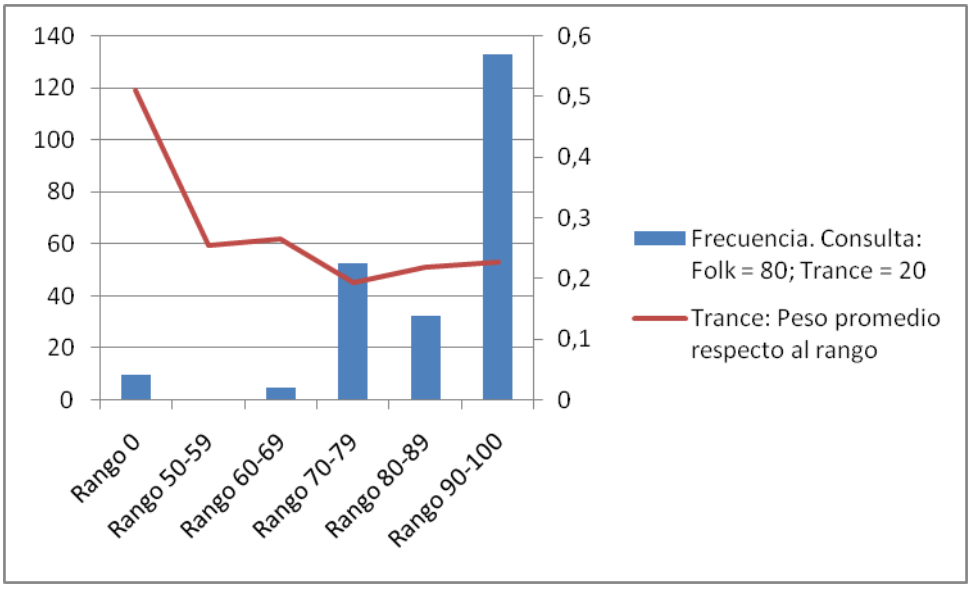


Figura 22. Frecuencia. Consulta: Folk = 80; Trance = 20

La saturación nos ayuda a completar el cuadro que nos ha dejado la frecuencia con un dato que ya de por sí es suficiente para ver la importancia de la relevancia. La saturación del rango 0 que estaba en un escaso 35% cuando la relevancia de Trance era 50 debido, como ya comentábamos, a la mayor duración de los discos de este rango, se despeña completamente hasta caer al 10%.

Una bajada de 25 puntos que ya no se le puede atribuir a la duración pues obviamente los discos son los mismos que en la anterior consulta. Una disminución de 30 puntos en la relevancia del término unida a un aumento por la misma cantidad de la relevancia de Folk hace que la saturación en la solución del género trance pierda dos veces y media la cantidad en la que acaba quedando, pasando de ser el segundo rango con mayor presencia de la solución a ser el tercero, solo por encima de los rangos de 50-59 y 70-79 con un 0% y un 6,7% respectivamente.

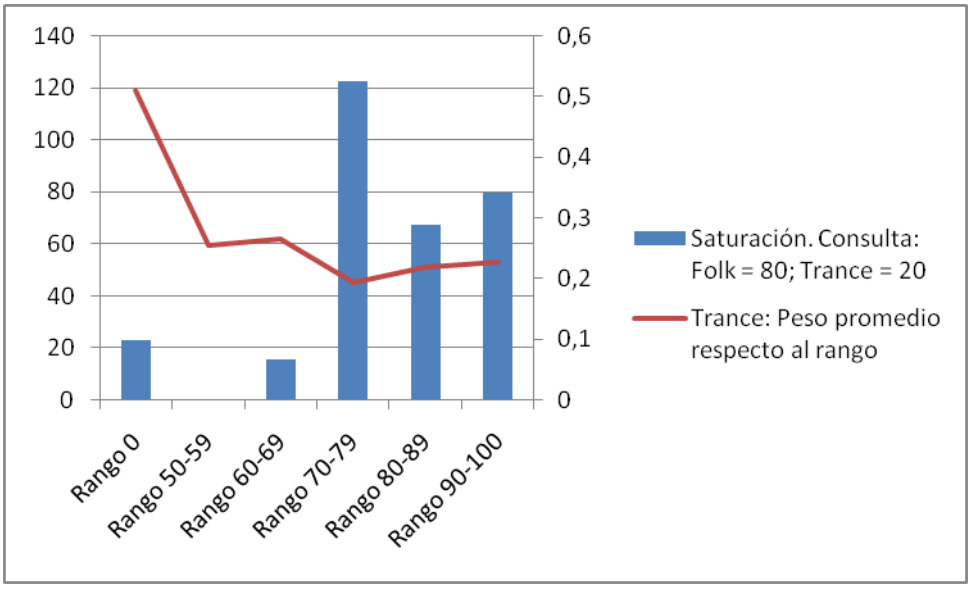


Figura 23. Saturación. Consulta: Folk = 80; Trance = 20

5.3.1.1.5 Consulta: Folk = 50; Javier Krahe = 50

Una vez hemos vista la influencia de los pesos y la relevancia cuando dos factores de idéntico criterio se encuentran, ahora probaremos una consulta con dos factores, uno del criterio género y otro del criterio artista.

A pesar de que no hay relación entre ambos criterios en lo que al cálculo de las distancias se refiere (ambos tienen su propia tabla de distancias independiente) resulta evidente que aquellos artistas que comparten género musical se encuentren a mucha menor distancia que aquellos que tengan un género completamente distinto. De esta manera, esperamos que las soluciones de las dos primeras consultas sean mucho más homogéneas que aquellas de las dos consultas finales en las que el artista del criterio se aleja completamente del género elegido para la consulta.

Esta consulta tiene como primer factor Folk, del criterio género, con relevancia igual a 50, mientras que el segundo factor es Javier Krahe, del criterio artista, con igual relevancia. Javier Krahe tiene 4 discos en el repositorio, todos ellos dentro de los 9 discos del género Folk. Como única nota destacar que uno de estos cuatro discos tiene una duración muy por encima de los otros, superando en más de 20 minutos al siguiente. Comprobaremos si esto también influye en los resultados. Para empezar mostraremos las gráficas de los resultados desde el punto de vista del primer factor, Folk = 50.

De los 16/17 discos que tiene en promedio la solución, un 82,9% de ellos se incluyen en los rangos 0 o 50-59, y solo un 1,2% están en el rango final 90-100, un resultado bastante homogéneo que ya esperábamos. De los nueve discos del rango 0, se incluyen en la solución más de siete en promedio y no llegan a tres los discos con rango de 60 o superior que formar parte de la solución.

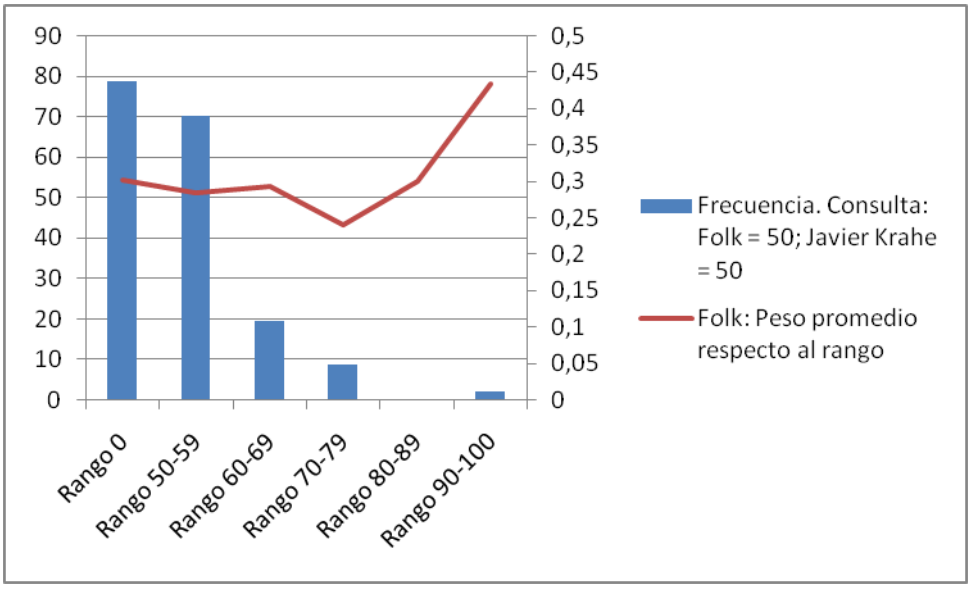


Figura 24. Frecuencia. Consulta: Folk = 50; Javier Krahe = 50

Pasamos ahora a analizar la saturación para ver si, aún a pesar de la homogeneidad de la solución, aún nos han quedado muchos discos pertenecientes a rangos inferiores fuera de la lista de reproducción generada.

Los resultados son igualmente buenos, aunque es cierto que nos sorprende que la saturación para el rango 0 sea inferior a aquella que encontramos en la consulta Folk = 50; Trance = 50. Aún así un 80% es una saturación altísima, y la diferencia entre ambas puede ser causada simplemente por la varianza en las soluciones. El rango 50-59 también tiene una saturación muy alta, con un 37,7% mientras que los rangos de 80-89 y 90-100 tienen una saturación del 0% y del 0,9% respectivamente. Básicamente no tienen presencia alguna. Una confirmación más de la validez y homogeneidad de las soluciones obtenidas para esta consulta.

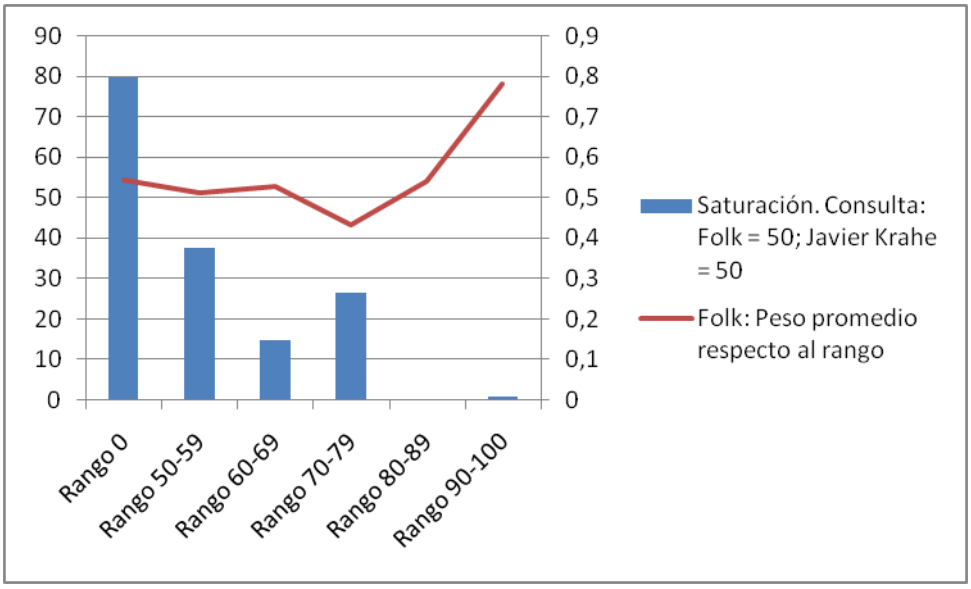


Figura 25. Saturación. Consulta: Folk = 50; Javier Krahe = 50

Cuando pasamos a comprobar los resultados desde el punto de vista del segundo término de la consulta, Javier Krahe = 50, hemos de tener en cuenta que por primera vez pasamos a trabajar sobre un término con pocas ocurrencias. Hasta el momento tanto Trance como Folk habían contado con prácticamente diez ocurrencias, pero Javier Krahe solo tiene cuatro. Esto se nota en especial cuando vemos la frecuencia del rango 0. Sobre los más de 16 discos de la solución promedia, el rango 0 no podía aspirar ni tan siquiera a llegar a un 25% de frecuencia. Conociendo esto, por tanto, vemos que su frecuencia del 23,2% es muy elevada. De hecho en prácticamente todas las soluciones se han incluido los cuatro discos de Javier Krahe. Siendo el disco de mayor duración del que hablamos más arriba el único que quedó fuera menos de un 20% de las veces.

El único problema que podría haber es que la escasez de discos en el rango 0 hiciese que la solución se esparciese a lo largo de los otros rangos, pero como vemos en la gráfica no ha sido exactamente así, y es el rango siguiente, 50-59 el que cuenta con el grueso de la frecuencia con un 61%. Un 84% de los discos de la solución están, por tanto, en estos primeros dos rangos.

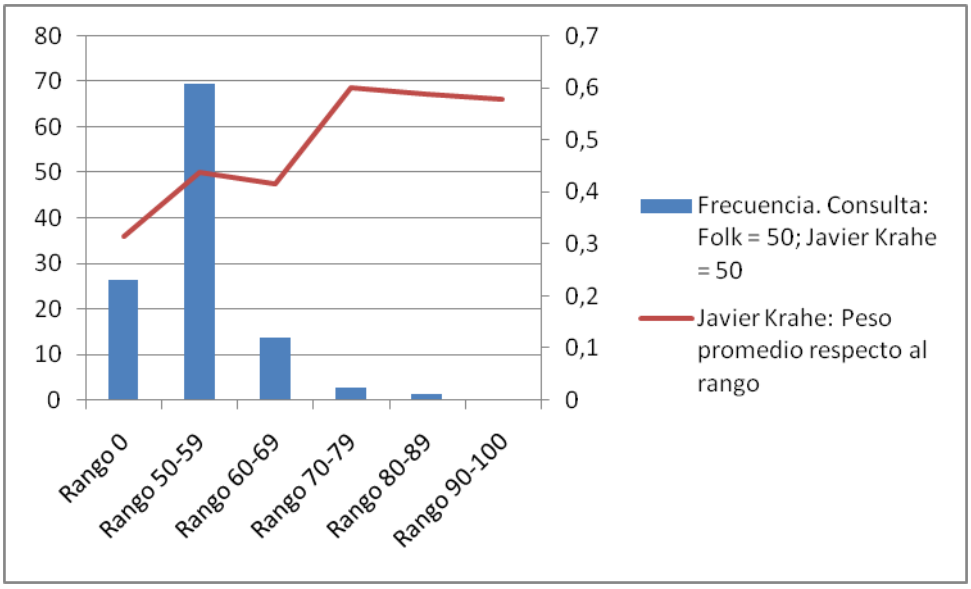


Figura 26. Frecuencia. Consulta: Folk = 50; Javier Krahe = 50

La saturación en cambio sí que es un punto bastante relevante. Teniendo en cuenta que además de tener un artista con tan solo cuatro discos, cuando las soluciones promedio tienen 16 discos, éste artista tiene todos sus discos pertenecientes al género que es el primer término de la consulta, el Folk.

Por este motivo, parece lógico esperar que la saturación de Javier Krahe sea cercana al 100%. Y así se muestra en la Figura 27: un 95% de saturación que unida al 50% de saturación del rango 50-59 confirman el comentario anterior al respecto de la homogeneidad de las soluciones obtenidas con esta consulta.

Observando las saturaciones del resto de los rangos vemos como éstas disminuyen drásticamente a medida que nos distanciamos del rango 0. Un 18% para el rango 60-69 y apenas un 3 y un 2% para los rangos 70-79 y 80-89 para terminar con un 0% en el rango final, 90-100.

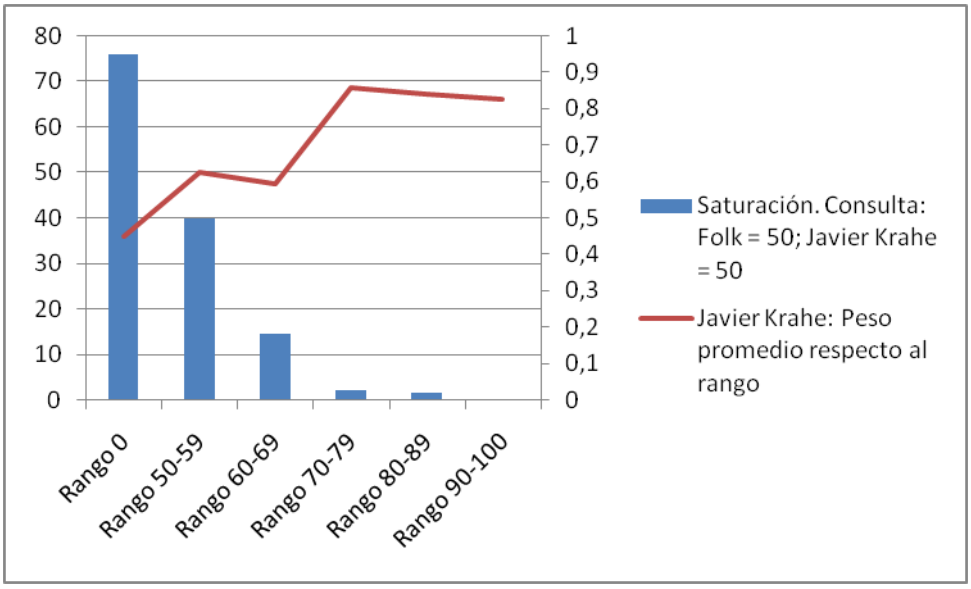


Figura 27. Saturación. Consulta: Folk = 50; Javier Krahe = 50

5.3.2 Recomendaciones sensibles al contexto en sistemas de navegación

Una vez comprobado el funcionamiento del algoritmo OCH con datos semánticos, es el turno de evaluar el funcionamiento de la propuesta al completo, es decir, incluyendo las secuencias contextuales con el objetivo de utilizarlas como modelo histórico sobre el que basar la selección de situaciones contextuales más probables en el futuro. En este caso el dominio de aplicación son los sistemas de navegación basados en mapas. Hasta el momento podemos distinguir 4 generaciones de sistemas de navegación tomando como criterio de clasificación el uso que hacen de la información de contexto y cómo visualizan sobre el mapa la información adicional:

- La 1ª generación está formada por aquellos sistemas de navegación de mapas que no disponen de ninguna capacidad para detectar y mantener el contexto del usuario del sistema, como por ejemplo su ubicación geográfica actual, por lo que el propio usuario es responsable de suministrar y mantener su información contextual mientras hace uso del sistema.
- La 2ª generación de sistemas es capaz de detectar y mostrar sobre un mapa la localización actual del usuario; para ello puede hacer uso de un dispositivo GPS, giroscopios, acelerómetros, triangulación de una señal WIFI o 3G o una combinación de varios de estos métodos. En cualquier caso, la posición del usuario es el único dato contextual que es capturado automáticamente por el sistema. A partir de este dato el sistema de

navegación muestra sobre el mapa puntos de interés (POI) cercanos, como por ejemplo museos, gasolineras, hospitales, hoteles, supermercados, etc.

- La característica diferenciadora de la 3ª generación de sistemas de navegación de mapas es su capacidad de realizar un filtrado de los POI disponibles en el sistema mediante la utilización de la información contextual que el usuario introduzca manualmente. Esta información aportada por el usuario puede expresarse en forma de una selección de categorías de POI preferidas, una distancia máxima desde su ubicación actual, un conjunto de palabras clave que describan semánticamente los POI preferidos, u otro tipo de información contextual. La mayoría de los sistemas de navegación basados en mapas modernos pertenecen a esta generación.
- La 4ª generación de estos sistemas es capaz de aprender las preferencias de los usuarios e inferir el contexto actual para personalizar la experiencia del usuario.

Sin embargo, ninguno de estos sistemas muestra información relacionada con las áreas por las que el usuario pasará muy probablemente en un futuro cercano teniendo en cuenta su comportamiento histórico, lo que permitiría anticiparse a sus necesidades de información y servicios.

En este punto es donde pensamos que nuestra propuesta algorítmica puede ser de utilidad para mejorar qué información mostrar sobre los mapas y cómo hacerlo. Si las secuencias contextuales que define nuestra propuesta representasen las rutas que sigue un usuario, y por lo tanto tendrían entre sus metadatos de contexto capturados las coordenadas espaciales de las rutas, podríamos tener un modelo semántico de los viajes realizados por el usuario. De esta manera, un ambiente inteligente podría hacer uso de nuestro algoritmo OCH para prever cuáles son las rutas o destinos más probables a partir del contexto actual y el modelo semántico de las secuencias contextuales correspondientes a pasados viajes y, a partir de esta información, mostrar información y ofrecer servicios personalizados a la situación contextual actual y futura del usuario. Un sistema con estas características representaría la 5ª generación en los sistemas de navegación de mapas y permitiría que se hiciese realidad un escenario similar al planteado en el punto 1.1 de esta tesis.

5.3.2.1 Resultados experimentales

El modelo utilizado en este experimento se creó a partir de aproximadamente 120 viajes diferentes que, aplicando el proceso de adición de secuencias contextuales, dieron lugar a alrededor de 15 rutas diferentes con información semántica asociada. Algunas de estas rutas representan las rutas habitualmente realizadas por el usuario y se formaron a partir de varias secuencias, aunque

también había rutas hechas sólo una vez. La información contextual recogida por cada nodo de las secuencias contextuales creadas para cada viaje realizado fue la siguiente:

- Actividad: indica el tipo de actividad que realizaba el usuario cuando la información fue capturada; por ejemplo, ocio o trabajo.
- Hora: hora del día en la que la información contextual fue recogida.
- Coordenadas: la ubicación del usuario cuando la información fue capturada, expresada en latitud y longitud.

Para poder interactuar con el algoritmo y visualizar sobre un mapa los resultados de su ejecución, desarrollamos una aplicación de escritorio. La aplicación permite cargar el modelo semántico de los viajes realizados por un usuario y crear consultas a partir de la introducción de valores para algún término de la ontología y de un factor de relevancia asociado, tal y como puede verse en la Figura 28. La aplicación también permite una ligera personalización adicional de la visualización en el mapa de las rutas devueltas: es posible especificar el número de rutas, así como el número de nodos intermedios por ruta a mostrar sobre el mapa.

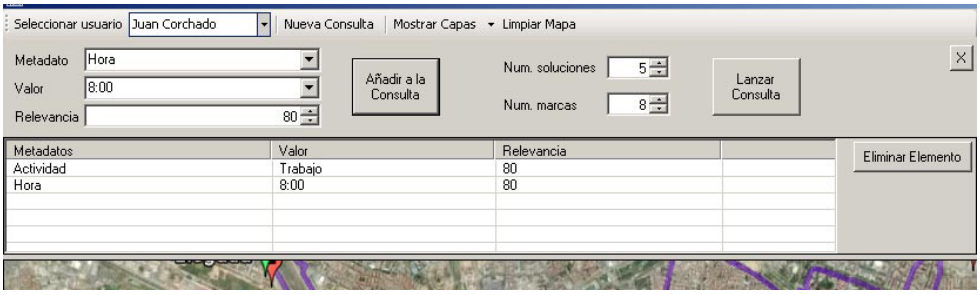


Figura 28. Introducción de valores y sus factores de relevancia asociados para la creación de una consulta.

En la Figura 29, que podemos ver a continuación, observamos cómo se visualizan sobre el mapa las rutas devueltas por el algoritmo.

Figura 29. Visualización de las rutas devueltas para una determinada consulta.

Como parte del experimento, y al igual que en el presentado anteriormente, se definieron varias funciones para calcular las distancias semánticas entre los valores de los términos de la ontología utilizada. El proceso de asignación dinámica de *score* hace uso de estas funciones.

En los siguientes puntos mostraremos las consultas utilizadas para buscar en el modelo y evaluar el funcionamiento del algoritmo. Asimismo mostraremos una lista ordenada de mayor a menor *score* con las rutas devueltas, de las que mostraremos también su distancia semántica con la consulta correspondiente. Una medida adicional de calidad comprobada en cada consulta es el porcentaje de rutas que hubiese en el modelo con una distancia semántica inferior a la distancia semántica de la peor solución y que no formasen parte de las soluciones devueltas por el algoritmo. De esta manera se comprueba si la implementación realizada pasa por alto buenas soluciones que sí deberían formar parte del conjunto de soluciones devueltas. Cuanto más cercano a 0 sea este porcentaje, mejor será el comportamiento del algoritmo. Aunque a primera vista pudiese parecer que el valor de esta medida debiera ser siempre 0, es decir, sólo se devuelven las mejores soluciones, la naturaleza estocástica del

proceso de selección de nodos junto con un posible alto número de bifurcaciones dentro del espacio del problema, pueden hacer que el algoritmo obtenga soluciones que sean óptimos locales. Para minimizar este efecto el algoritmo hace uso de varias hormigas y realizar varias iteraciones por hormiga.

5.3.2.1.1 Experimento 1. Consulta: {(actividad = trabajo, relevancia = 80), (hora = 8.00, relevancia = 80)}

Para esta consulta el porcentaje de rutas con una distancia semántica menor que 30.375 fue 0, aunque sí que hubo otras rutas que tenían términos coincidentes con los de la consulta pero, en todos los casos, con distancias semánticas mayores.

	Actividad	Hora	Distancia semántica	Duración
Ruta 1	trabajo	8:00:02	5.564	14 min
Ruta 2	trabajo	8:08:21	19.044	30 min
Ruta 3	trabajo	8:01:21	21.864	48 min
Ruta 4	trabajo	8:13:41	23.674	34 min
Ruta 5	trabajo	8:05:58	30.375	1h 1min

5.3.2.1.2 Experimento 2: Consulta: {(actividad = ocio, relevancia = 80), (hora = 22.00, relevancia = 80)}

En este caso, el porcentaje de de rutas del espacio de búsqueda con una distancia menor que 165.011 también fue 0, aunque también hubo rutas con correspondencia con esta consulta, aunque con que distancias semánticas mayores.

	Actividad	Hora	Distancia semántica	Duración
Ruta 1	ocio	22:20:07	25.275	22 min
Ruta 2	ocio	18:06:41	100.694	4h 57 min
Ruta 3	ocio	19:09:13	120.738	40 min
Ruta 4	ocio	17:27:41	160.899	1h 33 min
Ruta 5	ocio	16:18:34	165.011	3h 16min

5.3.2.1.3 Experimento 3: Consulta: {(hora = 14.30, relevancia = 80)}

El porcentaje de rutas con una distancia semántica inferior a la de la mayor de las soluciones devueltas fue 0. Para esta consulta también se encontraron más rutas que se correspondían con la consulta, aunque con distancias mayores.

	Actividad	Hora	Distancia semántica	Duración
Ruta 1	trabajo	10:43:54	143.481	33 min
Ruta 2	ocio	16:57:32	146.397	10 min
Ruta 3	trabajo	10:09:55	158.360	59 min
Ruta 4	ocio	17:01:36	159.620	36 min
Ruta 5	ocio	16:59:48	160.835	38 min

5.3.2.1.4 Experimento 4: Consulta: {(coordenadas = (39.469794,-0.377655), relevancia = 80)}

Para esta consulta esta es la lista de soluciones devueltas y el porcentaje de rutas con una distancia semántica con la consulta menor que 523.421 fue de 0. Como en otros casos el espacio de búsqueda contenía más rutas coincidentes, aunque con distancias semánticas mayores.

	Actividad	Hora	Distancia semántica	Duración
Ruta 1	ocio	16:32:55	88.884	3h 29 min
Ruta 2	ocio	18:06:41	109.770	4h 57 min
Ruta 3	trabajo	10:09:55	127.665	59 min
Ruta 4	trabajo	8:05:58	236.014	1h 1 min
Ruta 5	ocio	22:20:07	523.421	22 min

5.3.2.1.5 Experimento 5: Consulta: {(hora = 14:30, relevancia = 80), (coordenadas = (39.469794,-0.377655), relevancia = 80)}

En este último caso, el porcentaje de de rutas del espacio de búsqueda con una distancia menor que 908.696 también fue 0, aunque también hubo rutas con correspondencia con esta consulta, aunque con que distancias semánticas mayores.

	Actividad	Hora	Distancia semántica	Duración
Ruta 1	ocio	16:32:55	263.616	3h 29 min
Ruta 2	trabajo	10:09:55	310.026	4h 57 min
Ruta 3	ocio	18:06:41	401.652	59 min
Ruta 4	trabajo	8:05:58	517.638	1h 1 min
Ruta 5	ocio	22:20:07	908.696	22 min

5.4 Conclusiones

En este capítulo hemos reincidido en la importancia del contexto para los ambientes inteligentes, en como el tratamiento semántico de la información contextual capturada es el mecanismo adecuado para permitir que los humanos sigan pudiendo manipular la información y que los distintos dispositivos y sistemas que forman el ambiente puedan procesar y compartir cualquier nuevo conocimiento inferido a partir de la información contextual disponible, y en que el uso de la información contextual histórica es el camino para conseguir que los ambientes inteligentes tengan un comportamiento más adaptado a las características y sucesos del entorno en el que se encuentra. Después presentamos nuestra propuesta algorítmica basada en colonias de hormigas como la herramienta para sacar el mejor partido de la información contextual histórica y explicamos su funcionamiento.

Para demostrar el funcionamiento de nuestra propuesta y evaluar su correcto comportamiento, aplicamos nuestra propuesta a 2 dominios de aplicación diferentes y realizamos diversos experimentos. Con la aplicación del algoritmo al dominio de la generación automática de listas de reproducción musicales pretendíamos demostrar una funcionalidad básica de nuestra propuesta: la capacidad de nuestra propuesta de algoritmo OCH para utilizar datos heterogéneos pertenecientes a una ontología en su funcionamiento. Los resultados experimentales confirman dicha capacidad, se generaron listas de reproducción que respondían a las características especificadas en la consulta, así como que soporta consultas con más de un término y sus relevancias asociadas.

Una vez comprobadas las ideas básicas de la propuesta planteada, elegimos el dominio de los sistemas de navegación basados en mapas para comprobar el funcionamiento de la característica de predicción de situaciones contextuales futuras a partir de un modelo histórico de secuencias contextuales de un cierto

entorno. El objetivo en este caso sería prever cuáles son las rutas o destinos más probables a partir del contexto actual y el modelo semántico de las secuencias contextuales correspondientes a pasados viajes y, a partir de esta información, se podría mostrar información relevante sobre el mapa y ofrecer servicios personalizados a la situación contextual actual y a la futura prevista. Los resultados experimentales muestran como consistentemente el algoritmo devuelve aquellas rutas con una mayor proximidad semántica a la consulta especificada.

"Paint the future", por Andrew Judd (2009)

Capítulo 6

Conclusiones y trabajos futuros

En el primer capítulo de esta tesis hemos hecho una introducción al área de HCI y su excesiva concentración en mejorar la interacción entre los usuarios y el software de su entorno con el que interactúan mediante la reducción de la frustración mediante la simplificación al extremo de cualquier tarea que debieran realizar. Esto, según la teoría de *flow*, termina generando aburrimiento y apatía y también otros estados emocionales que alejan al usuario de cualquier interés intrínseco por utilizar el software. Además, hemos destacado como la creciente abundancia de dispositivos electrónicos a nuestro alrededor ha traído consigo la aparición de una nueva generación de sistemas de información, conocidos como ambientes inteligentes, que aglutinan los dispositivos y objetos del entorno en un medioambiente electrónico, que se adaptan al contexto del entorno en el que se encuentran automáticamente y están ideados para responder de una manera inteligente y no intrusiva a la presencia de las personas en su entorno.

Sin embargo, también hemos destacado como las propuestas de ambientes inteligentes desarrolladas hasta el momento no tienen la información histórica de contexto recopilada para mejorar su adaptación, con lo que pierden una oportunidad valiosa de anticiparse a situaciones del entorno que volverán a tener lugar en un futuro cercano con una alta probabilidad. Así, finalmente, planteamos la necesidad de inclusión de modelos afectivos en los ambientes inteligentes para que puedan "tomar decisiones" para adaptarse al estado emocional del usuario, así como la necesidad de recopilar y utilizar información contextual del entorno a lo largo del tiempo con el fin de conocer qué situaciones contextuales se darán con mayor probabilidad en el futuro más próximo y poder así tomar las decisiones más adecuadas.

En el capítulo 2 hemos presentado el estado actual tanto en sistemas de toma de decisiones influenciados por emociones como en aprendizaje de secuencias contextuales. En el primer caso, hemos revisado diversos mecanismos que se están utilizando, o que podrían ser utilizados para crear sistemas de toma de decisiones influenciados por emociones. Como hemos podido comprobar, los sistemas multiagente son frecuentemente utilizados para dotar de comportamiento emocional a personajes o entidades virtuales. Desafortunadamente, y a pesar de la variedad de propuestas de arquitecturas de agentes emocionales, estos sistemas serán tan buenos generando comportamiento emocional como lo sea la técnica, algoritmo o método empleado realmente para seleccionar las acciones a realizar. Los métodos más tradicionales de *decision making* no los tenemos en cuenta puesto que: muchos de ellos son específicos a determinados dominios, otros no cuentan siquiera con un modelo computacional (no pasan de ser simples especificaciones de ideas o teorías), y en la mayoría sería un verdadero problema intentar modificarlos para que contemplasen la influencia de estados emocionales. Otros mecanismos, como los basados ampliamente en probabilidades y cálculos estadísticos, presentan los inconvenientes de no utilizar el conocimiento adquirido para mejorar la recompensa obtenida a medio plazo, o de elegir de forma inadecuada la mejor decisión con el único objetivo de reducir el tiempo de decisión. Este último inconveniente se podría abordar, por ejemplo, paralelizando el sistema pero el propio diseño de estas propuestas (y de muchas otras) hace casi inviable o poco provechoso el intento.

Respecto al aprendizaje de secuencias contextuales, hemos realizado un análisis de las aproximaciones existentes que indica que existen una gran variedad de posibles soluciones algorítmicas al problema desde un punto de vista teórico. Uno de los primeros aspectos a tener en cuenta es que existe una gran variedad de tareas relacionadas con el problema del aprendizaje de secuencias como son: la clasificación de las mismas, la predicción de series temporales, la estimación de distribuciones probabilistas sobre dominios de secuencias y la transducción

de secuencias. Dada la complejidad y variedad de problemas en el ámbito del aprendizaje de secuencias sería ingenuo pensar que existe una única aproximación que domina el campo de investigación. Según la naturaleza de los problemas a resolver se han estudiado diferentes aproximaciones que varían desde las redes neuronales recurrentes, los modelos de Markov ocultos, el aprendizaje con refuerzo, la computación evolutiva, los sistemas basados en reglas y los sistemas difusos, entre otros. Aunque recientemente parece haber un sentimiento en la comunidad investigadora sobre la necesidad de aproximaciones híbridas para resolver problemas reales, no existe por desgracia una base sistemática que describa de forma rigurosa como proceder para combinar las distintas aproximaciones existentes. También hemos hablado de los algoritmos genéticos y de las colonias de hormigas como ejemplos de métodos metaheurísticos. Este tipo de métodos son generalistas y pueden ser aplicados a infinidad de dominios, aunque presentan un problema y es que las soluciones que proporcionan no son óptimas ya que, de forma nativa, estos métodos no buscan en todo el espacio del problema, razón por la cual son utilizados para resolver problemas con espacios de búsqueda enormes. De entre estos dos métodos, las colonias de hormigas presentan una mayor adecuación a las características de los dos dominios abordados. En el caso de la toma de decisiones, hemos de tener en cuenta que una de sus aplicaciones más habituales es en la resolución de caminos en grafos, por lo que tanto la representación del espacio del problema como la obtención de la mejor secuencia de decisiones para un futuro cercano son inherentes. Además, su sistema de aprendizaje basado en el depósito y actualización de las feromonas en aquellos caminos (secuencia de decisiones) más beneficiosos hace que sea capaz de adaptarse a cambios en las condiciones de su entorno. En cuanto al ámbito de las secuencias contextuales, algoritmos de refuerzo como los de colonias de hormigas presentan la ventaja de construir el modelo de forma progresiva a medida que se van obteniendo secuencias del fenómeno a analizar. La propia estructura en forma de grafo de representación del problema en dichos algoritmos permite representar de forma natural las secuencias temporales. Además, al tratarse de algoritmos de refuerzo, y utilizando métricas de similitud entre secuencias, es posible reforzar mediante algún tipo de medida escalar (feromonas en el caso de las colonias de hormigas) aquellos caminos (secuencias) que se presentan con mayor frecuencia. Dado que los nodos pueden representar situaciones contextuales de cualquier complejidad en cuanto al número de variables presentes en el problema, es posible transformar los problemas de clusterización de secuencias (totales o parciales) y de estimación de secuencias futuras como problemas de optimización sobre dichos espacios anotados mediante mecanismos de refuerzo. Además, es conveniente destacar que la propia naturaleza de los algoritmos de colonias de hormigas permite el diseño de forma natural de algoritmos de naturaleza no

secuencial, ya que hay muchos agentes artificiales (hormigas) obteniendo soluciones de forma colectiva.

Por lo tanto, vistas las ventajas que presenta respecto a la capacidad de tratar problemas de talla grande y obtener buenas soluciones en un tiempo razonable, y las facilidades que aporta para realizar algunas modificaciones y mejoras, decidimos utilizar las colonias de hormigas como punto de partida para la creación de una estrategia capaz de abordar problemas de ambos dominios. Los retos a afrontar estriban en poder definir mecanismos abstractos de computación basados en algoritmos OCH que permitan obtener soluciones para problemas cuyo modelo (variables involucradas, métricas de similitud y funciones a optimizar) pueda ser proporcionado de forma específica para cada problema concreto a resolver sin tener que recodificar la estrategia algorítmica.

En el capítulo 3 hemos descrito con mayor detalle la técnica metaheurística de optimización basada en colonias de hormigas, OCH, explicando en primer lugar la capacidad natural de las hormigas para encontrar el camino más corto entre su nido y una fuente de comida, además de los experimentos que confirman este comportamiento. Luego hemos expuesto las características matemáticas tanto de los problemas que se pueden resolver empleando esta técnica, como del modelo de hormiga artificial utilizado para simular el comportamiento de las hormigas naturales en algoritmos OCH. Más tarde hemos explicado el funcionamiento de un algoritmo OCH genérico y presentado cuatro de los diversos algoritmos existentes en la literatura, uno de los cuales (el sistema de colonias de hormigas) tomamos como base para nuestra propuesta.

También hemos presentado los dominios de problema más habituales en los que se aplica la metaheurística OCH, así como una lista no exhaustiva de propuestas concretas de algoritmos OCH diseñadas para abordar tipos de problema específicos. A continuación hemos presentado algunas de las líneas abiertas de investigación que más interés suscitan, explicando las características de los problemas a los que van dirigidas o que características de la metaheurística pretenden mejorar, y nombrado algunas propuestas realizadas y que demuestran la potencia y versatilidad de OCH, ya que han conseguido en muchos casos convertirse en las aproximaciones de referencia en sus dominios, tanto por la calidad de sus soluciones como por el rendimiento demostrado (en términos de tiempo empleado y/o tamaño del problema abordado). Sin embargo, ninguna de las propuestas existentes cubre las características que planteábamos como clave para las futuras generaciones de ambientes inteligentes; a saber, utilizar la información semántica contextual recopilada para prever situaciones contextuales en el futuro cercano, y añadir el estado emocional del usuario del ambiente a la información semántica contextual

recopilada y tenerlo en cuenta en el proceso de búsqueda y selección de acciones a desencadenar por parte del ambiente inteligente. Con estas características, lo que pretendemos es que el ambiente sea capaz de adelantarse a situaciones futuras que vayan a darse con alta probabilidad y tome las decisiones más adecuadas para conseguir mantener al usuario en un estado cercano al estado de *flow*.

Debido a que ninguna de las propuestas existentes nos resultaba válida, en los siguientes capítulos presentamos nuestras propuestas algorítmicas basadas en OCH y que han sido diseñadas para trabajar con información semántica con datos heterogéneos, de entre los cuáles prestará especial atención al estado emocional para que la decisión sobre qué acción realizar esté influida por el objetivo de acercar al usuario al estado de *flow* o experiencia óptima.

En el capítulo 4 hemos explicado en detalle la propuesta *hierarchical network of affective-cognitive decisions*, ya que es la aproximación de la que tomamos los modelos probabilistas a partir de los cuales simular y aprender el efecto cognitivo y emocional de las decisiones que se toman, y que más tarde influirán en el proceso de selección de nuevas decisiones.

También hemos presentado nuestra propuesta de sistema para la toma de decisiones influenciada por emociones utilizando como base algorítmica un sistema de colonias de hormigas, y como mecanismo de generación de la influencia cognitiva y emocional hemos partido de la propuesta mencionada y hemos realizado una serie de modificaciones para adaptarla a las características de un sistema de colonias de hormigas (por ejemplo, hemos añadido algunos modelos y una fórmula que es el objetivo a maximizar por las hormigas, y utilizamos la componente $Q_{DM}(\hat{e},c,d)$ como la matriz de atracción y feromonas de las hormigas). Las principales ventajas de nuestra propuesta incluyen la capacidad de abordar espacios de búsqueda grandes (conjunto de decisiones y estados cognitivos y emocionales) gracias al uso de una metaheurística, la mayor eficiencia que proporciona la paralelización intrínseca del uso de hormigas artificiales, y la previsión de secuencias de decisiones que puedan reportar el mayor beneficio en el futuro cercano más probable.

Al final del capítulo hemos presentado eCoology, un juego educativo para niños que utiliza realidad aumentada para mostrar entidades virtuales que tienen emociones en un entorno real, el caso de estudio en el que hemos probado nuestra propuesta. El dominio de aplicación elegido ha sido la resolución de caminos que las entidades deben realizar para desplazarse por el escenario desde su ubicación actual a un destino y teniendo en cuenta sus emociones. Después de presentar qué es eCoology, hemos explicado cómo funciona el mecanismo de *pathfinding* implementado en eCoology basado en un sistema de partículas y algunas ideas de algoritmos de grafos, y por último

hemos presentado la implementación de nuestra propuesta para este dominio y la aplicación con la que hemos comprobado su correcto funcionamiento.

Finalmente, en el capítulo 5, hemos reincidido en la importancia del contexto para los ambientes inteligentes, en como el tratamiento semántico de la información contextual capturada es el mecanismo adecuado para permitir que los humanos sigan pudiendo manipular la información y que los distintos dispositivos y sistemas que forman el ambiente puedan procesar y compartir cualquier nuevo conocimiento inferido a partir de la información contextual disponible, y en que el uso de la información contextual histórica es el camino para conseguir que los ambientes inteligentes tengan un comportamiento más adaptado a las características y sucesos del entorno en el que se encuentra. Después presentamos nuestra propuesta algorítmica basada en colonias de hormigas como la herramienta para sacar el mejor partido de la información contextual histórica y explicamos su funcionamiento.

Para demostrar el funcionamiento de nuestra propuesta y evaluar su correcto comportamiento, aplicamos nuestra propuesta a 2 dominios de aplicación diferentes y realizamos diversos experimentos. Con la aplicación del algoritmo al dominio de la generación automática de listas de reproducción musicales pretendíamos demostrar una funcionalidad básica de nuestra propuesta: la capacidad de nuestra propuesta de algoritmo OCH para utilizar datos heterogéneos pertenecientes a una ontología en su funcionamiento. Los resultados experimentales confirman dicha capacidad, se generaron listas de reproducción que respondían a las características especificadas en la consulta, así como que soporta consultas con más de un término y sus relevancias asociadas.

Una vez comprobadas las ideas básicas de la propuesta planteada, elegimos el dominio de los sistemas de navegación basados en mapas para comprobar el funcionamiento de la característica de predicción de situaciones contextuales futuras a partir de un modelo histórico de secuencias contextuales de un cierto entorno. El objetivo en este caso sería prever cuáles son las rutas o destinos más probables a partir del contexto actual y el modelo semántico de las secuencias contextuales correspondientes a pasados viajes y, a partir de esta información, se podría mostrar información relevante sobre el mapa y ofrecer servicios personalizados a la situación contextual actual y a la futura prevista. Los resultados experimentales muestran como consistentemente el algoritmo devuelve aquellas rutas con una mayor proximidad semántica a la consulta especificada.

6.1 Trabajos futuros

Aunque con las propuestas presentadas en esta tesis se cubren las características identificadas como esenciales para las futuras generaciones de ambientes inteligentes, hemos identificado una serie de extensiones posibles al trabajo ya expuesto:

- Añadir soporte para operadores de lógica temporal a la propuesta de algoritmo OCH semántico para la predicción de situaciones contextuales. Puesto que forma parte esencial de esta propuesta el uso de secuencias de situaciones contextuales pasadas para predecir las situaciones contextuales futuras más probables, creemos que dar soporte a operadores temporales enriquecería y otorgaría mayor flexibilidad a la propuesta. De esta manera, podríamos plantearnos la creación de consultas en las que establezcamos una cierta secuencia temporal entre términos (o conjuntos de términos). Por ejemplo, tomando como dominio las recomendaciones sensibles al contexto en sistemas de navegación, podríamos crear una consulta que buscase rutas en las que a las 17 de la tarde estuviésemos en una cierta ubicación y, más tarde (aquí haríamos uso de un operador temporal del tipo "después" o "en el futuro"), estuviésemos en otra localización.

- Realizar un estudio de las características de escalabilidad de las propuestas presentadas. Unas de las razones por las que elegimos la metaheurística OCH como base para el diseño de nuestras propuestas fueron que tanto la escalabilidad como la paralelización son unas de sus características intrínsecas. Sin embargo, los modelos utilizados en los experimentos mostrados en esta tesis fueron creados para que fuesen simples y facilitasen la evaluación de la corrección del funcionamiento de los algoritmos, así como el estudio de los resultados de los experimentos.

- Integrar ambas propuestas en una única y realizar una implementación para evaluar que la propuesta unificada sigue siendo capaz de resolver los mismos tipos de problemas, obteniendo unos resultados similares. Las decisiones posibles, los estados cognitivos y los emocionales pueden verse como términos de una ontología que define un contexto muy específico de un entorno, y puesto que también se realizan predicciones sobre cuál es la mejor decisión a tomar o acción a llevar a cabo, intuitivamente parece que la propuesta de algoritmo OCH para la toma de decisiones influenciada por emociones debe formar parte de la propuesta para la toma de decisiones en base a información semántica. En cualquier caso, el reto estriba en cómo incluir todos los modelos probabilistas y las tareas de simulación realizadas por las hormigas en la primera propuesta en el diseño de la segunda.

Apéndice A. Palabras Clave

Apéndice B. Acrónimos

Bibliografía

(Acosta, 2006) Acosta, R., Esteve J.M., Mocholí J.A., Jaén J. Ecoology: An Emotional Augmented Reality Edutaiment Application. En: International Conference on Cognition and Exploratory Learning in Digital Age. Barcelona, pp. 19-26, 2006.

(Adolphs, 1996) Adolphs, R. et al. Neuropsychological Approaches to Reasoning and Decision Making. In: Damasio, A., et al. Eds. Neurobiology of Decision-Making. Berlin, Springer-Verlag, 1996.

(Ahn, 2005) Ahn, H.I., Picard, R.W. Affective Cognitive Learning and Decision Making: A Motivational Reward Framework For Affective Agents. The 1st International Conference on Affective Computing and Intelligent Interaction, 2005, Beijing, China.

(Anderson, 1995) Anderson, J. An Introduction to Neural Networks. Cambridge, Mass, MIT Press, 1995.

(Avison, 1999) Avison, D.E., Lau, F., Myers, M.D., Nielsen, P.A. Action research. Commun. ACM, vol 42, 1, pp. 94-97, 1999.

(Azuma, 1997) Azuma, R. A Survey of Augmented Reality. Teleoperators and Virtual Environments, pp. 355-385, 1997.

(Baldauf, 2007) Baldauf, M., Dustdar, S., Rosenberg, F. A survey on context-aware systems. Int. Journal of Ad Hoc and Ubiquitous Computing, 2(4), pp. 263–277, 2007.

(Baskerville, 1999) Baskerville, R., Pries-Heje, J. Grounded action research: a method for understanding IT in practice, Accounting, Management and Information Technologies, vol 9, 1, 1999.

(Bauer, 1999) Bauer, A., Bullnheimer, B., Hartl, R.F., Strauss, C. An ant colony optimization approach for the single machine total tardiness problem. En: Proceedings of the 1999 Congress on Evolutionary Computation, IEEE Press, pp. 1445–1450, 1999.

(Berthouze, 2006) Berthouze, L., Tijsseling, A. A Neural Model for Context-dependent Sequence Learning. Neural Process. Lett. 23, 1, 2006

(Besten, 2000) den Besten, M.L., Stützle, T., Dorigo, M. Ant colony optimization for the total weighted tardiness problem. En: M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J.J. Merelo and H.-P. Schwefel (eds.), Proceedings of PPSN-VI, The 6th International Conf. on Parallel Problem Solving from Nature, volume 1917 of LNCS. Springer-Verlag, , pp. 611–620, 2000.

(Biegel, 2004) Biegel, G., Cahill, V. A framework for developing mobile, context-aware applications. En: Proceedings of the 2nd IEEE Conference on Pervasive Computing and Communication, pp. 361–365, 2004.

(Billinghurst, 2002) Billinghurst, M., Hirokazu, K. Collaborative Augmented Reality. Communication of the ACM, Vol.45, No.7, pp. 64-70, 2002.

(Blum, 2005a) Blum, C. Beam-ACO—Hybridizing ant colony optimization with beam search: An application to open shop scheduling. Computers & Operations Research, vol. 32, no. 6, pp. 1565–1591, 2005.

(Blum, 2005b) Blum, C., Blesa, M. J. New metaheuristic approaches for the edge-weighted k-cardinality tree problem.Computers & Operations Research, vol. 32, 6, pp 1355–1377, 2005.

(Bolondi, 1993) Bolondi, M., Bondanza, M. Parallelizzazione di un algoritmo per la risoluzione del problema del commesso viaggiatore. Master's thesis, Dipartimento di Elettronica, Politecnico di Milano. 1993.

(Bullnheimer, 1998) Bullnheimer, B., Hartl, R. F., Strauss, C. Parallelization Strategies for the Ant System. En: R. De Leone, A. Murli, P. Pardalos, and G. Toraldo (eds.), High Performance Algorithms and Software in Nonlinear Optimization; Series: Applied Optimization, vol. 24, 1998.

(Bullnheimer, 1999a) Bullnheimer, B., Hartl, R. F., Strauss, C. A new rank-based version of the Ant System: A computational study. Central European Journal for Operations

Research and Economics, vol 7, 1, pp 25-38, 1999.

(Bullnheimer, 1999b) Bullnheimer, B., Hartl, R. F., Strauss, C. Applying the Ant System to the vehicle routing problem. En: S. Voß, S. Martello, I.H. Osman and C. Roucairol (eds.), Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization. Kluwer Academic Publishers, Dordrecht, The Netherlands, pp. 285–296, 1999.

(Burg, 2000) Van Der Burg, J. Building and advanced particle system. *http://www.gamasutra.com/view/feature/3157/building_an_advanced_particle_.php*. Última visita el 23/12/2010.

(Campos, 2002) de Campos, L. M., Fernández-Luna, J. M., Gámez, J. A., Puerta, J. M. Ant colony optimization for learning Bayesian networks. International Journal of Approximate Reasoning, vol. 31, no. 3, pp. 291–311, 2002.

(Cañamero, 1997) Cañamero, D. A Hormonal model of emotions for behaviour control. In Fourth European conference on artificial life, ECAL' 97, pp 1–10, Brighton, UK, 1997.

(Castelfranchi, 1995) Castelfranchi, C. Guarantees for autonomy in cognitive agent architecture. In Wooldridge, M. and Jennings, N. R. eds., Intelligent Agents: Theories, Architectures, and Languages (LNAI Volume 890), pp 56–70. Springer-Verlag: Heidelberg, Germany. 1995.

(Castro, 2002a) de Castro, L. N., Von Zuben, F. J. Learning and optimization using the clonal selection principle. IEEE Transactions on Evolutionary Computation, 6(3), pp. 239–251, 2002.

(Castro, 2002b) de Castro, L.N., Timmis, J. Artificial Immune Systems: A New Computational Intelligence Approach. London, Springer, 2002.

(Catala, 2007) Catala, A., Jaen, J., Mocholi, J.A. Strategies for accelerating ant colony optimization algorithms on graphical processing units. En: Proceedings of the. IEEE Congress on Evolutionary Computation, pp.492-500, 2007.

(Chen, 2004) Chen, H. An Intelligent Broker Architecture for Pervasive Context-Aware Systems. PhD Thesis, University of Maryland, Baltimore County, USA, 2004.

(Chow, 1968) Chow, C.K., Liu, C.N. Approximating Discrete Probability Distributions with Dependence Trees. IEEE Transactions on Information Theory, vol. IT-14, No. 3, 1968.

(Clore, 1994) Clore, G. L., Schwarz, N., Conway, M. Affective causes and consequences of social information processing. In: Wyer, R.S. & Srull, K. (eds.): Handbook of social cognition, 2nd ed., pp 323-417. Erlbaum, 1994.

(Coello, 2008) Coello, C. A. Introducción a la computación evolutiva. 2008. Disponible en: http://delta.cs.cinvestav.mx/~ccoello/compevol/clase5-2008.pdf

(Colorni, 1994) Colorni, A., Dorigo, M., Maniezzo, V., Trubian, M. Ant System for job shop scheduling. JORBEL - Belgian Journal of Operations Research, Statistics and Computer Science, 34(1), 39–53, 1994.

(Cordón, 2000) Cordón, O., Fernández de Viana, I., Herrera, F., Moreno, L. A new ACO model integrating evolutionary computation concepts: The Best-Worst Ant System. En M. Dorigo, M. Middendorf, y T. Stützle, editores, Abstract proceedings of ANTS2000 - From Ant Colonies to Artificial Ants: A series of International Workshops on Ant Algorithms, pp 22-29, 2000.

(Costa, 1997) Costa, D., Hertz, A. Ants can colour graphs. Journal of the Operational Research Society, vol. 48, pp. 295–305, 1997.

(Cowie, 2001) Cowie, R.; Douglas-Cowie, E.; Tsapatsoulis, N.; Votsis, G.; Kollias, S.; Fellenz, W.; Taylor, J.G. Emotion recognition in human-computer interaction. IEEE Journal on Signal Processing Magazine, vol 18, 1, pp 32-80, 2001.

(Csikszentmihalyi, 1985) Csikszentmihalyi, M., Massimini, F. On the psychological selection of bio-cultural information. New Ideas in Psychology, 3, pp 115-138, 1985.

(Csikszentmihalyi, 1988) Csikszentmihalyi, M., Csikszentmihalyi, I.: Optimal Experience. Psychological Studies of Flow in Consciousness. Cambridge University Press, 1988.

(Damasio, 1994) Damasio, A. Descartes' Error: Emotion, Reason, and the Human Brain. New York: Gosset/Putnam, 1994.

(Darwin, 1859) Darwin, C. On the Origin of Species by Means of Natural Selection, Murray, London, 1859.

(Davalo, 1991) Davalo, E., Naim, P. Neural Networks. New York, Macmillan, 1991.

(Davis, 1991) Davis, L. (ed). Handbook of Genetic Algorithms. New York, Van Nostrand Reinhold, 1991.

(Davison, 2004) Davison, R., Martinsons, M., Kock, N. Principles of canonical action research. Journal of Information Systems, 14, pp 65-86, 2004.

(Deci, 1985) Deci, E.L., Ryan, R. M.: Intrinsic Motivation and Self-Determination in Human Behavior. Plenum Press, 1985.

(Dey, 2000) Dey, A.K., Abowd, G.D. Towards a Better Understanding of Context and Context-Awareness. CHI 2000 Workshop on the What, Who, Where, When, and How of Context-Awareness, 2000.

(Di Caro, 1998) Di Caro, G., Dorigo, M. AntNet: Distributed stigmergetic control for communications networks. Journal of Artificial Intelligence Research, vol. 9, pp. 317–365, 1998.

(Doerner, 2003) Doerner, K., Hartl, R. F., Reimann, M. CompetAnts for problem solving – the case of full truckload transportation. Central European Journal for Operations Research and Economics, vol. 11, no. 2, pp. 115–141, 2003.

(Doerner, 2006) Doerner, K., Gutjahr, W. J., Hartl, R. F., Strauss, C., Stummer, C. Pareto ant colony optimization in multiobjective project portfolio selection with ILP preprocessing. European Journal of Operational Research, vol. 171, no. 3, pp. 830–841, 2006.

(Dorigo, 1991) Dorigo, M., Maniezzo, V., Colorni, A. Positive feedback as a search strategy. Dipartimento di

Elettronica, Politecnico di Milano, Italy. Tech. Rep. 91-016, 1991.

(Dorigo, 1992) Dorigo, M. Optimization, Learning and Natural Algorithms, Ph.D. Thesis, 1992.

(Dorigo, 1996) Dorigo, M., Maniezzo, V., Colorni, A. The Ant System: Optimization by a Colony of Cooperating Agents. IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, vol. 26, no. 1, 1996.

(Dorigo, 1997a) Dorigo, M., Gambardella, L. M. Ant Colonies for the Travelling Salesman Problem. BioSystems, vol. 43, pp. 73-81, 1997.

(Dorigo, 1997b) Dorigo, M., Gambardella, L. M. Ant Colony System: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation, 1(1), 53–66, 1997.

(Dorigo, 1999) Dorigo, M., Di Caro, G. The Ant Colony Optimization Meta-Heuristic. En D. Corne, M. Dorigo and F. Glover (eds.), New Ideas in Optimization, pp 11-32, 1999.

(Dorigo, 2003) Dorigo, M., Stützle, T. The ant colony optimization metaheuristic: Algorithms, applications and advances. En: F. Glover and G. Kochenberger, eds., Handbook of Metaheuristics, pp 251-285. Kluwer Academic Publishers, 2003.

(Dorigo, 2004) Dorigo, M., Stützle, T. Ant colony optimization. The MIT Press, 2004.

(Dorigo, 2006) Dorigo, M., Birattari, M., Stützle, T. Ant Colony Optimization: Artificial Ants as a Computational Intelligence Technique. IEEE Computational Intelligence Magazine, vol. 1, 9, pp. 28-39, 2006.

(Ducatelle, 2005) Ducatelle, F., Di Caro, G., Gambardella, L. M. Using ant agents to combine reactive and proactive strategies for routing in mobile ad hoc networks. International Journal of Computational Intelligence and Applications, vol. 5, no. 2, pp. 169–184, 2005.

(Eberly, 2003) Eberly, D.H. Game Physics. Morgan Kaufmann, 2003.

(El-Nasr, 1999) El-Nasr, M. S., Ioerger, T., Yen, J., Parke, F., House, D. Emotionally expressive agents. In Proceedings of Computer Animation, pp 48–57, Geneva, Switzerland, 1999.

(El-Nasr, 2000) El-Nasr, M. S., Yen, J., Ioerger, T.R. FLAME – Fuzzy logic adaptive model of emotions. In Autonomous agents and multi-agent systems, vol 3, Kluwer Academic Publishers, Netherlands, pp. 219–257, 2000.

(Englebart, 1963) Englebart, D. A conceptual framework for the augmentation of man's intellect. In P. W. Howerton, D. C. Weeds (eds.), Vistas in information handling, vol. 1, pp 1-29. Washington, DC, 1963.

(Fahy, 2004) Fahy, P., Clarke, S. CASS – a middleware for mobile context-aware applications. En: Proc. Mobisys 2004, Workshop on Context Awareness, 2004.

(Fenet, 2003) Fenet, S., Solnon, C. Searching for maximum cliques with ant colony optimization. En: Applications of Evolutionary Computing, Proceedings of EvoWorkshops 2003, LNCS, G. R. Raidl et al., Eds., vol. 2611. Springer Verlag, pp. 236–245, 2003.

(Forrest, 1994) Forrest, S., Perelson, A.S., Allen, L., Cherukuri, R. Self-nonself discrimination in a computer. En: Proceedings of the IEEE Symposium on Research in Security and Privacy, 1994.

(Fredrickson, 2001) Fredrickson, B. L., Branigan, C.: Positive Emotions. In: Mayne, T.J. & Bonnano, G.A. (eds.): Emotion: Current issues and future directions, pp 123-151. Guilford Press, 2001.

(Fülöp, 1998) Fülöp, J. Introduction to Decision Making Methods. Evergreen State College, 2008.

(Gambardella, 1995) Gambardella, L.M., Dorigo, M. Ant-Q: A reinforcement learning approach to the traveling salesman problem. En: A. Prieditis and S. Russell (eds.), Proceedings of the Twelfth International Conference on Machine Learning (ML-95), Morgan Kaufmann Publishers, Palo Alto, CA, pp. 252–260, 1995.

(Gambardella, 1996) Gambardella, L.M., Dorigo, M. Solving symmetric and asymmetric TSPs by ant colonies. En: Proceedings of the 1996 IEEE International Conference on Evolutionary Computation (ICEC'96), IEEE Press, Piscataway, NJ, pp. 622–627, 1996.

(Gambardella, 1997) Gambardella, L.M., Dorigo, M. HAS-SOP: An hybrid Ant System for the sequential ordering problem. Technical Report IDSIA-11 -97, 1997.

(Gambardella, 1999a) Gambardella, L.M., Taillard, É. D., Agazzi, G. MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows. En: D. Corne, M. Dorigo and F. Glover (eds.), New Ideas in Optimization. McGraw Hill, UK, pp. 63–76, 1999.

(Gambardella, 1999b) Gambardella, L.M., Taillard, É. D., Dorigo, M. Ant colonies for the quadratic assignment problem. Journal of the Operational Research Society, 50(2), 167–176, 1999.

(Genesereth, 1994) Genesereth, M. R., Ketchpel, S. P. Software agents. Communications of the ACM, vol 37, 7, pp 48–53, 1994.

(Glover, 1997) Glover, F., Laguna, M. Tabu Search. Boston, Kluwer Academic Publishers.

(Goldberg, 1989) Goldberg, D. Genetic Algorithms in Search, Optimization and Machine Learning. Reading, Mass, Addison-Wesley, 1989.

(Goss, 1989) Goss, S., Aron, S., Deneubourg, J. L., Pasteels, J. M. Self-organized shortcuts in the Argentine ant. Naturwissenschaften, 76, pp 579-581, 1989.

(Graf, 2003) Graf, C. Digital Characters in the Real World. Report, University of Magdeburg, Germany, 2003.

(Graf, 2005) Graf C. Digital Characters as Affective Interfaces. In Peter, C., Beale, R., Crane, E., Axelrod, L., Blyth, G. (eds.): Emotion in HCI, Joint Proceedings of the 2005, 2006, and 2007 International Workshops, pp 34, 2008.

(Grasse, 1944) Grasse, P.-P. Recherches sur la biologie des termites champignonnistes (Macrotermitina). Ann. Sc. Nat., Zool. Biol. anim., 6, 97, 1944.

(Grasse, 1959) Grasse, P.-P. La reconstruction du nid et les coordinations interindividuelles chez Bellicositermes Natalensis et Cubitermes Sp. La theorie de la stigmergie: essai d'interpretation du comportement des termites constructeurs, Insectes Sociaux, 6, 41, 1959.

(Greene, 1988) Greene, T. R., Noice, H. Influence of positive affect upon creative thinking and problem solving in children. Psychological Reports, 63, pp 895-898, 1988.

(Grudin, 2008) Grudin, J. A moving target: the evolution of HCI. En Jacko, J.A., Sears, A. (eds.) The Human-computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications. Lawrence Erlbaum Associates, pp 1-24, 2008.

(Gruninger, 2002) Gruninger, M., Lee, J. Ontology applications and design. Communications of the ACM, 45(2), pp. 39–41, 2002.

(Gu, 2004) Gu, T., Pung, H.K., Zhang, D.Q. A middleware for building context-aware mobile services. En: Proceedings of IEEE Vehicular Technology Conference (VTC 2004), Milan, Italy, 2004.

(Hansen, 1999) Hansen, P. Mladenovic, N. An introduction to variable neighborhood search. En: S. Voss, S.Martello, I. H. Osman, and C. Roucairol (Eds.), MetaHeuristics - Advances and Trends in Local Search Paradigms for Optimization. Kluver Academic Publishers, 1999.

(Harris, 1998) Harris, R. Introduction to Decision Making, VirtualSalt, 1998. Disponible en: http://www.virtualsalt.com/crebook5.htm

(Herbon, 2006) Herbon, A., Oehme, A., Zentsch, E. Emotions in ambient intelligence–an experiment on how to measure affective states. In: 20th BCS HCI Group conference, London, 2006.

(Holland, 1975) Holland, J. Adaptation in Natural and Artificial Systems. The University of Michigan, 1975.

(Humaine, 2007) Humaine. D3k Pre-completion report on blueprint volume. Workpackage 3 deliverable, 2007. Disponible en: http://emotion-

research.net/projects/humaine/deliverables

(Ichishita, 2007) Ichishita, T., Fujii, R. H. Performance Evaluation of a Temporal Sequence Learning Spiking Neural Network. En: Proceedings of the 7th IEEE international Conference on Computer and information Technology 2007.

(Inoue, 1996) Inoue, K., Kawabata, K., Kobayashi, H. On a Decision Making System with Emotion. IEEE Int. Workshop on Robot and Human Communication, Tokyo, Japan, pp. 461-465, 1996.

(Isen, 2000) Isen, A. M. Positive affect and decision making. In: Lewis, M., Haviland, J. M. (eds.): Handbook of emotions, 2nd ed., pp 417-435. Guilford Press, 2000.

(Kishner, 2005) Kishner, S. Modeling of multivariate time series using hidden Markov models. PhD Thesis Dissertation, 2005.

(Kock, 2001) Kock, N., Lau, F. Information Systems Action Research: Serving Two Demanding Masters. Information Technology & People (Special Issue on Action Research in Information Systems), (14)1, pp 6-11, 2001.

(Korb, 2006) Korb, O., Stützle, T., Exner, T. E. PLANTS: Application of ant colony optimization to structure-based drug design. En: Ant Colony Optimization and Swarm Intelligence, 5th International Workshop, ANTS 2006, LNCS, M. Dorigo et al., Eds., vol. 4150. Springer Verlag, pp. 247–258, 2006.

(Kruchten, 2000) Kruchten, P. The Rational Unified Process: An Introduction. Addison-Wesley, 2000.

(Krüger, 1998) Krüger, F., Merkle, D., Middendorf, M. Studies on parallel ant system for the BSP model. Manuscrito no publicado. 1998.

(Lazzaro, 2008) Lazzaro, N. Why we play: Affect and the fun of games: Designing emotions for games, entertainment interfaces and interactive products. The Human-computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications. Eds. Julie A. Jacko, Andrew Sears. Lawrence Erlbaum

Associates pp 679-700, 2008.

(LeDoux, 1996) LeDoux, J. E. The emotional brain: The mysterious underpinnings of emotional life. New York: Simon & Schuster, 1996.

(Leguizamón, 1999) Leguizamón, G., Michalewicz, Z. A new version of Ant System for subset problems. En: Proceedings of CEC'99. IEEE Press, pp. 1459–1464, 1999.

(Lessing, 2004) Lessing, L., Dumitrescu, I., Stützle, T. A comparison between ACO algorithms for the set covering problem. En: The Fourth International Workshop on Ant Algorithms and Swarm Intelligence, LNCS, M. Dorigo et al., Eds., vol. 3172. Springer Verlag, pp. 1–12, 2004.

(Liang, 2003) Liang, Y. C., Smith, A. E. An Ant Colony Approach to the Orienteering Problem. Journal of the Chinese Institute of Industrial Engineers. 2003.

(Maniezzo, 1994) Maniezzo, V., Colorni, A., Dorigo, M. The Ant System applied to the quadratic assignment problem. Technical Report IRIDIA/94-28, IRIDIA, Université Libre de Bruxelles, Belgium, 1994.

(Maniezzo, 1999) Maniezzo, V. Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem. INFORMS Journal on Computing, 11(4), 358–369, 1999.

(Martens, 2006) Martens, D., Backer, M. D., Haesen, R., Baesens, B., Mues, C., Vanthienen, J. Ant-based approach to the knowledge fusion problem. En: Ant Colony Optimization and Swarm Intelligence, 5th International Workshop, ANTS 2006, LNCS, M. Dorigo et al., Eds., vol. 4150. Springer Verlag, pp. 84–95, 2006.

(Marzano, 2003) Marzano, S., Aarts, E. The New Everyday View on Ambient Intelligence. Uitgeverij 010 Publishers, 2003.

(Maurer, 2005) Maurer, A., Hersch, M., Billard, A. Extended Hopfield Network for Sequence Learning: Application to Gesture Recognition. Artificial Neural Networks: Biological Inspirations – ICANN 2005. Lecture Notes

in Computer Science, 2005.

(Merkle, 2002) Merkle, D., Middendorf, M., Schmeck, H. Ant colony optimization for resource-constrained project scheduling. IEEE Transactions on Evolutionary Computation, vol. 6, no. 4, pp. 333–346, 2002.

(Merkle, 2003) Merkle, D., Middendorf, M. Ant colony optimization with global pheromone evaluation for scheduling a single machine. Applied Intelligence, vol. 18, no. 1, pp. 105–111, 2003.

(Michaliewicz, 1992) Michaliewicz, Z. Genetic Algorithms + Data Structures = Evolutionary Programs. Springer-Verlag, 1992.

(Michel, 1998) Michel, R., Middendorf, M. An island model based ant system with lookahead for the shortest supersequence problem. En A. E. Eiben, T. Back, M. Schoenauer, and H.-P. Schwefel, editors, Proceedings of PPSN-V, Fifth International Conference on Parallel Problem Solving from Nature. Springer-Verlag, 1998.

(Middendorf, 2000) Middendorf, M., Reischle, F., Schmeck, H. Information Exchange en Multi Colony Ant Algorithms. Parallel and Distributed Computing: Proceedings of the 15th IPDPS 2000 Workshops, the 3rd Workshop on Biologically Inspired Solutions to Parallel Processing Problems (BioSP3), LNCS 1800, 2000.

(Middendorf, 2002) Middendorf, M., Reischle, F., Schmeck, H. Multi colony ant algorithms. Journal of Heuristics, 8: 3, 2002.

(Miyoshi, 2004) Miyoshi, S., Yanai, H., Okada, M. Associative memory by recurrent neural networks with delay elements. Neural Networks 17, 2004.

(Mocholi, 2005) Mocholi, J.A., Jaen, J., Canos, J.H. A grid ant colony algorithm for the orienteering problem. En: Proceedings of the 2005 IEEE Congress on Evolutionary Computation, vol.1, pp.942-949, 2005.

(Mocholi, 2006) Mocholi, J. A., Esteve, J. M., Jaén, J., Acosta, R., Xech, P. L. An Emotional Path Finding Mechanism for Augmented Reality Applications. En: 5th International

Conference on Entertainment Computing, 2006.

(Moga, 2002) Moga, S., Gaussier, P. Artificial Neural Network for Sequence Learning. En: Proceedings of International Joint Conference on Artificial Intelligence, 2002.

(Money, 2008) Money, A. G., Agius, H. Video Playing with our emotions. Emotion in HCI: Joint Proceedings of the 2005, 2006, and 2007 International Workshops, pp 168-171, 2008.

(Navarro, 1999) Navarro, G., Sinclair, M.C. Ant colony optimisation for virtual wavelength- path routing and wavelength allocation. En: Proceedings of the 1999 Congress on Evolutionary Computation (CEC'99). IEEE Press, Piscataway, NJ, pp. 1809–1816, 1999.

(Newell, 1972) Newell, A., Simon, H.A. Human problem solving. Englewood Cliffs, NJ. Prentice-Hall, 1972.

(Norman, 1982) Norman, D. A. Steps toward a cognitive engineering: Design rules based on analyses of human error. Proceedings of the Conference on Human Factors in Computing Systems, pp 378-382. New York, ACM, 1982.

(Norman, 1983) Norman, D. A. Design principles for human computer interfaces. Proceedings of Computer Human Interaction (CHI'83), pp 1-10. New York, ACM, 1983.

(Norman, 1988) Norman, D. A. Psychology of everyday things. New York, Basic Books, 1988.

(Norman, 2002) Norman, D. A. Emotion and design: Attractive things work better. Interactions Magazine, ix (4), 36-42, 2002.

(Norman, 2004) Norman, D. A. Emotional Design: Why we love (or hate) everyday things. New York, Basic Books, 2004.

(Ortony, 1988) Ortony, A., Clore, G.L., Collins, A. The cognitive structure of emotions. Cambridge University Press, Cambridge, UK, 1988.

(Padak, 1994) Padak, N., Padak, G. Guidelines for Planning Action Research Projects. Research to Practice, Ohio Literacy Resource Center, 1994. Disponible en http://archon.educ.kent.edu/Oasis/Pubs/0200-

08.html.

(Parpinelli, 2002) Parpinelli, R. S., Lopes, H. S., Freitas, A. A. Data mining with an ant colony optimization algorithm. IEEE Transactions on Evolutionary Computation, vol. 6, no. 4, pp. 321–332, 2002.

(Pasteels, 1987) Pasteels, J. M., Deneubourg, J.-L., Goss, S. Self-organization mechanisms in ant societes (I): Trail recruitment to newly discovered food sources. Experientia Supplementum, 54, 1987

(Pew, 2003) Pew, R.W. Evolution of Human-Computer Interaction: from memex to Bluetooth and beyond. En Jacko, J.A., Sears, A. (eds.) The Human-computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications. Lawrence Erlbaum Associates, pp 1-18, 2003.

(Picard, 1997) Picard, R. Affective Computing. MIT Press, 1997.

(Picard, 1999) Picard, R. Affective Computing for HCI. Proceedings HCI. Munich, Germany, 1999.

(Picard, 2002) Picard, R., Wexelblat, A., Nass, C. Future interfaces: social and emotional. CHI '02: CHI '02 extended abstracts on Human factors in computing systems, pp 698-699, 2002.

(Poggio, 1990) Poggio, T., Girosi, F. Networks for Approximation and Learning. Proceedings IEEE, vol.78, 9, pp 1481-1497, 1990.

(Pohle, 2005) Pohle, T., Pampalk, E., Widmer G. Generating Similarity-Based Playlists Using Traveling Salesman Algorithms. En: Proc. of the 8th Int. Conference on Digital Audio Effects (DAFx'05), Madrid, Spain, 2005.

(Pomerol, 1997) Pomerol, J.C. Artificial Intelligence And Human Decision Making. European Journal of Operational Research, vol. 99, 1, pp 3-25, 1997.

(Ramalhinho, 1998) Ramalhinho, H., Serra, D. Adaptive Approach Heuristics for the Generalized Assignment Problem. Economics and Business Working Papers Series, artículo nº 288, 1998.

(Reeves, 1993) Reeves, C. Modern Heuristic Techniques for Combinatorial Problems, Blackwell Scientific Publications, 1993.

(Reimann, 2004) Reimann, M., Doerner, K., Hartl, R. F. D-ants: Savings based ants divide and conquer the vehicle routing problems. Computers & Operations Research, vol. 31, no. 4, pp. 563–591, 2004.

(Salber, 1999) Salber, D., Dey, A.K., Abowd, G.D. The context toolkit: aiding the development of context-aware applications. En: Proceedings of the ACM CHI, Pittsburgh, PA, pp.434-441, 1999.

(Schwarz, 1990) Schwarz, N. Feelings as information: Informational and motivational functions of affective states. In: Higgins, E. T., Sorrentino, R. (eds.): Handbook of motivation and cognition: Foundations of social behavior, vol 2, pp 527-561. Guilford, 1990.

(Schwarz, 1996) Schwarz, N., Clore, G.L. Feelings and phenomenal experiences. In: Higgins, E.T., Kruglanski, A. (eds.): Social psychology: Handbook of basic principles, pp 433-465. Guilford, 1996).

(Sellen, 2009) Sellen, A., Rogers, Y., Harper, R., Rodden, T. Reflecting Human Values in the Digital Age. Communications of the ACM, vol 52, 3, pp 58-66, 2009.

(Settimi, 1998) Settimi, R., Smith, J.Q. On the geometry of Bayesian graphical models with hidden variables. En: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, pp. 472-479. Morgan Kaufman, San Mateo, CA, 1998.

(Shmygelska, 2005) Shmygelska, A., Hoos, H. H. An ant colony optimisation algorithm for the 2d and 3d hydrophobic polar protein folding problem. BMC Bioinformatics, vol. 6:30, 2005.

(Shneiderman, 2004) Shneiderman, B. Designing for fun: how can we design user interfaces to be more fun?. Interactions, vol.11, 5, pp 48-50, 2004

(Simon, 1967) Simon, H. A. Motivational and emotional controls of

cognition. Psychology Review, 74, pp 29-39, 1967.

(Socha, 2002) Socha, K., Knowles, J., Sampels, M. A MAX-MIN ant system for the university timetabling problem. En: Ant Algorithms, 3rd International Workshop, ANTS 2002, LNCS, M. Dorigo et al. (Eds.), vol. 2463. Berlin, Germany: Springer Verlag, p. 1, 2002.

(Solnon, 2000) Solnon, C. Solving permutation constraint satisfaction problems with artificial ants. En: Proceedings of ECAI'2000. Amsterdam, The Netherlands: IOS Press, pp. 118–122, 2000.

(Stützle, 1997) Stützle, T., Hoos, H. H. The Ant System and local search for the traveling salesman problem. En: T. Bäck, Z. Michalewicz and X. Yao (eds.), Proceedings of the 1997 IEEE International Conference on Evolutionary Computation, IEEE Press, Piscataway, NJ, pp. 309–314, 1997.

(Stützle, 1998a) Stützle, T. An ant approach to the flow shop problem. En: Proceedings of the 6th European Congress on Intelligent Techniques & Soft Computing (EUFIT'98), vol. 3, Verlag Mainz, pp. 1560–1564, 1998.

(Stützle, 1998b) Stützle, T. Parallelization strategies for ant colony optimization. En: A. E. Eiben, T. Back, M. Schoenauer, and H.-P. Schwefel (eds.), Proceedings of PPSN-V, Fifth International Conference on Parallel Problem Solving from Nature, Springer-Verlag, 1998.

(Stützle, 2000) Stützle, T., Hoos, H. H. MAX-MIN Ant System. Future Generation Computer Systems, vol 16, 8, pp 889-914, 2000.

(Talbi, 1999) Talbi, E.G., Roux, O., Fonlupt, C., Robilliard, D. Parallel Ant Colonies for Combinatorial Optimization Problems. IPPS/SPDP Workshops, 1999.

(T'kindt, 2002) T'kindt, V., Monmarché, N., Tercinet, F., Laügt, D. An ant colony optimization algorithm to solve a 2-machine bicriteria flowshop scheduling problem. European Journal of Operational Research, vol. 142, no. 2, pp. 250–257, 2002.

(Tyrrell, 1993) Tyrrell, T. Computational Mechanisms for Action

Selection. PhD Thesis. University of Edinburgh, 1993.

(Velasquez, 1996) Velasquez, J. D. Cathexis: A computational model for the generation of emotions and their influence in the behaviour of autonomous agents. S.M. Thesis. Department of Electrical Engineering and Computer Science, MIT, 1996.

(Velasquez, 1997) Velasquez, J. D. Modelling emotions and other motivations in synthetic agents. In Proceedings of American Association for Artificial Intelligence, pp 10–15, 1997.

(Velasquez, 1998a) Velasquez, J. Modeling Emotion-Based Decision Making. In L. D. Cañamero ed., Emotional and Intelligent: The Tangled of Knot of Cognition, AAAI Fall Symposium Series, pp 164-169, Orlando, FL., 1998.

(Velasquez, 1998b) Velasquez, J. A computational framework for emotion-based control. In Proceedings of the Grounding Emotions in Adaptive Systems workshop, SAB '98, Zurich, Switzerland, 1998.

(Wadsworth, 1998) Wadsworth, Y. What is Participatory Action Research? Action Research International, Paper 2. 1998. Disponible en: http://www.scu.edu.au/schools/gcm/ar/ari/p-ywadsworth98.html

(Wang, 2003) Wang, D.: Temporal pattern processing. The Handbook of Brain Theory and Neural Network 2, 2003.

(Weiser, 1991) Weiser, M. The computer for the 21st century. Scientific American, pp. 94-10, September, 1991.

(Wooldridge, 1995) Wooldridge, M., Jennings, N. Intelligent Agents: Theory and Practice. The Knowledge Engineering Review, vol 10, 2, pp 115-152, 1995.