Memoria del Programa Oficial de Postgrado
Tecnologías Informáticas

Máster en Ingeniería del Software e Inteligencia Artificial

2006/2007

José Manuel García Nieto
email: jnieto@lcc.uma.es
www.garcianieto.net

Universidad de Málaga, Septiembre 2007

Índice general

Introducción

En este documento se resumen los conocimientos adquiridos y trabajos realizados por el autor para la obtención del Máster de postgrado titulado "Ingeniería del Software e Inteligencia Artificial". Este Máster ha sido desarrollado por el departamento de Lenguajes y Ciencias de la Computación de la Universidad de Málaga durante el periodo 2006/2007, bajo la coordinación del Doctor Catedrático Don José Muñoz Pérez.

El objetivo principal de este programa consiste en orientar al alumno e inculcarle una base de conocimientos lo suficientemente sólida, con la intención de introducirle en el mundo de la investigación científica y la posterior consecución de su proyecto de tesis doctoral. Si bien, puede estar dirigido a personal tanto del ámbito académico como del mundo empresarial, ya que se muestra una recopilación de los avances más significativos realizados dentro del campo de estudio, las tendencias actuales y las líneas de investigación más prometedoras. La oferta de asignaturas es lo suficientemente amplia para que el alumno pueda elegir una formación exclusiva en Ingeniería de Software o en Inteligencia Artificial, campos en los que el departamento organizador del Máster tiene más experiencia investigadora y docente. Del mismo modo, tras haber cursado dichas asignaturas, el alumno realiza un trabajo de investigación tutelado en relación con alguna o algunas líneas de trabajo presentado en las asignaturas. Como complemento al programa de Máster, de manera paralela a su desarrollo, se realizan cursos de especialización, conferencias y charlas con las que se profundiza en las materias estudiadas y se exponen los últimos avances realizados.

Perfil del Alumno

El estudiante de Máster José Manuel García Nieto realizó sus estudios de Ingeniería Informática en la Escuela Técnica Superior de Ingeniería Informática de la Universidad de Málaga, obteniendo el correspondiente título en Septiembre de 2006. Realizó el Proyecto de Fin de Carrera titulado "Algoritmos Basados en Cúmulos de Partículas Para la Resolución de Problemas Complejos" bajo la dirección del Doctor Don Enrique Alba Torres, con quien continúa trabajando, siendo su director del presente Máster.

Durante el periodo 2004/2005 trabajó en tareas de colaboración con el Departamento de Lenguajes y Ciencias de la Computación mediante una beca de colaboración con el departamento que se le concedió en Noviembre de 2004 por parte del Ministerio de Educación y Ciencia.

En Diciembre de 2003 obtuvo el título de Ingeniero Técnico en Informática de Gestión por la misma escuela. Realizó en Proyecto de Fin de Carrera titulado "Diseño de un Servicio de Optimización en Internet", dirigido por el Doctor Don Enrique Alba Torres.

Actualmente, está desarrollando su trabajo de investigación colaborando en los proyectos OPLINK [16] (TIN2005-08818-C04-01) de ámbito nacional y CARLINK [10] (CP3-005) de ámbito europeo. Los principales temas de investigación del estudiante de Máster están relacionados con la aplicación de técnicas metaheurísticas para la resolución de problemas obtenidos de diversos dominios como la bioinformática y las telecomunicaciones.

Las Asignaturas cursadas por el alumno en el programa de Máster fueron las siguentes:

- Bases Metodológicas de los Sistemas Software. Un Enfoque Basado en Coordinación
- Métodos Para la Construcción de Software Fiable
- Técnicas de Bases de Datos y de Programación Distribuida Para la Web
- Fundamentos Teóricos de la Inteligencia Artificial
- Algoritmos Evolutivos
- Programación de Sistemas Multiagentes
- Sistemas Neuronales y Neurodifusos
- Servicios Avanzados Multimedia Basados en Componentes

Contenido del Documento

En este documento se presenta la memoria del programa de Máster "Ingeniería del Software e Inteligencia Artificial" (2006/2007), en el que el alumno ha estado cursando las asignaturas anteriormente listadas y ha realizado las labores de investigación tutorizadas.

La memoria se encuentra dividida en dos partes. La primera parte resume los contenidos adquiridos durante el periodo de docencia y los trabajos realizados en las asignaturas. En esta parte se dedica un capítulo a cada asignatura donde se describen los objetivos principales de la asignatura, el contenido docente y los trabajos realizados para su superación. En la segunda parte se describen las labores de investigación llevadas a cabo por el alumno durante la realización del Máster. En esta parte, además del trabajo de investigación tutelado se incluyen artículos, informes técnicos, estancias realizadas en otros centros de investigación y proyectos en los que ha participado el alumno.

Parte I

Docencia

Capítulo 1

Bases Metodológicas de los Sistemas Software. Un Enfoque Basado en Coordinación

Profesores:	José María Troya Linero y Bartolomé Rubio Muñoz		
Créditos:	5	**Cuatrimestre:**	1
Tipo:	Metodológico	**Carácter:**	Obligatoria

1.1. Objetivos y organización

En esta asignatura se presentan las bases y conceptos iniciales sobre el diseño y desarrollo aplicaciones sobre sistemas paralelos y distribuidos bajo el paradigma de la coordinación. En una primera parte de la asignatura se imparte, por parte de los profesores, la materia correspondiente a la docencia, resumida a continuación:

- Computación e Interacción. El Paradigma de la Coordinación
- Modelos y Lenguajes de Coordinación. Una Clasificación
- La Coordinación en las Nuevas Tecnologías Software
- Entornos de Programación para Computación de Alto Rendimiento
- Sistemas Multiagentes
- Middleware para Redes Ad-Hoc
- Trabajos relacionados dentro del grupo GISUM. Entre los que se encuentra el modelo de coordinación TCM y el Lenguaje de coordinación BCL/DIP

En la segunda parte de la asignatura, el alumno elabora un trabajo relacionado con los modelos de coordinación y sistemas distribuidos a partir de material e información suministrados por el profesor. Finalmente, cada trabajo se expone en clase durante las últimas sesiones.

1.2. Docencia

En la actualidad, debido a los grandes avances realizados en el campo de las telecomunicaciones y especialmente al surgimiento de Internet, existen una gran cantidad de aplicaciones informáticas que demandan el uso de sistemas paralelos y distribuidos en los que un conjunto de subsistemas o "actores" colaboran para realizar una tarea común. Esta colaboración, realizada mediante las interacciones entre los actores independientes requiere de una *coordinación*. La idea clave es la separación de la computación y la coordinación, siendo un modelo de coordinación el "pegamento" que une las diferentes piezas activas en la construcción de un programa.

Se pueden clasificar tres categorías principales de modelos de coordinación: Los modelos *dirigidos por datos*, los modelos *dirigidos por control* y los *modelos híbridos*, dependiendo cada uno del grado de acceso que tengan las entidades coordinadoras a los datos intercambiados entre ellas. Inicialmente surgió el modelo dirigido por datos *Linda*, siendo fuente de inspiración para la creación de otros modelos (Linda-Like) como ShaDe, Gamma, PCN, CC++, Synchronisers y ActorSpaces. Linda está basado en un espacio de tuplas donde las entidades que colaboran pueden depositar o retirar tuplas. Por medio de las tuplas las entidades se comunican entre sí y se coordinan para realizar una tarea determinada. Dentro de los modelos dirigidos por control podemos mencionar IWIN, Manifold o TOOLBUS. Finalmente, mediante el modelo híbrido se trata de combinar los dirigidos por datos y los dirigidos por control. Como ejemplos podemos mencionar TuCSoN, LuCe y MARS.

La coordinación se viene aplicando con éxito en diferentes campos dentro de las nuevas tecnologías software como los Entornos de Programación para la Computación de Alto Rendimiento (CAR), Los Sistemas Multiagentes y el Middleware para Redes Ad-Hoc. Tecnologías en las que se utilizan conceptos de unidades software como los componentes y los agentes con cierto grado de autonomía y poder de reacción. Estos componentes/agentes deben tener la capacidad de comunicarse e interaccionar bajo una determinada coordinación. Del mismo modo, tras la aparición de los diferentes modelos de redes ad-hoc, como Peer-to-Peer, las redes MANETs, las redes de Sensores, etc, se hace imprescindible el desarrollo de software middleware para la monitorización y control de los diferentes nodos de las redes.

Por último, cabe mencionar una serie de modelos de coordinación desarrollados por el grupo de investigación *GISUM* de la Universidad de Málaga. En primer lugar, el modelo de coordinación *TCM* aporta un mecanismo de comunicación y sincronización heredado de la Programación Lógica Concurrente. Realiza la interconexión de canales de forma dinámica con la capacidad de ma-

nejo de excepciones, control y monitorización de procesos. En segundo lugar, el modelo *BCL*, que mediante el uso de dominios ofrece un lenguaje apropiado para la resolución de problemas numéricos. Finalmente, el modelo *DIP*, extensión del anterior, utiliza un patrón de canales PIPE para el encauzamiento de tareas.

1.3. Trabajo Realizado: New Research Challenges in Ad Hoc Wireless Networks

El trabajo llevado a cabo y expuesto en clase, tuvo como objetivo la realización de una recopilación sobre los nuevos avances realizados en las redes ad-hoc inalámbricas (wireless), así como la descripción de los nuevos tipos de redes que están surgiendo y clasificaciones de las mismas. La literatura recomendada en este trabajo consta principalmente de los trabajos desarrollados por el grupo de investigación de Mario Gerla y en sitios web de aplicaciones y proyectos científicos relacionados con esta materia.

Una red inalámbrica ad hoc (Wireless Ad Hoc Network) consiste en una colección de nodos (terminales) autónomos que se comunican entre ellos mediante enlaces inalámbricos (wireless) formando una red de multiples saltos (multihop) y manteniendo la conectividad de manera descentralizada. Esta es la definición dada por el Instituto Nacional de Estándares y Tecnología de los Estados Unidos (NIST) [6], en el apartado de tecnologías de redes avanzadas. Debido a que los nodos en este tipo de red se comunican a través de enlaces inalámbricos, están sujetos a los efectos de la comunicación por radio, como pueden ser: el ruido, la pérdida de intensidad, las interferencias, etc. Además, típicamente los enlaces inalámbricos tienen menor ancho de banda que los enlaces en una red cableada. Por otra parte, cada nodo en una red wireless ad hoc tiene funciones tanto de host terminal como de nodo encaminador (router), por lo que el control de la red está completamente distribuido entre los nodos. Esto hace que la topología de este tipo de redes sea generalmente dinámica, ya que la conectividad entre nodos puede variar en el tiempo por eventuales salidas y entradas de nodos a la red, e incluso la posibilidad de la existencia de nodos móviles.

Podemos clasificar los diferentes tipos de redes wireless ad hoc desde el punto de vista de la movilidad de los dispositivos que integran la red, el entorno donde trabaja e incluso la aplicabilidad para la que se crea. En función de estos parámetros, diferenciamos en este trabajo las redes Mobile Ad Hoc Networks (MANETs), Wireless Ad Hoc Sensor Networks (WAHSN), Wireless Mesh Networks y las Redes Híbridas.

Las redes MANETs (Mobile Ad Hoc Networks), son un tipo de redes inalámbricas ad hoc autoconfigurables compuestas por elementos móviles. Dichos elementos actúan como routers y hosts con una topología arbitraria. La característica fundamental es la movilidad de sus elementos con lo que la topología cambia rápidamente, así que el poder de autogestión y el mantenimiento de la conexión es clave para su funcionamiento. Con el uso generalizado de dispositivos como

ordenadores portátiles, PDAs, teléfonos móviles, etc., y las tecnologías de interconexión 802.11/WI-FI y bluetooth se está extendiendo el uso de las redes MANETs y ya existen aplicaciones comerciales. Una derivación de estas son las redes ad-hoc vehiculares (VANETs), que están siendo foco de gran interés por parte de la industria. Una red inalámbrica de sensores ah hoc (WAHSN) consiste en un número de sensores repartidos en un área geográfica, en la que cada sensor tiene capacidad de comunicación wireless y está dotado de cierta inteligencia para el proceso y la distribución de la información sensorizada. La mayoría de los artículos encontrados en la literatura denominan a los dispositivos de la red como "motas" con capacidad sensorial y comunicativa. El tercer tipo consiste en las redes wireless de malla (Mesh), mediante las cuales se lleva a cabo la comunicación a través de enlaces ad hoc (salto a salto) en una infraestructura totalmente descentralizada, relativamente de bajo coste y potencialmente resistente.

Para clasificar los nuevos avances realizados en este campo nos basamos en los siguientes casos de estudio: las redes vehiculares de sensores (VSNs) y las redes de sensores subacuáticas (UNSN), finalizando con la descripción de los simuladores y aplicaciones reales más novedosas.

Las redes vehiculares de sensores (VSNs) se presentan como un híbrido entre VANETs con capacidad sensorial, es decir, formando redes de sensores con las características implícitas en éstas. Los sensores están incorporados en los automóviles y proporcionan información como: el número de ocupantes de un vehículo (por análisis de la imagen), la existencia de gases tóxicos, etc. Al contrario de las redes de sensores tradicionales, las VSNs no están sujetas a restricciones estrictas de memoria, capacidad de proceso y energía. Sin embargo, las escalas típicas de VSN en grandes áreas geográficas (posiblemente millones de nodos), el volumen de datos generados y la movilidad de los vehículos hace imposible adoptar el esquema típico de las redes de sensores, donde la información obtenida tiende a ser sistemáticamente entregada en nodos centrales utilizando protocolos como la difusión directa.

Las redes ad-hoc de sensores submarinas o Underwater Sensor Network (UWSN) permiten aplicaciones para tareas tan diversas como la recolección de información oceanográfica, el control de la contaminación marina, la exploración de profundidades abisales y la prevención de desastres. Los dispositivos empleados en este tipo de redes consiste en sensores subacuáticos (distribuidos en el fondo o a diferentes profundidades) y pasarelas de superficie (boyas), que establecen la comunicación entre los anteriores y los nodos exteriores. Los nodos exteriores actúan como centros de control y sumideros de la información recolectada por la red. Como ejemplo significativo de redes UWSN que operan actualmente es el proyecto ARGO.

Finalmente, se menciona una serie de aplicaciones reales entre las que se encuentran en primer lugar simuladores de redes ad-hoc, como Qualnet o Mad-Hoc, que facilitan la creación de escenarios donde se desarrollan redes de sensores, MANETs y VANETs. Además, existen aplicaciones para el intercambio de ficheros entre vehículos como CarTorrent y FSF, sin dejar de mencionar los juegos online ad-hoc como algunas versiones de Puzzle Bubble y Medal of Honor.

Capítulo 2

Métodos para la Construcción de Software Fiable

Profesores:	Pedro Merino Gómez, Mª Mar Gallardo Melgarejo y Ernesto Pimentel Sánchez		
Créditos:	5	**Cuatrimestre:**	1
Tipo:	Fundamental	**Carácter:**	Optativa

2.1. Objetivos y organización

En esta asignatura se presenta la descripción y el uso de técnicas formales, basándose en alguna lógica o notación matemática, para el modelado y análisis del software con el objetivo final de su verificación y construcción con ausencia de errores. Los contenidos de la parte de docencia se resumen a continuación:

- Las técnicas de descripción formal en el análisis de errores software
- Modelado operacional de sistemas distribuidos
- Extensión de las técnicas de modelado operacional
- Fundamentos de la comprobación de Modelos
- Técnicas de análisis estadístico clásicas
- Introducción a la interpretación abstracta
- Integración de las técnicas de análisis estadístico y dinámico
- Herramientas basadas en la comprobación de modelos
- Aplicaciones. Fiabilidad en Lenguajes de programación y en UML

Una segunda parte de la asignatura consistió en la elaboración y exposición en clase de un trabajo por parte del alumno junto a un compañero de curso, en este caso junto a Juan José Durillo Barrionuevo.

2.2. Docencia

La presencia del errores en el software es una constante desde que se diseñaron los primeros lenguajes de programación y su depuración siempre ha sido uno de los mayores problemas en el desarrollo software. Además, esta tarea se complica a medida que las aplicaciones software se hacen más complejas. Si bien, los fallos en el software pueden ocasionar grandes pérdidas de dinero en la industria y en los procesos productivos hay algunos escenarios en los que la fiabilidad puede ser crítica. Algunos ejemplos de estos escenarios se pueden encontrar en la industria de armamento militar, las comunicaciones, la sanidad, la aviación e incluso en proyectos medioambientales. En esta asignatura se estudian técnicas para la verificación, validación y construcción de software fiable que ayude a paliar los problemas y futuros errores que se puedan producir en el software construido.

Existen diferentes clasificaciones de técnicas de verificación basadas en las definiciones de los modelos para representar el software y el tipo de análisis que desarrollen. Por ejemplo, técnicas basadas en el álgebra de procesos, técnicas de interpretación abstracta, análisis de flujo de datos, análisis de alcanzabilidad, chequeo de propiedades, análisis a priori o a posteriori etc. En este curso se destacan principalmente dos técnicas de análisis: las basadas en análisis de flujo de datos e interpretación abstracta, llamadas de *análisis estático* y la técnica de *model checking*, de *análisis dinámico*.

En el análisis estático no es necesario ejecutar el programa. El análisis se hace a partir del código del mismo. Dentro del *análisis de flujo de datos* se pueden estudiar cosas como el ámbito de las definiciones, las expresiones disponibles, las expresiones muy ocupadas o las variables libres. Por otro lado, la *interpretación abstracta* se basa en sustituir los dominios de los datos por otros dominios de menor cardinalidad conocidos como dominios abstractos de tal forma que a partir de una ejecución del programa usando estos dominios abstractos (*ejecución abstracta*) se saquen conclusiones de las ejecuciones estándar.

Como técnica de análisis dinámico tenemos el *model-checking*. Esta técnica consiste en explorar exhaustivamente todos los estados por los que puede pasar el sistema para encontrar una ejecución que incumpla una propiedad. Esta propiedad se describe mediante una fórmula de lógica temporal (LTL, CTL, CTL*). Si se encuentra, el *model-checker* devuelve esa traza, es decir, un contraejemplo de la propiedad. En la realización de dicho proceso, se representa el modelo del sistema y la negación de la fórmula mediante autómatas de Büchi. Tras realizar el producto síncrono de ambos autómatas, se busca en el nuevo autómata para encontrar un contraejemplo de la propiedad. Para llevar a cabo este análisis, una de las herramientas más populares y evolucionadas en el campo del model-checking es SPIN [13]. Esta aplicación trabaja con la especificación del sistema

a verificar en PROMELA. Existen otras herramientas como EMC, CAESAR y SMV que también realizan análisis de model checking, aunque en el mundo académico, SPIN es la más utilizada.

No obstante, el *model-checking* presenta problemas debido a la necesidad de analizar de manera exhaustiva todos los posibles estados posibles y almacenarlos. Este sobreanálisis provoca la explosión de estados que puede llegar a hacer la verificación inabordable. En este sentido, la *interpretación abstracta* permite construir un modelo más pequeño del sistema que, si bien no es exactamente igual, posee ciertas propiedades en común con el modelo original. En el *model-checking abstracto* se realiza una abstracción del modelo objeto de estudio que mantenga la propiedad que se desea verificar. En el grupo de investigación GISUM de la Universidad de Málaga se está desarrollando una herramienta denominada αSPIN basada en SPIN que realiza model-checking abstracto.

2.3. Trabajo Realizado: Técnicas Metaheurísticas Aplicadas a la Verificación de Programas Usando Model Checking

Para poder aplicar un *model checker* a un programa es necesario modelar el sistema y la propiedad que se quiere verificar (escrita en LTL) como *autómatas de Büchi*. La técnica consiste en explorar exhaustivamente todos y cada uno de los estados por los que puede pasar el sistema, con el objetivo de encontrar una ejecución que incumpla la propiedad. Esta búsqueda se realiza sobre el mencionado autómata de Büchi, puesto que éste modela tanto al sistema como a la propiedad. El problema del model checking puede simplificarse, por tanto, como un problema de búsqueda sobre un grafo.

Existen algunas herramientas como *SPIN* (véase la sección anterior) que permiten aplicar Model Checking. SPIN requiere como entradas el sistema modelado en el lenguaje *PROMELA* y la fórmula que se quiere verificar expresada en lógica temporal. La herramienta transforma el sistema y la negación de la fórmula, en los autómatas de Büchi correspondientes. Por último, realiza el producto síncrono de ambos autómatas y explora el autómata resulante en busca de un ciclo de estados alcanzables desde el estado inicial y que contenga un estado de aceptación. Si lo encuentra, existe una ejecución infinita del sistema que incumple la propiedad dada. Si no lo encuentra la verificación acaba con éxito.

El principal problema que nos encontramos cuando tratamos con este tipo de técnicas es lo que se conoce como explosión de estados: el número de estados a explorar puede ser extremadamente grande, incluso infinito. Este elevado número de estados hace que la exploración del grafo completo de estados sea inviable. Algunas técnicas han sido propuestas para tratar este tipo de problemas, entre ellas podemos mencionar métodos de orden parcial, verificación simbólica y métodos basados en simetría. Aunque estas técnicas han resuelto parcialmente el problema, y permiten aplicar Model Checking a problemas de un mayor

orden de magnitud, los problemas reales siguen siendo lo suficientemente complejos y la explosión de estados sigue siendo un problema para la verificación automática.

Cuando un problema es computacionalmente duro de resolver usando técnicas exactas, es habitual en el ámbito de las ciencias de la computación explorar el uso de heurísticas para encontrar soluciones aproximadas o converger de manera rápida hacia una solución. Quizás de manera sorpresiva, la aplicación de este típo de técnicas al model checking ha recibido bastante poca atención hasta el momento. Esto puede ser debido fundamentalmente a dos razones. Por un lado, el model checking no es un problema de optimización: el objetivo principal no es encontrar una mejor solución. Por otro lado la historia del model checking se caracteriza por el estudio exhaustivo o completitud: el objetivo principal es explorar exhaustivamente todos los posibles estados alcanzables por el sistema.

En este sentido, en base al trabajo realizado por Alba y Chicano [3], se ha estudiado la utilización de una técnica metaheurística para la verificación de software mediante model checking basada en *colonias de hormigas* (ACOs) [7], de gran impacto en la actualidad y que viene siento aplicada en un gran número de problemas reales. ACO (Ant Colony Optimization) se han aplicado con éxito a problemas de optimización combinatoria que pueden transformarse en una búsqueda dentro de un grafo. Las hormigas artificiales construyen la solución paso a paso añadiendo componentes que se representan mediante nodos del grafo. Este modelo es adecuado cuando el grafo no es muy grande (miles de nodos) pero no es viable cuando el tamaño del grafo supone un desafío para la memoria de un ordenador y no puede generarse ni almacenarse completamente en ella.

En el trabajo estudiado, se aplica el algoritmo ACO a la búsqueda de interbloqueos en el conocido *problema de los filósofos* de Edsger Dijkstra. Este problema se puede modelar en un ordenador usando n procesos que toman "n" palillos para "comer" y los liberan tras saciarse, permitiendo dejar comer a otros filósofos (problema de gran aplicación en programación concurrente). En el modelo descrito se produce un interbloqueo cuando todos los filósofos toman el palillo de su izquierda y esperan a que su compañero suelte el palillo de su derecha. En concreto se realizan cuatro pruebas con cuatro instancias distintas del problema, con 8, 12, 16 y 20 filósofos. Finalmente, se realizan comparaciones con otros algoritmos de búsqueda como búsqueda en profundidad (DFS) y búsqueda en anchura (BFS), obteniendo ACO resultados mucho más eficientes.

Capítulo 3

Técnicas de Bases de Datos y de Programación Distribuida para la Web

Profesores:	Antonio J. Nebro Urbaneja y José F. Aldana Montes		
Créditos:	5	**Cuatrimestre:**	1
Tipo:	Fundamental	**Carácter:**	Optativa

3.1. Objetivos y organización

Esta asignatura se compone de dos partes, una centrada en la aplicación de base de datos en la Web y otra en la programación distribuida. La organización del curso consta a su vez de dos partes. En la primera parte, se imparte la docencia referente a las dos partes de la asignatura, mientras que en la segunda parte, el alumno realiza un trabajo para ser expuesto en las sesiones finales. El esquema seguido para la parte de docencia es el siguiente:

- *Técnicas de Bases de Datos para la Web.*
  - El Problema de la Integración de la Información
  - Estructuración de la Información en la Web (XML)
  - Procesamiento de Consultas sobre la Web
  - La Web Semántica
- *Técnicas de Programación Distribuida.*
  - Introducción
  - Evolución de los Sistemas Distribuidos
  - Modelos, Lenguajes y Herramientas

3.2. Docencia

3.2.1. Técnicas de bases de datos para la Web

La aplicación de la tecnología de bases de datos a la web semántica trae consigo grandes retos en la investigación actual cuyo objetivo final es la representación del conocimiento y el razonamiento aplicado a la web. Se basa en la introducción de una semántica que describa y estructure de manera explícita la información y los servicios disponibles en la red, de forma que favorezca la automatización de la gestión de la información. De este modo, se consigue facilitar la integración de los recursos y su rápida localización, mediante nuevos métodos y algoritmos que agilizan la búsqueda distribuida.

Mediante la web semántica se intenta abordar el problema de la sobrecarga de la información y la heterogeneidad de fuentes resultante en problemas de interoperabilidad. A través de las *ontologías*, es posible alcanzar un entendimiento entre las partes (usuarios, desarrolladores, programas) que participan de este conocimiento común. Para lograr esos objetivos, ya existen multitud de herramientas, siendo una de las más importantes XML y sus tecnologías asociadas, ya que facilita la creación de ontologías dedicadas a cualquier temática tratada de manera estructurada, y lo que es más importante, con un formato estándar.

3.2.2. Técnicas de programación distribuida

Debido a las mejoras producidas en el hardware y sobre todo al decremento sufrido por los precios de los componentes informáticos, la utilización de grandes redes de ordenadores representa una alternativa muy atractiva en el campo de la computación paralela y distribuida. No obstante, la gran demanda tanto de computación como de espacio y gestión de almacenamiento requeridos por un gran número de aplicaciones que gestionan grandes cantidades de datos y han de hacerlo de forma eficiente, exige el uso de nuevas tecnologías, como es el caso de la computación grid o *grid computing*.

Las redes de ordenadores ofrecen la posibilidad de sumar las capacidades de cómputo de las distintas máquinas que las componen para obtener mayores prestaciones. Las rápidas y sencillas comunicaciones entre máquinas dentro del entorno local así como la accesibilidad de las mismas facilitan el proceso. Sin embargo, las necesidades actuales de cómputo han superado rápidamente las posibilidades ofrecidas por estas redes. Proyectos como la búsqueda de vida inteligente extraterrestre (SETI) o el análisis del genoma humano requieren cantidades astronómicas de cómputo, muy fuera del alcance de una red local.

El concepto de *grid computing*, surgido en los años 90, tiene como base el uso de un gran número de máquinas que funcionen como una única unidad de proceso. Esto conlleva un gran número de complicaciones que no existían para la computación en red local. Para empezar las máquinas serán mucho más heterogeneas. También es fácil que alguna máquina falle, o bien que aparezcan o desaparezcan máquinas. Este nuevo paradigma propone el uso de un entorno

muy dinámico de computación. Además debe garantizarse la seguridad de todo el proceso, cosa nada fácil cuando las máquinas pertenecen a distintos dominios administrativos.

Los mecanismos disponibles para realizar comunicaciones entre máquinas han ido aumentando con el paso de los años. En el nivel más bajo se tienen los *sockets*, herramienta flexible y eficiente pero que únicamente proporciona la conectividad 'cruda'. Un mayor nivel de abstracción suponen los mecanismos *PVM* (Parallel Virtual Machine) o *MPI* (Message Passing Interface). Estas bibliotecas siguen resultando difíciles de emplear en sistemas heterogéneos, por lo que se han propuesto sistemas de más alto nivel como *RPC*.

Más recientemente ha emergido la herramienta Globus como estándar de facto para la capa middleware de la rejilla (*grid*). Globus es capaz de realizar gestión de recursos de la rejilla, descubrimiento y monitorización, y gestión de datos en rejilla. Hay que comentar el importante papel jugado por el lenguaje XML en el desarrollo de la computación *grid*. El *OGSA* (Open Grid Services Architecture) define una arquitectura que presenta la computación *grid* como un servicio Web; el Globus Toolkit 3.0 será una implementación acorde con esta arquitectura.

3.3. Trabajo: Implementación de una Versión Paralela Multiobjetivo del Algoritmo PSO

En este trabajo se dispuso implementar y evaluar una versión multiobjetivo (MO) de un algoritmo basado en *Cúmulos de Partículas* Paralelizado.

La gran mayoría de problemas de naturaleza combinatoria presentados en la industria son de optimización. Estos problemas suelen tener una complejidad NP-duros, por lo que se vienen abordando mediante técnicas aproximadas. Un gran número de estos problemas requieren optimizar más de una función. Además, es habitual que dichas funciones a optimizar tengan direcciones contrapuestas, es decir, mientras una función se debe maximizar, la otra es de minimización. A estos problemas se les denomina como *Problemas de Optimización Multiobjetivo* (MOP). En estos, una solución viene dada por un conjunto de puntos *no dominados* situados en un frente (frente de *Pareto*). En este caso, se analizan propiedades como la diversidad y la convergencia del frente solución. La diversidad hace referencia a la distribución de las soluciones del frente de pareto. Una buena distribución será de mayor utilidad a un diseñador experto en el dominio del problema cuando tiene un mayor conjunto de soluciones distintas donde elegir, mientras que un frente con mala diversidad ofrecerá al diseñador un menor abanico de posibilidades donde elegir.

Las técnicas aproximadas entre las que se encuentran las Metaheurísticas, permiten obtener un conjunto de soluciones de calidad, en un tiempo y número de recursos razonable. Algunos ejemplos de este tipo de técnicas algorítmicas son: EAs (Evolutionary Algorithms), ACO (Ant Colony Optimization) y PSO (Particle Swarm Optimization). En este trabajo nos centramos en MO Particle Swarm Optimization (MOPSO).

El algoritmo *Particle Swarm Optimization* (PSO) es una metaheurística poblacional bioinspirada (*Kennedy & Eberhart, 1995*). Se inspira en el vuelo de las bandadas de pájaros. En la implementación del algoritmo existe un Swarm de partículas moviéndose hacia un óptimo en el espacio de búsqueda. Cada partícula tiene una posición y una velocidad y realiza un seguimiento del gradiente del espacio de búsqueda influenciada por su conocimiento anterior (influencia individual) y por su conocimiento sobre las partículas vecinas (influencia social).

Existen varias propuestas de MOPSO *(Coello et al, 2002) (Fieldsend, 2004)*. En estas se utiliza un archivo de soluciones no dominadas. Cada partícula se mueve influenciada por su conocimiento histórico y por soluciones del archivo. En la versión implementada en este trabajo, la elección de las partículas del archivo se basa en vecindarios. Además, acotamos la velocidad de las partículas y utilizamos un factor de *inercia adaptativa* para ajustar la exploración-explotación.

Por otra parte, la paralelización en sistemas Grid agiliza de manera considerable el cálculo y resolución de este tipo de problemas. Tareas como la evaluación de individuos y algunos operadores pueden ser realizadas en paralelo. Sin embargo, la necesidad de proporcionar frentes bien distribuidos dificulta la labor de paralelización. Existen distintos niveles de paralelización de metaheurísticas:

- nivel de metaheurística
- nivel de población/vecindario
- nivel de solución/función a optimizar

Para esto debemos aportar un mecanismo para la paralelización de metaheurísticas multiobjetivo eficiente y robusto. Sin embargo, se plantean serios inconvenientes. La idea es dividir el frente solución, es decir, dividir en varios subproblemas el problema inicial, aunque inicialmente no conocemos el frente óptimo y es difícil encontrar una correspondencia entre espacio de búsqueda y espacio objetivo. La estrategia tomada consistió en una división del frente basada en centroides (proporcionados por el *k-means*). Sin embargo, durante la evaluación del algoritmo muestra que esta técnica no funciona.

La razón se debe a que no produce un balanceo correcto del frente, es decir, se generaban muchas soluciones pertenecientes a un subproblema y pocas a otro. Otras estrategias como la partición respecto a una de las funciones objetivo o la partición geométrica del espacio de búsqueda reportan el mismo problema. Una posible solución consiste en generar nuevas soluciones a partir de las que ya disponemos subfrentes con pocas soluciones.

Se puede concluir diciendo que la paralelización de metaheurísticas multiobjetivo presentan la dificultad de encontrar un frente con buena diversidad. Las técnicas de subdivisión del frente hasta ahora no funcionan. Es un problema difícil, lo cual se ve refrendado debido al bajo número de publicaciones. Por tanto, es necesario estudiar la naturaleza del problema ya que únicamente es aplicable a problemas complejos que requieran una gran cantidad de recursos computacionales.

Capítulo 4

Fundamentos Teóricos de Inteligencia Artificial

Profesores:	José Luis Pérez de la Cruz y Rafael Morales		
Créditos:	5	**Cuatrimestre:**	1
Tipo:	Fundamental	**Carácter:**	Obligatoria

4.1. Objetivos y organización

En esta asignatura se da a conocer al alumno una serie de conceptos y técnicas fundamentales que sirven de base para la investigación en Inteligencia Artificial. En cada sesión, el profesor plantea una cuestión en relación al temario cuya respuesta debe meditar y responder el alumno. Cada reflexión o respuesta se redacta en un informe que será entregado en la sesión siguiente. De este modo, el profesor elige un informe al azar de todos los entregados y se debate en clase. La estructura de la asignatura se resume a continuación:

- Introducción Histórica y Conceptual
- Calculabilidad y Complejidad
- Agentes Reactivos y Agentes Deliberativos
- El Problema de la Comunicación
- El Problema de la Coordinación
- El Problema del Aprendizaje

4.2. Docencia

El campo de la Inteligencia Artificial (IA) comprende el conjunto de estudios intelectuales y tecnológicos para la creación de programas para máquinas que

imiten el comportamiento y la comprensión humana. La investigación en el campo de la IA se caracteriza por la producción de máquinas para la automatización de tareas que requieran un comportamiento inteligente. La IA es un concepto ampliamente estudiado en distintos ámbitos de la ciencia, la naturaleza y la sociedad y ha sido en últimas décadas, y sigue siendo, un campo importante de la investigación científica. Entre los objetivos principales de la IA se pueden extraer los siguientes:

- construcción de máquinas inteligentes, si bien, estas no tienen porqué operar de la misma forma a la que lo hacen los humanos,
- formalización del conocimiento y mecanización de los métodos de razonamiento,
- uso de modelos computacionales para el entendimiento de la psicología y el comportamiento de las personas, animales y agentes artificiales y
- facilitar el trabajo con las computadoras de manera que la interacción con ellas simule el trabajo cooperativo con gente habituada en el tema y posiblemente experta.

En la construcción de un sistema, se puede predecir la necesidad de cierta inteligencia cuando sea necesario procesos como: la toma de una decisión ante una situación imprevista, en presencia de un volumen de información enorme e indeterminado, que en gran parte es información imprecisa, insegura y conjeturada, tomando en consideración restricciones, metas y objetivos en gran medida fijados por el mismo agente que decide, resultando dicha decisión tomada satisfactoria.

De esta forma, si se consideran problemas bien estructurados, la IA obtendrá grandes logros, como los obtenidos hasta la actualidad en el campo de los juegos, la demostración automática y el *scheduling* (planificación). Así, previsiblemente, las máquinas irán superando a los humanos en más y más campos en los que la inteligencia se solía considerar imprescindible.

Dentro del campo de la IA, un *agente* se define como un ente que percibe ciertas características del medio que le rodea y lleva a cabos ciertas acciones en dicho medio. Los agentes más sencillos son los denominados *reactivos*, los cuales carecen de un modelo simbólico interno y actúan ligando ciertas combinaciones de estímulos a ciertas respuestas. Por otra parte, los agentes *deliberativos* contienen un modelo simbólico del mundo y toma sus decisiones utilizando mecanismos de razonamiento simbólico. En este caso, se introduce la *búsqueda* como paradigma general de representación y deliberación, ya que se relaciona la inteligencia como la capacidad de encontrar soluciones lo más rápidamente posible. *Newell* y Simon introdujeron la *hipótesis de búsqueda heurística* según la cual *las soluciones a los problemas se representan mediante estructuras simbólicas. Un sistema simbólico físico ejerce su inteligencia en la resolución de problemas mediante búsqueda, es decir, generando y modificando progresivamente estructuras simbólicas hasta que se produce una estructura solución.*

Por último, según el enfoque logicista de la IA, se utiliza la lógica como herramienta de análisis y control, herramienta de representación y herramienta de calculo. Puesto que la teoría es independiente a la implementación, la lógica puede utilizarse para orientar la implementación de un sistema inteligente. Para la representación, surge el concepto de *ontología*, mediante la que se define los términos y relaciones básicas que componen el vocabulario de un dominio, así como las reglas que permiten combinar dichos términos y relaciones para definir extensiones en el vocabulario (*Neches*).

4.3. Breves Resúmenes de los Trabajos

A lo largo del curso se formulan una serie de cuestiones por parte del profesor relacionadas con el contenido de cada sesión, debiendo ser meditadas y redactadas en trabajos cortos por el alumno. Las preguntas formuladas fueron las siguientes:

1. ¿Cómo se puede evaluar la calidad de una investigación científico-técnica en el ámbito de la Informática?
2. "El objetivo fundamental de la investigación en IA durante los próximos 15 años debe ser la construcción de un sistema capaz de superar el test de Turing"
3. "Modelos de cálculo aparecidos paralelamente y equivalentes a la Máquina de Turing"
4. "Situación actual (técnicas, aprendizaje, nivel alcanzado) del juego GO"
5. "Razones a favor y en contra del enfoque logicista de la IA"
6. "Obligations in Multiagent Systems", resumen del artículo
7. "Agentes, Servicios Web y Componentes: ¿En qué se parecen? ¿Cómo se diferencian?"
8. "Complejidad Espacial de una MT Determinista en la multiplicación y de una MT No Determinista en el cálculo de números primos"
9. "Ant Colony Optimization (ACO)"
10. "Un agente artificial debe diseñarse de forma que maximize su función de utilidad"

Capítulo 5

Algoritmos Evolutivos

Profesores:	Enrique Alba Torres y Carlos Cotta Porras		
Créditos:	5	**Cuatrimestre:**	1
Tipo:	Fundamental	**Carácter:**	Optativa

5.1. Objetivos y organización

En esta asignatura se presentan los fundamentos teóricos y prácticos de los Algoritmos Evolutivos como técnicas para la resolución de problemas de elevada complejidad. Se explica su funcionamiento, las clasificaciones en diferentes familias de algoritmos y la teoría subyacente. Todos estos conocimientos se tratan en las sesiones de docencia. El alumno realizó un trabajo relacionado con los algoritmos evolutivos para afianzar los conceptos y fundamentos de la asignatura. La docencia se estructura mediante el siguiente temario:

- Introducción a los Algoritmos Evolutivos
- Diseño de EAs y otras Metaheurísticas
- Algoritmos Evolutivos Descentralizados.
- Algoritmos Evolutivos Paralelos.
- Aplicación de los Algoritmos Evolutivos (I).
- Hibridación: Necesidad y Mecanismos.
- Hibridación Fuerte.
- Hibridación Débil.
- Aplicación de los Algoritmos Evolutivos (II).

Durante varias sesiones, el Doctor Pablo Moscato dio una charla sobre algoritmos evolutivos Meméticos. De forma adicional, el alumno asistió al curso de especialización titulado Optimización "Combinatoria y Paralela: del Diseño y la Implementación a las Aplicaciones", impartido por el Prof. Doctor El-Ghazali Talbi.

5.2. Docencia

Las técnicas Metaheurísticas son métodos heurísticos para la resolución de problemas que no tienen un algoritmo o heurística especifica que dé soluciones satisfactorias en tiempos razonables; o bien cuando sea imposible implementar ese método óptimo. La mayoría de las metaheurísticas tienen como objetivo la resolución de problemas de optimización combinatoria. Es decir, la optimización de un objeto matemático finito (por ejemplo, un vector de bits o permutación) que maximize (o minimize) una función objetivo, comúnmente llamada función de *fitness*. Si bien, las metaheurísticas no aseguran encontrar la solución óptima, son capaces de proporcionar soluciones satisfactorias.

Dentro de las metaheurísticas, se encuentran los *Algoritmos Evolutivos* (*Evolutionary Algorithm*, EA en adelante), un conjunto de técnicas inspiradas en el proceso de selección natural de las especies tomada de la teoría de evolución Darwiniana. El funcionamiento básico de los EAs consiste en la aplicación de una serie de operadores a los individuos de una población para la generación de individuos descendientes. Estos individuos representan las soluciones al problema. Los operadores más comunes son la selección, el cruce, la mutación y el reemplazo; así que tras su aplicación, se van creando nuevas soluciones y eliminando otras, de manera que la población en conjunto evoluciona. Esta evolución tiene por objeto que la población esté formada cada vez por soluciones mejores y en última instancia, contenga la solución óptima.

Dependiendo de la forma en la que se aplican los operadores y de la representación de los individuos, se pueden clasificar diferentes subclases de los EAs. Entre los más populares se encuentran los *Algorimos Genéticos*, que representa los individuos mediante una cadena binaria, las *Estrategias Evolutivas*, con representación real y la *Programación Genética*, en la que los individuos son árboles representando fragmentos de código programado.

Otra forma de clasificar los EAs depende de la estructura de la población. Es decir, mientras en el EA básico existe una única población en la que no existen restricciones a la hora de seleccionar individuos (población en panmixia), es posible estructurar la población de forma que a no todos los individuos les sean aplicados los operadores dentro de una generación. Esta subclase de los EAs, recibe el nombre de descentralizados, y son especialmente adecuados para ser paralelizados. Como máximos exponentes de esta subclase encontramos los EAs *distribuidos* (dEA) y los EAs *celulares* (cEA). En un dEA, un número pequeño de subalgoritmos ejecutan EAs independientes que, con una cierta frecuencia, intercambian individuos. De manera diferente, un cEA asigna un vecindario (topológico) a cada individuo, de manera que cada uno de estos sólo puede

interactuar con sus vecinos. Atendiendo a la "granularidad" en la división de la población: los dEAs tienen pocas subpoblaciones de gran tamaño mientras que los cEAs tienen muchas subpoblaciones de pequeño tamaño.

Todas estas modificaciones y versiones de los EAs traen en consecuencia la mejora de la búsqueda y un equilibrio entre la diversificación (explorar zonas distintas) e intensificación (explorar en profundidad zonas prometedoras). En este sentido, el teorema de "*No Free Lunch*" establece que no existe un algoritmo que sea mejor que otro en todos los problemas. No obstante, incorporando información del problema dentro del algoritmo es posible conseguir mejoras considerables. Esta estrategia de diseño se le conoce como *hibridación*. Podemos distinguir dos tipos de hibridación. El primer tipo es la *hibridación fuerte*, en la que la información sobre el problema se incorpora en la representación de las soluciones y en la adaptación de los operadores. El segundo tipo es la *hibridación devil*, que consiste en la incorporación de distintos algoritmos en el proceso evolutivo del EA.

Los EAs vienen siendo aplicados con éxito a numerosos problemas tanto del ámbito académico como real. Podemos encontrar algunos ejemplos en diferentes dominios de aplicación como las telecomunicaciones, bioinformática, logística y la ingeniería de los cuales destacamos: corte unidimensional, corte de patrones (2-D), asignación de frecuencias de radio, diseño molecular, clasificación en microarrays de ADN y enrutado de vehículos.

5.3. Trabajo: Aplicación de PSO y HNN al Problema de la Gestión de Localización de Terminales Móviles

El trabajo desarrollado en esta signatura trata de la aplicación del dos algoritmos al problema de la Localización de Terminales Móviles en redes GSM de Telecomunicaciones, el la literatura Location Area Management (LA). El primero de estos algoritmos consiste en una red neuronal de Hopfield hibridizado (fuerte) con una técnica de Ball Dropping (HNN+BD), mientras que el segundo consiste en una modificación del algoritmo Particle Swarm Optimization (PSO) basado en entornos geométricos [15] (hibridación devil).

En las redes de comunicaciones GSM, los ámbitos de las estaciones base se organizan en celdas. Estas a su vez se agrupan en regiones para establecer el espectro de frecuencias y para localizar estaciones de gestión de servicio (véase la Figura 5.1). Básicamente, el problema LA (de complejidad NP-Dificil) consiste en dos funciones principales: la gestión de movimientos (o actualizaciones de los terminales) y gestión de operaciones dentro de una celda. De esta forma, la función de *fitness* viene dada por la siguiente ecuación:

$$Coste = \beta \times \sum_{i \epsilon S} N_{LU}(i) + \sum_{i=0}^{N} N_P(i) \times V(i) \tag{5.1}$$

donde, $N_{LU}(i)$ es el número de actualizaciones en la celda i, $N_P(i)$ es el número de llamadas entrantes en la celda i, $V(i)$ es el factor de vecindad de la celda i, S es el número de celdas de gestión, y N es el número total de celdas de la red.

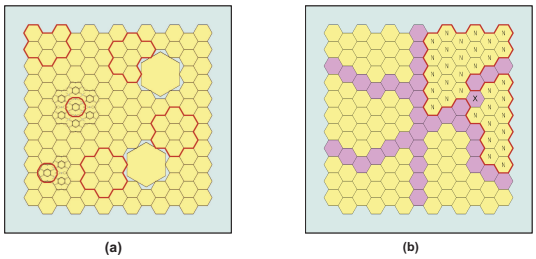

Figura 5.1: (a) Ejemplo de red de celdas GSM (b) agrupación de celdas de gestión

La forma de codificar una solución se realiza mediante un vector binario con las dimensiones de la red. Una celda i será codificada no lo es.

Para esto, se aplica la versión binaria del algoritmo PSO Geométrico el cual utiliza operadores originales de los EAs como pueden ser el cruce y la mutación. Mediante la red Hopfield, cada neurona codifica el estado de una celda. Para la experimentación, se utilizaron doce instancias de redes cuadradas GSM de diferentes dimensiones: 4×4, 6×6, 8×8 y 10×10. Estas instancias fueron generadas mediante simulador, aunque tomando escenarios realistas.

Para comprobar la confianza de los resultados realizamos 10 ejecuciones independientes de los dos algoritmos con cada instancia. Como conclusiones principales se observaron resultados similares en cuanto a la calidad del fitness obtenido y tiempos de computo. La diferencia radica en la naturaleza de los algoritmos, es decir, mientras que el algoritmo de HNN+BD es una modificación muy adaptada a las necesidades del problema, el algoritmo PSO es mucho más generalizado y sencillo en su diseño. La contribución principal fue que el algoritmo PSO con mínimas modificaciones, supone una técnica evolutiva capaz de igualar e incluso batir en algunos casos al algoritmo HNN+BD más específico al problema.

5.4. Charla: Doctor Pablo Moscato

El doctor Pablo Moscato de la universidad de Newcastle Callaghan (Australia) ofreció una charla sobre algoritmos Meméticos.

Los algoritmos Meméticos (MAs) son técnicas de optimización basados en la combinación sinérgica de ideas tomadas de otras metaheurísticas, como los algoritmos evolutivos y la búsqueda local (técnica de gradiente). Se basan en técnicas de hibridación fuerte para el aporte de información sobre el problema y la mejora del equilibrio entre exploración y explotación de la búsqueda.

Capítulo 6

Programación de Sistemas Multiagentes

Profesores:	Ricardo Conejo Muñoz y Mª Victoria Belmonte Martinez		
Créditos:	5	**Cuatrimestre:**	2
Tipo:	Fundamental	**Carácter:**	Optativa

6.1. Objetivos y Organización

El objetivo principal de este asignatura consiste en mostrar los fundamentos básicos de los *sistemas multiagente* (SMA). Desde el concepto de *agente* software como una abstracción para el diseño y la construcción de sistemas, hasta los modelos y arquitecturas de sistemas multiagente complejos. El contenido docente se ha estructurado en dos partes. En la primera parte, se han visto los conceptos teóricos sobre la tecnología SMA. En la segunda parte, de carácter práctico, se estudia y se trabaja con la plataforma de programación de agentes JADE. En programa de la asignatura consta de los siguientes temas:

- Fundamentos de los sistemas multiagente.
- Modelos y arquitecturas de agentes.
- Comunicación entre agentes.
- Programación de agentes con JADE.

De manera adicional, se ofreció la realización de forma voluntaria de dos trabajos por parte del alumno. Un trabajo relacionado con la parte teórica sobre "Agentes Económicos y Algoritmos Genéticos", además de otro trabajo práctico de programación de agentes con la plataforma JADE.

6.2. Docencia

Los Agentes software o sistemas Multi-Agente (SMA) han pasado de ser un sub-campo relativamente pequeño de la inteligencia artificial, a ser una tecnología computacional genérica, jugando hoy en día un papel muy importante en aplicaciones como la *computación grid*, la tecnología de *e-bussiness* e incluso la *web semántica*.

Básicamente, los agentes ofrecen una metáfora o una herramienta de abstracción, para el diseño y la construcción de sistemas complejos con múltiples componentes distintos e independientes. Estas abstracciones pueden ser usadas en el diseño y desarrollo de grandes sistemas, agentes individuales o distintas formas en las que agentes pueden interactuar para dar soporte a estos conceptos.

La investigación en los SMA está relacionada con el estudio, comportamiento y construcción de una colección de agentes autónomos, posiblemente preexistentes, que interactúan los unos con los otros y con sus entornos. El estudio de tales sistemas va más allá del estudio de la inteligencia individual, para considerar además los componentes sociales de la resolución de problemas. De este modo, un SMA se puede definir como "una red débilmente acoplada de resolvedores de problemas que interactúan para resolver problemas que están más allá de las capacidades individuales o de los conocimientos de cada uno de los resolvedores individuales de problemas".

Esta definición implica operaciones de *comunicación* entre agentes. La comunicación permite la interacción e interoperación entre agentes, esto es sincronizar acciones, enviar y recibir conocimiento, resolver conflictos en la resolución de tareas, etc. Al comunicarse, un agente influye en el conocimiento y acciones de otros agentes a través de un acto comunicativo. Por tanto son necesarios componentes como: un protocolo de transporte (TCP/IP, SMTP, HTTP, ...), un lenguaje común (pregunta, aceptación de petición, solicitud de ejecución,...) y un protocolo de interacción, es decir, patrones que suele seguir la secuencia de mensajes intercambiados entre agentes.

Los Lenguajes de Comunicación entre Agentes (ACL) permiten a los agentes interactuar, mientras ocultan los detalles internos. Ejemplos de estos lenguajes son: KQML (Knowledge Query and Manipulation Language) y FIPA ACL. Este último es el lenguaje de comunicación propuesto por FIPA (Foundation for Intelligent Physical Agents). El propósito de FIPA consiste en promover el desarrollo de especificaciones de tecnologías de agentes genéricos que maximicen la interoperabilidad dentro y entre diferentes sistemas multiagente. Algunos marcos de trabajo que permiten la implementación de SMA compatibles con el estándar FIPA son: FIPA-OS, ZEUS y JADE.

Como ejemplo de implementación del modelo FIPA se presentó el entorno de trabajo JADE. Implementado en Java, facilita la programación de agentes y su comunicación de forma transparente al usuario, e implementa todos los protocolos FIPA. JADE es concurrente, distribuido e implementa algunos agentes de servicio básico. Proporciona una interfaz gráfica para supervisar los eventos que ocurrente en el sistema multiagente y tiene herramientas auxiliares de diagnóstico.

6.3. Trabajo I: Agentes Económicos y Algoritmos Genéticos: Una Revisión del Estado del Arte

En este trabajo, se realizó una revisión del estado del arte sobre la aplicación de agentes, conjuntamente con Algoritmos Genéticos (y Algoritmos Evolutivos en general), bajo un punto de vista de económico-social.

Los algoritmos genéticos (AGs) describen el proceso de transformación de los individuos de una población, definidos mediante un vector de características, de acuerdo con fundamentos teóricos (Holland, 1975; Goldberg, 1989) que se basan en los procesos genéticos de formación, aprendizaje, adaptación y evolución de los organismos biológicos, y, especialmente, en el principio de selección natural o supervivencia de los mejores (término acuñado por Charles Darwin en The Origin of Species de 1859), y en los resultados de los intercambios genéticos. En la naturaleza, los procesos de selección natural y alteración genética ocurren en una generación, y luego en su descendencia, y a continuación en la descendencia de ésta, y así sucesivamente. Después de cada generación, o al menos así se desea, la población será mejor (o más adaptada a su entorno) que las anteriores, es decir, los individuos estarán más evolucionados (Moreno y Moreno, 1999).

Tradicionalmente, los algoritmos genéticos han sido considerados y aplicados como algoritmos de búsqueda de soluciones mejores (no necesariamente óptimas), especialmente útiles en problemas específicos de gran dimensión. Esta técnica difiere de los procedimientos tradicionales de optimización en varios aspectos que contribuyen a la robustez del algoritmo gen ético en cierto tipo de problemas y le proporcionan una ventaja sobre otras técnicas de optimización. Pero, además de su uso con fines de optimización, los algoritmos genéticos podrían considerarse como radiografías del proceso dinámico de transformación interna de la población bajo estudio. Es decir, más que determinar el punto final de un proceso iterativo hasta que se alcanza la convergencia, puede ser interesante observar la forma en que la población se adapta a su entorno en cada momento, cómo sus individuos se enfrentan mejor al contexto cambiante que les rodea, y cómo, en suma, la población evoluciona; lo que, en última instancia, sugiere un segundo uso alternativo de un algoritmo genético, especialmente útil en el contexto económico: reflejar el modo en que se transforma tal población, o lo que es lo mismo, servir como herramienta de predicción sobre la composición de la población.

En este sentido, la utilización de sistemas multiagentes para modelar los individuos de una población en evolución, viene siendo aplicado en diversos casos que encontramos en la literatura. La interacción entre agentes y la existencia de plataformas para el desarrollo de estos sistemas propicia la interrelación entre ambas estrategias. Actualmente existen protocolos de actuación y comunicación de agentes validados, además de un gran número de plataformas software desarrolladas y disponibles que facilitan la utilización y la construcción de sistemas inteligentes basados en algoritmos genéticos. Algunos ejemplos, vistos en este trabajo como casos de estudio son: aplicaciones para la configuración de AGs

mediante el uso de agentes (+CARPS), modelado de GAs paralelos basado en sistemas multiagentes, sistemas para el estudio y la evaluación del comportamiento de agentes económicos-Buying [14] (proporcionado por el profesor como base de este trabajo) y GAs multiobjetivo para la construcción de agentes de mercado.

6.4. Trabajo II: Práctica de Programación de Agentes

Esta práctica consistió en la implementación de un agente *jugador* para el conocido juego de las *Siete y Media*. Se usaron las clases del núcleo de la aplicación y la implementación del agente *crupier* que reparte las cartas y forma la partida. Además, se facilitaron las especificaciones de los mensajes que se transmiten los agentes. El agente jugador debe interactuar con el sistema multiagente y realizar una partida completa.

Capítulo 7

Sistemas Neuronales y Neurodifusos

Profesores:	José Muñoz Pérez, Francisco J. Vico Vela		
Créditos:	5	**Cuatrimestre:**	2
Tipo:	Fundamental	**Carácter:**	Optativa

7.1. Objetivos y organización

El objetivo de esta asignatura consiste en mostrar los conceptos básicos de las Redes Neuronales Artificial y su inspiración biológica. Se analizan los modelos y propiedades más relevantes y su aplicación en el procesamiento de la información. El programa docente resumido de esta asignatura consiste en los siguientes puntos:

- Introducción
- Red de Hopfield. Extensiones
- Redes Multivaluadas y Recurrentes
- Codificación y Evaluación de Redes Neuronales
- Aplicación a Problemas
- El Sistema Nervioso Animal

Al final de este bloque se pidió la elaboración de un pequeño trabajo de investigación con redes de este tipo. Como actividad adicional, el alumno asistió a la conferencia impartida por la Doctora Maria Victora Sánchez Vives del Instituto de Neurociencias de Alicante y por el propio profesor Fancisco Vico.

7.2. Docencia

Las Redes Neuronales Artificiales o *Artificial Neural Networks* (ANNs) son algoritmos basados en modelos matemáticos inspirados en el funcionamiento de las redes de neuronas biológicas. Las ANNs están formadas por unidades de memoria en analogía con las neuronas del cerebro conectadas unas con otras formando una red. Estas conexiones simulan las conexiones de las dendritas y los axones en los sistemas nerviosos biológicos.

Existen diferentes tipos de redes neuronales atendiendo a su arquitectura, funcionamiento y función de activación. En primer lugar, la red neuronal de Hopfiel (HNN) es uno de los tipos de redes básicas. Una HNN está formada está formada por un conjunto de neuronas N (unidades de procesamiento) con dos posibles estados, activada o desactivada. De este modo, sea una neurona i, la variable $S_i(t)$ nos da el estado de dicha neurona en el instante de tiempo t. La dinámica de computación viene dada por la siguiente ecuación:

$$S_i(t+1) = \begin{cases} +1 & si \ \sum_{j=1}^{N} w_{ij} S_j(t) > \theta_i \\ S_i(t) & si \ \sum_{j=1}^{N} w_{ij} S_j(t) = \theta_i \\ -1 & si \ \sum_{j=1}^{N} w_{ij} S_j(t) < \theta_i \end{cases} \tag{7.1}$$

Cada conexión entre dos neuronas (i, j) lleva asociada un peso sináptico w_{ij} representando la correlación entre ellas. EL valor umbral θ_i de cada neurona es una entrada externa que tiene asociada un peso -1 y que favorece la activación o desactivación de la neurona según su valor. Un tipo de las HNN son las *multi-valuadas* en las que el número de valores que puede tomar una neurona en la salida es mayor a la binaria (+1,-1 ó 0,1).

Las redes neuronales tienen la capacidad de aprender a lo largo de su funcionamiento, mediante un proceso adaptativo a través del cual se actualizan los pesos sinápticos entre neuronas con el objetivo de afinar el comportamiento de la red. La regla de actualización del peso sináptico correspondiente a la conexión entre el par de neuronas (i, j) se representa mediante la ecuación 7.2.

$$w_{ij}(k+1) = w_{ij}(k) + \Delta w_{ij}(k) \tag{7.2}$$

Por último, se pueden distinguir varios paradigmas de aprendizaje. Entre ellos tenemos el aprendizaje *supervidado*, en el que se dispone de un conjunto de patrones de entrenamiento para los que se conoce perfectamente la salida deseada de la red. Un objetivo para diseñar la regla de aprendizaje supervisada podrá ser minimizar el error cometido entre las salidas (respuestas) de la red y las salidas (respuestas) deseadas. Por otro lado, mediante el aprendizaje *no supervisado* (competitivo o autoorganizado), se dispone de un conjunto de patrones de entrenamiento pero no se conocen las salidas deseadas de la red. La red por sí misma buscará su comportamiento más adecuado atendiendo a cierto criterio y encontrará estructuras o prototipos en el conjunto de patrones de entrenamiento. Por último, el aprendizaje *por refuerzo* se basa en un proceso de prueba y error que busca maximizar el valor esperado de una función criterio

conocida como una *señal de refuerzo*. La idea de este paradigma surge en la psicología en relación con el estudio del aprendizaje en los animales. Si una acción supone una mejora en el comportamiento entonces la tendencia a producir esta acción se refuerza y en caso contrario de debilita.

7.3. Trabajo: Red Neuronal Recurrente Multivaluada Para el Problema de las N-Reinas

Como trabajo en esta signatura, se realizó una aplicación en MATLAB para resolver el problema de las n-reinas basándose de las redes neuronales recurrentes multivaluadas. Se realizaron una serie de pruebas para determinar los parámetros iniciales como el tiempo medio de ejecución y el número medio de iteraciones, de forma que puedan ser utilizados para compararse con otras implementaciones existentes. Además, se ha programó una forma de representar gráficamente las soluciones proporcionadas por la red neuronal, gracias a la cual ha sido posible la verificación visual de la validez de dichas soluciones. La red neuronal se ha diseñado utilizando una fórmula (ecuación 7.3) que indica cuando una neurona se encuentra en estado de jaque.

$$g(s_i, s_j) = \begin{cases} 1 & si\ s_i == s_j\ OR\ |s_i - s_j| == |i - j| \\ 0 & en\ otro\ caso \end{cases} \tag{7.3}$$

Este diseño es una generalización del modelo de Hopfield, y se ha podido observar en los resultados que el planteamiento realizado es correcto, ya que la red es capaz de encontrar la solución para un valor dado N de reinas. En la experimentación, se realizaron 20 ejecuciones independientes, inicializando el problema con diferentes dimensiones (20, 40, 80 y 160 reinas). Por tanto, se puede afirmar que los objetivos iniciales han sido cubiertos. Aunque el diseño soluciona el problema, podría ser una buena mejora al mismo reducir el número de iteraciones necesario para encontrar la solución.

Este trabajo fue realizado en grupo por Cristina Alcaraz Tello, Juanjo Durillo Barrionuevo, Jaime Gálvez Cordero y el autor de esta memoria.

7.4. Conferencia: Doctora Maria Victora Sánchez Vives

Esta conferencia se estructuró en dos sesiones, la primera de ellas titulada como *Actividad rítmica generada por la edad neuronal de la corteza cerebral. Observaciones experimentales y modelo computacional.* La segunda de sesión trató los *Mecanismos de control de la velocidad de propagación en la red cortical. Propiedades estadísticas estructurales de la red en distintas áreas de la corteza.*

Capítulo 8

Servicios Avanzados Multimedia Basados en Componentes

Profesores:	Antonio Vallecillo Moreno, Carlos Canal Velasco y Lidia Fuentes Fernández		
Créditos:	5	**Cuatrimestre:**	2
Tipo:	Fundamental	**Carácter:**	Optativa

8.1. Objetivos y organización

En esta asignatura se introducen los conceptos fundamentales en el uso de componentes para el diseño y construcción de aplicaciones distribuidas y abiertas. Se hizo una presentación del estado del arte, las principales líneas de investigación, paradigmas y herramientas de diseño. La metodología decente consiste en sesiones de docencia de carácter teórico impartidas por los profesores de la asignatura. Por parte del alumno, se realizó una presentación basada en un artículo asignado, sometiéndose a una rueda de preguntas y debate emulando una sesión de conferencia. Como suplemento al temario docente, el alumno asistió a una serie de charlas y conferencias resumidas al final de este capítulo. El programa de la asignatura se estructura en los siguientes temas:

1. Arquitectura, marcos de trabajo y patrones
2. Adaptación de componentes
3. Viewpoint Modelling
4. Modelado de aplicaciones software mediante MDA
5. Desarrollo basado en componentes y orientado a aspectos

8.2. Docencia

En la actualidad, en el desarrollo software es frecuente el empleo de técnicas que facilitan su construcción, además de el uso de nuevos puntos de vista que permiten abordarlo de manera completamente distinta. Una definición bastante aceptada del concepto de arquitectura software es la siguiente: *Estructura o estructuras de un sistema, lo que incluye sus componentes de software, las propiedades observables de dichos componentes y las relaciones entre ellos (Bass, Clements y Kazman, 1998)*. La arquitectura del software recoge las metodologías y estructuras empleadas en la producción de software. Una arquitectura software contiene por tanto un *conjunto de patrones y abstracciones que proporcionan un marco de referencia para guiar la producción de software*.

Los *lenguajes de descripción de arquitectura* (LDA) son herramientas diseñadas específicamente para describir la arquitectura de los sistemas de software. Sus elementos o primitivas básicas son los *componentes*, los *conectores* y las *configuraciones*. Los componentes son elementos computacionales y de datos, descritos mediante los papeles abstractos (*roles*) que desempeñan. Los conectores son mecanismos de interacción flexibles y variados. Por último, las configuraciones son combinaciones entre componentes y conectores interconectados. Existen algunos ejemplos de LDAs como Wright, Darwin, UniCon, Rapide, C2 y LEDA.

Utilizando el concepto de componente como unidad software reutilizable con funcionalidad específica y un interfaz de comunicación definido, se establece el *desarrollo basado en componentes*. Se basa en la composición y no en la implementación, es decir, pretende crear una serie de repositorios o mercado común de componentes reutilizables, hechos por terceros y accesibles, los cuales pueden ser adquiridos como piezas y utilizados para la construcción software (componentes COTS o *Commercial Off-The-Shelf*).

En el desarrollo de componentes entra en juego el concepto de *adaptación software*, que pretende solventar las posibles incompatibilidades entre los componentes y la adaptación de sus interfaces. La Adaptación es una nueva disciplina cuyo objetivos fundamentales son la construcción automática de adaptadores y la conexión de interfaces. Existen varios niveles y escenarios de adaptación dependiendo del nivel de la conexión: nivel de parámetros y nombres, nivel de comportamiento y nivel de servicio.

Por otra parte, cuando se enfoca el desarrollo software bajo un punto de vista modular se habla de desarrollo dirigido por modelos(*Model Driven Development*, MDD). Este tipo de desarrollo pretende alcanzar el producto final evolucionando de manera gradual desde un modelo inicial. Existen muchos tipos de modelo (metamodelos) como el archiconocido UML. Sin embargo, hay características propias del software que lo vuelven especialmente difícil de desarrollar y sobre todo de modificar. Una de las principales es la dificultad de modularización. En muchas situaciones hay funcionalidades (como el acceso a memoria o el login) que no se encuentran recogidas en un módulo, sino que se encuentran *enmarañadas* por todo el código; entonces un cambio en esta funcionalidad provocaría una modificación masiva del código.

La *programación orientada a aspectos* (POA) captura los diferentes conceptos que componen una aplicación en entidades bien definidas, eliminando las dependencias inherentes entre cada uno de los módulos.

8.3. Trabajo: Theme, An Approach for Aspect-Oriented Analysis and Design

Theme es una herramienta que permite la identificación de aspectos dentro del código y su modelado. Theme permite, mediante distintas vistas, representar un sistema desde sus requisitos hasta el modelado del mismo, identificando los aspectos y manteniendo la trazabilidad de los mismos. Esta herramienta está compuesta por diferentes *vistas de diseño*.

En primer lugar, en *action view* se representan los requisitos del sistema una vez identificadas las entidades claves y las acciones clave. En esta representación se establecen las dependencias entre acciones tal y como se definen en los requisitos Además, en esta vista se identifican qué acciones enmarañan el código y se muestran a continuación en la *clipped action view*. El la *theme view*, se realiza una representación de los requisitos del sistema identificando tanto entidades como acciones. Se realiza una representación por cada acción clave que incorpora todas las cláusulas de requisitos que le afectan. Esta representación permite el paso directo a *theme UML* que es ya un modelo del sistema.

Theme pretende ser una herramienta eficaz de apoyo a la identificación de aspectos en un sistema gracias a sus representaciones de los requisitos. De esta forma es también muy fácil conocer el cubrimiento de requisitos y realizar una traza de los aspectos desde los requisitos hasta el modelado del sistema. No obstante, se trata de un diseño preliminar y todavía no existe implementaciones.

8.4. Charla/Curso: Coral Calero

La Doctora Coral Calero, de la Universidad de Castilla-La Mancha, dio una charla titulada *Modelos de Calidad y Medición software*. Esta charla formó parte de un curso de especialización con el mismo título al cual el alumno asistió en su totalidad.

8.5. Charla: Pascal Poizat

El Doctor Pascal Poizat, del INRIA, dio una charla titulada *Adaptation of Open Component-Based Systems*.

8.6. Sesiones: Jornadas JICT 2007

Una de las sesiones previstas en la asignatura consistió en la asistencia a las *Jornadas Internacionales de Ciencia y Tecnologías* (JICT'07), de cuyas charlas se realizó resumes de contenidos. Los títulos de estas charlas se listan a continuación:

- Estado del Arte en Técnicas de Selección de Componentes, desde un punto de vista de Ingeniería de Requisitos.
- El procesador gráfico como acelerador de la transformada Hough.
- Desarrollo de Aplicaciones Científicas Basadas en Componentes Software, Esqueletos y Aspectos.
- ADAPTOR: Adaptación dinámica de componentes mediante Vectores de Sincronización y Expresiones Regulares.
- Another Co*cryption Method.
- Pour une composition paramétrable et automatique de services Web.

Parte II

Investigación

Capítulo 9

Trabajo de Investigación Tutelado

Tutor: Doctor Enrique Alba Torres

Algoritmos Basados en Inteligencia Colectiva para la Resolución de Problemas de Bioinformática y Telecomunicaciones

El trabajo de investigación tutelado se encuadra dentro de la principal línea de investigación del alumno, el empleo de técnicas metaheurísticas bioinspiradas para la resolución de problemas de optimización. Estos problemas consisten en la "selección y clasificación de genes en Microarrays de ADN', del campo de la bioinformática y la "gestión de la localización de terminales móviles de redes GSM", del campo de las telecomunicaciones. Este trabajo se presenta como un informe en un documento anexo a esta memoria.

Capítulo 10

Artículos Publicados

En este capítulo se presentan los trabajos publicados por el alumno hasta la fecha. Entre estos trabajos se encuentran artículos en revistas y congresos internacionales, artículos en congresos nacionales e informes técnicos.

10.1. MALLBA: A Software Library To Design Efficient Optimization Algorithms

Autores:	E. Alba, G. Luque, J. García-Nieto, G. Ordóñez y G. Leguizamón
Año:	2007
Publicación:	*International Journal of Innovative Computing and Applications (IJICA)*
Tipo:	Revista Internacional

En este articulo se hace una presentación de los nuevos algoritmos incorporados a la biblioteca MALLBA [1], una herramienta software para la resolución de problemas de optimización combinatoria mediante algoritmos genéricos implementados con un esquema de esqueletos (software skeletons) en C++. Todos los esqueletos en la biblioteca MALLBA implementan un método de optimización (exactos, metaheurísticas e híbridos) y ofrece tres implementaciones para cada uno: secuencial, paralela para redes de área local (LAN) y paralela para redes de área extensa. Este artículo introduce varios aspectos sobre el diseño software de la biblioteca, detalles sobre las implementaciones de esqueletos más recientes y los resultados computacionales obtenidos tras la resolución del problema de planificación mediante el que se ilustra la utilización de la biblioteca.

10.2. A Comparison of PSO and GA Approaches for Gene Selection and Classification of Microarray Data

Autores:	J. García-Nieto, L. Jourdan, E. Alba y E. G. Talbi
Año:	2007
Publicación:	*IEEE Congress on Evolutionary Computation, (IEEE CEC'2007), Singapore, Sep*
Tipo:	Congreso Internacional

En este trabajo (con referencia [9]) se estudia la utilización de una versión muy novedosa del algoritmo Particle Swarm Optimization a la selección y clasificación de genes en Microarrays de ADN de grandes dimensiones. Esta nueva versión, llamada Geometric PSO (GPSO) [15] se evalúa empíricamente pro primera vez en este trabajo utilizando una representación binaria de soluciones en espacios de Hamming. El algoritmo se utiliza para encontrar subconjuntos reducidos de genes informativos de bases de datos con decenas de millares de ellos. Para esto, se utiliza el algoritmo para clasificación Support Vector Machines (SVM) junto con 10-fold cross-validation para evaluar la adecuación de las soluciones encontradas. Además, se realiza una implementación, siguiendo la misma metodología, de un algoritmo genético y se realizan comparaciones. Los algoritmos desarrollados se evaluaron con seis bases de datos públicas relativas de microarrays de cánceres reales entre los que se encuentran: *AML ALL Leukemia*, *cancer de pecho, colon, ovarios, próstata y pulmón.*

10.3. Gene Selection in Cancer Classification using PSO/SVM and GA/SVM Hybrid Algorithms

Autores:	J. García-Nieto, L. Jourdan, E. Alba y E. G. Talbi
Año:	2007
Publicación:	*Genetic and Evolutionary Computation Conference (GECCO'07), London, UK*
Tipo:	Congreso Internacional

Este trabajo (con referencia [12]) constituye una aproximación *a priori* al trabajo anterior. Aquí se aborda el problema de selección y clasificación de genes en Microarrays de ADN mediante un algoritmo de Particle Swarm en una versión basada en umbrales. Del mismo modo, se realizan comparaciones con un algoritmo genético, además de un gran número de técnicas encontradas en el estado del arte. Las bases de datos evaluadas constituyen los mismos Microarrays que en el trabajo anterior.

10.4. Using Metaheuristics Algorithms Via ROS

Autores:	E. Alba, J. García-Nieto y F. Chicano
Año:	2007
Publicación:	*Genetic and Evolutionary Computation Conference, (GECCO'07), London, UK*
Tipo:	Congreso Internacional

En este trabajo (con referencia [11]) se presenta un servicio que ofrece a sus usuarios la posibilidad de utilización de una serie de algoritmos de optimización (posiblemente acoplados en bibliotecas) implementados en diferentes lenguajes de programación. Además, se provee de una plataforma de ejecución de los algoritmos sin la necesidad de grandes recursos computacionales. Este servicio, llamado ROS (Remote Optimization Service), es accesible remotamente a través de Internet, de manera que un potencial usuario no necesita implementar el algoritmo ni ejecutarlo en sus máquinas. En este artículo se describe la arquitectura de ROS y se detalla la forma en la que sus diferentes componentes interactúan entre ellos. Además, se describe la estructura de los datos intercambiados por los diferentes servidores: mediante ficheros XML dedicados a la especificación de algoritmos metaheurísticos. Finalmente, se evalúa numéricamente el funcionamiento de ROS bajo diferentes configuraciones y escenarios.

10.5. Algoritmos Basados en Cúmulos de Partículas para el Análisis de Microarrays de ADN

Autores:	E. Alba, J. Garcia-Nieto y G. Luque
Año:	2007
Publicación:	*Actas del V congreso sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB'07), Tenerife, España*
Tipo:	Congreso Nacional

Como primer acercamiento a la línea de investigación sobre bioinformática, el alumno realizó este trabajo (con referencia en [2]) en el que se estudia la aplicación de los Algoritmos Basados en Cúmulos de Partículas (o Particle Swarm Optimization - PSO) al problema de ordenación de genes en microarrays de ADN, un problema NP-duro con fuertes implicaciones en Biomedicina. Este problema consiste en la ordenación de un conjunto de genes, agrupando los que presenten comportamientos similares. El algoritmo PSO propuesto trabaja con representación de soluciones mediante permutaciones de enteros, y utiliza operadores adaptados a este tipo de representación. Además, se han desarrollado versiones secuenciales y paralelas del mismo integradas en la biblioteca MALLBA. La evaluación experimental sobre tres instancias reales demuestra la eficiencia y competitividad real de esta aproximación.

10.6. Sélection d'attributs de puce à ADN par essaim de particules

Autores:	El-Ghazali Talbi, Laeticia Jourdan, José García-Nieto y Enrique Alba
Año:	2007
Publicación:	*In Optimisation par Essaim Particulaire (OEP 2007) Paris, France*
Tipo:	Congreso Nacional

En este trabajo se describe la metodología utilizada en la aplicación de algoritmos de Particle Swarm al problema de selección de características y la clasificación en minería de datos. Está orientado a la selección en grandes bases de datos referentes a Microarrays de ADN. Se presenta el mecanismo para la hibridación de este tipo de metaheurísticas con algoritmos de clasificación como las *máquinas de vectores de soporte* (support vector machines) y los *k vecinos más cercanos*. Dichos algoritmos se evalúan mediante una serie de bases de datos de cánceres disponibles en la red. Se referencia como [8].

10.7. ROS: Servicio de Optimización Remota

Autores:	E. Alba, J.García-Nieto y F. Chicano
Año:	2006
Publicación:	Actas de las *Jornadas de Ingeniería del Software y Bases de Datos JISBD'06*
Tipo:	Congreso Nacional

En este artículo (con referencia [4]), se describe la arquitectura de ROS (*Remote Optimization Service*) desde la perspectiva de la *Ingeniería del Software*. Se presenta su arquitectura, componentes y las dos versiones de comunicación de servidores mediante SOAP y WebStream. Se detalla la estructura de clases del mecanismo de *Wrapper* para la anexión de algoritmos externos. Finalmente, se realiza una descripción del esquema XML diseñado para especificar todos las facetas y características relativas a las técnicas metaheurísticas.

10.8. On the Configuration of Optimization Algorithms by Using XML Files

Autores: E. Alba, J.García-Nieto y A. Nebro
Año: 2005
Publicación: *LCC ITI 2005-08*
Tipo: Informe Técnico

Existe diferentes formas para controlar y configurar un algoritmo. En primer lugar, la codificación directa de los valores de los parámetros implementados en el programa. Este método es bastante inflexible debido a la necesidad de recompilación cuando se realiza un mínimo cambio. Por lo tanto, la mayoría de los programadores prefieren utilizar otra solución consistente en diseñar ficheros de configuración externos que contengan los parámetros a introducir en un algoritmo. Sin embargo, la mayoría de los programadores utilizan ficheros ASCII de manera que los valores son escritos siguiendo reglas ad-hoc, sin ningún tipo de método estándar, difícil de entender para un usuario externo y con información propensa a errores. En este informe, presentamos diferentes enfoques sobre la configuración de algoritmos mediante ficheros de configuración. Esto es un factor no poco importante, ya que la configuración puede llevar a la estandarización, facilita la interacción entre algoritmos y minimiza los errores en la cooperación de estos. En este caso, nos situamos en el contexto de los algoritmos de optimización y en el diseño de servicios web para la optimización. Este propósito debe alojar y hacer interoperables algoritmos heterogéneos, los cuales se configuran siguiendo un esquema XML basado en un documento de validación común DTD.

Capítulo 11

Estancias en Otros Centros

En esta sección se presentan las estancias de investigación realizadas por el alumno en otros centros de investigación en el extranjero durante el periodo de Máster.

11.1. Universidad de Lille, INRIA Futurs I

El alumno realizó una estancia de un mes (desde el 10/10/2006 al 10/11/2006) en el *Laboratoire d'Informatique Fondamentale de Lille, Université des Sciences et Technologies de Lille*, Villeneuve d'Ascq Cédex, Francia.

Durante esta estancia se inició la línea de investigación sobre selección y clasificación de genes en Microarrays de ADN y bases de datos de cánceres. En este periodo se aplicaron diversos algoritmos de optimización (PSO y GA) al problema y se definieron los objetivos para el trabajo de colaboración que continua actualmente. En el desarrollo de los algoritmos se utilizó tanto la biblioteca MALLBA como el marco de trabajo Paradis-EO desarrollado por el equipo Dolphin del INRIA, equipo con el que se colaboró.

Esta estancia fue tutelada por el Doctor El-Ghazali Talbi y la Doctora Laetitia Jourdan de dicho centro. Los artículos resumidos en el capítulo anterior reflejan el alto grado de colaboración y la continuidad en la línea de investigación conseguidos en esta estancia.

11.2. Universidad de Lille, INRIA Futurs II

Como continuación de la primera estancia en Lille, el alumno realizó una segunda una estancia de dos meses (desde el 1/06/2007 al 1/08/2007) en el *Laboratoire d'Informatique Fondamentale de Lille, Université des Sciences et Technologies de Lille*, Villeneuve d'Ascq Cédex, Francia.

Durante esta estancia se abordó el problema de selección y clasificación de genes en Microarrays de ADN bajo un enfoque Multiobjetivo. Además, se realizó una implementación paralela del mismo para trabajar con grandes bases de datos sobre una plataforma *grid computing*. Esta implementación se evalúa en el *grid5000* [5] organizada por el INRIA (Francia).

Capítulo 12

Cursos y Seminarios

Este capítulo contiene los cursos y seminarios realizados por el alumno hasta la fecha. Se incluye un resumen de los mismos con el objetivo de mostrar que todos están relacionados con las líneas de investigación abiertas durante el programa de Máster.

12.1. Scatter Search y Path Relinking

Ponente. Dr. Manuel Laguna, Universidad de Colorado.

Resumen. La técnica de Scatter Search ó búsqueda dispersa es un tipo de técnica de optimización que hace uso de un conjunto de soluciones (población) cuyos individuos van mejorando durante la ejecución: el conjunto de referencia. La técnica proporciona una plantilla general en la que se implementan varios procedimientos dependientes del problema. Path Relinking es un algoritmo que se basa en generar por medio de algún sistema de interpolación soluciones intermedias en una trayectoria trazada entre dos soluciones.

12.2. Swarm Intelligence and Reconfigurable Computation

Ponente. Dr. Martin Middendorf, Universidad de Leipzig.

Resumen. La técnica de optimización de tipo colonia de hormigas se inspiran en el comportamiento de las hormigas reales en busca de alimento. Éstas dejan una traza de feromona que indica el camino que recorren. Las hormigas tienden a escoger los caminos con mayor rastro de feromona, de tal forma que de manera natural el mejor camino se verá reforzado y al final será el que las hormigas escojan. Esta optimización pertenece al tipo de enjambre de partículas.

12.3. Curso de Sistemas Complejos, Algoritmos Evolutivos y Bioinspirados

Resumen. En este curso se presentan las bases y características de los sistemas complejos, y de las técnicas de resolución de problemas tales como los algoritmos evolutivos y demás algoritmos bioinspirados (colonia de hormigas, enjambre de partículas etc). Este curso tubo lugar en Baeza, sede de la *Universidad Internacional de Andalucía*.

12.4. Curso Metaheurísticas: Introducción y Tendencias Recientes

Ponente. Dr. Chris Blume, Universidad Politécnica de Cataluña.

Resumen. Las metaheurísticas constituyen un tipo de técnica de optimización que se ha popularizado recientemente, dando lugar a numerosas tendencias. Inspiradas mayoritariamente en procesos observados en el mundo natural, algunas de las técnicas clásicas dentro de este dominio, como el recocido simulado (*Simulated Annealing*, SA) o el algoritmo genético (*Genetic Algorithm*, GA) imitan los procesos de templado de un metal y de la evolución natural de las especies respectivamente. Recientemente han aparecido nuevos tipos de algoritmos, como los de colonia de hormigas (*Ant Colony*, AC) y enjambre de partículas (*Particle Swarm*, PS) que se inspiran en el comportamiento de las hormigas en busca de alimento y de las bandadas de aves.

12.5. Descubrimiento de Conocimiento en Bases de Datos y Minería de Datos

Ponente. Dr. Dimitris A. Dervos, Technological Educational Institute of Thessaloniki (Grecia).

Resumen. Bajo el nombre de minería de datos se engloban un conjunto de técnicas encaminadas a la extracción de çonocimiento"procesable implícito en las bases de datos de las . Las bases de la minería de datos se encuentran en la inteligencia artificial y en el análisis estadístico. Mediante los modelos extraídos utilizando técnicas de minería de datos se aborda la solución a problemas de predicción, clasificación y segmentación. Un proceso típico de minería de datos parte de la selección del conjunto de datos, tanto en lo que se refiere a las variables dependientes, como a las variables objetivo, como posiblemente al muestreo de los registros disponibles. En este curso se estudiaron los conceptos básicos sobre la extracción del conocimiento el bases de datos y se hicieron prácticas de aplicación con la herramienta "DB2 Intelligent Miner".

12.6. Parallel Combinatorial Optimization: From Design and Implementation to Applications

Ponente. Dr. El-Ghazali Talbi, Universidad de Lille (Francia).
Resumen. Las técnicas de optimización combinatoria requieren generalmente un gran esfuerzo computacional para resolver problemas complejos. Se impone pues el uso de técnicas que permitan acelerar la realización de esa computación. El paralelismo permite la distribución de la carga computacional entre distintos procesadores, dividiendo de esta manera la duración temporal de cada una. Paradis-EO es una biblioteca de algoritmos de optimización que incorpora mecanismos para su ejecución en paralelo, permitiendo al usuario la aplicación de las principales técnicas metaheurísticas de optimización en paralelo sin tener que implementar la paralelización del código.

12.7. Modelos de Calidad y Medición software

Ponente. Dra. Coral Calero, Universidad de Castilla - La Mancha.
Resumen. Las métricas son un buen medio para entender, monitorizar, controlar, predecir y probar el desarrollo software y los proyectos de mantenimiento. Sin embargo, la medición software ha sido ignorada completamente dentro de la Ingeniería del Software. En este curso se estudian los modelos, métodos y métricas para la medición de la calidad de un producto software. Se realizaron ejercicios prácticos de validación de sistemas web actualmente accesibles en la web.

Bibliografía

[1] E. Alba, F. Almeida, M. Blesa, C. Cotta, M. Díaz, I. Dorta, J. Gabarró, C. León, G. Luque, J. Petit, C. Rodríguez, A. Rojas, and F. Xhafa. Efficient parallel LAN/WAN algorithms for optimization: the MALLBA project. *Parallel Comput.*, 32(5):415–440, 2006.

[2] E. Alba, J. Garcia-Nieto, and G. Luque. Algoritmos basados en cúmulos de partículas para el análisis de microarrays de ADN. In *Actas del V Congreso sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB'07)*, pages 419–426, Tenerife (Spain), Feb 2007.

[3] Enrique Alba and Francisco Chicano. Una versión de ACO para problemas con grafos de muy gran extensión. In *Actas del Quinto Congreso Español de Metaheurísticas, Algoritmos Evolutivos y Bioinspirados, MAEB 2007*, pages 741–748, Puerto de la Cruz, Tenerife, Spain, February 2007.

[4] Enrique Alba, Jose G. Nieto, and Francisco Chicano. ROS: Servicio de Optimización Remota. In *XI Jornadas de Ingeniería del Software y Bases de Datos (JISBD2006)*, pages 508–513, Sitges, Barcelona, Spain, October 2006.

[5] Raphaël Bolze, Franck Cappello, Eddy Caron, Michel Daydé, Frédéric Desprez, Emmanuel Jeannot, Yvon Jégou, Stephane Lanteri, Julien Leduc, Noredine Melab, Guillaume Mornet, Raymond Namyst, Pascale Primet, Benjamin Quetier, Olivier Richard, El-Ghazali Talbi, and Iréa Touche. Grid'5000: A large scale and highly reconfigurable experimental grid testbed. *Int. J. High Perform. Comput. Appl.*, 20(4):481–494, 2006.

[6] Advanced Network Topology Division. Wireless ad-hoc networks, itl(nist). http://www.antd.nist.gov/wahn_home.shtml.

[7] M. Dorigo and T. Stützle. Ant colony optimization. *The MIT Press*, 2004.

[8] Jose García-Nieto El-Ghazali Talbi, Laeticia Jourdan and Enrique Alba. Sélection d'attributs de puce à adn par essaim de particules. In *Optimisation par Essaim Particulaire (OEP 2007)*, Paris, France, April 2007.

[9] Laetitia Jourdan Enrique Alba, José Garcá-Nieto and El-Ghazali Talbi. Gene Selection in Cancer Classification using PSO/SVM and GA/SVM

Hybrid Algorithms. In *IEEE Congress on Evolutionary Computation CEC-07*, Singapore, Sep 2007.

[10] CARLINK: Wireless Traffic Service Platform for Linking. Celtic: Eureka. http://www.celtic-initiative.org/Projects/CARLINK/default.asp.

[11] José García-Nieto, Enrique Alba, and Francisco Chicano. Using metaheuristic algorithms remotely via ROS. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 1510–1510, New York, NY, USA, 2007. ACM Press.

[12] José García-Nieto, Enrique Alba, Laetitia Jourdan, and El-Ghazali Talbi. A comparison of PSO and GA approaches for gene selection and classification of microarray data. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 427–427, New York, NY, USA, 2007. ACM Press.

[13] G. J. Holzmann. The model checker SPIN. *IEEE Trans. on Soft. Eng.*, 23(5):1–17, 1997.

[14] Tokuro Matsuo Masaki Hyodo and Takayuki Ito. An optimal coalition formation among buyer agents based on a genetic algorithm. In *Developments in Applied Artifcial Intelligence*, volume 2718 of *Lecture Notes in Computer Science*, pages 151–157, 2003.

[15] A. Moraglio, C. Di Chio, and R. Poli. Geometric Particle Swarm Optimization. In *10th European conference on Genetic Programming (EuroGP 2007)*, volume 4445 of *Lecture Notes in Computer Science*. Springer, Abril 2007.

[16] OPLINK: Net Centric Optimization. Tin2005-08818-c04-01 - ministerio de educación y ciencia (spain). http://oplink.lcc.uma.es/.