UNIVERSIDADE DA CORUÑA

FACULTADE DE INFORMÁTICA

Departamento de Tecnologías de la Información y de las Comunicaciones

PROYECTO FIN DE CARRERA DE INGENIERÍA INFORMÁTICA

Optimización y Análisis del Rendimiento de Sistemas Conexionistas NeuroGliales

Autor: **Pablo Mesejo Santiago**

Directores: **Ana Belén Porto Pazos**

Óscar Ibáñez Panizo

A Coruña, septiembre de 2009

Título del Proyecto:

OPTIMIZACIÓN Y ANÁLISIS DEL RENDIMIENTO DE SISTEMAS CONEXIONISTAS NEUROGLIALES

Clase: Proyecto de investigación y desarrollo

Nombre del Alumno: Pablo Mesejo Santiago

Nombre de los directores: Ana Belén Porto Pazos

Óscar Ibáñez Panizo

Miembros del Tribunal
Fecha de Lectura y Defensa
Calificación

A mi madre, por todo

AGRADECIMIENTOS

Este proyecto fin de carrera bajo ningún concepto se hubiese podido realizar sin la ayuda, dirección y apoyo de Ana Belén Porto y Óscar Ibáñez y, por supuesto, sin la colaboración y el trabajo de Noha Veiguela. Me siento orgulloso de poder afirmar que mis directores y colaboradores son, además, buenos amigos.

Agradecer al Centro de Supercomputación de Galicia (CESGA) tanto la posibilidad de emplear sus recursos como la excelente disposición a la hora de ofrecer soporte y ayuda. También agradecer a Ricardo Cao sus generosos y útiles consejos en lo relativo a la parte estadística de este trabajo.

Es justo mostrar agradecimiento no sólo ante aquellas personas que han colaborado directamente en la realización de este proyecto, sino también ante aquellas otras que, de una u otra forma, lo han hecho posible o le han dado sentido a hacerlo posible. Me refiero a mi familia: mi abuelo, mi abuela, Rosita, Tono, mis tíos y primos; y mis amigos, especialmente Juan, Dini y Miguel. Finalmente, quería hacer mención a dos personas con las que he contraído una deuda especial: mi madre y Sofía. Quería agradecerles su paciencia y apoyo en todo momento, su dedicación y consejo siempre que lo he necesitado. Estoy en deuda con vosotras y lo estaré siempre.

Resumen

Este proyecto consta de dos fases diferenciadas, pero íntimamente relacionadas, en las que se han realizado labores de investigación y desarrollo en Inteligencia Artificial y en Neurociencia.

En una primera fase, se ha llevado a cabo un estudio comparativo entre Redes NeuroGliales Artificiales y Redes de Neuronas Artificiales a la hora de resolver cuatro problemas de diferente complejidad: clasificación de flores de Iris y de señales de la Ionosfera, y predicción de cáncer de mama y de enfermedades coronarias. Las Redes NeuroGliales Artificiales son Sistemas Conexionistas que incorporan, además de neuronas artificiales, elementos de control que simulan el comportamiento de células del Sistema Glial denominadas astrocitos. Se ha descubierto recientemente que los astrocitos participan de modo activo en el procesamiento de la información en el cerebro. El estudio aquí realizado ha permitido demostrar la influencia de los astrocitos artificiales en el procesado de la información, así como encontrar un mejor método de resolución de problemas de clasificación y predicción emulando el funcionamiento del Sistema Nervioso. Se ha desarrollado una aplicación que permite el diseño y construcción de los sistemas conexionistas a comparar, así como la ejecución masiva de pruebas.

Además, en una segunda fase del proyecto y como parte del paradigma de aprendizaje de este nuevo modelo de sistema conexionista, se ha diseñado e implementado un método de optimización empleando Algoritmos Genéticos para obtener la mejor configuración de parámetros inherentes a la acción de los astrocitos artificiales en las Redes Neurogliales Artificiales. Finalmente, se ha probado la capacidad de aprendizaje y generalización de estas redes optimizadas mediante los Algoritmos Genéticos, aplicándolas a problemas sencillos de clasificación y predicción.

Palabras clave

Redes NeuroGliales Artificiales, Redes de Neuronas Artificiales, Sistema Glial, Algoritmos Genéticos.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. INTRODUCCIÓN

1.1. Motivación

El análisis de los modelos de Redes de Neuronas Artificiales (en adelante, RNA) desarrollados hasta ahora, ha permitido observar que presentan ciertas limitaciones como paradigmas del procesado de información [FREE 93; LAPE 88]. Estas limitaciones podrían ser debidas a que dichas RNA no reflejan ciertos comportamientos de las neuronas y no consideran la participación de elementos que no sean neuronas artificiales. Dado que las RNA pretenden emular el Sistema Nervioso (en adelante, SN), los investigadores han tratado de reflejar en ellas la importancia que desde siempre se les ha otorgado a las neuronas. Sin embargo, durante las últimas décadas el avance en la investigación en Neurociencia ha revelado que el papel del Sistema Glial (en adelante, SG) en el procesado de la información en el SN es mucho más relevante de lo que se creía hasta ahora. Esto ha llevado a pensar en la utilidad de incorporar a los Sistemas Conexionistas (en adelante, SC) nuevos elementos que emulen las células del SG y más concretamente, un tipo particular de estas células conocidas como *astrocitos*. Estos elementos, que hasta ahora no habían sido considerados en los modelos artificiales y que recientemente han sido identificados como células fundamentales en las sinapsis cerebrales [PERE 07], podrían facilitar un funcionamiento más óptimo de los SC. Considerando lo anterior, han sido creados hace pocos años SC que integran RNA multicapa con astrocitos artificiales que se han denominado *Redes NeuroGliales Artificiales* (en adelante, RNGA) [PORT 04].

Las RNGA aúnan la utilización de neuronas artificiales, la implementación de astrocitos artificiales y el uso de Algoritmos Genéticos (en adelante, AG), de tal forma que se combina la utilización de SC con métodos de Computación Evolutiva (en adelante, CE). Este proyecto se enmarca por tanto en el campo de la Inteligencia Artificial (en adelante, IA) y en el de la Neurociencia. En general, la unión de estas dos áreas científicas pretende alcanzar unos objetivos que beneficien el avance en ambas, objetivos que son los siguientes:

1. Comprender cómo funciona el SN, intentando reproducir en el laboratorio diferentes fenómenos y comportamientos de los circuitos cerebrales de los seres vivos.
2. Mejorar los modelos conexionistas actuales.

Alcanzar el primer objetivo, "Comprender cómo funciona el SN", beneficia directamente a la Neurociencia porque permite profundizar en el funcionamiento del SN, pero también beneficia a la IA, pues cuantos más conocimientos se posean sobre el SN más fielmente se podrán plasmar estos en la construcción de sistemas y máquinas con comportamientos más inteligentes.

Conseguir el segundo objetivo, "Mejorar los modelos conexionistas actuales", beneficia directamente a la IA, ya que se conseguirían sistemas con mayores capacidades de procesamiento. Pero además, beneficia indirectamente a la Neurociencia, ya que cuanto más evolucionen estos modelos se contará con mayores facilidades para estudiar el SN: por un lado, utilizando estos modelos, se podrán probar más fielmente las hipótesis que se planteen en los laboratorios de Neurociencia, y por otro, puede que computacionalmente se expliquen fenómenos biológicos no entendidos todavía.

En este proyecto se ha llevado a cabo un análisis de la eficiencia de estas recientemente creadas RNGA con respecto a RNA multicapa entrenadas mediante AG, para lo cual se ha realizado un estudio comparativo entre los resultados obtenidos por ambos tipos de redes con distintas arquitecturas en problemas de diferente complejidad. Se trata de los problemas de clasificación de flores de Iris, clasificación de señales de la Ionosfera, y los problemas de predicción de cáncer de mama y de enfermedades de corazón (enfermedades coronarias). El objetivo del análisis es doble: por un lado, a nivel biológico, se estudia la influencia de los astrocitos artificiales en el procesado de la información mediante el modelo computacional de las RNGA, y por otro, se intenta encontrar un mejor método de resolución de problemas de clasificación y predicción emulando para ello el funcionamiento del SN, teniendo en cuenta el papel fundamental de la glía según los últimos descubrimientos realizados en este campo [PERE 04; PERE 07].

Para realizar este trabajo de investigación se ha partido de una herramienta de simulación previamente desarrollada [PORT 04; ALVA 07], que ha sido adaptada para

poder realizar pruebas masivas en el Centro de Supercomputación de Galicia (CESGA), todo ello con el propósito de realizar el mayor número de pruebas posibles combinando distintos parámetros para los problemas aquí tratados. Así, se han realizado simulaciones con distintas arquitecturas de red, diferentes algoritmos gliales y diversas combinaciones de parámetros, empleando la técnica de validación *cross-validation* [STON 78; CANTU 05], uno de los métodos más utilizados para la valoración de la precisión de algoritmos de clasificación y predicción.

Esta memoria se articula en seis capítulos que se estructuran de la forma siguiente:

- El primer capítulo, *Introducción*, consta de la motivación del trabajo, la exposición de los objetivos a alcanzar y una presentación de los antecedentes.
- El segundo capítulo, *Fundamentos Teóricos*, trata toda la fundamentación teórica y conceptual que subyace en este estudio.
- El tercer capítulo, *Desarrollo del Entorno de Simulación*, describe las herramientas desarrolladas para la realización de las simulaciones y el análisis de los datos, así como las plataformas utilizadas, junto con la estructura ideada para la realización masiva de pruebas.
- En el cuarto capítulo, *Experimentación*, se comentará pormenorizadamente la parte experimental junto con los resultados obtenidos y la discusión de los mismos.
- En el quinto capítulo se presentan las *Conclusiones* del proyecto realizado.
- El sexto contiene los *Trabajos Futuros* a desarrollar en la línea de investigación abordada.
- Al término de estos capítulos se encuentra la *Bibliografía* referenciada.

1.2. Objetivos

El objetivo general de este trabajo de investigación es por tanto, la validación y análisis de la eficiencia de las RNGA con respecto a RNA multicapa entrenadas mediante AG. Dicho análisis persigue la obtención de un doble beneficio: contribuir a la investigación de los SC, tratando de mejorar los modelos de procesado de información

actuales; y beneficiar a la Neurociencia, ya que comprobar la utilidad de los astrocitos artificiales permitiría validar las distintas hipótesis que se han formulado acerca del papel de estas células en el SN. Además, al contribuir a la investigación del SN, debido a que el uso de modelos conexionistas facilita cuantificar el funcionamiento de los astrocitos naturales, beneficiaría a su vez y de nuevo a la IA, pues un mayor conocimiento de los astrocitos permitiría una mejor emulación de estas células en los SC.

Para alcanzar este objetivo general se incorpora el comportamiento de astrocitos a redes multicapa orientadas hacia delante y sin conexiones laterales. Dicho objetivo general se descompone en los siguientes objetivos específicos:

- Estudiar la influencia de los astrocitos artificiales en el procesamiento de la información realizado mediante Sistemas Conexionistas multicapa.
- Analizar dicha influencia cuando aumenta la complejidad de los problemas abordados y de la arquitectura de las redes analizadas.
- Realizar una selección automática, mediante métodos de CE, de la mejor configuración de los parámetros característicos de la Glía artificial en estos Sistemas Conexionistas Neurogliales, y analizar el rendimiento de las RNGA optimizadas mediante CE, en la resolución de distintos tipos de problemas.

1.3. Antecedentes

Desde los inicios de la rama conexionista de la IA, hace más de 65 años [MCCU 43], se han desarrollado diferentes tipos e implementaciones de RNA, las cuales varían significativamente en topología, dinámica, alimentación y funciones internas de sus elementos de procesado (en adelante, EP). Las RNA emulan las redes de neuronas biológicas y permiten resolver problemas del mundo real no abordables mediante sistemas de informática convencional, o de la rama simbólica de la IA, como son los problemas de clasificación, *clustering* o predicción [PAZO 96]. Aunque se ha avanzado en el diseño de estos sistemas, así como en sus algoritmos de aprendizaje, desde las primeras aportaciones que hiciera Hebb [HEBB 49], todavía existen muchas limitaciones en cuanto a su velocidad y complejidad computacionales.

Las mejoras que están siendo logradas en los modelos conexionistas se obtienen siguiendo dos vías diferentes de investigación: por un lado, algunos investigadores

construyen SC basados en modelos matemáticos con diversas ecuaciones que controlan su funcionamiento [CORT 95; HAYK 99]; por otro, están los investigadores que creen que las limitaciones en el procesado de la información de los modelos de SC actuales pueden deberse a que estos no plasman fielmente el funcionamiento del sistema nervioso [DORA 99; PORT 07].

El trabajo de investigación que aquí se presenta está enmarcado en esta segunda vía de investigación, es decir, se parte de la idea de que las limitaciones existentes en las RNA actuales pueden ser debidas, bien a que no implementan ciertas capacidades de las neuronas biológicas, o bien a que no se han incluido en las redes artificiales elementos que sí participan en el procesamiento de la información en los sistemas biológicos. Este último supuesto se ha visto reforzado por los descubrimientos de las últimas décadas en el campo de la Neurociencia [ARAQ 99; ARAQ 01; HAYD 01; HAYD 02; PERE 02; PERE 04], descubrimientos que han puesto de manifiesto la importancia de las funciones del SG y que llevan a pensar que la participación de los astrocitos en el procesamiento de la información en los sistemas biológicos tiene un papel más relevante del que se le había adjudicado. De ser así, sería interesante integrar elementos en las RNA que simulen el funcionamiento de estos astrocitos del SG, que es lo que se pretende con las RNGA. Por lo que se sabe, los astrocitos podrían ser los encargados de realizar tareas como: escoger los mejores caminos para la transmisión, determinar qué elementos van a constituir un determinado circuito cerebral, acelerar la transmisión de los impulsos, realizar un procesamiento "heurístico" de la información, comunicar los resultados del procesamiento a otros circuitos que no intervengan directamente en el procesamiento de determinada información (cuya salida constituirá la entrada de este circuito para el procesamiento de información con el fin de realizar otra tarea distinta), etc.

Los trabajos previos existentes en esta área de estudio son los realizados por la directora de este trabajo, A. Porto [PORT 04; PORT 05a; PORT 05b; PORT 07] y han sido desarrollados con la colaboración del laboratorio de "Fisiología celular de astrocitos y neuronas" del Departamento de Plasticidad Neural, situado en el Instituto de Neurobiología Ramón y Cajal del Consejo Superior de Investigaciones Científicas (CSIC) en Madrid.

No se han encontrado publicaciones de otros grupos de investigación que estén trabajando en esta línea (implementación de astrocitos artificiales), pues aunque surgen

los primeros intentos de incorporar la función de los astrocitos, estos son, de momento, modelizaciones matemáticas, como por ejemplo el trabajo de Xi Shen y Philippe De Wilde [SHEN 06], que modeliza el aumento del flujo sanguíneo de los capilares cerebrales en función de la actividad neuronal y el emparejamiento entre neuronas y astrocitos. También, fuera de los modelos conexionistas, en [NADK 07] se elabora un modelo matemático de las interacciones sinápticas que tienen lugar entre las neuronas y los astrocitos, teniendo en cuenta el concepto de sinapsis tripartita [ARAQ 99]. Este modelo teórico intenta proporcionar una base conceptual para protocolos experimentales en Neurociencia dependiendo de la actuación de los astrocitos. En ambos casos, se trata sólo de una modelización matemática, y no de la integración directa, como en el caso del presente estudio, del comportamiento de los astrocitos en las RNA.

El presente estudio analiza la eficiencia de las RNGA resultantes de la integración de nuevos elementos, los astrocitos, en las RNA clásicas. Tal como se ha implementado esta aproximación, no complica de forma excesiva el proceso de entrenamiento. Se aplica un método híbrido de entrenamiento que combina aprendizaje supervisado y no supervisado. Este método híbrido [PORT 04; PORT 07], que se describirá en profundidad en el capítulo de *Fundamentos Teóricos*, se diferencia de otros métodos híbridos en que, aunque utiliza técnicas de CE para el entrenamiento, la técnica de búsqueda local empleada se fundamenta en el SG.

Para la realización de las simulaciones que se han llevado a cabo en este trabajo, se contaba inicialmente con una aplicación compuesta por una interfaz gráfica en Delphi, donde el código de los algoritmos de entrenamiento, test, etc., se había desarrollado en C. Con el fin de realizar el mayor número de pruebas posibles en paralelo, se decidió adaptar esta aplicación a los requisitos de ejecución del Centro de Supercomputación de Galicia (CESGA). Para ello fue necesario migrar toda la aplicación al lenguaje de programación C e idear una estructura de directorios, archivos de parámetros y *scripts* de ejecución, con el fin de organizar y automatizar lo máximo posible las simulaciones llevadas a cabo, tal y como se explicará de forma detallada en el capítulo 3 de esta memoria. Estos scripts y este código en C fueron optimizados al máximo con vistas a minimizar los tiempos de ejecución en el CESGA y facilitar toda la parte experimental.

Por último, destacar que la innovación de los modelos existentes de RNA hacia el desarrollo de nuevas arquitecturas está condicionada por la necesidad de integrar nuevos parámetros en los algoritmos de aprendizaje, de forma que ellos puedan ajustar sus valores. La mayor parte de los avances publicados en investigación en RNA se refieren a optimizaciones de los algoritmos más frecuentemente empleados, optimizaciones que incrementan la potencia de los cálculos y que trabajan básicamente en el lado computacional del algoritmo. Es mucho más infrecuente que se consigan nuevos parámetros que provean de nuevas funcionalidades a los modelos de RNA actuales, que es lo que se estaría logrando con la incorporación de los astrocitos en las RNGA resultantes.

2. FUNDAMENTOS TEÓRICOS

2.1. Redes de Neuronas Artificiales

La Inteligencia Artificial es una ciencia que busca solucionar problemas imitando los mecanismos que los seres humanos (o cualquier ser inteligente) emplean para resolverlos (estas soluciones pueden ser tanto software como hardware). La IA se divide en dos escuelas de pensamiento, la IA convencional y la IA computacional. La IA convencional (o simbólica) intenta modelar matemáticamente los procesos de razonamiento; dentro de esta rama se encuentran, por ejemplo, los Sistemas Expertos y las Redes Bayesianas. La IA computacional implica un aprendizaje iterativo; partiendo de la representación del problema mediante datos empíricos, la estructura del sistema o algún parámetro de éste varía, buscando con cada iteración una mejora del sistema en la resolución del problema, esto es, un menor error. En esta rama se engloban, entre otras disciplinas, las RNA, los Sistemas Difusos y la CE.

Las RNA son un paradigma de aprendizaje y procesamiento automático de información que forman parte de la rama computacional de la IA. Se trata de un sistema de interconexión de elementos de procesado o neuronas artificiales en una red, los cuales colaboran para producir un estímulo de salida.

El primer modelo de neurona artificial fue propuesto por McCulloch y Pitts [MCCU 43], quienes modelizaron una estructura y un funcionamiento simplificado de las neuronas del cerebro, considerándolas como dispositivos con *m* entradas, una única salida y sólo dos estados posibles: activa o inactiva. Una RNA era, en ese planteamiento inicial, una colección de neuronas de McCulloch y Pitts, todas sincronizadas, donde las salidas de unas neuronas estaban conectadas a las entradas de otras. Algunos de los planteamientos de McCulloch y Pitts se han mantenido desde 1943 sin modificaciones, otros por el contrario han ido evolucionando, pero todas las formalizaciones matemáticas sobre RNA que se han realizado desde entonces, aún sin pretender ser una modelización exacta de las redes de neuronas biológicas, sí han resultado un punto de partida útil para el estudio de las mismas. El reciente resurgir de las RNA es, en gran parte, debido a la presentación de ciertos modelos fuertemente inspirados por los biólogos [HOPF 82; HOPF 89].

Según la definición propuesta por Kohonen [KOHO 88] las RNA son "*conjuntos de elementos de cálculo simples, usualmente adaptativos, interconectados masivamente*

en paralelo y con una organización jerárquica que les permite interactuar con algún sistema del mismo modo que lo hace el sistema nervioso biológico". Su aprendizaje adaptativo, carácter autoorganizativo, capacidad de generalización y tolerancia a fallos, operación en tiempo real y fácil inserción dentro de la tecnología existente, han hecho que su utilización se haya extendido en áreas como la biológica, financiera, industrial, medio ambiental, militar, salud, etc. [HILE 95]. Están funcionando en aplicaciones que incluyen identificación de procesos [GONZ 98], detección de fallos en sistemas de control [ALDR 95], modelación de dinámicas no lineales [MEER 98; WANG 98], control de sistemas no lineales [BLOC 97; LEVI 93], recuperación de la información [MOYA 98] y optimización de procesos [OLLE 98; ALTI 98; AGUI 98]. Entre los modelos de RNA que han sido propuestos hasta ahora podemos citar: Perceptrón (1957), Adaline y Madaline (1960), Avalancha (1967), Retropropagación (1974), Neocognitrón (1980), SOM (1980), Hopfield (1982), ART (1986), etc.

Una RNA se compone por tanto de unidades artificiales llamadas neuronas (ver figuras 1 y 2). Se considera una neurona como un elemento formal o módulo o unidad básica de la red que recibe información de otros módulos o del entorno, la integra, la computa y emite una única salida que se va a transmitir a múltiples neuronas posteriores [WASSE 89]. En las RNA cada conexión tiene asignado un peso, que determina la importancia de dicha conexión dentro del grupo de entradas de una neurona.

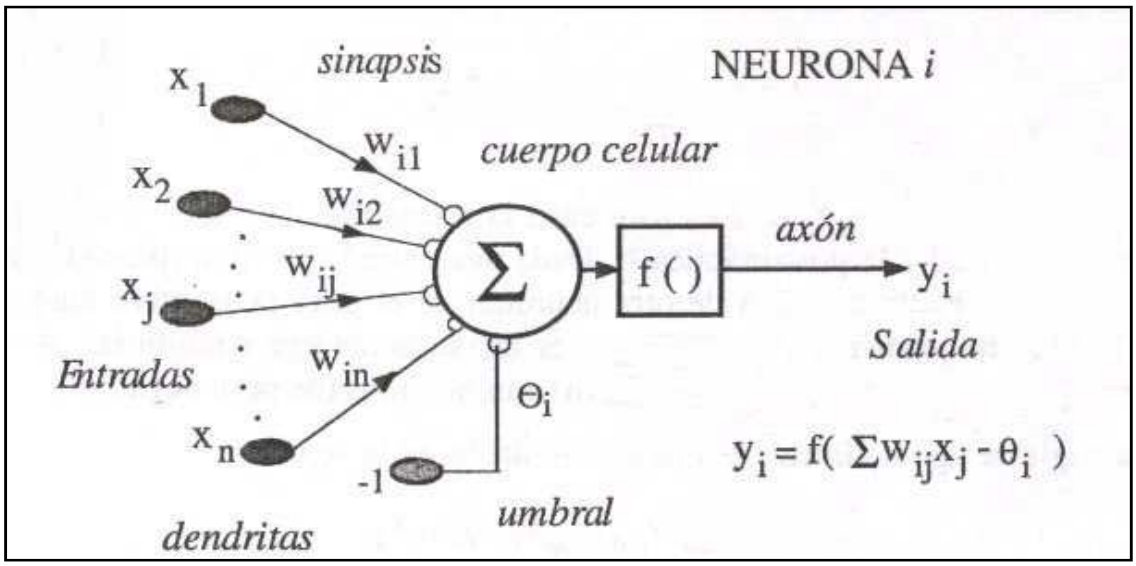


Figura 1: Modelo de Neurona Estándar [MART 06]

El peso de las conexiones se denota de la siguiente manera:

Wij = *Peso de la conexión entre la neurona j (que emite) y la neurona i (que recibe).*

A través de estas conexiones, cada una con un valor de peso asociado, llegan las entradas a la neurona. A partir de dichas entradas la neurona emite una salida, que viene dada por 3 funciones, que se aplican de forma consecutiva: propagación (o valor neto), activación y transferencia [ISAS 04]. Cada Neurona Artificial o Elemento de Procesado (en adelante, EP) calcula su valor neto en base a sus entradas y la fuerza de las conexiones asociadas a las mismas. A partir del valor neto se calcula el valor de activación, y a partir de este último se calcula el valor de salida aplicando una función de transferencia, limitador de rango dinámico, sobre la activación de la neurona.

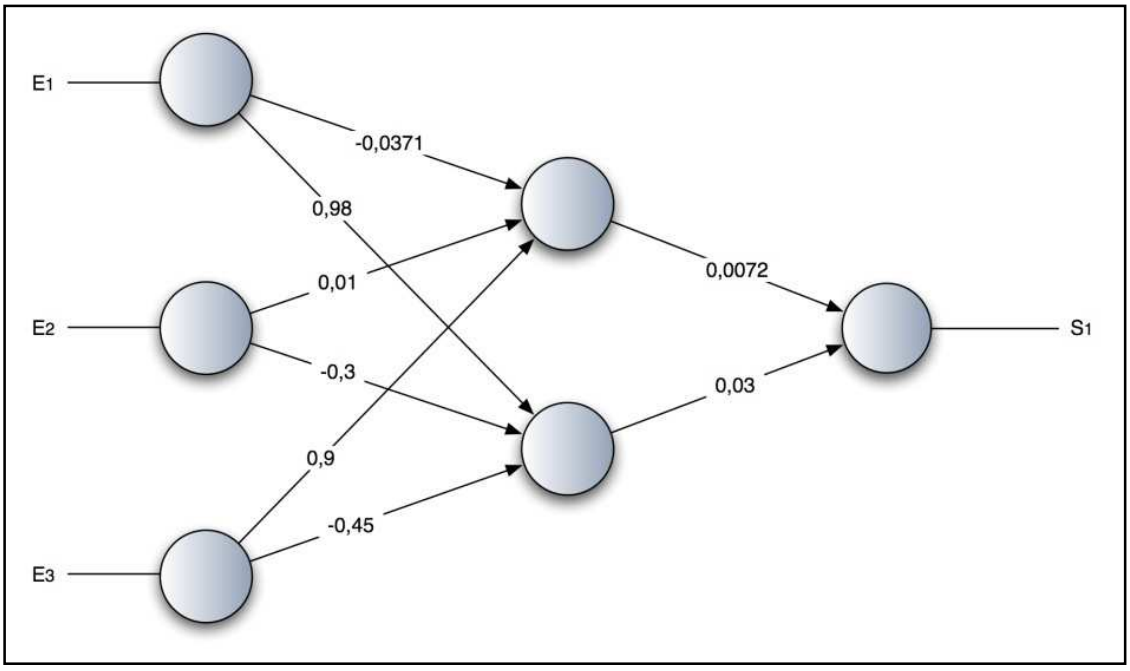


Figura 2: Ejemplo de RNA: elementos de procesado interconectados mediante pesos sinápticos, que reciben unas entradas y procesan una salida.

El funcionamiento básico de las RNA abarca dos grandes etapas. La primera comprende las fases de creación y desarrollo de la red, la segunda correspondería a la fase de funcionamiento real o fase de ejecución.

1. **Fase de Creación.** En ella se llevarán a cabo las siguientes tareas:

- **Diseño de la arquitectura**: consistirá en determinar el número de capas de la red, el número de neuronas dentro de cada capa, y la conectividad entre ellas, además de las funciones de activación y transferencia que se utilizarán. Esta etapa es crítica, ya que la topología de la red determina la capacidad de representatividad de la misma, y por lo tanto la capacidad de conocimiento que puede albergar. No existen técnicas que determinen cual es la mejor topología para un determinado problema, por lo que habrá que recurrir al método de *ensayo-error*.
- **Entrenamiento de la red**: una vez diseñada la arquitectura de la red, habrá que entrenar la red para que "aprenda" el comportamiento que debe tener; es decir, para que aprenda a dar la respuesta adecuada a la configuración de estímulos o patrones de entrada que se le presente [PAZO 96]. Esto se realiza presentando a la red un subconjunto de los posibles valores de entrada y/o salida del problema que se está tratando de resolver, subconjunto que debe ser suficientemente representativo de la casuística del problema, ya que su elección condicionará el desarrollo del entrenamiento, el cual consistirá en determinar los valores de los pesos de las conexiones de la red. Para la realización de dicha tarea existen diferentes algoritmos, el más común de ellos el de *backpropagation* (BP) o retropropagación del error [RUME 86], que se aplica a redes alimentadas hacia adelante, dispuestas en varias capas, y con una conectividad total entre las neuronas de una capa y las de la capa siguiente.

  Para conseguir que la red generalice bien, los datos usados para el entrenamiento deben cubrir un rango de hechos suficientemente amplio. En general, cuando aumenta el tamaño y variedad del conjunto de entrenamiento disminuye la necesidad de que los datos de entrada durante la fase de trabajo normal se parezcan mucho a los patrones de este conjunto, es decir, la red generalizará mejor. Pero si los datos de un problema se diferencian demasiado de todos los patrones con los que se ha entrenado la red, ésta tendrá dificultades para encontrar la respuesta correcta.

  A la hora de entrenar una red, existen dos grandes tipos de aprendizaje: supervisado y no supervisado.

El aprendizaje supervisado se caracteriza por la existencia de un "supervisor" que le indica a la red, durante el proceso de entrenamiento, cuál es la salida que debe ofrecer, y, por lo tanto, también le indica qué error está cometiendo. En este tipo de aprendizaje, el entrenamiento consiste en presentarle a la red reiteradamente patrones de estímulos de entrada pertenecientes al conjunto de entrenamiento, y a partir de las salidas que ofrece la red, calcular un valor de error con respecto a las respuestas que se desean. Es decir, se comparan las respuestas de la red con las salidas deseadas del conjunto de entrenamiento. De acuerdo al resultado de esta comparación se reajustan los pesos de las conexiones, para lo cual existen varios algoritmos, siendo el más común el mencionado anteriormente, BP. El reajuste de los pesos de las conexiones está orientado a que, ante el patrón de entrada, la red se acerque cada vez más a la respuesta correcta. Después de sucesivas presentaciones de todos los patrones del juego de ensayo, los pesos de las conexiones de todas las neuronas se habrán estabilizado en torno a unos valores óptimos, es decir, el algoritmo de aprendizaje converge, y se puede considerar entonces a la red entrenada y dar por terminada la fase de aprendizaje.

El aprendizaje no supervisado se caracteriza por la no existencia de ese elemento "supervisor", es decir, para un conjunto de entradas no existen unas salidas deseadas. En estos casos es cuando se desea que la red encuentre relaciones y dependencias entre los datos de entrada. Ejemplos de estas redes son los mapas autoorganizativos (*selforganizing maps*, SOM) [KOHO 88], cuyo objetivo es, a partir de un conjunto de datos, realizar una tarea de clasificación o *clusterización* de los mismos.

Estas dos categorías de aprendizaje definen los modelos básicos de entrenamiento; mezclando las características correspondientes a cada modelo o aplicando las dos categorías de entrenamiento en distintas fases, se pueden obtener otros tipos de entrenamiento:

Entrenamiento híbrido: Se trata de una combinación del aprendizaje supervisado y del no supervisado. Parte de los pesos se ajustan por medio de un esquema de aprendizaje supervisado, y el resto se obtienen por medio de un aprendizaje no supervisado. Dentro de este grupo se pueden incluir aquellas redes formadas por capas distintas, aunque podrían ser incluidas en aprendizaje

supervisado ya que en su conjunto funcionan como tal. En este tipo de entrenamiento se incluyen métodos como el *Learning Vector Quantizer* (LVQ) o la Red de Contrapropagación (CPN) [OHM. 06].

Entrenamiento evolutivo: Consiste en utilizar métodos de CE para optimizar el proceso de aprendizaje de la RNA. Permiten también la búsqueda en paralelo de las mejores opciones en cada uno de los aspectos del diseño de la red. Se intenta de este modo emular el comportamiento de los sistemas naturales acercando más los sistemas artificiales a los naturales, a la vez que sirve para demostrar la potencialidad del entrenamiento evolutivo a la hora de evaluar, de una forma fácil, nuevos desarrollos en el campo de las RNA.

- **Validación de la red**: el proceso de validación de una red suele realizarse de forma conjunta al entrenamiento, con un conjunto de patrones distinto al de entrenamiento y test. El objetivo es impedir que la red caiga en alguno de los problemas habituales del entrenamiento, como son el de que la red no se entrene con precisión suficiente (es decir, que tenga un alto porcentaje de "fallos" que no se reduce por más veces que se le pase el juego de ensayo, o que la red tarde mucho tiempo en entrenarse) o el de sobreentrenamiento. La validación, por tanto, asegura que la red ha sido entrenada de forma satisfactoria.
- **Test:** finalmente, cuando se tiene una red entrenada y validada, se realiza un test de la misma en unas condiciones que no han sido presentadas durante el entrenamiento, para evaluar su capacidad de generalización y su comportamiento en casos que la red no ha "visto", con lo que se tiene una medida real de cómo se va a comportar la red con el problema para el cual ha sido creada.

2. Fase de Ejecución

Tras la fase de entrenamiento viene la fase de ejecución, durante la cual se le pedirá a la RNA que responda a estímulos diferentes a los presentados durante la fase de entrenamiento. Gracias a los ejemplos aprendidos del juego de ensayo, la RNA deberá ser capaz de generalizar, es decir, ante entradas similares a las de su juego de ensayo, producirá salidas correctas.

Para operar con una RNA entrenada, el proceso es el mismo que cuando se realizaba el entrenamiento: se le sigue suministrando información de entrada a la red,

sólo que ahora no se realizará ningún ajuste en los pesos de las conexiones. La red reconocerá, evaluará y dará una respuesta.

2.2. Algoritmos Genéticos

Como ya se ha dicho, las RNA intentan crear sistemas con capacidad de aprendizaje imitando al cerebro (mecanismo natural a nivel de individuo); otra rama de la IA computacional, la Computación Evolutiva, intenta crear sistemas con capacidad de aprendizaje mediante la simulación de la evolución (mecanismo natural a nivel de especie). La CE se basa en realizar modelos de ciertas características de la naturaleza, fundamentalmente de la capacidad que tienen los seres vivos para adaptarse a su ambiente, lo que ya había sido tomado como base por Darwin para desarrollar su teoría de la evolución según el principio de selección natural en 1859 [DARW 59; WALL 58; WALL 55]. Basándose en este principio, las técnicas de CE desarrollan algoritmos que son capaces de adaptarse al problema que se pretende solucionar. La CE engloba varios paradigmas muy relacionados entre sí [YAO 99], siendo uno de ellos los AG.

2.2.1. Definición de AG

Un AG es un algoritmo de búsqueda basado en la siguiente premisa: la reproducción y el principio de supervivencia del más apto permiten a las especies biológicas adaptarse a su ambiente y competir por los recursos.

Según la definición propuesta por Koza [KOZA 92]: "*Es un algoritmo matemático altamente paralelo que transforma un conjunto de objetos matemáticos individuales con respecto al tiempo usando operaciones modeladas de acuerdo al principio Darwiniano de reproducción y supervivencia del más apto, y tras haberse presentado de forma natural una serie de operaciones genéticas de entre las que destaca la recombinación sexual. Cada uno de estos objetos matemáticos suele ser una cadena de caracteres (letras o números) de longitud fija que se ajusta al modelo de las cadenas de cromosomas, y se les asocia con una cierta función matemática que refleja su aptitud*".

Más formalmente, y siguiendo la definición dada por Goldberg [GOLD 1989], *"Los Algoritmos Genéticos son algoritmos de búsqueda basados en la mecánica de*

selección natural y de la genética natural. Combinan la supervivencia del más apto entre estructuras de secuencias con un intercambio de información estructurado, aunque aleatorizado, para constituir así un algoritmo de búsqueda que tenga algo de las genialidades de las búsquedas humanas".

En un AG el conjunto de todos los parámetros cuyos valores constituyen la solución al problema (*genes* en la terminología de AG) se codifican en una cadena de valores denominada *cromosoma*. El conjunto de los parámetros representado por un cromosoma particular recibe el nombre de *genotipo*. El genotipo contiene la información necesaria para la construcción del organismo, es decir, la solución real al problema, denominada *fenotipo*. Desde los primeros trabajos de John Holland [HOLL 75] la codificación suele hacerse mediante valores binarios, aunque durante la última década son igualmente comunes las representaciones que codifican directamente cada parámetro con un valor entero, real o en punto flotante. Si se hace mediante valores binarios, a cada parámetro se le asigna un determinado número de bits, número que puede ser distinto para cada parámetro. Cada uno de los bits pertenecientes a un gen suele recibir el nombre de *alelo.*

Para alcanzar la solución a un problema, los AG parten de un conjunto inicial de individuos, llamado población, generado de manera aleatoria. Cada uno de los individuos de la población tendrá asociado un valor de ajuste o *fitness* acerca de su adecuación para resolver el problema planteado. Este valor, que se obtendrá de una manera específica para cada uno de los problemas, es la información cuantitativa que empleará el algoritmo evolutivo para guiar el proceso de búsqueda, y representa lo bien que el fenotipo del individuo soluciona el problema actual. Goldberg describió esta función de ajuste como *"una medida de beneficio, utilidad o bondad que queremos maximizar"* [GOLD 89].

El funcionamiento genérico de un Algoritmo Genético puede apreciarse en el siguiente pseudocódigo:

```
Inicializar población actual aleatoriamente
MIENTRAS  no se cumpla el criterio de terminación
        crear población temporal vacía
        MIENTRAS  población temporal no llena
                seleccionar padres
```

cruzar padres con probabilidad Pc
SI se ha producido el cruce
mutar uno de los descendientes con probabilidad Pm
evaluar descendientes
añadir descendientes a la población temporal
SINO
añadir padres a la población temporal
FIN SI
FIN MIENTRAS
aumentar contador generaciones
establecer como nueva población actual la población temporal
FIN MIENTRAS

Una generación se obtiene a partir de la anterior por medio de los operadores de reproducción. Existen 2 tipos:

- Cruce. Se trata de una reproducción de tipo sexual. Se genera una descendencia a partir del mismo número de individuos (usualmente 2) de la generación anterior. Las tasas de cruce con las que se suele trabajar rondan el 90%.
- Copia. Se trata de una reproducción de tipo asexual. Un determinado número de individuos pasa sin sufrir ninguna variación directamente a la siguiente generación.

Una vez generados los nuevos individuos se realiza la mutación con una probabilidad *Pm* (por lo general entre 0.5% y el 2%). Se sale de este proceso cuando se alcanza alguno de los criterios de parada fijados. Los más usuales suelen ser:

- Los mejores individuos de la población representan soluciones suficientemente buenas para el problema que se desea resolver.
- La población ha convergido. Un gen ha convergido cuando el 95% de la población tiene el mismo valor para él, en el caso de trabajar con codificaciones binarias, o valores dentro de un rango especificado, en el caso de trabajar con otro tipo de codificaciones. Una vez que todos los genes alcanzan la convergencia se dice que la población ha convergido. Cuando

esto ocurre la media de bondad de la población se aproxima a la bondad del mejor individuo.

- Se ha alcanzado el número de generaciones máximo especificado.

Los individuos de la población evolucionarán, por lo tanto, tomando como base los esquemas propuestos por Darwin y Wallace sobre la selección natural, y tras el paso de cada generación se irán adaptando cada vez más a la solución requerida. La evolución de estos individuos se realizará de forma análoga a como se realiza en el mundo natural: se seleccionará un conjunto de individuos que se combinarán entre sí (cruce, ver figura 3), y su descendencia se insertará en la población. Con una probabilidad muy baja, cuando se inserte un individuo nuevo en la población, éste sufrirá una mutación (figura 4). Con la mutación lo que se pretende es conseguir individuos a los que no se podría llegar usando simplemente la combinación de otros, ya que ésta genera nuevos valores para los genes mutados. El proceso continuará mientras no se alcance un criterio de parada predeterminado, como pueda ser que la solución alcance un determinado error umbral o bien que se llegue a un número fijado de generaciones.

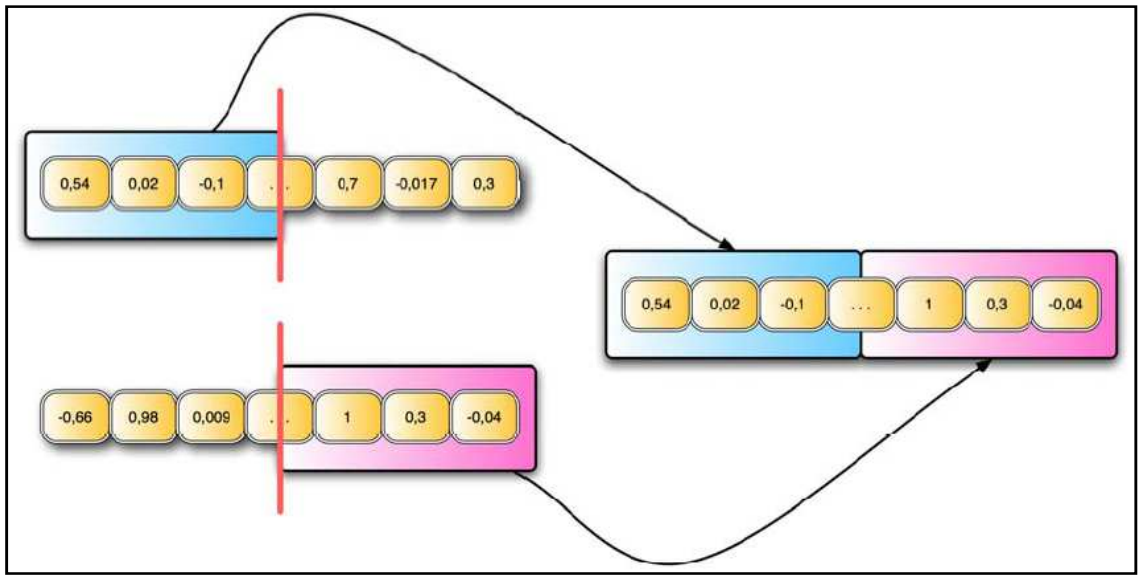


Figura 3: Cruce de Individuos

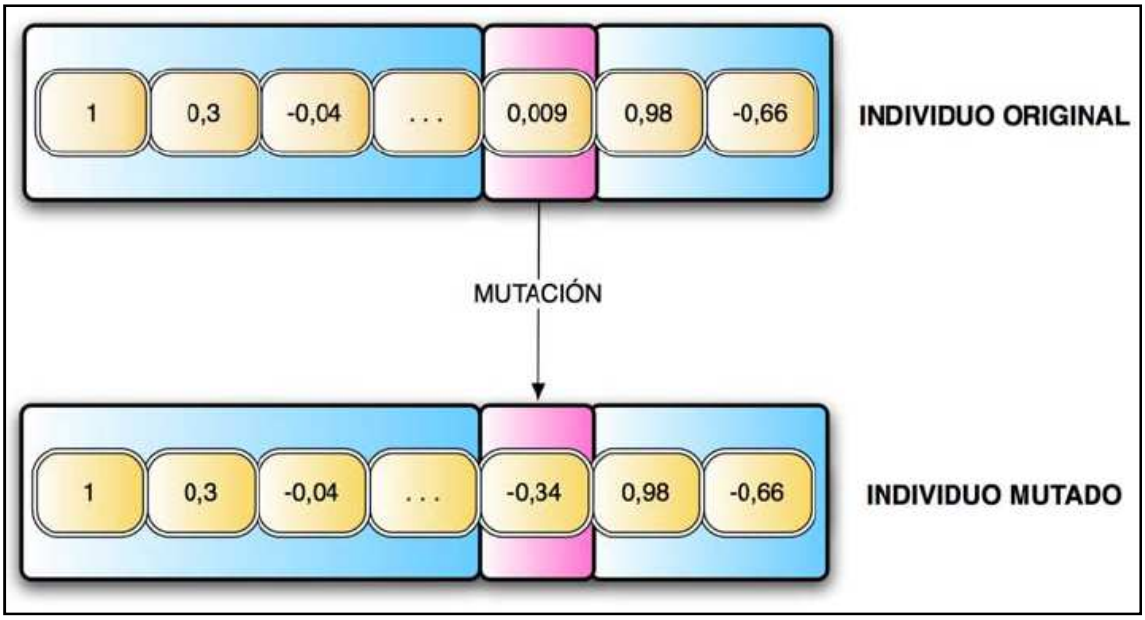


Figura 4: Mutación de Individuos

2.2.2. Algoritmos Genéticos para el entrenamiento de Redes de Neuronas Artificiales

En el caso de las RNA, los AG se han utilizado a lo largo de los años para optimizar diferentes aspectos: arquitectura de la red, funciones, pesos, etc. [DORA 99]. Dado que en este trabajo se emplean para colaborar en la optimización de los pesos de las conexiones y en la selección de los valores de los parámetros inherentes a la glía artificial, se va a describir el modo en que se produce el entrenamiento de una RNA mediante AG.

Para comenzar han de elegirse los elementos que van a constituir los "genes" de los individuos que conformarán la población. Generalmente estos serán los pesos de las conexiones de la red (si todas las redes de la población comparten la misma estructura y parámetros), o los pesos junto con otros parámetros, en caso de tener una población con individuos (redes) heterogéneos en cuanto a sus parámetros; se genera entonces la población inicial de individuos, normalmente mediante la generación aleatoria de los valores de los genes (dentro de los márgenes que establece la estructura elegida para las redes). A continuación se evalúa la red, y se ordena la población según el error de los individuos, tras lo cual se procede a la selección de los individuos de acuerdo al algoritmo de selección que se desee aplicar (torneo [WETZ 83], ruleta [DEJO 75], etc.). Tiene lugar entonces el cruce, para lo cual se dividen los individuos seleccionados en un número de cortes determinado por el algoritmo de cruce elegido (punto, multipunto o

uniforme) y se procede a mezclar los genes de los progenitores para obtener la descendencia, la cual se evalúa para obtener el error de los nuevos individuos generados. Algunos de estos individuos sufrirán mutaciones, generalmente con una probabilidad muy baja. Los nuevos individuos así generados se incluirán en la población mediante el algoritmo de sustitución que se haya elegido (sustitución del peor individuo, de los individuos con ajuste parecido, los hijos sustituyen a los padres, etc.). Se siguen estos pasos de mezcla de genes (cruce), mutación y evaluación hasta que el error obtenido sea el esperado o se haya alcanzado el número de generaciones determinado. En la figura 5 se puede observar el procedimiento explicado.

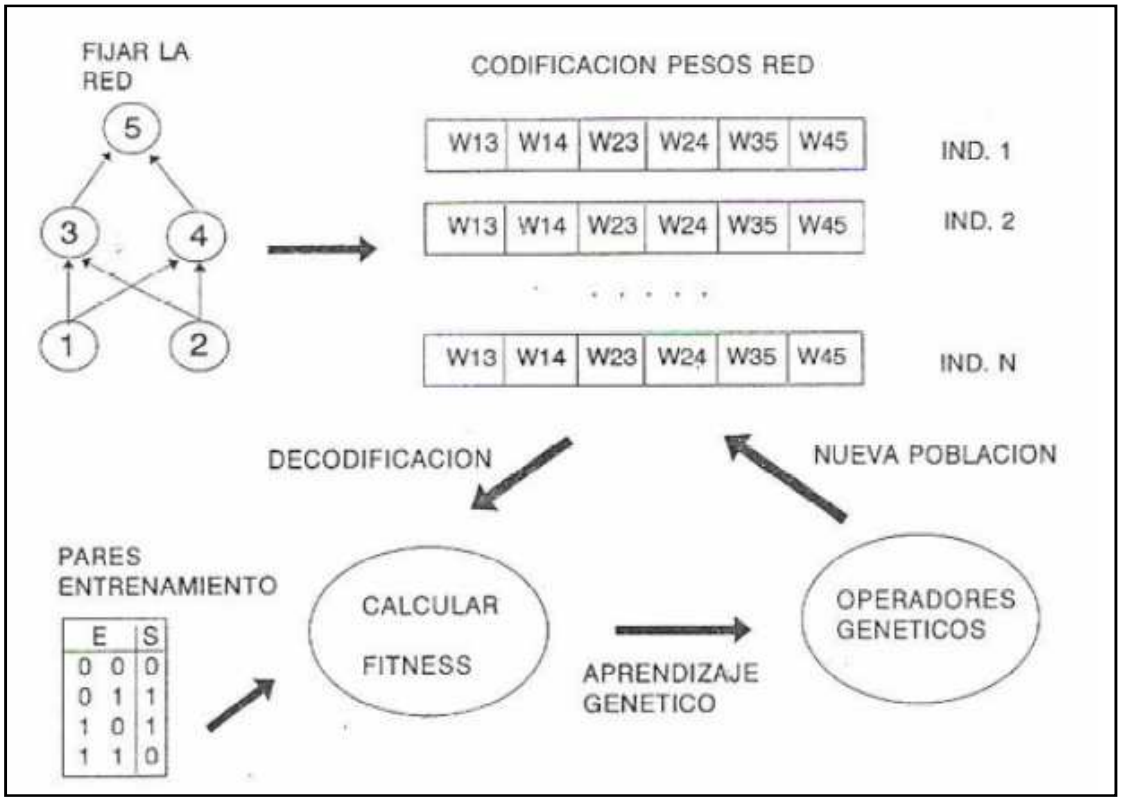


Figura 5: Esquema de la optimización de RNA mediante AG

2.2.3. Algoritmos Genéticos Coevolutivos

Dentro de la teoría de la Computación Evolutiva [MICH 96] ha surgido recientemente un nuevo paradigma, los algoritmos coevolutivos [PARE 95]. Estos algoritmos están recibiendo un interés creciente gracias a su habilidad para desenvolverse en problemas descomponibles con espacios de búsqueda muy grandes.

Los algoritmos coevolutivos se componen de dos o más especies (poblaciones) que interactúan permanentemente entre ellas mediante una función de adaptación conjunta. De ese modo, a pesar de que cada especie tiene su propio esquema de codificación y operadores genéticos, a la hora de evaluar un individuo, su bondad se calcula considerando algunos individuos de las otras especies. Esta coevolución hace más fácil encontrar buenas soluciones en problemas complejos.

Se pueden considerar diferentes tipos de interacciones entre las especies según las dependencias existentes entre los subcomponentes de la solución. En general, podemos mencionar dos tipos distintos de interacción:

- *Algoritmos coevolutivos competitivos* [ROSI 97], donde cada especie compite contra el resto (por ejemplo, para obtener exclusividad de un recurso limitado). En este caso, el incremento de la adecuación de un individuo de una especie implica una disminución de la adecuación en el resto de las especies, es decir, el éxito ajeno supone el fracaso personal.
- *Algoritmos coevolutivos cooperativos* o *simbióticos* [POTT 00], donde todas las especies cooperan para construir una solución al problema. En ese caso, la adecuación de un individuo dependerá de su capacidad para colaborar con individuos de otras especies. Este enfoque es el que ha resultado adecuado para la optimización realizada en la segunda fase de este proyecto.

La Coevolución Cooperativa es recomendable cuando el problema a resolver tiene las siguientes características [PEÑA 01]:

1. el espacio de búsqueda es complejo,
2. el problema, por definición, se puede descomponer,
3. se manejan diferentes tipos de valores y conceptos, y
4. hay una fuerte interdependencia entre los componentes de la solución.

En los algoritmos cooperativos coevolutivos, la evolución de las distintas especies se realiza según el esquema mostrado en la figura 6. En cada generación, y para cada población, se selecciona un conjunto de cooperadores. Para evaluar un individuo en una determinada especie, se combina su información con los cooperadores del resto de especies, formando así una solución al problema. El valor de adaptación para ese

individuo será el resultado de agregar (o calcular la media, el mínimo,…) la evaluación de cada una de las soluciones generadas.

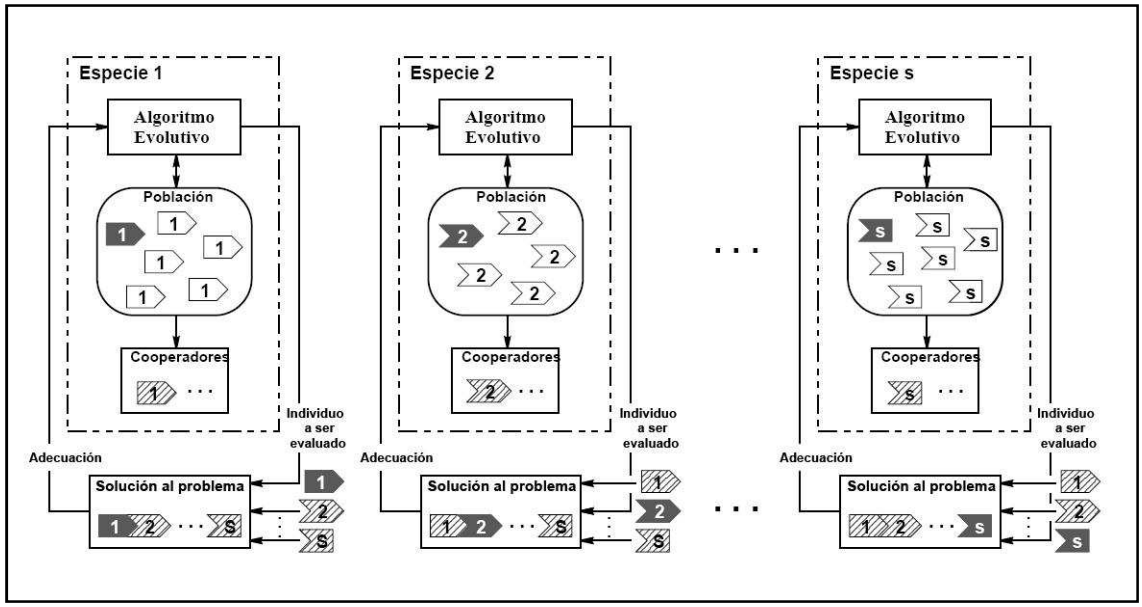


Figura 6: Coevolución Cooperativa [CASI 01]

2.2.3.1. Esquema de interacción cooperativa considerada

Sea F_{ij} la composición de los subcomponentes codificados en los cromosomas *i* y *j* de la especie 1 y 2, respectivamente. El objetivo será minimizar el error cuadrático medio:

$$\mathrm{ECM_{ij}} = \frac{1}{N} \sum_{l=1}^{N} (F_{ij}(x^l) - y^l)^2,$$

donde $F_{ij}(x^l)$ es la salida obtenida en la red para el patrón *l*, y^l es la salida esperada para el mismo patrón, y N es el número de patrones del entrenamiento [CASI 01]. Cada individuo de las especies 1 ó 2 se evalúa con la correspondiente **función de adaptación** f_1 o f_2, que se definen de la siguiente forma:

$$f_1(i) = \min \mathrm{ECM_{ij}} \text{ con } j \in R_2 \cup P_2$$

$$f_2(j) = \min \mathrm{ECM_{ij}} \text{ con } i \in R_1 \cup P_1$$

siendo *i* y *j* los individuos de las especies 1 y 2 respectivamente, *R*1 y *R*2 conjuntos formados por los individuos con el mayor grado de adaptación en la generación previa de las especies 1 y 2 respectivamente, y *P*1 y *P*2 conjuntos de

individuos seleccionados aleatoriamente de la población previa de las especies 1 y 2 respectivamente. La figura 7 ilustra gráficamente el esquema de interacción propuesto. Mientras que los conjuntos $R_{1|2}$ permiten a los mejores individuos influir en el proceso guiando la búsqueda a través de buenas soluciones, los conjuntos $P_{1|2}$ introducen diversidad en la búsqueda. El uso combinado de ambos tipos de conjuntos brinda al algoritmo un buen equilibrio entre explotación ($R_{1|2}$) y exploración ($P_{1|2}$). El diseñador será quien defina previamente la cardinalidad de los conjuntos $R_{1|2}$ y $P_{1|2}$.

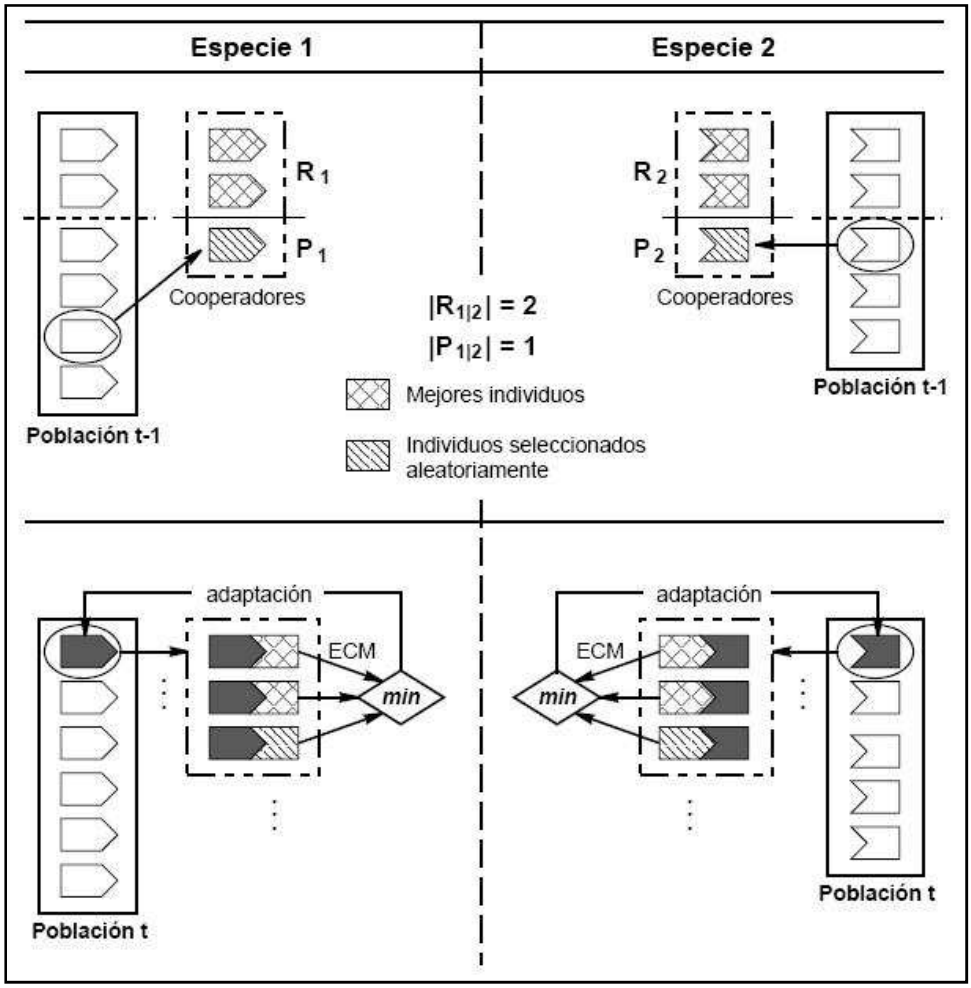


Figura 7: Esquema de interacción considerado en el método de aprendizaje cooperativo [CASI 01]

2.3. Redes Neurogliales Artificiales

2.3.1. Sistema Glial

El SN posee esencialmente dos tipos diferenciados de células, las neuronas y las células gliales [CAJA 04]. Al conjunto de células gliales se las denomina genéricamente

glía o neuroglía, que etimológicamente significa “pegamento de los nervios”. Hay alrededor de 10 a 50 veces más células gliales que neuronas (en los seres más evolucionados en la escala filogenética se observa una proporción mayor). La glía cumple entre otras, funciones de sostén, nutrición y se encarga de la reparación de las lesiones del SN. Esto es debido a que en el sistema nervioso no existe tejido conjuntivo, que es el encargado de la función primordial de sostén e integración sistémica en el resto del organismo. Estas células han seguido un desarrollo embriogénico (desarrollo durante la fase embrionaria del organismo) y ontogénico (desarrollo de un organismo desde el óvulo fertilizado hasta su forma adulta), tanto en forma como en funcionamiento, diferente al de las neuronas. Debido a que son menos diferenciadas que las neuronas, conservan la capacidad mitótica (se reproducen dividiéndose en dos células genéticamente idénticas).

2.3.2. Astrocitos

Los astrocitos son uno de los tipos de células que conforman la glía; forman parte del SN Central (en adelante, SNC) y proporcionan soporte estructural a las células nerviosas además de ayudar a controlar su ambiente extracelular químico e iónico. Estas células se caracterizan por sus largas prolongaciones radiales, algunas de las cuales están en contacto con un vaso sanguíneo (capilar) o rodeando las sinapsis nerviosas (figura 8).

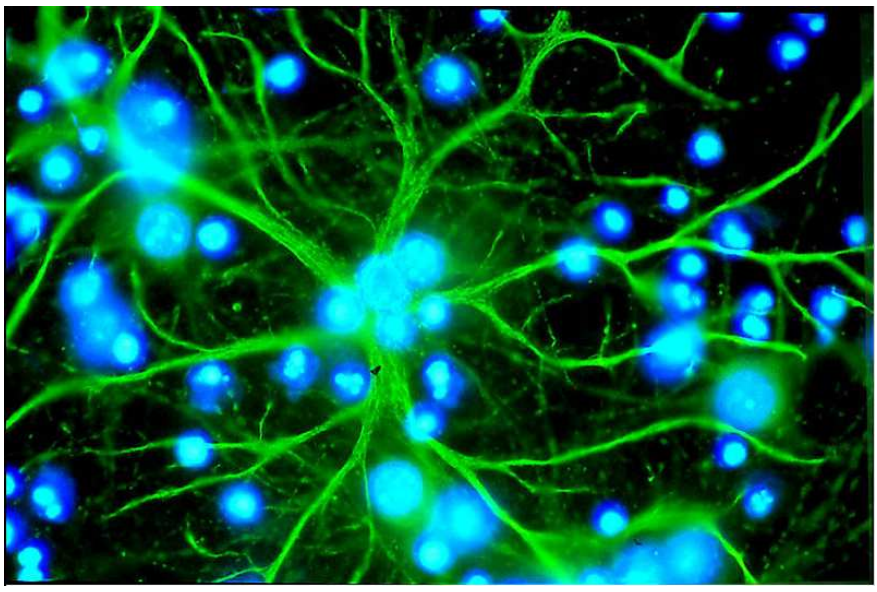


Figura 8: Ejemplo de Astrocito biológico (en verde)

Hasta hace pocos años se creía que la principal tarea de los astrocitos era unir las neuronas a los capilares sanguíneos, así como también la de mantener una concentración equilibrada entre el medio extracelular y el intracelular, previniendo el ingreso de determinadas sustancias potencialmente nocivas. Además, se sabía que participaban en los procesos de regeneración de lesiones en el SN, aumentando su tamaño y enviando sus proyecciones para rellenar la zona dañada. Desde finales de los 80, la aplicación de técnicas celulares y fisiológicas novedosas y refinadamente desarrolladas (como patch-clamp, imagen con fluorescencia sensitiva a los iones, microscopía confocal y biología molecular) a los estudios gliales ha desafiado la idea clásica de que los astrocitos simplemente proporcionaban soporte estructural y trófico a las neuronas, sugiriendo que los astrocitos jugaban papeles más activos en la fisiología del SNC. Por tanto, nuevos descubrimientos sugieren que la glía está íntimamente ligada al control activo de la actividad neuronal y participa en la regulación de la neurotransmisión sináptica [PERE 07]. Se sabe que los astrocitos desempeñan funciones metabólicas, estructurales y homeostáticas muy importantes y que juegan papeles críticos en el desarrollo y fisiología del SNC. Están involucrados en aspectos clave de la función neuronal, tales como: soporte trófico [CAJA 11], supervivencia neuronal y diferenciación [RAFF 93], guía neuronal [KUWA 86; RAKI 90], crecimiento externo de neuritas [LERO 94] y eficacia sináptica [MAUC 01; PFRI 97]. Además, los astrocitos contribuyen a la homeostasis del cerebro, regulando las concentraciones locales de iones [LARG 96] y las sustancias neuroactivas [MENN 94; LARG 96].

Recientemente se han descubierto nuevas funciones de los astrocitos [HAYD 02]: se cree que los astrocitos tienen su propia red en el cerebro (se comunican con ondas de calcio), y que esta red interviene en el procesamiento de la información. Las ondas de calcio pueden propagarse a través de redes de astrocitos o dirigirse a un grupo de astrocitos cercanos, es decir, la información en la glía también tiene sus rutas favoritas. Además, se ha observado que la comunicación entre los astrocitos y las neuronas es bidireccional y compleja [HAYD 01], lo cual ha llevado a proponer un nuevo concepto en la fisiología de la sinapsis, *la sinapsis tripartita*, que consiste en tres elementos funcionales: los elementos presinápticos, los postsinápticos y los astrocitos que los rodean [PERE 07].

El papel de los astrocitos como moduladores de la sinapsis se explica de la forma siguiente: los astrocitos son muy sensibles al nivel de actividad neuronal debido a sus

posiciones dentro del cerebro y a su sensibilidad a cambios en el medio químico compartido por neuronas y astrocitos. Las ondas de calcio pueden ser activadas por la liberación de neurotransmisores y la frecuencia de dichas oscilaciones de calcio puede cambiar acorde al nivel de actividad sináptica. Se ha visto que los astrocitos activados pueden controlar la transmisión sináptica regulando la liberación de neurotransmisores. Esta regulación puede ser tanto excitatoria (secretando el mismo neurotransmisor) como inhibitoria (absorbiendo el neurotransmisor). Si la regulación de la actividad sináptica es el efecto a corto plazo de los astrocitos, también pueden modificar la fuerza de la actividad sináptica mediante la liberación de moléculas que causan que el axón de la neurona incremente o decremente la cantidad de neurotransmisor que libera. Recientes experimentos desvelan los caminos de las señales entre las redes de neuronas y las redes de astrocitos [PERE 02], por lo que es posible modelar un sistema que incluya esta relación. Las neuronas se comunican con otras neuronas, mientras que los astrocitos "escuchan" esta comunicación, regulando la actividad de las neuronas y comunicándose con otros astrocitos basándose en lo que "escuchan".

2.3.3. Funcionamiento de las RNGA

Dadas las evidencias existentes sobre la importancia de los astrocitos en el procesado de la información en el cerebro, se decidió, tal como se ha indicado, construir redes artificiales que incluyesen no sólo neuronas sino también elementos de control que simulasen la influencia astrocítica, las RNGA. La construcción y funcionamiento de una RNGA sigue todos los pasos de un SC, comenzando con el diseño de la arquitectura de red, seguido por el entrenamiento, validación y test y fase de ejecución, fases que se detallan a continuación.

FASE DE DISEÑO

El diseño considerado hasta el momento para una RNGA está basado en arquitecturas multicapa conectadas hacia delante sin conexiones laterales, sin retroalimentación y totalmente conectadas, y orientadas hacia la resolución de problemas de clasificación y reconocimiento de patrones.

FASE DE ENTRENAMIENTO

Se trata de un entrenamiento híbrido, que combina aprendizaje no supervisado (primera fase) con aprendizaje supervisado utilizando AG (segunda fase).

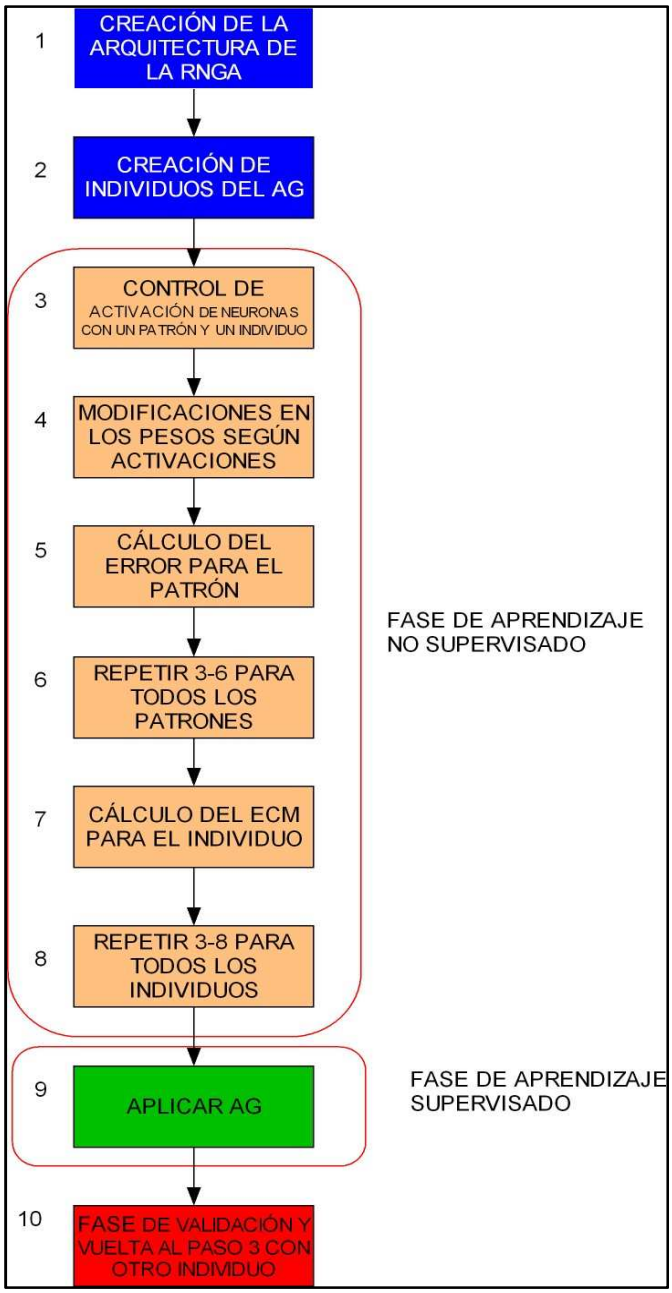


Figura 9: Esquema del entrenamiento híbrido de la RNGA

Fase de aprendizaje no supervisado

La primera fase, tal como muestra el esquema de la figura 9, consiste en un aprendizaje no supervisado basado en el comportamiento de los astrocitos. Dado que el

AG trabaja sobre un conjunto de individuos, el primer paso será crear este conjunto. Cada individuo del AG representará una RNGA y consistirá en tantos valores como pesos de las conexiones en las RNGA, de forma que cada conjunto arbitrario de valores de todos los pesos constituirá un individuo diferente. Cada uno de estos individuos será modificado cada vez que un patrón de entrada pase por la red de acuerdo a la actividad de las neuronas durante el paso de ese patrón. Para cada individuo, cada patrón se presenta a la red durante un número dado de veces o iteraciones. Estas iteraciones representan la lentitud de actuación de los astrocitos y constituyen un ciclo de ese patrón. Durante cada iteración del ciclo, las conexiones son modificadas por los astrocitos artificiales de acuerdo a ciertas reglas establecidas según lo observado en los trabajos de A. Porto y A. Alvarellos [PORT 04; PORT 05a; PORT 05b; ALVA 07], que se basan en la frecuencia de activación de las neuronas.

El funcionamiento de los astrocitos artificiales es el siguiente: la red se evalúa **X** iteraciones con un mismo ejemplo de entrenamiento. Si una neurona se ha activado **Y** veces durante las **X** iteraciones, la glía lo tendrá en cuenta para modificar sus pesos (activarse implica que su salida sea mayor que un umbral determinado). Las modificaciones genéricas realizadas por los astrocitos artificiales en la experimentación de este proyecto, consisten en:

- Incrementar en un 25% el peso de las conexiones que salen de las neuronas activadas Y veces durante X iteraciones.
- Disminuir en un 50% el peso de las conexiones que salen de las neuronas no activadas nunca Y veces durante X iteraciones.

Por ejemplo, si en un problema se trata de una combinación iteración-activación 6-3, se presentaría este mismo patrón 6 veces a la red y se tendría en cuenta si cada neurona se ha activado más o menos de 3 veces. Se incrementaría un 25% el peso de las conexiones que salen de las neuronas activadas 3 veces y se decrementarían un 50% en caso contrario.

Se considera que si una neurona se activa durante las iteraciones definidas, el astrocito artificial que controla la capa de dicha neurona se activa e influye directamente sobre las conexiones que tiene esa neurona con sus respectivas postsinápticas. Del mismo modo, si la actividad de una neurona no llega a la activación requerida, el astrocito artificial no es excitado por esa neurona y se debilitan las conexiones que salen de dicha neurona.

Esas dos o tres iteraciones de activación pretenden representar una gran actividad neuronal (alta frecuencia de estimulación) en la realidad cerebral. Se vio que los porcentajes de incremento y decremento de 25% y 50% proporcionaban resultados satisfactorios tras probar varias combinaciones, aunque no todas las posibles, porque ello condicionaría una cantidad inabordable de pruebas [PORT 04]. Los fundamentos biológicos justificaron además esta elección, ya que si el incremento por refuerzo es menor que el decremento cuando no hay actividad constante, solamente permanecerán reforzadas aquellas conexiones de neuronas que presenten una actividad continuada [ARAQ 02].

Antes de comenzar las simulaciones, y basándose en resultados previos [PORT 04] que comprobaron que más de 8 iteraciones incrementan mucho el tiempo de simulación y que con menos de dos no da tiempo a que la glía actúe, se eligieron las combinaciones de iteraciones y activaciones con las que se realizarían las pruebas y que se muestran en la tabla 1.

Activaciones	**Iteraciones**
2	4
	6
3	6
	8

Tabla 1: Combinaciones analizadas de activación-iteración

Para cada uno de los problemas aquí tratados se han realizado simulaciones con estas cuatro posibles combinaciones de iteración-activación, es decir, para cada una de estas combinaciones (4 iteraciones-2 activaciones, 6 iteraciones-2 activaciones, 6 iteraciones-3 activaciones, 8 iteraciones-3 activaciones) se han realizado 100 test distintos (10 conjuntos de entrenamiento distintos **x** 10 poblaciones iniciales distintas). En total 400 simulaciones de RNGA para cada problema. En el caso de las RNA se han realizado 100 test (10 x 10), sin glía artificial en idénticas condiciones (mismos parámetros de la red, conjuntos de entrenamiento y test, mismas poblaciones iniciales, mismos parámetros del algoritmo genético, etc.).

Continuando con la primera fase de entrenamiento, una vez que el ciclo del patrón haya terminado, se calculará el error de esa red para ese patrón dado con el fin de

encontrar la diferencia entre la salida obtenida y la deseada. El error de la red para cada patrón será almacenado y una vez que todos los patrones hayan pasado por la red, se calculará el error cuadrático medio (ECM) para ese individuo. El proceso será el mismo para todos los individuos. Esta fase constituye un entrenamiento no supervisado, porque las modificaciones de los pesos de las conexiones no consideran el error de la salida, sino que tienen lugar en el tiempo de acuerdo a la frecuencia de activación de cada neurona, simulando los refuerzos e inhibiciones que son probablemente provocados por los astrocitos en el cerebro.

A modo de resumen, considerar que se parte de un individuo generado aleatoriamente, que contiene los pesos de las conexiones de la red. Se introduce un patrón y se calculan en cada neurona de la primera capa, el valor neto y el valor de activación. A continuación se hace lo mismo para cada neurona de cada capa hasta obtener la salida. Los astrocitos artificiales contabilizan el número de activaciones de cada neurona. En este proceso actúa solamente la glía artificial, que es la única que va modificando los pesos de las conexiones. Estas modificaciones son temporales, pues al introducir el siguiente patrón el individuo es el mismo que al comienzo, no han perdurado los cambios realizados en las conexiones. Una vez terminado el ciclo de presentación a la red del mismo patrón, (ej. si la combinación es 6-3 habría que pasarlo 6 iteraciones), se obtiene la salida. Se calcula la diferencia entre salida deseada y obtenida, esto es, el error para el individuo 1 con el patrón 1. Se hace lo mismo con todos los patrones de los que se dispone, para finalmente calcular el ECM del individuo 1. Después se repite el proceso con todos los individuos hasta que se disponga de un array con todos los individuos con su ECM asociado y ordenados en base a este valor. Este array, de tantas posiciones como individuos, es el que usará el Algoritmo Genético para realizar mutaciones y cruces. Hasta aquí llegaría el Aprendizaje No Supervisado, pues para modificar pesos no se tiene en cuenta el error de la salida sino las activaciones de las neuronas controladas por los astrocitos artificiales.

Fase de aprendizaje supervisado

La segunda fase del entrenamiento es un entrenamiento supervisado. Consiste en aplicar los AG a los individuos considerando su ECM, el cual fue almacenado durante la primera fase de entrenamiento. El AG en esta segunda fase emplea los correspondientes operadores genéticos (cruces y mutaciones) y selecciona los nuevos

individuos, con los cuales la primera y segunda fases serán repetidas hasta que se alcance el mínimo ECM posible o se ejecuten un número de generaciones concreto, previamente determinado. La segunda fase se considera entrenamiento supervisado porque el AG tiene en cuenta el error cometido por la red para seleccionar los individuos que serán cruzados y mutados, es decir, hace los cambios en los pesos de acuerdo a ese error.

FASE DE VALIDACIÓN

Como ya se comentó en el capítulo de *Fundamentos Teóricos* dedicado a Redes de Neuronas Artificiales, el proceso de validación se realiza de forma conjunta al entrenamiento, con un conjunto de patrones distinto al de entrenamiento y test, con el fin de asegurarse que la red se está entrenando de forma satisfactoria.

El proceso de parada temprana se emplea para detener el entrenamiento en el momento óptimo, obteniendo la configuración más adecuada de una red y evitando así el sobreentrenamiento. Al disponer para un mismo problema de 3 conjuntos diferentes de patrones (entrenamiento, validación y test), se va validando la red a utilizar, usando para ello el conjunto de validación, de forma simultánea al conjunto de entrenamiento, y guardando las configuraciones intermedias de la red para aquellos valores que hagan mínimo el error de validación. Después de finalizado el entrenamiento-validación se dispondrá de la configuración de la red con mínimo error de validación. Esta red se probará con el conjunto de test para comprobar si realmente funciona correctamente. En los problemas estudiados en este proyecto, debido a no disponer de un número suficiente de ejemplos como para establecer 3 conjuntos de patrones, se ha empleado el conjunto de validación como conjunto de test.

En la figura 10 se puede observar una gráfica donde se muestra la evolución del error de entrenamiento, y cómo, a partir de un cierto valor de éste, el error en validación no mejora sino que empeora. Esta gráfica muestra la situación idealizada, en la práctica la gráfica del error de validación presentaría numerosos mínimos locales.

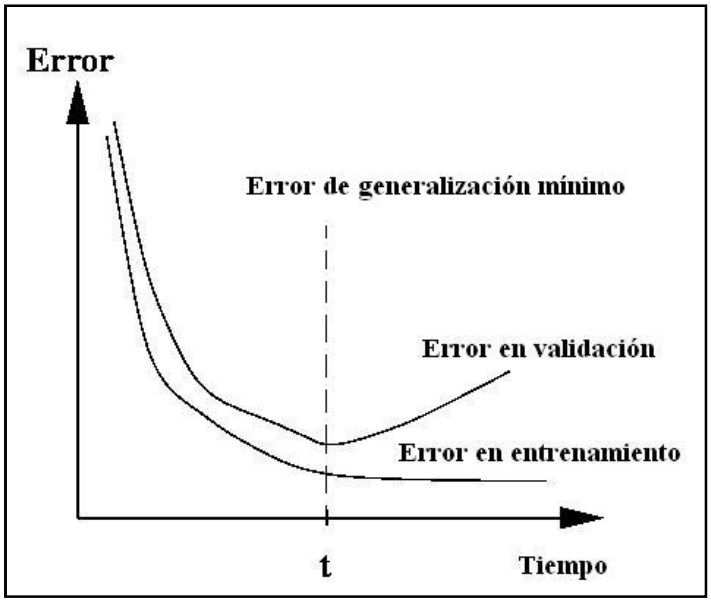


Figura 10: Evolución del error de aprendizaje y del error de generalización [PREC 98]

FASES DE TEST Y EJECUCIÓN

El entrenamiento de la RNGA proporciona individuos cuyos pesos permiten obtener el menor error posible en la salida. Durante la fase de test se usarán estos individuos para comprobar si la salida obtenida por el modelo es correcta, es decir, si la capacidad de generalización de la RNGA es correcta con patrones de entrada distintos de aquellos usados durante la fase de entrenamiento y validación.

A diferencia de lo que ocurre con otros modelos de redes, con RNGA en la fase de test y en el curso de todas las ejecuciones subsiguientes, los astrocitos artificiales continúan actuando de la misma manera que durante la fase de entrenamiento no supervisada. Estos elementos incorporados serán, por tanto, parte del modelo en todas sus etapas y participarán directamente en el procesado de información.

2.3.4. Justificación del método de funcionamiento de las RNGA

El método de entrenamiento de las RNGA que se ha descrito es el utilizado hasta el momento por A. Porto [PORT 05; PORT 07] y es el que se ha utilizado en este proyecto. Este método surge en respuesta a las dos siguientes observaciones: por un lado se sabe que el algoritmo BP y otros métodos de gradiente presentan dificultades para entrenar RNA con elementos de procesamiento diferentes, o más complejos, que los usados tradicionalmente [DORA 99; DORA 00; PAZO 99]. Por otra parte, los

Algoritmos Evolutivos son particularmente útiles para tratar con cierta clase de problemas y han sido utilizados ya en diversos trabajos para entrenar RNA con elementos de procesamiento modificados [DORA 99; PORT 04; RABU 04]. Aunque es menos probable que estos algoritmos caigan en mínimos locales que los métodos de gradiente, son, sin embargo, bastante ineficientes en el ajuste local a la solución [YAO 99]. Diversos autores han propuesto aprovechar la eficacia de la CE y enmendar su defecto incorporando al proceso de entrenamiento un método de búsqueda local para conseguir un buen ajuste. Por ejemplo, la aproximación híbrida AG/Retropropagación emplea los Algoritmos Genéticos para encontrar un conjunto inicial de pesos óptimos para las conexiones, para después usar Retropropagación para realizar una búsqueda local desde estos pesos iniciales. El entrenamiento híbrido ha sido empleado exitosamente en muchas áreas de aplicación [KINN 94; TAHA 95; YAN 97; ZHAN 95].

Teniendo en cuenta lo mencionado más arriba, el método híbrido que se utiliza en este proyecto ha demostrado ser eficiente porque los AG son eficaces en búsquedas globales [WHIT 95; YAO 99] y su uso ha permitido incorporar fácilmente los astrocitos artificiales que facilitan un buen ajuste local en la búsqueda de la solución. La diferencia fundamental con respecto a otros métodos híbridos [ERKM 97; LEE 96; YAN 97; ZHAN 95] reside en que no se usa el AG para la búsqueda de los pesos iniciales de las conexiones, sino que la técnica de búsqueda local empleada se realiza en base al comportamiento biológico observado en los astrocitos.

Las RNGA construidas con este método de entrenamiento se compararán en este trabajo con RNA multicapa entrenadas solamente con AG (usando los mismos parámetros en los AG en ambos casos, incluso se ha partido en las pruebas de las mismas poblaciones iniciales y de las mismas semillas de inicialización de operaciones aleatorias como el cruce o la mutación), para determinar la eficacia de los astrocitos artificiales.

2.4. Método de Comparación de resultados: Cross-Validation o Validación Cruzada

A la hora de comparar algoritmos de clasificación, el método más utilizado es el de *cross validation* o validación cruzada para estimar la precisión de los algoritmos. La validación cruzada se emplea para evitar la influencia de los conjuntos de datos en los resultados de un problema, esto es, para comprobar hasta qué punto los resultados son deudores de un determinado conjunto de patrones.

En el método de validación cruzada, el conjunto D de datos se divide en k conjuntos $D_1,\ldots, D_k$ no solapados (*k-fold cross validation*). En cada iteración i (que varía de 1 a k), el algoritmo se entrena con el conjunto $D\backslash D_i$ y se hace test en D_i. Sin embargo, algunos estudios han mostrado que la comparación de algoritmos utilizando estos *t-tests* en validación cruzada conlleva un ligero aumento de lo que se denomina error tipo I [HERR 04], es decir, detectar una diferencia cuando no existe.

Dietterich analizó el comportamiento del método *k-fold cross validation*, combinado con el empleo de un test t [DIET 98]. En ese trabajo se propone modificar el estadístico utilizado y se justifica que es más efectivo realizar k/2 ejecuciones de un test *2-fold cross validation* con diferentes permutaciones de los datos, que realizar un test *k-fold cross validation*. Como solución de compromiso entre la potencia del test y el tiempo de cálculo, propone realizar 5 ejecuciones de un test de validación cruzada con k=2, de ahí el nombre 5x2cv, lo cual también es propuesto por Alpaydin en [ALPA 99]. En cada una de las 5 iteraciones, los datos se dividen aleatoriamente en dos mitades. Cada una de las mitades se toma como entrada del algoritmo y la otra se utiliza para hacer un test de la solución final, con lo que en total se tienen 10 tests distintos (5 iteraciones, 2 resultados por cada una) [CANT 05]. Este es el método que se utiliza en este trabajo, y ya empleado en [IBÁÑ 08; VEIG 08], para comparar los resultados de las RNGA con los obtenidos por las RNA. En todos los casos, la base de la comparación es la media de los 10 tests realizados por el método 5x2cv, que realiza una medida de la precisión obtenida en los tests. Además, para independizar los resultados de la inicialización de los datos, se han utilizado 10 poblaciones iniciales distintas, con cada una de las cuales se han realizado estos 10 test. Es decir, en este trabajo se han realizado y analizado 100 test distintos para cada problema en el caso de RNA y 400 en el caso de RNGA, pues como ya se ha indicado, cada uno de estos 100 test se ha realizado con 4 combinaciones distintas de los parámetros iteración-activación (4-2, 6-2, 6-3, 8-3).

Un problema del método 5x2cv como base para comparar métodos es que este método exige particionar el conjunto de patrones en dos mitades. Esto puede no representar ningún problema cuando se trabaja con conjuntos lo suficientemente amplios. Sin embargo, en los problemas tratados en este trabajo los conjuntos de datos disponibles son bastante pequeños. Es por ello que, al no haber datos suficientes para realizar entrenamiento, validación y test, no se ha realizado validación, pues ello exigiría dividir en dos el conjunto de entrenamiento, el cual es a su vez la mitad del conjunto de patrones. Esta última división provocaría que, o bien el entrenamiento o bien la validación se produjese con un número muy reducido de patrones, que seguramente no sería representativo del espacio de búsqueda que se estuviese explorando, por lo que la red no generalizaría bien y se comportaría mal en el test, pues las redes creadas sólo representarían una parte del conocimiento deseado. Para que no ocurra esto, hay que evitar trabajar con conjuntos de patrones demasiado pequeños (poco representativos). En este caso, esto se hace al no dividir el conjunto de entrenamiento en entrenamiento y validación, es decir, no realizar validación. Al no hacerla, se está realizando un sobreentrenamiento de las redes generadas. Sin embargo en [RIVE 07a; RIVE 07b], se muestra que la precisión obtenida es mayor cuando esas redes se sobreentrenan que cuando son generadas con conjuntos de entrenamiento y validación pequeños. En caso de contar con un conjunto de patrones lo suficientemente amplio, lo más recomendable sería particionar éste en tres partes: entrenamiento, test y validación. En este caso, la precisión ofrecida por el sistema sin validación (sobreentrenando) sería menor que al usar validación. Sin embargo, al no ser esto posible, se divide sólo en entrenamiento y test.

2.5. Contrastes de Hipótesis

Una hipótesis estadística es cualquier conjetura sobre una o varias características de interés de un modelo de probabilidad. Formulada ésta, un contraste o test de hipótesis es una técnica de inferencia que permite comprobar si la información proporcionada por una muestra concuerda con la hipótesis planteada. Las hipótesis estadísticas pueden ser:

- Paramétricas: afirmación sobre los valores de los parámetros poblacionales desconocidos. Será simple si especifica un único valor para cada parámetro

poblacional desconocido, y será compuesta si asigna un conjunto de valores posibles a parámetros poblacionales desconocidos.

- No paramétricas: afirmación sobre alguna característica estadística de la población en estudio. Por ejemplo, las observaciones son independientes, la distribución de la variable es normal, la distribución es simétrica, etc. [VILA 03]

La hipótesis que se contrasta se denomina Hipótesis Nula (H_0). La hipótesis que se acepta si se rechaza H_0 es la Hipótesis Alternativa (H_1). "*H_0 debe ser la hipótesis que el experimentador asume como correcta y que no necesita ser probada. […] La aceptación de H_0 no implica que ésta sea correcta o que haya sido probada sino que los datos no han proporcionado evidencia suficiente como para refutarla. […] Si el experimentador quiere respaldar con contundencia un determinado argumento es debido a que éste no puede ser asumido gratuitamente y, por tanto, sólo podrá ser defendido a través del rechazo del argumento contrario (el establecido en H_0)*" [CAO 98]. La medida de discrepancia entre la información de la muestra y la hipótesis H_0 se denomina estadístico de contraste, que debe seguir una distribución conocida cuando H_0 sea cierta, de modo que se pueda distinguir una discrepancia grande que tenga una probabilidad muy pequeña de ocurrir cuando H_0 sea cierta, y una discrepancia pequeña que tenga una probabilidad grande de ocurrir cuando H_0 sea cierta. A continuación, se debe decidir qué discrepancias se consideran inadmisibles cuando H_0 sea correcta (valor o nivel de significación del contraste (α = P (rechazar H_0 | H_0 es cierta)). Finalmente, se toma la muestra y se calcula el valor del estadístico $\hat{d}$ asociado a la muestra (valor crítico del contraste), de manera que si $\hat{d}$ es pequeño (pertenece a la región de aceptación) se acepta H_0, y si $\hat{d}$ es grande (pertenece a la región de rechazo) se rechaza H_0.

Se denomina nivel crítico o p-valor a la probabilidad p de obtener una discrepancia con H_0 mayor o igual que el valor crítico $\hat{d}$ cuando H_0 es correcta; de esta forma, si $\alpha \geq$ p-valor entonces se procede a rechazar H_0. Con carácter general, y de manera orientativa, un p-valor inferior a 0,01 sugiere el rechazo de H_0, superior a 0,1 indica su aceptación, y uno comprendido entre 0,01 y 0,1 no suele considerarse determinante.

En el presente estudio H_0 sería: "La media de los resultados obtenidos con RNA es menor o igual a la media de los resultados obtenidos con RNGA". Se trata de una hipótesis estadística paramétrica y compuesta, que determina un contraste de hipótesis

unilateral por la derecha (de una cola). En este caso las muestras son apareadas, pues aparecen como distintas observaciones realizadas sobre los mismos datos. En este proyecto, las observaciones apareadas consisten en considerar las 100 combinaciones del cross-validation (10 conjuntos x 10 poblaciones) a las que se les aplica por un lado una RNA (variable *X*) y, por otro, una RNGA (variable *Y*) como sistemas de resolución. No es posible considerar a *X* e *Y* como variables independientes ya que va a existir una dependencia clara entre las dos variables. Si se quiere contrastar que los resultados han experimentado o no una mejoría con el empleo de RNGA, se pueden emplear diferentes test de hipótesis que veremos a continuación.

2.5.1. Test paramétrico T de Student sobre la diferencia de medias con muestras apareadas

El test t de Student es un contraste en el que el estadístico utilizado sigue una distribución t de Student [GOSS 08] si la hipótesis nula es cierta. Si se denomina d_i a la diferencia entre las observaciones antes y después de incluir la glía artificial

$$d_i = x_i - y_i$$

Para contrastar la hipótesis:

$$H_0 : \mu_x = \mu_y \leftrightarrow \mu_x - \mu_y = 0 \leftrightarrow \mu_d = 0$$

en el caso en que H_0 fuese cierta tendríamos que el estadístico de contraste a emplear es

$$d = \frac{\text{đ}}{\frac{\hat{S}_d}{\sqrt{n}}} \sim t_{n-1}$$

donde đ es la media muestral de las diferencias *di* y $\hat{S}_d$ es la cuasivarianza muestral de las mismas.

La aplicación del test t de Student requiere la normalidad de las observaciones para cada uno de los grupos. La comprobación de esta hipótesis puede realizarse tanto por medios gráficos (histogramas, diagramas de cajas o gráficos de normalidad) como mediante tests estadísticos (test de Kolmogorov-Smirnoff Lilliefors, test de Shapiro-Wilks) [PERT 01]. Para la contrastación de la normalidad en este trabajo se ha empleado el test de Kolmogorv-Smirnov Lilliefors. Éste supone una adaptación del test de Kolmogorov-Smirnov empleada para contrastar la hipótesis nula de que los datos provienen de una población distribuida normalmente.

Por otro lado, "*si las muestras son grandes, el Teorema Central del Límite garantiza que la aproximación normal para las distribuciones de la media de X y la media de Y (caso de muestras independientes), o de la media de D (caso de muestras apareadas) es adecuada y por consiguiente la violación de la hipótesis de normalidad no es especialmente preocupante. Más aún, se ha demostrado que el test t de Student es muy robusto (insensible) a las desviaciones de la normalidad y por todo ello los estadísticos utilizados proporcionan regiones críticas razonablemente válidas para tamaños muestrales grandes (pongamos n > 30).*

Si las muestras son pequeñas se dispone de contrastes no paramétricos como el contraste de los rangos de Wilcoxon para muestras independientes y una adaptación de éste para muestras apareadas (conocido también como contraste de los rangos con signo)." [CAO 98]

2.5.2. Test no paramétrico de Wilcoxon de rangos signados

El contraste de Wilcoxon [WILC 45] es una técnica no paramétrica apropiada cuando la suposición de normalidad no es válida. Igualmente se dispone de *n* parejas de valores (x_i, y_i) que se pueden considerar como una variable medida con cada conjunto de datos en cada una de las dos arquitecturas (sin glía y con glía). En ocasiones, las hipótesis necesarias para el test paramétrico (normalidad de las diferencias apareadas, d_i) no se verifican y es estrictamente necesario realizar el contraste que ahora se presenta.

El procedimiento consiste en [MOOR 02; WALP 98]:

1. Ordenar todos los valores absolutos de las diferencias ($|d_i|$) entre las observaciones pareadas de menor a mayor.

2. Asignar a la diferencia en valor absoluto más pequeña rango 1, a la siguiente en tamaño, 2, y así sucesivamente, sin tener en cuenta el signo. Cuando el valor absoluto de dos o más diferencias sea el mismo, se asignará a cada uno el promedio de los rangos que se asignarían si las diferencias se distinguieran.

3. A cada rango ($r(|d_i|)$*:* valor del orden del dato en el conjunto) se le asigna el signo de la diferencia.

4. Calcular la suma de rangos positivos por un lado (w_+), y la de rangos negativos por otro (w_-).

5. La suma de los rangos con diferencias negativas omitido su signo es el estadístico de contraste que se suele denotar con la letra W (o T).

Las hipótesis manejadas por este contraste son las siguientes:

$$\begin{cases} H_0: No\ hay\ diferencia\ entre\ las\ medianas\ de\ las\ muestras\ apareadas\ (Med_{RNGA} = Med_{RNA}) \\ H_1: Sí\ hay\ diferencia\ entre\ las\ medianas\ de\ las\ muestras\ apareadas\ (Med_{RNGA} < Med_{RNA}) \end{cases}$$

Cuanto menor sea una suma con respecto a la otra, mayor será la evidencia de que las dos poblaciones difieren en su localización (son distintas) [MORI 08]. En este caso, la hipótesis nula se puede rechazar a favor de la alternativa sólo si w_+ es pequeña y w_- es grande [WALP 98].

Como se puede observar, la medida característica contrastada con este test es la mediana, medida de posición central robusta e interesante por su poca sensibilidad a la presencia de observaciones atípicas en la muestra. Por otra parte, remarcar que si la distribución es normal, la media y la mediana coinciden, motivo por el cual en una situación donde se verificase la hipótesis de normalidad el t-test ofrecería un contraste sobre ambas medidas.

Para realizar los correspondientes contrastes de hipótesis en este proyecto se ha empleado el siguiente software: MS Excel 2007 para realizar los test t de Student, MATLAB para realizar el test de normalidad de Kolmogorov-Smirnov Lilliefors, y R para el test de Wilcoxon de los rangos signados. A continuación, en la figura 11, se muestra el código de MATLAB empleado para realizar el test de normalidad.

```
function [] = Lilliefors_test (pestanas)

fid = fopen('Lilliefors.txt','w');

for i = 1:pestanas
    ndata = xlsread('nuevo.xls',i);
    [h p] = lillietest(ndata(:,1));
    fprintf(fid,'%.3f %.9f\n' ,h,p);
end

fclose(fid);
```

Figura 11: Código en MATLAB empleado para realizar el test de Kolmogorov-Smirnov Lilliefors

3. DESARROLLO DEL ENTORNO DE SIMULACIÓN

3.1. Metodología de desarrollo

Se ha desarrollado una aplicación que permite la construcción, entrenamiento, validación y test tanto de RNA como de RNGA, facilitando la posterior comparación de los resultados de ambos tipos de redes. Se han desarrollado además herramientas para permitir un análisis automático y semiautomático de resultados. Todo ello se ha llevado a cabo siguiendo una metodología de desarrollo de software en cascada [PRES 97], con las fases de análisis, diseño, codificación, pruebas, puesta en funcionamiento y posterior documentación y mantenimiento.

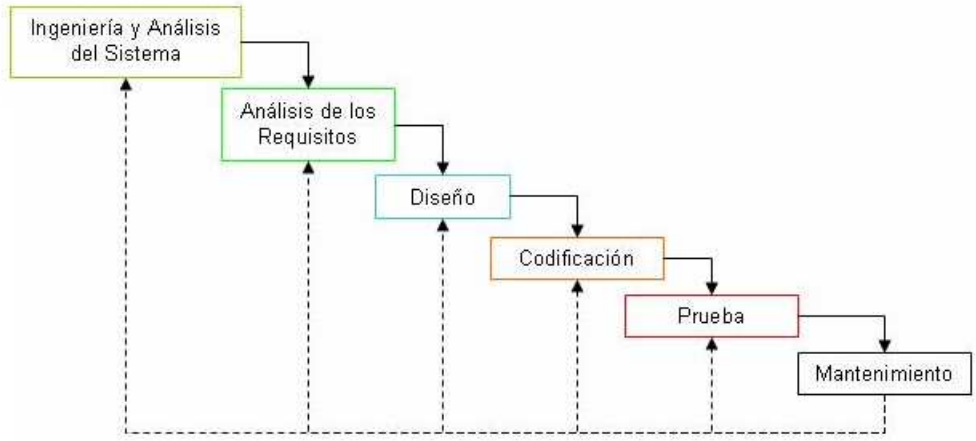


Figura 12: Metodología de desarrollo en cascada

El Modelo en cascada puro rara vez se utiliza tal cual, pues esto implicaría un previo y absoluto conocimiento de los requisitos, la no volatilidad de los mismos (o rigidez) y etapas subsiguientes libres de errores; ello sólo podría ser aplicable a escasos y pequeños desarrollos de sistemas. Para este proyecto, el modelo cascada empleado es uno de los actualmente más utilizados, por su eficacia y simplicidad, en software de pequeñas y medianas dimensiones. Se produce alguna realimentación entre etapas, lo que da oportunidad al desarrollo de productos software en los cuales hay ciertas incertidumbres, cambios o evoluciones durante el ciclo de vida. Así por ejemplo, una vez elicitados y especificados los requisitos, se puede pasar al diseño del sistema, pero durante esta última fase lo más probable es que se deban realizar ajustes en los requisitos (aunque sean mínimos), ya sea por errores detectados, ambigüedades o bien

porque los propios requisitos han cambiado o evolucionado; con lo cual se debe retornar a una etapa previa, hacer los pertinentes reajustes y luego continuar nuevamente con el diseño; esto último se conoce como realimentación.

Este modelo presenta un gran atractivo para el desarrollo de este trabajo por varios motivos:

- El proyecto presenta una alta rigidez (no se esperaban grandes cambios en su desarrollo).
- Requisitos y objetivos claros y bien especificados.
- Ante la aparición de imprevistos permite el retorno o avance a distintas fases de la metodología.

El análisis de requisitos se realizó tanto desde el punto de vista de la Inteligencia Artificial como desde el punto de vista de la Neurociencia, con la participación en el mismo de los investigadores del Instituto Cajal del CSIC para lo referente al comportamiento de los astrocitos artificiales.

En cuanto al diseño de la aplicación, se continuó con la arquitectura de diseño modular existente en la aplicación tomada como punto de partida. Dicha aplicación, desarrollada en lenguaje DELPHI (interfaz) + C (algoritmos), fue el resultado de los trabajos previos en este campo de A. Porto y A. Alvarellos [PORT 04; ALVA 07]. Mediante este proyecto se ha mejorando la entrada/salida de la herramienta, la rapidez de ejecución gracias al paralelismo de las pruebas, el análisis de resultados y el acceso a diversas funcionalidades, tal como se explicará más adelante en este capítulo.

Con respecto a la metodología de diseño y construcción de los Sistemas Conexionistas necesarios, se ha empleado una metodología propia, desarrollada en el seno del equipo de investigación de los directores del proyecto, al no haber una formalmente aceptada en la literatura científica, y tener el grupo una gran experiencia en la experimentación con Redes de Neuronas Artificiales. Destacar que en el marco del desarrollo de Sistemas Conexionistas no existe ninguna metodología bien definida, pero es aconsejable seguir un proceso metodológico explícito, para no saltarse pasos involuntariamente que pongan en peligro el éxito del desarrollo. La metodología que se ha seguido está basada, por tanto, en la propia experiencia y en la experiencia del uso de otras disciplinas relacionadas como por ejemplo la Ingeniería del Conocimiento [GOME 88]. La metodología define los siguientes pasos:

- Identificación del problema a resolver
- Adecuación - Justificación del uso del Sistema Conexionista (SC)
- Diseño y construcción del SC
- Preparación de los datos y construcción de los conjuntos de prueba y entrenamiento
- Entrenamiento del SC
- Organización y análisis de resultados
- Ajuste de parámetros
- Validación del SC

Se presentan en la figura 13 los casos de uso de la aplicación de simulación.

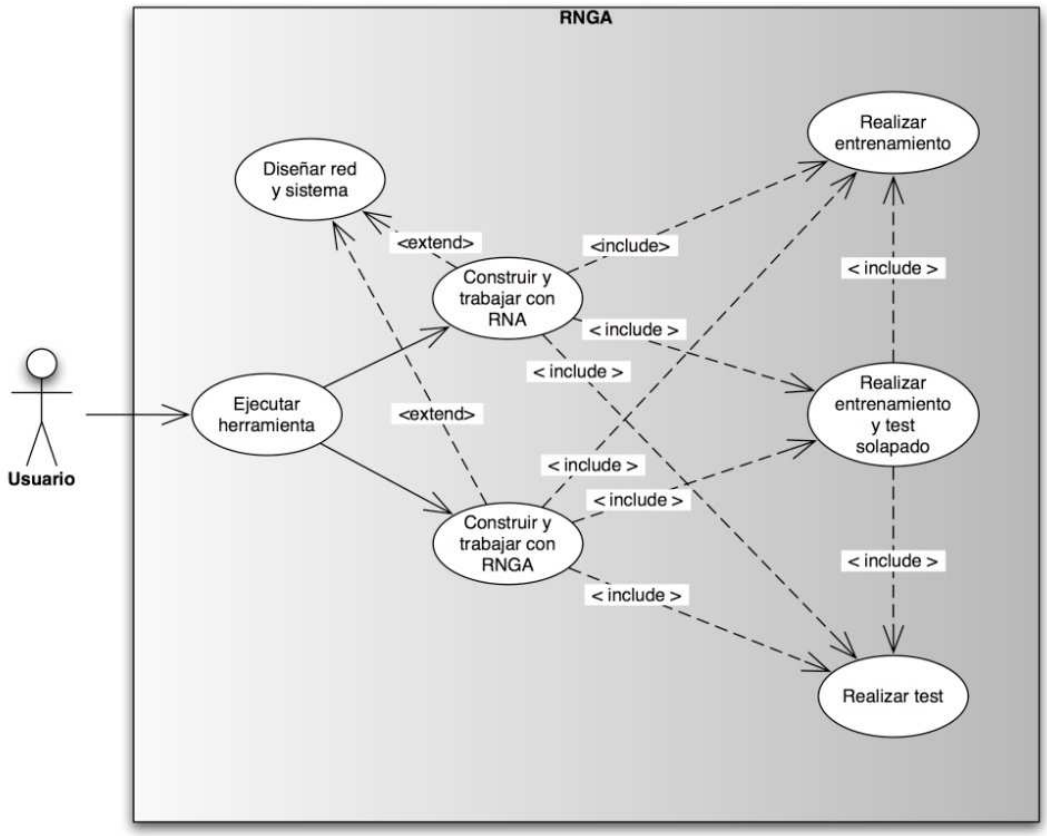


Figura 13: Casos de Uso de la Aplicación de Simulación

El diagrama de flujo principal de la aplicación se presenta en la figura 14.

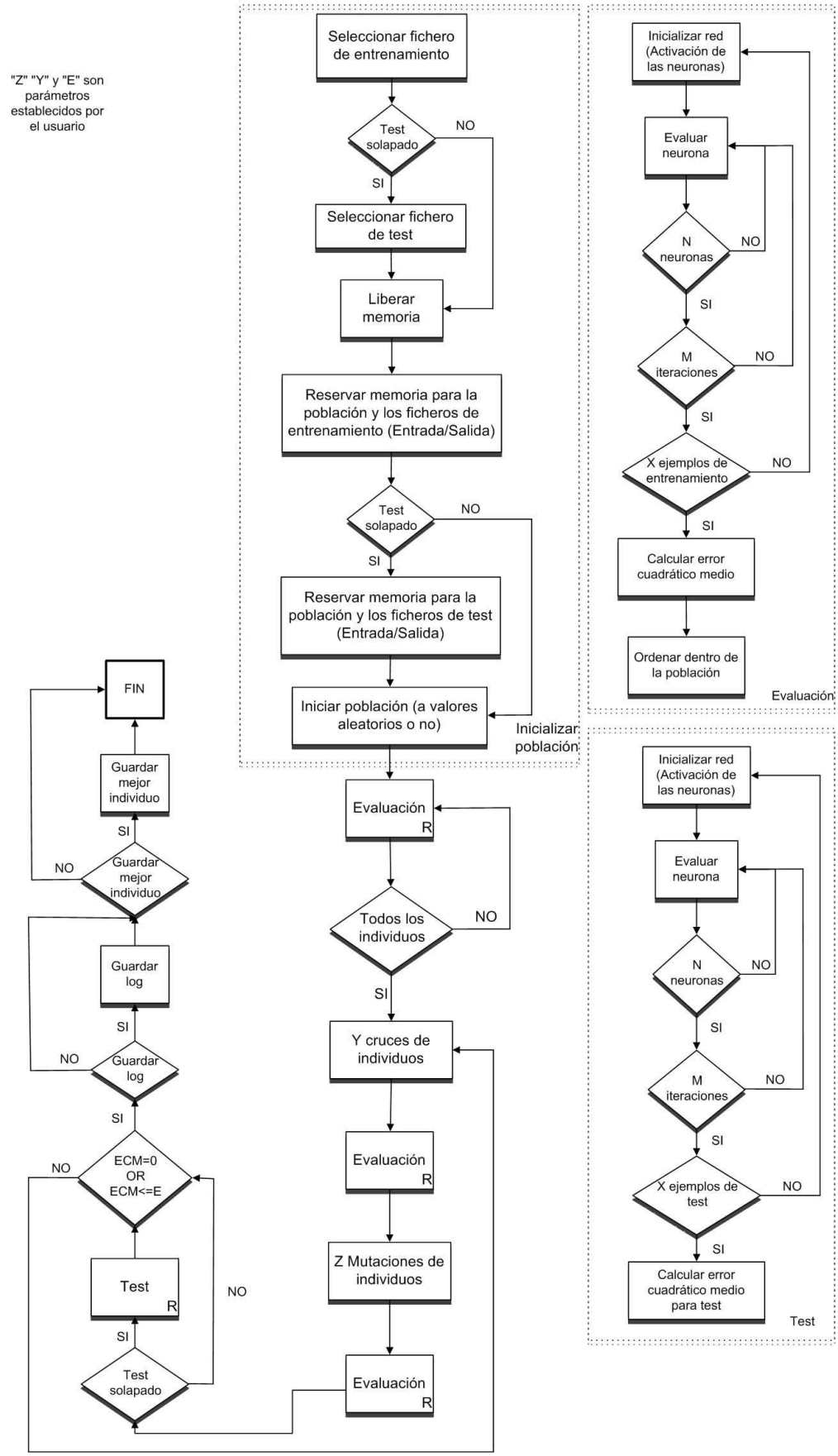


Figura 14: Diagrama principal de flujo de la aplicación

Tras las fases de análisis y diseño, se llevó a cabo la codificación, para lo cual se emplearon los lenguajes de programación MATLAB, C y Bourne Shell. El lenguaje de programación de la aplicación final desarrollada es C, ya que este lenguaje ofrece gran rapidez de ejecución y presenta gran versatilidad. Tras la realización de pruebas de funcionamiento del sistema se desarrollaron todas las simulaciones correspondientes a la parte experimental.

3.2. Planificación

Como ya se indicó, para el desarrollo de la aplicación de simulación se ha seguido un ciclo de vida en cascada realimentado. Se decidió emplear este modelo por la claridad y adecuada especificación de los requisitos y objetivos.

Se muestran dos figuras: en la 15 se presenta la planificación inicial, y en la 16 la planificación final. Cada una de ellas lleva asociada una tabla con su respectivo coste. La duración en días se ha realizado excluyendo los fines de semana, y para el coste total del proyecto se ha supuesto un coste por hora para un Ingeniero en Informática de 20 euros, y 7 horas diarias de trabajo.

Estimación Inicial	
Fecha de Inicio	12/1/2009
Fecha Fin	13/5/2009
Duración Días	88
Coste Total	12320

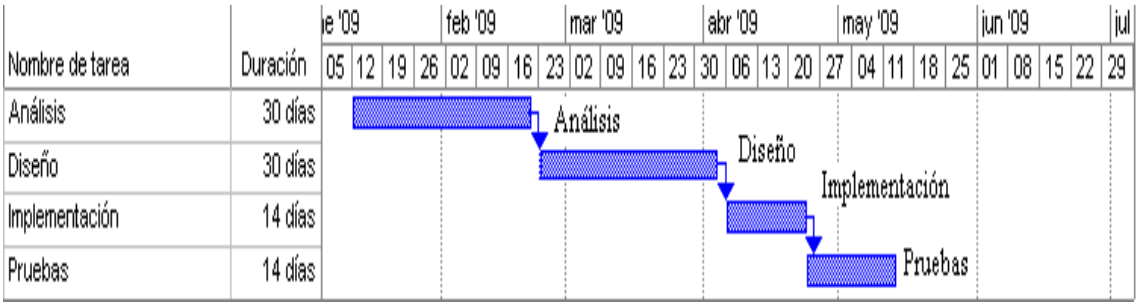


Figura 15: Planificación inicial de desarrollo de la aplicación

Las diferencias existentes entre la planificación estimada inicialmente y la final proceden de una ligera subestimación de los tiempos en las fases de Análisis, Implementación y Pruebas. Este retraso, equivalente a tres semanas, viene provocado por el tiempo empleado en analizar y comprender profundamente el sistema ya desarrollado y la forma de optimizarlo y ampliarlo adecuadamente, así como una prolongación del período de pruebas que asegurase el funcionamiento correcto de todo el sistema.

Estimación Final	
Fecha de Inicio	12/1/2009
Fecha Fin	9/6/2009
Duración Días	107
Coste Total	14980

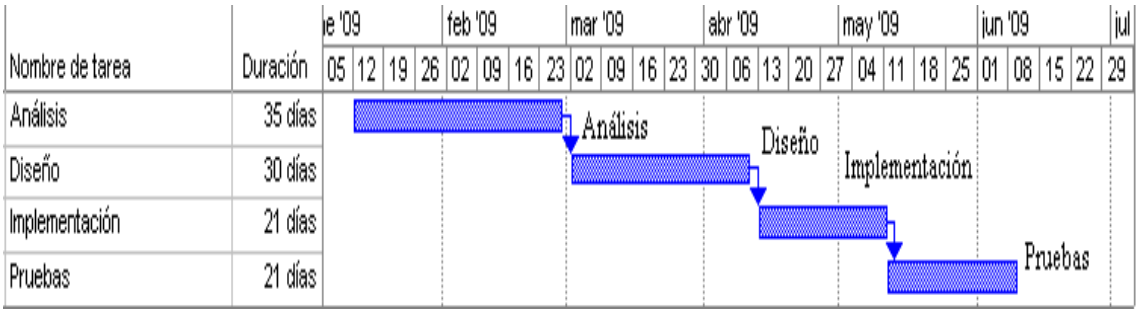


Figura 16: Planificación final de desarrollo de la aplicación

3.3. Plataforma de simulación

Como se detallará en el capítulo siguiente de esta memoria, denominado *Experimentación*, el número de simulaciones a realizar para el análisis de los resultados obtenidos por RNA y RNGA para todos los problemas que aquí se tratarán es muy elevado; más aún cuando se pretenden comparar los resultados de aplicar distintos algoritmos de entrenamiento y diferentes combinaciones de parámetros neurogliales. Hay que tener en cuenta además el alto coste computacional de estas simulaciones, que aumentará con la complejidad del problema y arquitectura de red a tratar. Por lo tanto, resulta necesario contar con un marco de trabajo que permita ejecutar procesos de forma

concurrente sobre máquinas con gran capacidad de procesado. Es por ello que se decidió adaptar la herramienta de simulación disponible, Delphi + C, y adecuarla para su ejecución en el Centro de Supercomputación de Galicia (CESGA), con el fin de automatizar y ejecutar el mayor número posible de pruebas en paralelo.

El CESGA es uno de los principales centros de supercomputación de España. Cuenta con 5 sistemas de computación: FINISTERRAE, SVG, HPC 320, SUPERDOME y GRID, siendo el superordenador FinisTerrae el 6º de España en capacidad de cómputo y el 427 del mundo [T500 08]. Las máquinas sobre las que se han llevado a cabo las pruebas de este trabajo han sido las de FinisTerrae y SVG, cuyas características se enumeran a continuación.

Máquina SVG:

- Arquitectura: granja de PC (de construcción propia), Beowulf Cluster
- Número de procesadores: 96
- Tipo de los procesadores: Intel Pentium III 1GHz, P4 3,2 GHz
- Pico de rendimiento: 528 GFLOPS (nodo CESGA)
- Interconexión: Myrinet y Gigabit Ethernet
- Memoria: 1GB - 2GB por cada nodo
- Disco: 160 GB por cada nodo (sobre 12TB en global)
- Sistema operativo: Linux
- Año de instalación: 2000 (primera etapa).

Máquina FINIS TERRAE:

- Arquitectura: Cluster SMP NUMA
- Número de procesadores: 2.528
- Tipo de procesadores: Intel IA 64 Itanium 2 Montvale Dual Core 1.600MHz(6.4 Gflops)
- Pico de rendimiento: 15.360 GFLOPS
- Interconexión: Infiniband 4x DDR 20 Gbps
- Memoria: 19.670 GB
- Disco: 390.000 GB
- Sistemas operativos: Unix, Linux, Windows

- Compiladores, librerías y herramientas de desarrollo instaladas: Intel C/C y Fortran, Intel MKL, Vtune, HP-MPI y HP UPC
- Año de instalación: 2007

El CESGA cuenta con un sistema de colas (Sun Grid Engine) para la ejecución de *scripts* en alguno de los nodos de la máquina seleccionada. Este sistema de colas permite encolar un número de procesos muy elevado (el número máximo de trabajos a encolar es 800 y 1000 en SVG y FinisTerrae, respectivamente), que irán entrando en ejecución en alguno de los nodos de la máquina seleccionada en base a un sistema de prioridades, que tiene en cuenta: el tiempo de espera en la cola, el tiempo de ejecución ya empleado por un mismo usuario, y los requerimientos de memoria y tiempo del mismo (cuanto más precisas y ajustadas sean las estimaciones antes entrará en ejecución).

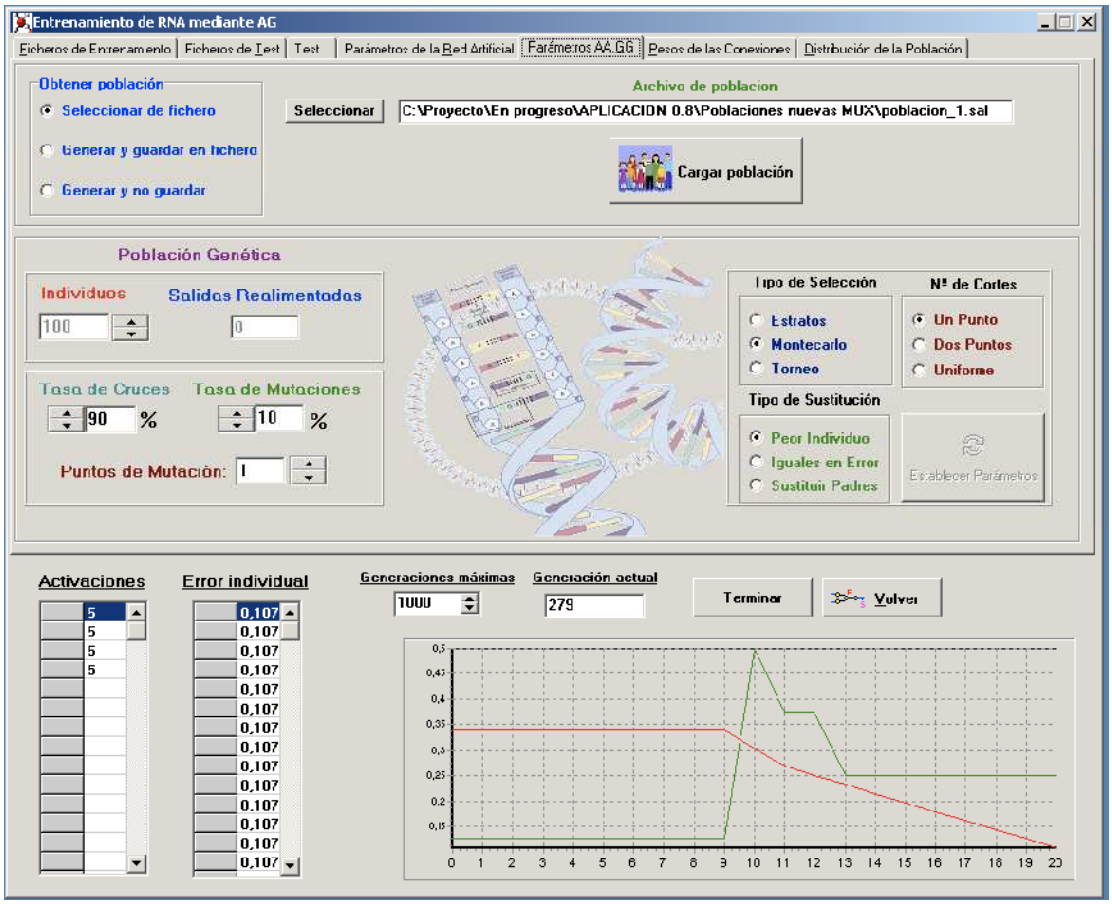


Figura 17: Interfaz de la herramienta de simulación en DELPHI

Dado que las máquinas del CESGA empleadas cuentan con sistema operativo Linux y, además, sólo se pueden encolar en ellas archivos binarios, se ha migrado la interfaz gráfica en Delphi (ver figura 17) a través de la cual se introducían y validaban los datos de entrada (conjuntos de entrenamiento y test, parámetros del AG, rutas ficheros de log y mejor individuo, etc.) a una aplicación con paso de parámetros por fichero, implementada íntegramente en lenguaje C. De esta forma modificando este fichero de parámetros con los valores deseados para cada simulación, podrán lanzarse en paralelo distintos procesos. Con esta migración se posibilita, por un lado, la ejecución de la herramienta en las supercomputadoras del CESGA y, por otro, el encolamiento de múltiples procesos.

3.4.Scripts de ejecución

Para la realización masiva de simulaciones se ideó una estructura de directorios, archivos de parámetros y *scripts* de encolamiento, que facilita realizar el mayor número de pruebas posibles en paralelo y el posterior análisis de resultados. Se partió de una primera versión de scripts que posteriormente se optimizó tal y como se detallará en este apartado.

Se ha creado un directorio genérico para albergar el código fuente, compilado y ejecutable de la herramienta de simulación y además para cada problema se ha ideado una estructura común de subdirectorios, todo lo cual permite reutilizar los *scripts* de encolamiento y lanzar las simulaciones para los distintos problemas de forma paralela. La estructura de directorios utilizada es la siguiente:

- **entrenar_linux**: directorio común que contiene los archivos fuentes, compilados y ejecutables del código de la aplicación de simulación, los cuales serán invocados desde los *scripts* de encolamiento de los diferentes problemas a tratar.

Para cada problema se crea un directorio de trabajo con los siguientes subdirectorios.

- **ejecutables**: contiene los diferentes scripts de encolamiento de procesos.
- **patrones**: este directorio se organiza a su vez en 10 subdirectorios correspondientes a cada uno de los conjuntos de entrenamiento/test resultantes de la división de los datos según la aplicación de la técnica *cross-validation*

5x2cv ya descrita en el capítulo de *Fundamentos Teóricos*. Dentro de cada subdirectorio se almacenan los ficheros de entrenamiento/test asociados.

- **fParametros**: en este directorio se almacenan los ficheros de parámetros que posibilitan parametrizar los scripts de ejecución para poder lanzar procesos en paralelo.
- **poblaciones**: aquí se ubican las diferentes poblaciones genéticas a utilizar, en el caso de las simulaciones que se han realizado en este trabajo se han empleado en cada problema 10 poblaciones de 150 individuos, a excepción del problema de clasificación de flores de Iris que se han empleado 10 poblaciones de 100 individuos.
- **resultados**: con la misma organización en 10 subdirectorios que el directorio patrones, almacenará dentro del subdirectorio correspondiente al conjunto de patrones tratado, los ficheros de resultados (*log*) con los valores de número de generación, error de test, error de entrenamiento y tiempo en alcanzarse, así como los mejores individuos resultantes de las simulaciones y las gráficas de comparación de resultados.
- **ScriptsPlots**: contiene los scripts que generan, para cada simulación, las gráficas, en formato PNG (ver figura 18), de comparación de los resultados de error de entrenamiento y test de RNGA frente a RNA según van transcurriendo las generaciones.

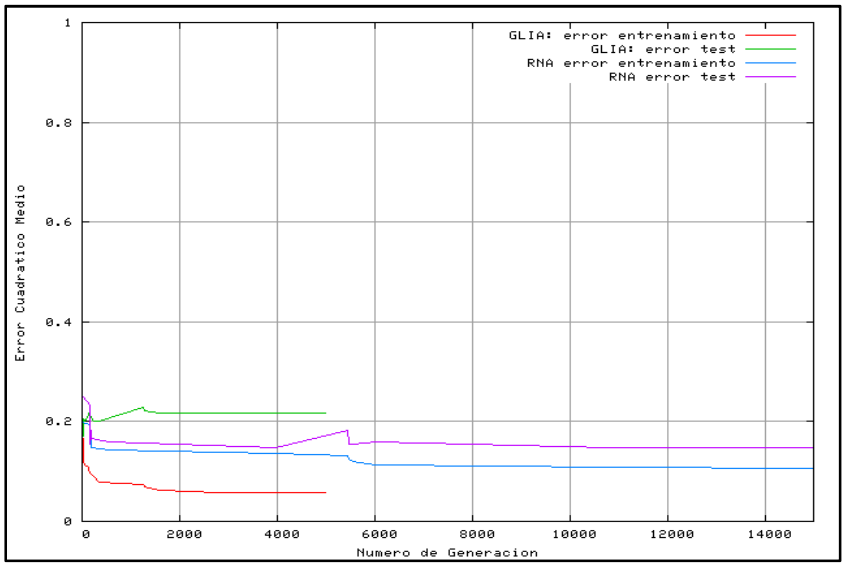


Figura 18: Ejemplo de gráfico de evolución del error generado automáticamente por el script

Primera Versión

A continuación se muestra un ejemplo de la versión inicial de un script de encolamiento (ver figura 19), que posteriormente fue mejorado, escrito en lenguaje Bourne Shell de Linux, donde se envía a la cola (instrucción *qsub*) 100 procesos, resultantes de la combinación de 10 poblaciones distintas y 10 conjuntos de patrones distintos que se ejecutarán durante 40 horas (*s_rt=40:00:00*) usando como fichero de parámetros el fichero *glia38capa9_4indi150.par* que se muestra en la figura 20 y utilizando el ejecutable *neuroGlial.out* contenido en la carpeta genérica entrenar_linux.

```
#!/bin/bash
conjunto=11
poblacion=11

for ((i=1; i < $conjunto ; i++))
do
      for ((j=1; j < $poblacion ; j++))
      do
            echo                "../../entrenar_linux/neuroGlial.out
            ../fParametros/glia38capa9_4indi150.par
            ../patrones/${i}/ ../resultados/${i}/
            pob${j}glia38capa9_4indi150.txt pob${j}_i150_9_4.pob" >>
ejecutable

            qsub -l num_proc=1,s_rt=40:00:00,s_vmem=180M,h_fsize=100M -
cwd ejecutable

            rm ejecutable
done
done
```

Figura 19: Script de encolamiento para la simulación 10 poblaciones x 10 conjuntos RNGA 8-3

En cada fichero de parámetros que se le pasa a la aplicación adaptada neuroGlial.out, (en este ejemplo *glia38capa9_4indi150.par*), se indican los parámetros que definen la arquitectura de la red artificial a tratar, los parámetros de la glía artificial y los parámetros propios del AG a utilizar. Por tanto, se indica:

- **Iteraciones:** indica el número de veces que el patrón o conjunto de patrones es presentado a la red durante la fase de entrenamiento no supervisado de las RNGA, fase en la cual actúa la glía. No debe confundirse este parámetro con cada fase de *cross-validation* en la que los datos de entrada se dividen en conjuntos de entrenamiento y test y que en la literatura referenciada es denominada de la misma forma.

- **Activaciones:** representa el número de iteraciones que tiene que activarse una neurona para que se vean reforzadas sus conexiones.
- **Función de activación:** función perteneciente a una neurona artificial, que tras operar sobre los valores de entrada del elemento, su valor de activación anterior, etc., determina, dependiendo del tipo de función, si éste se activa o no.
- **Salidas negativas:** este parámetro permite a la red artificial el uso de salidas con valores negativos en los elementos de procesado.

En lo que respecta a los parámetros de los AG se eligieron las opciones que a continuación se indican, siguiendo los trabajos de J. Rabuñal [RABU 98]. No se trata tanto de que estas opciones sean las que proporcionen en todos los casos los mejores resultados, sino que lo que se pretende es que dichos parámetros coincidan en los SC que se van a comparar.

- En cuanto al tamaño de la población, es importante que haya una cantidad de individuos suficiente como para que exista diversidad genética, pero no excesivo, ya que debe mantenerse un equilibrio que impida que se incremente demasiado el tiempo de convergencia del SC, que es lo que ocurre cuando hay muchos individuos. En sistemas sencillos no es necesaria una población muy grande para que exista diversidad, ya que los individuos son de menor tamaño y el espacio de búsqueda es entonces menor.
- Se empleará para la selección de individuos la técnica de "Montecarlo", que basa la probabilidad de selección en la adaptación de los individuos. En cuanto al método de sustitución, se ha elegido la sustitución Darwiniana o del peor individuo, en la que los individuos menos adaptados, es decir, aquellos que consiguen una valoración peor, sean los candidatos a dejar su sitio a la nueva descendencia. Se considera que una generación consiste en la realización de todos los cruces y mutaciones establecidos sobre los individuos de la población. En lo que respecta al cruce de individuos se ha considerado un solo punto de corte. La tasa de cruces se ha establecido en un 90% y la de mutaciones en un 10%.
- La función de activación considerada para todos los problemas fue la hiperbólica tangente en todas las capas. Esto es así excepto en la capa de salida que se

estableció la función umbral, con un valor del umbral 0.5, pues en el caso de los problemas aquí tratados la salida esperada siempre es 0 o 1.

Las opciones indicadas en el archivo de parámetros (.par) relativas a los parámetros de ubicación y nombre de archivos necesarios para cada simulación son:

- La ubicación y nombre de los ficheros de patrones de entrenamiento y test.
- Nombre del fichero de log que contendrá los resultados del error de entrenamiento, validación, generación y tiempo que tarda la red en conseguir tales errores. Este fichero se actualiza cada vez que hay una disminución del error de entrenamiento o un cambio en el error de test.
- Nombre del fichero que contiene el mejor individuo de pesos resultante, siendo éste el formado por el conjunto de pesos de todas las conexiones y parámetros de la arquitectura de la red que dan lugar a un error determinado. Este archivo permite realizar el test de la red artificial a partir de este individuo conseguido. El mejor individuo se guarda cada vez que se escribe una generación en el fichero de log.
- Nombre del fichero de población inicial de individuos de pesos a usar.

Así por ejemplo, el fichero *glia38capa9_4indi150.par* que se muestra en la figura 20, contiene los valores adecuados para simular una RNGA con dos capas ocultas de 9 y 4 neuronas, función de activación hiperbólica tangente (T), una población de 150 individuos y parámetros de iteración y activación astrocítica artificial de 8-3 respectivamente.

```
0
2                   •Numero de capas de la red.
9,4                 •Número de neuronas en cada capa (separadas por comas).
1                   •Limite para inicializar los pesos.
1                   •Pesos aleatorios. TRUE=1, FALSE=0
1.0                 •Limite para inicializar K o U.
0.004               •Limite del ERROR en evaluacion.
0                   •TRUE Con Retropropagacion (TRUE=1 o FALSE=0)
1                   •Conexiones Aleatorias. (TRUE=1 o FALSE=0)
0                   •Valor aleatorio. TRUE=1 Umbrales o K aleatorios
0                   •Pendiente de recta aleatoria. TRUE=1
1                   •Numero de ejemplos a agrupar.
1                   •Salida Booleana. Convierte la salida a 0 o 1. TRUE=1 o FLASE=0
0                   •Modo de entrenamiento (discreto=0 o contínuo=1).
0                   •Fijar arquitectura. 0=funciones de activacion aleatorias;1=funciones aleatorias de
forma externa
100                 •Valor maximo de activacion de las neuronas.
1                   •Permitir salidas de neurona negativas. TRUE=1,FALSE=0
0                   •Ciclos de realimentacion de la salida de las neuronas a la entrada
2                   •Numero de clasificaciones de las salidas
0.5                 •Valor del umbral para la clasificación de las salidas.
1.00                •Intervalo de las salidas clasificadas.
8                   •GLIA.Iteraciones a evaluar cada ejemplo.
3                   •GLIA.Iteraciones que activan la glia.
T                   •Funcion de activacion. Lineal=K,Umbral=U,H.Tangente=T,Sigmoide=E,Aleatoria=X
1                   •Limite Pendiente
150                 •Numero de Individuos (en test solo 1)
1                   •Test solapado. TRUE=1, FALSE=0
1                   •Crear Log. TRUE=1, FALSE=0
0                   •Modo generar poblacion. 0-->cargar desde fichero;1-->Se genera nueva y se guarda;2-
->Se genera nueva y no se guarda
/**********************PARAMETROS DEL ALGORITMO GENETICO**************************/
1                   •Numero de puntos de mutacion
0                   •Numero de puntos de cruce (si es cero se usa el cruce sbx)
80                  •Tasa de cruces
10                  •Tasa de mutaciones
0                   •Tipo de sustitucion/insercion(peor individuo,iguales en error,sustituir padres)
2                   •Tipo de seleccion (estratos,montecarlo,torneo)
2                   •Tamano de ventana (en caso de seleccion Torneo)
5000                •Numero maximo de generaciones
0.005               •Error maximo
/**********************FICHEROS DE PATRONES***************************************/
ionosphere_entradas_normalizado_entrenamiento     •Nombre del fichero de patrones de entrenamiento
con las entradas
ionosphere_salidas_normalizado_entrenamiento              •Nombre del fichero de patrones de
entrenamiento con las salidas
ionosphere_entradas_normalizado_test                      •Nombre del fichero de patrones de test
con las entradas
ionosphere_salidas_normalizado_test                       •Nombre del fichero de patrones de test
con las salidas
/**********************FICHEROS DE SALIDA*****************************************/
mejorInd_                                         •Fichero que contendrá el mejor individuo
LOG_                                              •Fichero que contendrá el LOG
../poblaciones/                                   •Fichero que contiene/contendrá la poblacion a
cargar/generar
500                                                       •Cada cuantas generaciones guardo el mejor
individuo a fichero
0.5                                                       •Valor de mu para SBX (real positivo, a
mayor valor mas se parecen los hijos a los padres)
```

Figura 20: Fichero de parámetros glia38capa9_4indi150.par de la herramienta de simulación adaptada al CESGA

Segunda Versión

Como se ha indicado, sobre estos primeros scripts y código desarrollado se fueron realizando una serie de mejoras y optimizaciones incrementales, que mejoraron sustancialmente tanto la comodidad y facilidad de ejecución de las pruebas como el rendimiento de las mismas. Tanto el script para obtener las gráficas como el de encolamiento se generalizaron para poder emplear un único fichero para lanzar cualquier ejecución. El script de la figura 20 sólo serviría para ejecutar las pruebas

relativas a dos capas ocultas con 9 y 4 neuronas, 150 individuos, 8 iteraciones y 3 activaciones. Se optimizó de modo que se emplease un único script al que se le introducen los parámetros concretos de cada simulación por línea de comandos.

Además, a diferencia de lo utilizado en las primeras pruebas llevadas a cabo en el CESGA, se realizaron correcciones en el modo de compilación [CESG 09]. De entrada, se pasó de emplear el compilador gcc a utilizar el compilador de Intel icc (con la consecuente carga dinámica del módulo correspondiente) por no estar optimizado el anterior compilador para Itanium y no generar un código EPIC óptimo. Gracias a este cambio, y a emplear la directiva de compilación –O3, el tiempo de ejecución se redujo considerablemente, pasando de no acabar algunas simulaciones en 50 horas a finalizar en 8-9 horas. Todas estas correcciones/optimizaciones se pueden observar en las figuras 21 y 22.

Los compiladores de GNU están disponibles pero presentan peor rendimiento que Intel en Itanium; en SVG no hay tanta diferencia en los tiempos. Por ello, se debe emplear icc. Para poder usarlo hay que cargar primero el módulo correspondiente mediante module load icc. Después se realiza un linkado dinámico por defecto, de modo que, una vez en el nodo de cómputo, se debe cargar el módulo icc (module load icc) y compilar. Al ejecutar, se debe también cargar el módulo para que estén las librerías disponibles, o bien compilar en estático. Para cargar el módulo en ejecución se debe añadir el "module load icc" en el script de ejecución. Para compilar en estático se debería añadir (-static -i-static).

En el caso de utilizar el SVG hay que indicar, tanto en el momento de compilar como de encolar los trabajos, la arquitectura (recurso *arch*), es decir, el tipo de procesador en el que se desea ejecutar el trabajo. Valores posibles para este recurso: 32, 64, opteron y bw.

```
#!/bin/bash
# ORDEN DE INTRODUCCION DE PARAMETROS:
# $1 = 00 (activacion-iteracion)
# $2 = 5 (neuronas capas ocultas)
# $3 = 100 (individuos)
# El penultimo parametro es solo la denominacion que servira para el LOG_ y para mejor_ind_ de *.par
conjunto=11
poblacion=11
for ((i=1; i < $conjunto ; i++))
do
        for ((j=1; j < $poblacion ; j++))
        do
                echo "module load icc" >> ejecutable
                echo "../../entrenar_linux/neuroGlial.out ../fParametros/glia$1capa$2indi$3.par ../patrones
/${i}/ ../resultados/${i}/ pob${j}glia$1capa$2indi$3.txt pob${j}_i$3_$2.pob" >> ejecutable
                qsub -l num_proc=1,s_rt=20:00:00,s_vmem=180M,h_fsize=100M,arch=64 -cwd ejecutable
                rm ejecutable
        done
done
~
~
```

Figura 21: Script de encolamiento para simulación genérica en SVG (gliaSVG.sh)

```
#!/bin/bash
# ORDEN DE INTRODUCCION DE PARAMETROS:
# $1 = 00 (activacion-iteracion)
# $2 = 5 (neuronas capas ocultas)
# $3 = 100 (individuos)
# El penultimo parametro es solo la denominacion que servira para el LOG_ y para mejor_ind_ de *.par
conjunto=11
poblacion=11
for ((i=1; i < $conjunto ; i++))
do
        for ((j=1; j < $poblacion ; j++))
        do
                echo "module load icc" >> ejecutable
                echo "../../entrenar_linux/neuroGlial.out ../fParametros/glia$1capa$2indi$3.par ../patrones
/${i}/ ../resultados/${i}/ pob${j}glia$1capa$2indi$3.txt pob${j}_i$3_$2.pob" >> ejecutable
                qsub -l num_proc=1,s_rt=20:00:00,s_vmem=180M,h_fsize=100M -cwd ejecutable
                rm ejecutable
        done
done
~
~
~
```

Figura 22: Script de encolamiento para simulación genérica en FinisTerrae (gliaFT.sh)

Para compilar en FinisTerrae hay que realizar la siguiente secuencia de comandos:

compute /* se accede a un nodo de cómputo para compilar allí. El motivo es que la arquitectura de los nodos de cómputo y el frontal no es la misma */

sh compilar.sh /* En este script estarían los comandos de compilación empleando el compilador icc y creando el ejecutable neuroGlial.out */

exit /* Se debe salir del nodo de cómputo y lanzar los trabajos desde el frontal */

En el caso de emplear el superordenador SVG hay que ejecutar la siguiente secuencia de comandos:

compilar -arch 64 /* Hay que indicar la arquitectura, que luego también se indicará en el qsub*/
sh compilar.sh
exit

Versión Array Jobs

Posteriormente, con la finalidad de que los trabajos permanezcan el menor tiempo posible en cola y reducir la carga de trabajo en el CESGA, se estudió e implementó una nueva forma de encolado de trabajos denominada *ArrayJobs*, que fue empleada en las últimas pruebas. Para problemas que requieran el lanzamiento de muchos trabajos de idénticos requerimientos y fácilmente parametrizables, el equipo del CESGA recomendó usar la mencionada técnica de *ArrayJobs*. Con esta técnica se trata de reducir la carga del sistema para que finalicen antes las tareas. Si se necesitan lanzar 1000 trabajos y se lanzasen 1000 qsub, se tendrían en cola 1000 trabajos, cada cual con una prioridad menor. En cambio, si se lanza un único objeto *ArrayJob* en cuyo interior se encuentran los 1000 trabajos, el sistema lo verá como un único trabajo y, en su interior, los "jobs" se irán ejecutando.

Esta técnica presenta las siguientes ventajas [GRID 09; SUN 09]:

- Se escribe un solo Shell script.
- Si al enviar un *ArrayJob* se percibe que hay un error en el script, basta con matar un solo job con su id, sin necesidad de suprimir múltiples jobs.
- Se ocupa mucho menos el máster (nodo con una configuración distinta al resto en el que se halla la copia maestra del sistema operativo) del Cluster, en términos de memoria. La carga de trabajo del sistema es menor.

La opción por la que se ha optado para implementar el *ArrayJobs* aplicado a este caso ha sido la siguiente:

Hasta este momento, se lanzaba, en el caso de FinisTerrae, el fichero llamado gliaFT.sh mostrado en la figura 22. Al lanzarlo, por ejemplo, con *sh gliaFT.sh 24 9_4 150,* lanzaba 100 qsub automáticamente. Se implementó un script que creaba un archivo de nombre *ejecutable_arrayJob*, que incluía todos los trabajos a encolar, de modo que contuviese algo parecido a lo siguiente:

module load icc
../../entrenar_linux/neuroGlial.out ../fParametros/glia24capa9_4indi150.par
../patrones/1/ ../resultados/prueba_arrayJob/1/ pob1glia24capa9_4indi150.txt
pob1_i150_9_4.pob
module load icc
../../entrenar_linux/neuroGlial.out ../fParametros/glia24capa9_4indi150.par
../patrones/1/ ../resultados/prueba_arrayJob/1/ pob2glia24capa9_4indi150.txt
pob2_i150_9_4.pob
module load icc
../../entrenar_linux/neuroGlial.out ../fParametros/glia24capa9_4indi150.par
../patrones/1/ ../resultados/prueba_arrayJob/1/ pob3glia24capa9_4indi150.txt
pob3_i150_9_4.pob
….

Así hasta las 200 líneas correspondientes a los 100 qsub. Más tarde, se implementó un fichero llamado *arrayJob.sh*, cuyo contenido es:

```
#!/bin/bash
#$ -S /bin/bash
SEEDFILE=./ejecutable_arrayJob
cat $SEEDFILE | head -$SGE_TASK_ID | tail -2 > run-$SGE_TASK_ID.sh
chmod +x run-$SGE_TASK_ID.sh
./run-$SGE_TASK_ID.sh
```

Empleando *qsub -t 2-200:2 -l num_proc=1, s_rt=20:00:00, s_vmem=180M, h_fsize=100M -cwd arrayJob.sh* se lanzan, encapsuladas en el interior de un *ArrayJob*, las 100 ejecuciones. De esta forma se van leyendo de dos en dos las líneas del fichero *ejecutable_arrayJob*, pues *$SGE_TASK_ID*, que es una variable en la que se almacena el número de tarea, iría de 2 a 200 cogiendo todos los elementos del fichero.

3.5. Herramientas de análisis automático de los datos generados

Se han implementado tres herramientas en MATLAB para el análisis automático de los ficheros de log generados en las simulaciones. Estas herramientas vuelcan automáticamente los datos a un fichero EXCEL en el que se pueden calcular todos los valores de interés para la investigación, crear gráficos y analizar contrastes estadísticos como la t de Student o el test de Wilcoxon, para poder saber si las diferencias en el rendimiento de redes sin Glía y con Glía son verdaderamente significativas. Los 3

ficheros de análisis se corresponden con los 3 modos empleados para estudiar los resultados:

MODO 1: se presentarán los mejores valores de test obtenidos por las RNGA y las RNA, es decir, para cada población se recogen los menores errores de test (en el caso de las RNGA, el mínimo error de test de las 4 combinaciones de iteración-activación simuladas para los algoritmos probados) alcanzados con cada uno de los 10 conjuntos de datos analizados. Este error mínimo no se utilizará para elegir la mejor red que finalmente se usaría para el modo ejecución de la red, sino para comprobar de modo preliminar si mediante RNGA se puede llegar a un mejor porcentaje de aciertos con datos no usados para el entrenamiento que con RNA entrenadas sólo con AG. Además, esta comprobación preliminar es significativa pues se hace eliminando la componente aleatoria implícita en un único conjunto de entrenamiento y test (lo cual se consigue gracias a tener los 10 conjuntos diferentes 5x2 usados para *cross-validation*). Con esto pretende mostrarse global e inicialmente cómo actúa la glía artificial, pues si en la mayor parte de los casos el error de test llegase a ser menor con glía artificial, parecería lógico pensar que la inclusión de la misma en las RNA sí tiene influencia en los resultados, produciendo variación en los mismos. En la figura 23 se puede observar un ejemplo de la salida del programa en MATLAB para el Modo 1 con un conjunto de entrenamiento-test y 10 poblaciones.

			1	
POB 1	*Con Glia*	*ECM Test - Act-It*	0,052786	24
		Gener-Tiempo(seg)	7	21
	Sin	*ECM Test*	0,096774	
		Generación-Tiempo	4005	2265
POB 2	*Con Glia*	*ECM Test - Act-It*	0,041056	38
		Generación-Tiempo	7	36
	Sin	*ECM Test*	0,102639	
		Generación-Tiempo	8	5
POB 3	*Con Glia*	*ECM Test - Act-It*	0,055718	38
		Generación-Tiempo	8	41
	Sin	*ECM Test*	0,096774	
		Generación-Tiempo	20	12
POB 4	*Con Glia*	*ECM Test - Act-It*	0,055718	24
		Generación-Tiempo	115	332
	Sin	*ECM Test*	0,099707	
		Generación-Tiempo	4	2,999992371
POB 5	*Con Glia*	*ECM Test - Act-It*	0,052786	38
		Generación-Tiempo	12	61
	Sin	*ECM Test*	0,085044	
		Generación	22271	12594
POB 6	*Con Glia*	*ECM Test - Act-It*	0,043988	26
		Generación-Tiempo	4938	20193
	Sin	*ECM Test*	0,087977	
		Generación	515	292
POB 7	*Con Glia*	*ECM Test - Act-It*	0,043988	38
		Generación-Tiempo	2692	13699
	Sin	*ECM Test*	0,105572	
		Generación	4	4
POB 8	*Con Glia*	*ECM Test - Act-It*	0,070381	26
		Generación-Tiempo	58	237
	Sin	*ECM Test*	0,111437	
		Generación	5	3,999992371
POB 9	*Con Glia*	*ECM Test - Act-It*	0,046921	38
		Generación-Tiempo	393	2007
	Sin	*ECM Test*	0,090909	
		Generación	4	2,000015259
POB 10	*Con Glia*	*ECM Test - Act-It*	0,046921	38
		Generación-Tiempo	270	1379
	Sin	*ECM Test*	0,099707	
		Generación	69	39,99999237
MEDIA/ITERAC				
	Con Glia	*ECM Test*	0,0510263	
		Generación-Tiempo	**850**	3800,6
	Sin	*ECM Test*	0,097654	
		Generación-Tiempo	**2690,5**	**1522,1**

Figura 23: Análisis de Fichero de log Modo 1

Para completar el fichero de análisis, a la figura anterior habría que añadirle otras nueve columnas correspondientes a los otros 9 conjuntos relativos al 5x2cv, así como otras dos columnas que incluirían las medias por población y los porcentajes medios de aciertos en test; en la figura 24, se muestran además las medias por conjunto de patrones calculadas automáticamente.

Dentro de cada conjunto y población, para las redes con glia (RNGA), la celda superior izquierda se corresponde con el mínimo ECM en Test de las cuatro combinaciones iteración-activación, junto con, a su derecha, la combinación con la que se consiguió dicho mínimo. Las celdas inmediatamente inferiores contienen la generación y el tiempo (en segundos) en los que se alcanzó ese valor. Para reflejar el comportamiento de las redes sin glía (RNA) se dispone de una celda que contiene el menor ECM en Test de la red sin glía, y en la fila inferior se encuentran la generación y el tiempo (en segundos) en los que se alcanzaron los mínimos.

Conjuntamente con lo mencionado, el fichero de análisis Modo 1 incluye también las tablas resumen que se pueden observar en la figura 24. Estas tablas resumen ofrecen una comparativa global del rendimiento de la RNGA respecto de la RNA. En ellas se muestran, entre otros datos, por cada población el porcentaje de aciertos en test, la media de generaciones y la media de tiempo por población.

	MEDIA ERROR TEST		% ACIERTOS TEST		DESVIACIÓN TÍPICA		MENOR ERROR TEST	
	RNGA	RNA	RNGA	RNA	RNGA	RNA	RNGA	RNA
POBLACIÓN 1	0,09	0,14	0,91	0,86	0,12	0,11	0,03	0,07
POBLACIÓN 2	0,09	0,15	0,91	0,85	0,12	0,11	0,02	0,08
POBLACIÓN 3	0,09	0,15	0,91	0,85	0,11	0,12	0,03	0,06
POBLACIÓN 4	0,09	0,14	0,91	0,86	0,11	0,11	0,03	0,07
POBLACIÓN 5	0,09	0,14	0,91	0,86	0,11	0,11	0,02	0,06
POBLACIÓN 6	0,09	0,14	0,91	0,86	0,11	0,11	0,02	0,08
POBLACIÓN 7	0,09	0,15	0,91	0,85	0,12	0,11	0,01	0,08
POBLACIÓN 8	0,09	0,15	0,91	0,85	0,12	0,12	0,01	0,08
POBLACIÓN 9	0,09	0,15	0,91	0,85	0,11	0,12	0,02	0,07
POBLACIÓN 1	0,09	0,14	0,91	0,86	0,11	0,11	0,02	0,07

	RNGA	RNA	
MEDIA ERROR TEST	0,09	0,14	
% ACIERTOS TEST	91,00%	85,64%	5,37%
DESVIACIÓN TÍPICA	0,11	0,11	
MENOR ERROR TEST	0,01	0,06	

	GENERACIÓN		TIEMPO (minutos)	
	RNGA	RNA	RNGA	RNA
POBLACIÓN	608,30	2988,4	3074,6	1690,7
POBLACIÓN	1155,80	4214	5768,3	2383,7
POBLACIÓN	532,90	4010,8	2156,8	2271,3
POBLACIÓN	912,20	82,3	4057,3	50,0000008
POBLACIÓN	1015,90	4730,8	4252,2	2673,9
POBLACIÓN	1590,00	2216,6	6387,9	1252,7
POBLACIÓN	1306,70	5547,1	6082,3	3139,9
POBLACIÓN	742,50	6586,9	3487,1	3716,8
POBLACIÓN	698,20	1782,9	2523	1007,5
POBLACIÓN	499,00	393,2	2431,3	225,6
MEDIA GLOE	906,15	3255,30	4022,08	1841,21

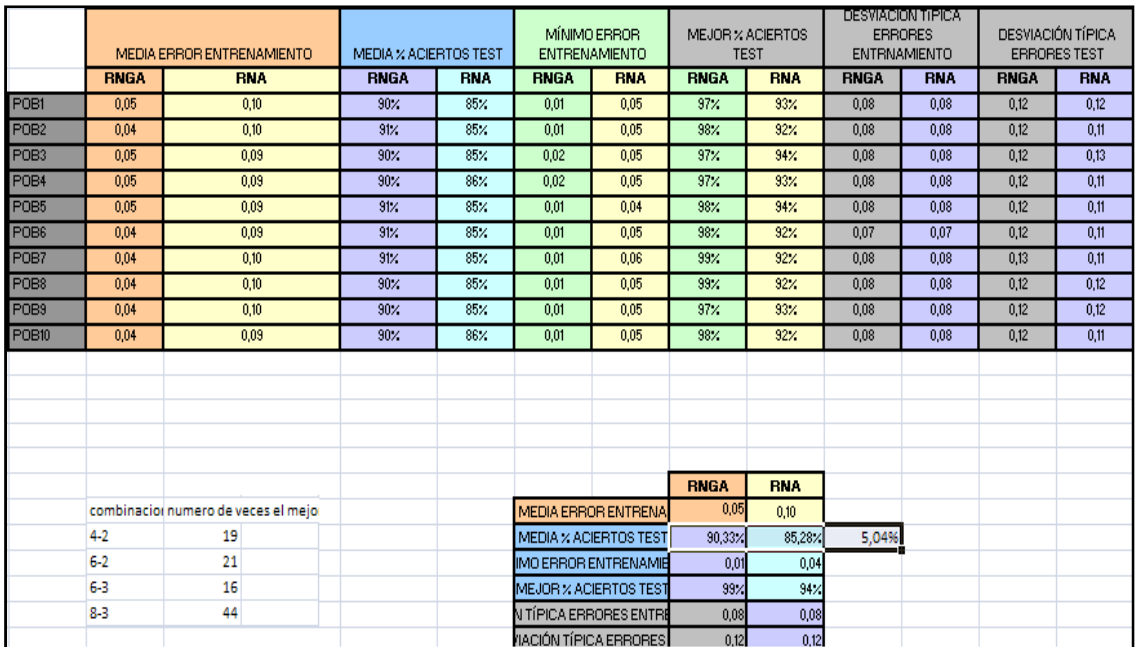
Figura 24: Tablas resumen incluidas en el Modo 1

MODO 2: Debido a que en la mayoría de los problemas tratados los conjuntos de entrenamiento y test de los que se dispone son de pequeño tamaño, los datos van a presentarse de este segundo modo: se fijará un tiempo límite, el mismo para las simulaciones de las RNA que de las RNGA, para comparar el error de entrenamiento y el error de test. Con el mejor individuo de pesos asociado a este tiempo límite y el conjunto de test correspondiente, se llevará a cabo el test. Esto quiere decir que se compararán las medias de los test con los mejores individuos obtenidos, transcurrido el mismo tiempo de entrenamiento, lo que permitirá comprobar si con las RNGA se llega mayoritariamente a un menor error de test que con las RNA. Se ha decidido llevar a cabo la comparación fijando un tiempo límite y no un número dado de generaciones, debido al diferente coste computacional (y su correspondiente tiempo de procesado) de una generación en los dos SC evaluados. En el caso de las RNGA la glía hace que el proceso de evaluación sea más lento, ya que cada patrón se evaluará **X** iteraciones para simular la lentitud de la influencia astrocítica. Si bien el número de generaciones máximas para el entrenamiento en el caso de las RNA ha sido fijado a 30000 frente a las 5000 de las RNGA, los errores a comparar serán considerados tras haber transcurrido el mismo tiempo. En la figura 25 se puede observar un pequeño fragmento de la salida de la herramienta en MATLAB para el Modo 2 correspondiente a 10 poblaciones y un conjunto. Para completar el resultado del análisis habría que incluir los otros 9 conjuntos del cross-validation, junto con las medias por población, así como los porcentajes medios de error en test y las tablas resumen que se muestran en una figura posterior.

		PUNTO DE PARADA A	420	min
			1	
POB 1	Con Glia	ECM Test	0,067449	38
		ECM Train	0,026316	
	Sin	ECM Test	0,102639	
		ECM Train	0,073099	
POB 2	Con Glia	ECM Test	0,055718	36
		ECM Train	0,026316	
	Sin	ECM Test	0,105572	
		ECM Train	0,076023	
POB 3	Con Glia	ECM Test	0,055718	38
		ECM Train	0,02924	
	Sin	ECM Test	0,099707	
		ECM Train	0,064327	
POB 4	Con Glia	ECM Test	0,061584	24
		ECM Train	0,026316	
	Sin	ECM Test	0,102639	
		ECM Train	0,076023	
POB 5	Con Glia	ECM Test	0,058651	24
		ECM Train	0,035088	
	Sin	ECM Test	0,096774	
		ECM Train	0,067251	
POB 6	Con Glia	ECM Test	0,043988	26
		ECM Train	0,02924	
	Sin	ECM Test	0,093842	
		ECM Train	0,073099	
POB 7	Con Glia	ECM Test	0,043988	38
		ECM Train	0,026316	
	Sin	ECM Test	0,108504	
		ECM Train	0,073099	
POB 8	Con Glia	ECM Test	0,076246	36
		ECM Train	0,032164	
	Sin	ECM Test	0,111437	
		ECM Train	0,078947	
POB 9	Con Glia	ECM Test	0,055718	24
		ECM Train	0,026316	
	Sin	ECM Test	0,111437	
		ECM Train	0,076023	
POB 10	Con Glia	ECM Test	0,064516	24
		ECM Train	0,023392	
	Sin	ECM Test	0,105572	
		ECM Train	0,073099	
MEDIA/ITERAC				
	Con Glia	ECM Test	0,0583576	
		ECM Train	0,0280704	
	Sin	ECM Test	0,1038123	
		ECM Train	0,073099	

Figura 25: Análisis de Fichero de log Modo 2

Previa selección por parte del usuario del tiempo de parada, la herramienta capturará el error en test y entrenamiento en ese mismo instante para la red sin glía. De la misma forma, para la RNGA obtendrá el ECM en test y entrenamiento para el tiempo dado, añadiendo la combinación iteración-activación correspondiente a esos valores. En la figura 26 se muestran las tablas resumen que incluyen el Análisis Modo 2.



	MEDIA ERROR ENTRENAMIENTO		MEDIA % ACIERTOS TEST		MÍNIMO ERROR ENTRENAMIENTO		MEJOR % ACIERTOS TEST		DESVIACIÓN TÍPICA ERRORES ENTRNAMIENTO		DESVIACIÓN TÍPICA ERRORES TEST	
	RNGA	RNA	RNGA	RNA	RNGA	RNA	RNGA	RNA	RNGA	RNA	RNGA	RNA
POB1	0,05	0,10	90%	85%	0,01	0,05	97%	93%	0,08	0,08	0,12	0,12
POB2	0,04	0,10	91%	85%	0,01	0,05	98%	92%	0,08	0,08	0,12	0,11
POB3	0,05	0,09	90%	85%	0,02	0,05	97%	94%	0,08	0,08	0,12	0,13
POB4	0,05	0,09	90%	86%	0,02	0,05	97%	93%	0,08	0,08	0,12	0,11
POB5	0,05	0,09	91%	85%	0,01	0,04	98%	94%	0,08	0,08	0,12	0,11
POB6	0,04	0,09	91%	85%	0,01	0,05	98%	92%	0,07	0,07	0,12	0,11
POB7	0,04	0,10	91%	85%	0,01	0,06	99%	92%	0,08	0,08	0,13	0,11
POB8	0,04	0,10	90%	85%	0,01	0,05	99%	92%	0,08	0,08	0,12	0,12
POB9	0,04	0,10	90%	85%	0,01	0,05	97%	93%	0,08	0,08	0,12	0,12
POB10	0,04	0,09	90%	86%	0,01	0,05	98%	92%	0,08	0,08	0,12	0,11

combinacio	numero de veces el mejo
4-2	19
6-2	21
6-3	16
8-3	44

	RNGA	RNA
MEDIA ERROR ENTRENA	0,05	0,10
MEDIA % ACIERTOS TEST	90,33%	85,28%
IMO ERROR ENTRENAMIE	0,01	0,04
MEJOR % ACIERTOS TEST	99%	94%
N TÍPICA ERRORES ENTR	0,08	0,08
IACIÓN TÍPICA ERRORES	0,12	0,12

5,04%

Figura 26: Tablas resumen incluidas en el Modo 2

MODO 3 o Modo Gráficas: Empleado para obtener los valores del error de test y del error de entrenamiento cada cierto tiempo, siendo este tiempo constante y fijado previamente. Permite saber en cada instante qué valores de error poseen la RNGA y la RNA. Se irán guardando en el fichero EXCEL los errores cada X tiempo, lo que facilitará la realización de gráficas de evolución con respecto al tiempo del error, obtenido por las RNGA con respecto a las RNA.

Si un usuario selecciona un tiempo de captura de 15 minutos, cada 15 minutos se escribe en las dos primeras hojas de un fichero Excel el valor correspondiente al error de test y de entrenamiento de la RNGA, y en las dos últimas hojas del archivo se escriben los errores de test y entrenamiento de la RNA.

ERROR ENTRENAMIENTO REDES SIN GLÍA (COMBINACIÓN 00)				
TIEMPO	15	minutos		
CONJ1-POB1	CONJ1-POB2	CONJ1-POB3	CONJ1-POB4	CONJ1-POB5
0,070175	0,070175	0,076023	0,076023	0,073099
0,070175	0,070175	0,076023	0,076023	0,073099
0,070175	0,067251	0,076023	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,067251	0,073099	0,076023	0,073099
0,070175	0,064327	0,073099	0,076023	0,073099

ERROR ENTRENAMIENTO REDES CON GLÍA (COMBINACIONES ITERACIÓN-ACTIVACIÓN)					
TIEMPO	15	minutos			
CONJ1-POB1	CONJ1-POB2	CONJ1-POB3	CONJ1-POB4	CONJ1-POB5	CONJ1-POB6
36	26	24	26	26	36
0,04386	0,032164	0,038012	0,035088	0,02924	0,026316
0,04386	0,032164	0,035088	0,02924	0,02924	0,026316
0,04386	0,032164	0,035088	0,02924	0,02924	0,026316
0,038012	0,02924	0,035088	0,02924	0,02924	0,026316
0,038012	0,02924	0,035088	0,026316	0,02924	0,026316
0,032164	0,02924	0,035088	0,026316	0,02924	0,026316
0,032164	0,026316	0,035088	0,026316	0,02924	0,026316
0,032164	0,026316	0,035088	0,026316	0,02924	0,020468
0,032164	0,026316	0,035088	0,026316	0,02924	0,020468
0,032164	0,026316	0,035088	0,026316	0,026316	0,020468
0,02924	0,026316	0,02924	0,026316	0,026316	0,020468
0,02924	0,023392	0,02924	0,026316	0,026316	0,020468
0,02924	0,023392	0,02924	0,026316	0,026316	0,020468
0,02924	0,023392	0,02924	0,023392	0,026316	0,020468
0,02924	0,023392	0,02924	0,023392	0,026316	0,020468
0,026316	0,023392	0,02924	0,023392	0,026316	0,020468
0,026316	0,023392	0,02924	0,020468	0,026316	0,020468
0,026316	0,023392	0,02924	0,020468	0,026316	0,020468
0,026316	0,023392		0,020468	0,026316	0,020468
0,026316	0,023392		0,020468	0,026316	0,020468
0,026316	0,023392		0,020468	0,026316	0,020468
0,026316	0,023392		0,020468	0,026316	0,020468
0,026316	0,023392		0,020468	0,026316	0,020468
0,026316	0,023392		0,020468	0,026316	0,020468
	0,023392		0,020468	0,026316	0,020468

Figura 27: Análisis de Fichero log Modo 3

En la figura 27 se puede observar un fragmento del resultado del análisis automático de los datos mediante la herramienta implementada con MATLAB, Modo 3. Se pueden observar los valores de error de entrenamiento para redes sin glía (izquierda, conjunto 1, 5 primeras poblaciones) y con glía (derecha, conjunto 1, 6 primeras poblaciones), incluyendo en este último caso también la combinación con la que se consiguieron dichos valores (aquella combinación iteración-activación con la que se obtuvo el mínimo ECM en el test).

4. OPTIMIZACIÓN DE LA GLÍA ARTIFICIAL EN RNGA MEDIANTE AG

Tal como se ha dicho en el capítulo de *Introducción*, la segunda fase de este proyecto ha consistido en diseñar un método automático de optimización de las RNGA. Se ha diseñado e implementado un método de optimización empleando Algoritmos Genéticos para obtener la mejor configuración de parámetros inherentes a la acción de los astrocitos artificiales en las Redes Neurogliales Artificiales.

4.1. Propuesta de Metodología para la Optimización

Como se ha visto en el capítulo 2 de esta memoria, en la propuesta del método de entrenamiento de las RNGA ideado en los trabajos de A. Porto [PORT 04; PORT 05a], se incluyen una serie de parámetros cuyos límites y/o combinaciones han sido establecidos en base a una amplia experimentación y teniendo en cuenta algunos de los fenómenos cerebrales observados y analizados por A. Araque [ARAQ 01; PERE 02]. Aún así, todavía no se han podido cubrir todas las posibles combinaciones debido al elevado número de posibilidades que surgen de combinar los parámetros que actualmente rigen el funcionamiento de la glía artificial, es decir: i) los valores de iteración-activación; ii) los porcentajes de incremento y decremento de los pesos de las conexiones que salen de las neuronas activadas y desactivadas, respectivamente, durante **Y** iteraciones; iii) el algoritmo glial a utilizar, ya que en los trabajos previos de A. Porto y A. Alvarellos [PORT 05; ALVA 07] se han definido 6 diferentes, de los cuales dos se han utilizado en este trabajo. Además, aunque se pudieran probar un número muy elevado de posibilidades, la mejor combinación de estos parámetros es muy probable que sea dependiente del problema, por lo que surge la necesidad de que estos parámetros se incluyan en el proceso adaptativo de entrenamiento supervisado, llevado a cabo mediante AG.

Para ello se propone en este trabajo un nuevo marco evolutivo con las siguientes características:

Los individuos de la población genética pasan a tener dos cromosomas. Por un lado, uno de los cromosomas contendrá en sus genes la información relativa a los pesos

de las conexiones, es decir, los cromosomas existentes hasta ahora. Por otro lado, debido a que la información relativa a los parámetros que rigen el funcionamiento de la glía artificial es de índole diferente a la contenida en los cromosomas actuales, se propone codificar esta información en un segundo tipo de cromosoma [DOPI 04] [DORA 99] [CORD 01a] [CORD 01b]. Este segundo cromosoma estará compuesto de los siguientes cinco genes:

- Dos genes que codifican los valores de activación e iteración. Estos genes solo podrán tener valores enteros y tienen que cumplir la restricción iteración ≥ activación. Hay que tener en cuenta que los valores de iteración deben limitarse al intervalo $3 \leq$ iteración ≤ 9, ya que basándose en experimentaciones previas [PORTO 04] se comprobó que más de 8 iteraciones incrementaban mucho el tiempo de simulación y que con menos de 3 no daba tiempo a que la glía actuase. En relación con estas dos restricciones, los posibles valores de activación se ven constreñidos al intervalo [2,8].
- Dos genes que codifican el porcentaje de incremento y decremento. Con valores enteros, múltiplos de 10, tomados entre 10 y 90. Basándose en la observación fisiológica estos porcentajes se habían establecido en un 25% y 50% respectivamente, pero se opta por no limitar el cambio a estos valores para observar los resultados alcanzados de esta manera. Se ha tenido en cuenta la restricción biológica, comentada en el capítulo de *Fundamentos Teóricos*, que determina porcentaje_incremento < porcentaje_decremento [ARAQ 02]. Debido a esta restricción, el porcentaje de incremento tomará valores entre 10 y 80, y el de decremento entre 20 y 90.
- Un gen que codifica el algoritmo glial que rige el funcionamiento de dicho sistema. Cada uno de los algoritmos disponibles se codificará en este gen. La optimización simultánea de algoritmo y el resto de parámetros es factible ya que todos los algoritmos propuestos hasta ahora para las RNGA [ALVA 07] utilizan los parámetros de porcentaje de incremento-decremento, iteraciones y activaciones, controlando si las activaciones son consecutivas o no.

En este punto surgen dos opciones. Por un lado, codificar estos cinco parámetros globalmente para toda la RNGA o, por otro lado, codificar estos cinco parámetros para cada neurona artificial o elemento de procesado que tenga la RNGA, con lo que la

longitud de este segundo cromosoma sería de 5 x NEP, siendo NEP el número de elementos de procesado de la red. De este modo, se consideraría un astrocito artificial para cada neurona artificial, es decir, podría tratarse de una red de elementos de control (astrocitos) total, o que sólo afectase a alguna de las neuronas.

Una vez vistas las posibles codificaciones del problema, surge también la necesidad de un nuevo flujo del AG. En este trabajo se plantean dos aproximaciones para el flujo de este AG:

- **Opción 1**: el flujo típico de un AG, en donde se tendrá una población inicial de individuos generando aleatoriamente los valores (teniendo en cuenta las restricciones ya indicadas) para sus dos cromosomas. A partir de esta población, se aplican los operadores genéticos de selección, cruce y mutación para cromosomas del mismo tipo, durante un número determinado de generaciones. Este esquema de flujo del AG, se puede ver en trabajos como [SCHA 92; SCHA 90], en donde se resuelve de una forma similar el hecho de tener cromosomas cuyos genes tienen un peso muy diferente en el proceso de búsqueda.
- **Opción 2**: se genera una población inicial de individuos con valores aleatorios (de acuerdo a las restricciones ya indicadas) para los dos tipos de cromosomas. Se aplica un Algoritmo Genético Coevolutivo, dadas las características de los mismos explicadas en el capítulo de *Fundamentos Teóricos*.

En la figura 28 se muestra un esquema resumen de la metodología propuesta:

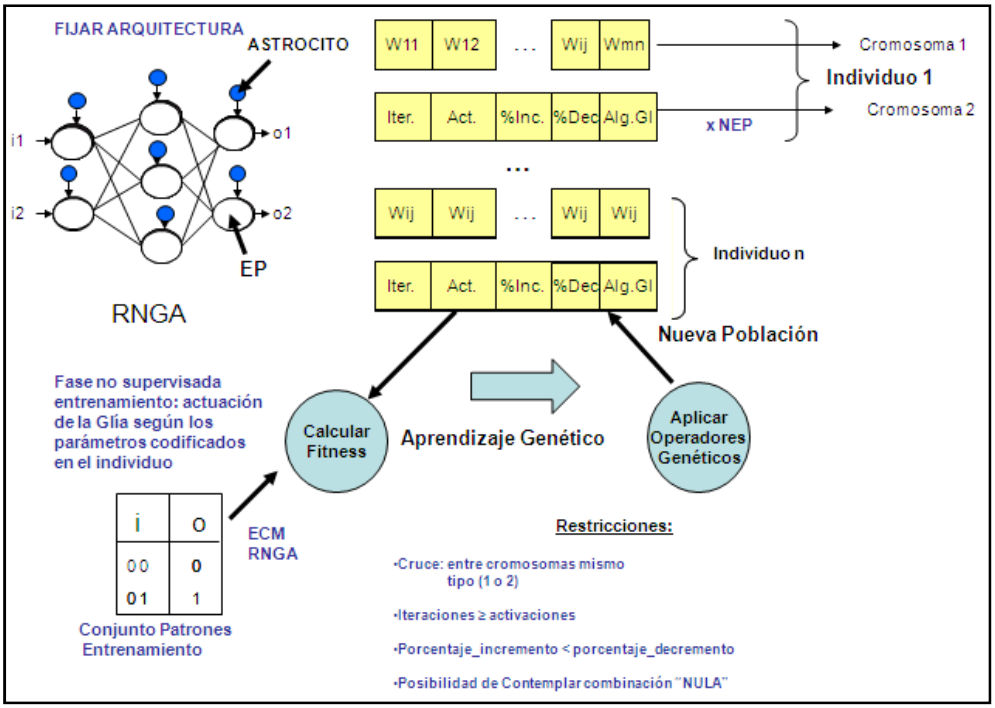


Figura 28: Esquema resumen de las dos opciones de optimización RNGA.

Una vez vistas las posibles codificaciones del problema, existen una serie de aspectos a tener en cuenta a la hora del diseño de los operadores de cruce y mutación para el cromosoma 2:

- **Operador de cruce**: a la hora de cruzar cromosomas del tipo 2, los cromosomas resultantes tienen que cumplir la restricción de *iteración* ≥ *activación*. Para ello, se pueden restringir los cruces a pares de individuos que cumplan que su descendencia cumplirá esta restricción, se puede restringir el operador de cruce de manera que el punto de cruce de este no pueda estar entre los genes que codifican iteración y activación, o, al realizar el cruce, en caso de generar un individuo "erróneo" se podría seleccionar automáticamente otro punto de corte distinto.
- **Operador de mutación**: a la hora de mutar el gen que codifica activación, el límite superior de las mutaciones vendrá acotado por el valor de iteración, de manera que siempre se cumpla que *iteración* ≥ *activación*.

Nótese, que teniendo en cuenta estas restricciones, no importa que se codifiquen los cinco parámetros para cada EP o para toda la red, los operados genéricos descritos son independientes de dicha codificación.

Una vez presentadas las distintas opciones de implementación de la metodología de optimización que suponen una primera aproximación al problema en cuestión, se explican a continuación las decisiones que han llevado a inclinarse por una de ellas. Se ha implementado la opción 2, empleando un AG Coevolutivo Cooperativo, por trabajar con dos poblaciones de individuos de naturaleza perfectamente diferenciable y cuya interacción es fundamental para la adecuada resolución del problema. El cromosoma 2, que define los parámetros de la glía artificial, es común a todos los astrocitos de la red. No se ha optado por un cromosoma por cada neurona esencialmente por dos motivos: primero, por ser la opción elegida una alternativa más sencilla y menos costosa computacionalmente, y segundo, al ser una primera aproximación a la optimización automática de estos parámetros, se optó por seleccionar los valores globales de la red al igual que se hacía previamente de forma manual. A la hora de implementar el operador de cruce se decidió controlar el cumplimiento de las restricciones escogiendo, en el caso de que el cruce de dos padres diese como resultado un individuo incorrecto, otro punto de corte diferente dentro de los mismos individuos. Se hizo de este modo porque siempre habrá, como mínimo, dos puntos de corte válidos en todo cruce: si se cruza por el algoritmo glial o si se corta entre la combinación iteración-activación y los porcentajes de incremento-decremento (ver figura 29). Los detalles de implementación se exponen en el apartado siguiente.

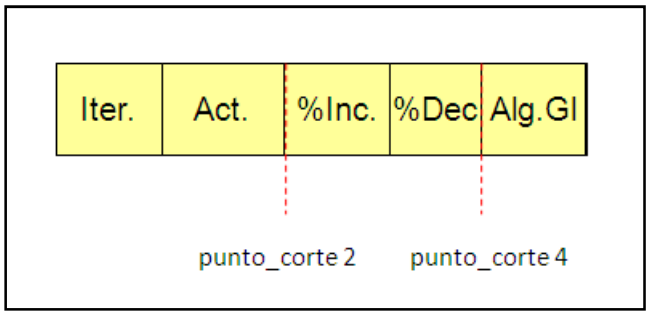


Figura 29: Puntos de corte válidos para todos los individuos

Quedan abiertas las otras posibilidades indicadas en la metodología para futuras implementaciones, así como otras aproximaciones cuando se tengan en cuenta más

parámetros cuya influencia pueda resultar interesante para estos nuevos Sistemas Conexionistas.

4.2. Implementación del Algoritmo Genético Coevolutivo Cooperativo

Para llevar a cabo la implementación del Algoritmo Genético Coevolutivo Cooperativo (en adelante, AGCC) se ha tenido que rediseñar el flujo del programa. Con este nuevo paradigma evolutivo cada vez que se realiza la evaluación de un individuo, de pesos o de parámetros, se combina éste con un número, determinado por el usuario, de individuos de la otra especie.

Inicialmente se evalúa cada individuo de la población de parámetros mediante la combinación con X_i individuos de la población de pesos. A continuación, se evalúa cada individuo de la población de pesos combinándolos con X_i individuos de la población de parámetros. La primera vez que se realiza esta evaluación, la población de pesos no se encuentra ordenada pues todavía no se ha llevado a cabo ninguna evaluación previa. De este modo, la primera evaluación de la población de parámetros se realiza con X_i individuos de la otra especie seleccionados aleatoriamente. En las sucesivas evaluaciones de cada individuo, se combina éste con R individuos aleatorios (correspondiente al valor R_{ij} de la nomenclatura empleada en el apartado 2.2.3.1) y P individuos con el mejor fitness (correspondiente al valor P_{ij} empleado en el mismo apartado del capítulo de *Fundamentos Teóricos*), habida cuenta que $R + P = X_i$. Por cada una de las X_i combinaciones se obtiene el correspondiente error de entrenamiento de la red resultante. El valor de ajuste de ese individuo será uno de los siguientes: el menor de los X_i valores, la media o el mayor. Para calcular el error correspondiente a la red que resulta de combinar un individuo de cada población, se tienen en cuenta los valores de los pesos contenidos en el individuo de pesos y los valores de iteración, activación, porcentaje de incremento, porcentaje de decremento y algoritmo glial, contenidos en el individuo de parámetros de la glía.

En la figura 30 se observa el flujo global del programa tras la implementación del AGCC en los términos mencionados. Así mismo, en la figura 31 se muestra un diagrama detallado de la evaluación, etapa conceptual y computacionalmente más compleja.

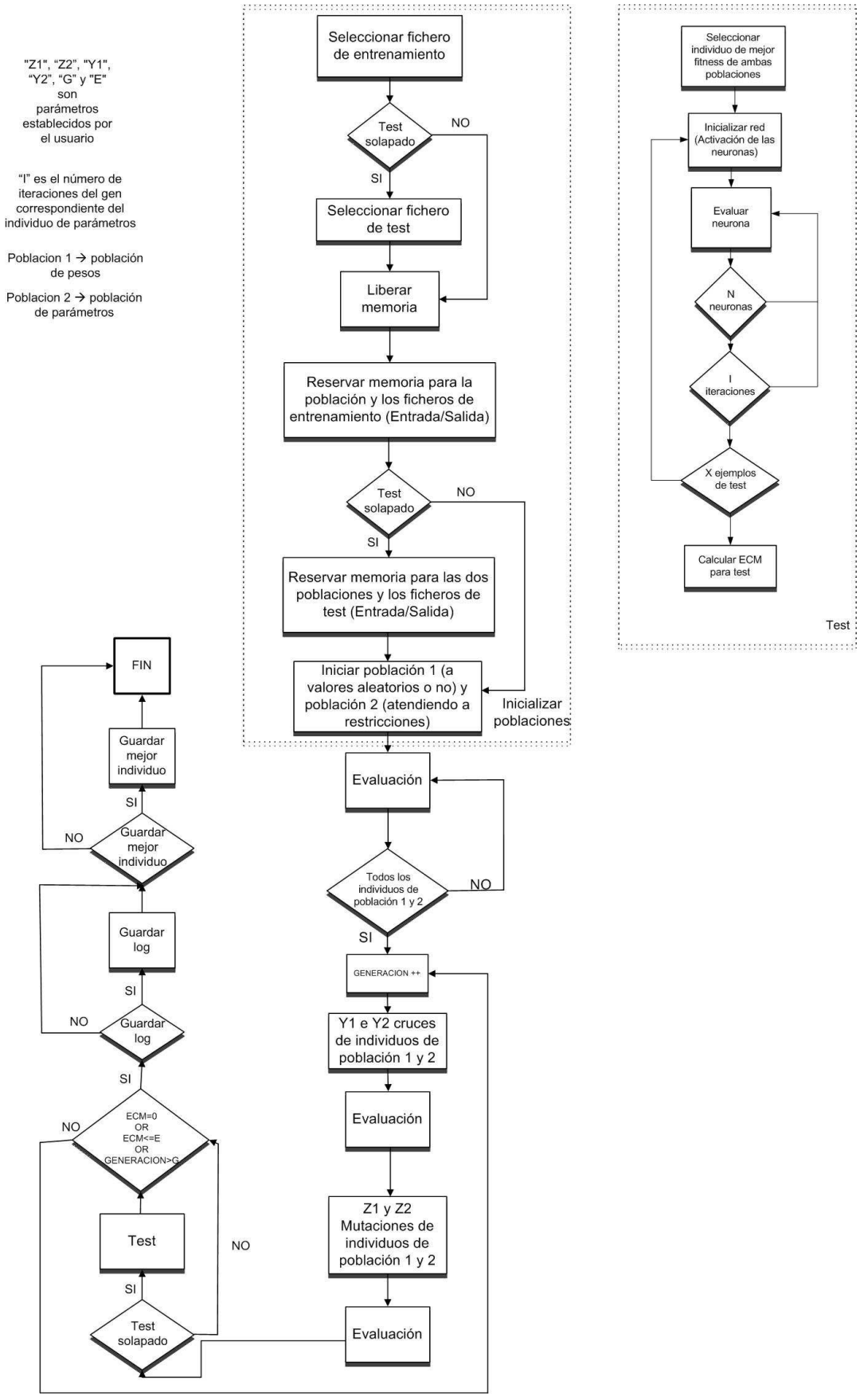


Figura 30: Flujo global de la aplicación empleando el AGCC

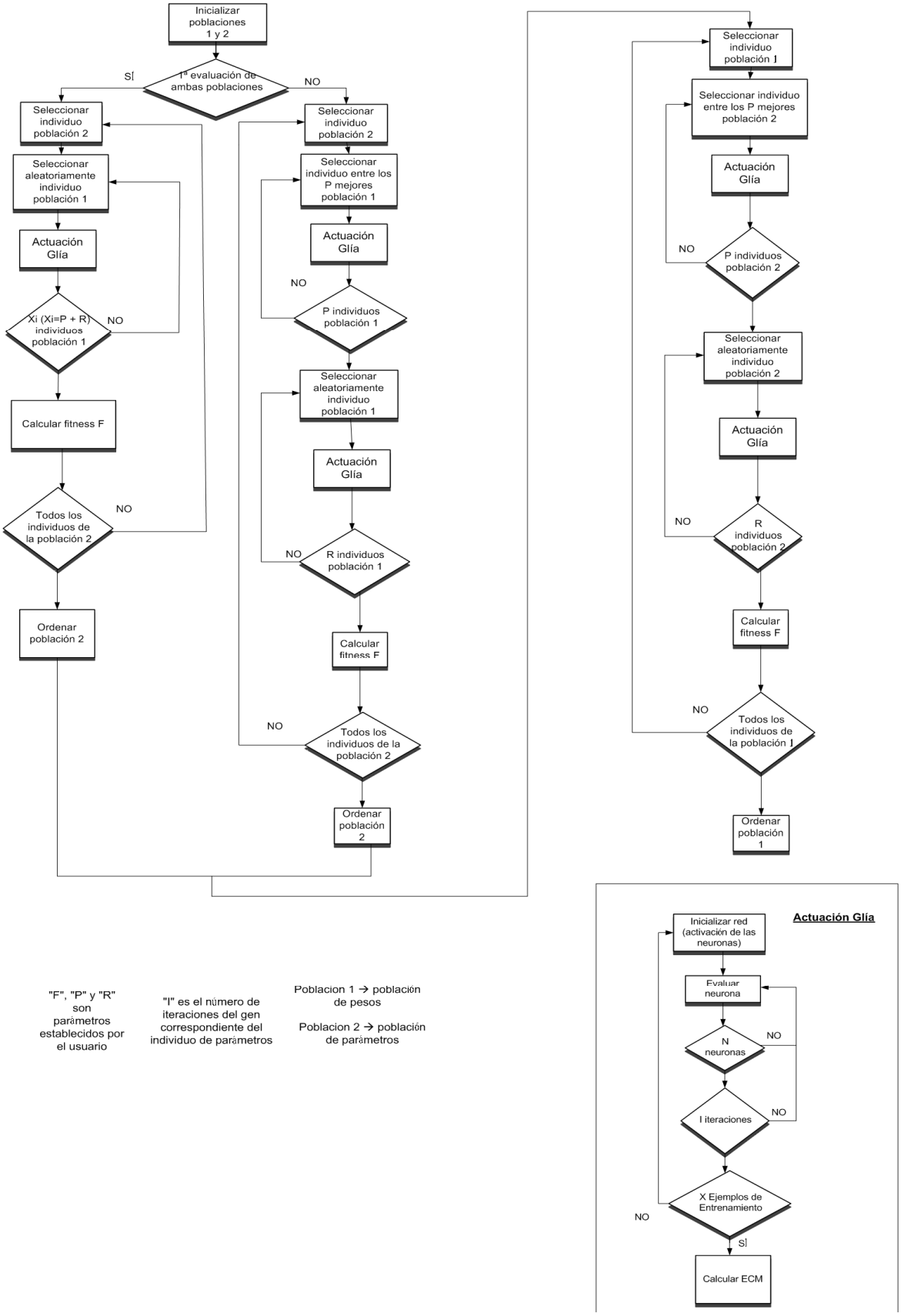


Figura 31: Proceso de evaluación del AGCC

Como se puede observar en las figuras 30 y 31, el número de parámetros introducidos por el usuario aumenta considerablemente. En los ficheros de parámetros se puede seleccionar el número R de individuos aleatorios y de P mejores para la evaluación, el tipo de ajuste a calcular (mínimo, media, máximo), el tamaño de la población de parámetros, el porcentaje de cruces y mutaciones para ambas poblaciones, la carpeta que contendrá las poblaciones de parámetros de la glía, y el modo de generar las nuevas poblaciones. En la figura 32 se pueden observar estas nuevas variables incorporadas en los ficheros de parámetros .par:

```
/**********************PARAMETROS DEL ALGORITMO GENETICO*****************************/
1               <95>Numero de puntos de mutacion
0               <95>Numero de puntos de cruce
80              <95>Tasa de cruces
10              <95>Tasa de mutaciones
0               <95>Tipo de sustitucion/insercion(peor individuo,iguales en error,sustituir padres)
2               <95>Tipo de seleccion (estratos,montecarlo,torneo)
2               <95>Tamano de ventana (en caso de seleccion Torneo)
5000            <95>Numero maximo de generaciones
0.005           <95>Error maximo
/**********************FICHEROS DE PATRONES*****************************************/
ionosphere_entradas_normalizado_entrenamiento   <95>Nombre del fichero de patrones de entrenamiento con las entradas
ionosphere_salidas_normalizado_entrenamiento            <95>Nombre del fichero de patrones de entrenamiento con las salidas
ionosphere_entradas_normalizado_test                    <95>Nombre del fichero de patrones de test con las entradas
ionosphere_salidas_normalizado_test                     <95>Nombre del fichero de patrones de test con las salidas
/**********************FICHEROS DE SALIDA*******************************************/
mejorInd_                                       <95>Fichero que contendrÃ¡ el mejor individuo
LOG_                                            <95>Fichero que contendrÃ¡ el LOG
../poblaciones/                                 <95>Fichero que contiene/contendrÃ¡ la poblacion a cargar/generar
500                                                     <95>Cada cuantas generaciones guardo el mejor individuo a fichero
0.3                             <95>Parametro del Cruce SBX (a mayor valor mas parecidos seran los hijos a los padres)
1                               <95>Parametro del Coevolutivo: numero de individuos aleatorios para la evaluacion
1                               <95>Parametro del Coevolutivo: numero de individuos mejores para la evaluacion
0                               <95>Forma de calcular el fitnes del Coevolutivo: 0 --> minimo, 1 --> Media, 2 --> Maximo
10                              <95>Parametro del Coevolutivo: tamanho de la poblacion de coevolutivo
90                              <95>Parametro del Coevolutivo: porcentaje de cruces
1                               <95>Parametro del Coevolutivo: porcentaje de mutaciones
poblaciones_glia/               <95>carpeta que contendra las poblaciones de parametros de la glia
0                               <95>MODO GENERAR POBLACION DE PARAMETROS (0 --> leer desde fichero, 1--> crear, guardar y leer)
```

Figura 32: Fichero de parámetros con las nuevas variables incluidas

A la hora de guardar los mejores individuos de redes, hay que tener en cuenta, además de los pesos, el individuo de parámetros con el que se consiguieron esos resultados. En el archivo denominado mejor individuo se registran también estos valores, como puede observarse en la figura 33:

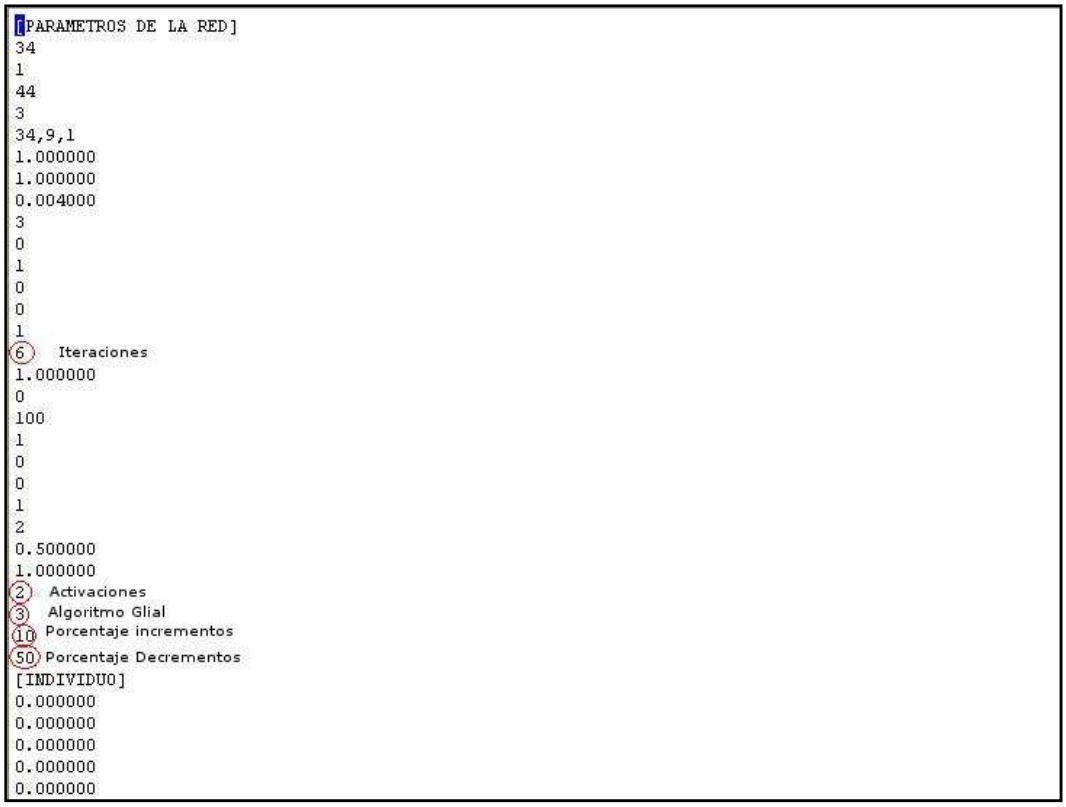


Figura 33: Archivo denominado “mejor individuo”, contenedor de la arquitectura y los parámetros con los que realizar el test. Almacena los parámetros característicos de la glía, además de los pesos de las conexiones y la configuración de la red.

5. EXPERIMENTACIÓN

5.1. Simulaciones para el análisis de la eficiencia de RNGA vs RNA

Con el objetivo de comparar la eficiencia de las RNGA respecto a las RNA entrenadas con AG, se han analizado los resultados obtenidos por ambos tipos de redes para cuatro problemas concretos de clasificación y predicción: Flor de Iris, Enfermedades Coronarias, Cáncer de Mama y Señales de la Ionosfera.

Las pruebas se han realizado sobre redes multicapa, conectadas hacia delante sin conexiones laterales, sin retroalimentación y totalmente conectadas (la salida de una neurona se conecta a la entrada de todas las neuronas de la capa siguiente). Las arquitecturas de las redes a comparar se han elegido considerando arquitecturas con un alto grado de adecuación, comprobado en trabajos previos del grupo RNASA/IMEDIR [RABU 98; RABU 04; RIVE 07b].

Para llevar a cabo la comparación se ha realizado el entrenamiento de las RNA y de las RNGA utilizando los mismos parámetros y partiendo de las mismas poblaciones iniciales, con el objetivo de que el análisis de la variación de los resultados obtenidos por uno y otro tipo de redes pueda explicarse por la influencia de los elementos que representan a los astrocitos artificiales en las redes y no por variaciones de los datos iniciales o parámetros del entrenamiento. La única variación residirá en el número de generaciones máximas que se dejarán transcurrir antes de dar por finalizado un entrenamiento; así, el número de generaciones máximas se ha fijado para todos los problemas en un número considerablemente superior en el entrenamiento de las RNA que en de las RNGA, 30000 frente a 5000; esto se ha hecho así porque en las RNGA el paso de una generación a la siguiente consume más tiempo, por lo que se ha intentado que el tiempo de entrenamiento para los dos tipos de redes sea el mismo, para poder llevar a cabo una comparación más objetiva de los resultados obtenidos.

Destacar que en el caso de las RNGA, en los problemas de Flor de Iris y Enfermedades Coronarias, se han realizado simulaciones utilizando dos algoritmos diferentes de funcionamiento de la glía artificial, algoritmos que se detallarán en el apartado correspondiente a dichos problemas.

En lo tocante a las características de los AG, tal como se ha indicado, las pruebas se llevaron a cabo manteniendo en cada problema las mismas poblaciones de

individuos, así como la misma semilla de generación de números aleatorios que originan la selección de los individuos a cruzar y las mutaciones a realizar. De este modo, se garantiza la validez tanto de la comparación entre las distintas posibilidades de aplicación de modificaciones según los comportamientos gliales simulados, como la comparación entre el funcionamiento de las RNA y las RNGA.

Los datos de todos los problemas tratados en este proyecto han sido tomados del UCI [MERT 02]. Ésta es una colección de bases de datos disponible en Internet, de acceso público, pensadas para ser usadas por la comunidad de investigadores en técnicas de aprendizaje máquina para el análisis empírico de los algoritmos que utilizan. Los atributos de todas estas bases de datos han sido normalizados entre 0 y 1.

Para cada problema los resultados se presentan teniendo en cuenta el porcentaje de aciertos en test y en entrenamiento tras un cierto tiempo de entrenamiento, idéntico para las RNGA y las RNA a comparar (correspondiente al Modo 2 de análisis ya expuesto anteriormente). El tiempo escogido para el Modo 2 proviene de una observación minuciosa de los tiempos del Modo 1, una vez que estos, tanto para RNA como para RNGA, se hayan estabilizado y se pueda afirmar que, dentro del intervalo marcado por el Modo 1, no mejora el rendimiento.

También se muestran gráficos que permiten observar la evolución a lo largo del tiempo de dichos porcentajes de aciertos, así como la evolución de la media de los mismos en los 100 casos simulados usando la técnica de comparación de cross-validation. Como ya se ha comentado, dicha técnica permite validar formalmente los resultados obtenidos. Junto con estos, se presentan también gráficos que permiten observar cuánto tiempo emplea cada tipo de red en alcanzar un porcentaje de aciertos adecuado a cada problema. Estos gráficos se obtienen, a partir del análisis del Modo 3, seleccionando un determinado porcentaje de error a alcanzar (suficientemente alto para que sea significativo, y suficientemente bajo para que la mayor parte de las simulaciones lo alcancen) y tomando como máximo el tiempo empleado en el Modo 2. Para cada simulación (correspondiente con una combinación conjunto-población) se busca el tiempo requerido para alcanzar el porcentaje de error deseado (en test o entrenamiento). De alcanzarse ese porcentaje, se almacena ese tiempo; de no alcanzarse, se le asigna el máximo (el empleado en el Modo 2). Finalmente, se realiza la media aritmética de los 100 valores obtenidos para cada simulación.

5.1.1. Problema de Clasificación de Flor de Iris

Este problema consiste en la clasificación de la especie de la flor en sus tres posibles variantes: Iris Setosa, Iris Versicolor e Iris Virginica. Para ello se cuenta con cuatro parámetros de naturaleza continua en los que se han recogido medidas en milímetros de cuatro características de las flores: longitud y anchura de pétalos y sépalos. Las medidas recogidas corresponden a 150 flores de las tres especies distintas de iris, 50 de cada tipo. Por lo tanto, se cuenta con 150 casos con 4 características cada uno. Las entradas de cada ejemplo son los cuatro parámetros necesarios para la clasificación de las flores (en milímetros) y las salidas son tres y booleanas, representando cada una de las especies del género. Usando tres salidas booleanas y no una múltiple se consiguen identificar las clasificaciones erróneas, ya que sólo una salida puede estar activada a la vez, por lo que si se activan dos o tres, se sabrá que el sistema ha hecho una mala clasificación. Los valores de las cuatro variables que constituirán las entradas se han normalizado al intervalo [0,1].

Se ha partido de una estructura de red óptima, a la cual el grupo RNASA había llegado en trabajos previos: 5 neuronas en la capa oculta, función de activación hiperbólica tangente y función umbral (0.5) en las neuronas de la capa de salida. Usando RNA con estas características, y entrenándolas exclusivamente mediante AG, J. Rabuñal [RABU 04] alcanzó mejores resultados que los obtenidos por la mejor red de A. Martínez [MART 01], usando BP y con una neurona más en la capa oculta. Estos buenos resultados demuestran la capacidad de los AG para simplificar y resolver este problema. Se ha comparado una RNGA con una RNA entrenada exclusivamente con AG.

De los diferentes algoritmos de la glía artificial desarrollados hasta el momento [PORTO 07; ALVA 07; PORTO 08], para estas pruebas se han elegido los dos algoritmos que en trabajos previos con este problema permitieron obtener mejores resultados. Los dos algoritmos gliales elegidos han sido los denominados *Glía Atenuación* y *Glía No Atenuación SLP (sin límite de pesos*). Se indica a continuación su funcionamiento.

5.1.1.1. Glía Atenuación

En este algoritmo la glía contempla tanto activaciones consecutivas de las neuronas como no consecutivas, y los pesos de las conexiones se incrementan y decrementan sin restricciones en el valor a alcanzar. El funcionamiento del algoritmo permite que las conexiones salientes de las neuronas que más se activen tengan un refuerzo mayor, de modo que el efecto de la glía perdure en el tiempo, atenuándose sus efectos sin desaparecer de repente [ALVA 07].

5.1.1.2. Glía No Atenuación SLP (sin límite de pesos)

Este algoritmo se diferencia del anterior básicamente en que el contador de activaciones de las neuronas será 0 cuando haya alcanzado las **Y** activaciones fijadas para que la glía artificial actúe. La glía artificial incrementará los pesos de las conexiones que salen de aquellas neuronas que se activaron un preestablecido número de veces. La función que controla el contador de activaciones incrementará este cada vez que la neurona se activa, si la neurona no se activa se decrementará [ALVA 07]. En este caso los pesos de las conexiones pueden también alcanzar cualquier valor.

5.1.1.3. Resultados y discusión

A continuación se exponen los resultados de las simulaciones realizadas.

Mediante el Modo 2 de análisis se han recogido, para cada población y para cada conjunto de cross-validation, los errores de entrenamiento y test trascurrido un tiempo de 5 minutos desde el inicio del entrenamiento. Este tiempo es el mismo para RNGA que para RNA y se ha elegido tras estudiar detenidamente el Modo 1 y observar que, después de ese tiempo, no variaban los resultados. La tabla 2 muestra para cada población:

- La media de los errores de entrenamiento de los 10 conjuntos de patrones tratados, teniendo en cuenta que en el caso de las RNGA el error de entrenamiento considerado es el menor obtenido entre las 4 combinaciones de iteración-activación analizadas para cada uno de los dos algoritmos gliales utilizados para el entrenamiento.
- Se muestra además en esta misma tabla el porcentaje medio de aciertos en el test. El test se realiza para cada conjunto con el individuo de pesos obtenido en la fase de entrenamiento transcurridos 5 minutos.

- En dicha tabla se muestran también el mínimo de los errores de entrenamiento y el máximo porcentaje de aciertos en el test entre los obtenidos con cada uno de los 10 conjuntos de patrones utilizado.

	MEDIA ERROR ENTRENAMIENTO		MEDIA % ACIERTOS TEST		MÍNIMO ERROR ENTRENAMIENTO		MEJOR % ACIERTOS TEST	
	RNGA	RNA	RNGA	RNA	RNGA	RNA	RNGA	RNA
POB1	0,44	0,45	44,80%	43,47%	0,28	0,33	54,67%	58,67%
POB2	0,40	0,46	47,07%	40,67%	0,19	0,31	72,00%	56,00%
POB3	0,43	0,46	45,33%	42,00%	0,20	0,32	52,00%	57,33%
POB4	0,44	0,47	44,80%	38,93%	0,29	0,32	54,67%	60,00%
POB5	0,41	0,47	46,00%	37,73%	0,23	0,35	54,67%	53,33%
POB6	0,43	0,44	47,47%	42,93%	0,20	0,28	60,00%	57,33%
POB7	0,41	0,48	42,27%	35,87%	0,25	0,39	57,33%	53,33%
POB8	0,42	0,44	43,60%	39,47%	0,31	0,33	58,67%	57,33%
POB9	0,42	0,44	46,53%	43,33%	0,32	0,29	64,00%	58,67%
POB10	0,43	0,48	43,07%	36,93%	0,23	0,33	61,33%	64,00%

Tabla 2: Análisis resultados flor de Iris por población transcurridos 5 minutos del entrenamiento.

A modo de resumen, se exponen las medias globales de los resultados obtenidos en los 900 test realizados para este problema: 400 con cada uno de los dos algoritmos gliales y 100 simulaciones de RNA.

	RNGA	RNA	DIFERENCIAS RNGA-RNA
MEDIA ERROR ENTRENAMIENTO	0,42	0,46	
MEDIA % ACIERTOS TEST	45,09%	40,13%	4,96%
MÍNIMO ERROR ENTRENAMIENTO	0,19	0,28	
MEJOR % ACIERTOS TEST	72,00%	64,00%	8,00%

Tabla 3: Análisis medias resultados flor de Iris por población transcurridos 5 minutos del entrenamiento.

Puede observarse que el error medio de entrenamiento tras 5 minutos es menor en las RNGA que en las RNA. Si usamos como indicador de eficacia la media del error de test obtenida tras 5 minutos de entrenamiento, los resultados para las RNGA son también claramente mejores que para las RNA, siendo el porcentaje medio de aciertos en el test para las RNGA un 4,96% mayor que en las RNA.

Para comprobar si las diferencias entre los resultados de los 100 casos simulados para este problema son estadísticamente significativas, se estudian el test t de Student y test de Wilcoxon. El contraste de normalidad de Kolmogorov-Smirnov Lilliefors resultó

significativo: se rechazó la hipótesis nula por lo que no se cumple el supuesto de normalidad; no obstante, por el Teorema Central del Límite, los resultados del t-test siguen siendo válidos. El p-valor obtenido tras aplicar la prueba t de Student fue de 1,855e-07 (tabla 4), lo que indica que las diferencias son muy significativas. Por otro lado, aplicando el test de Wilcoxon el p-valor obtenido es 1.045e-006 (figura 34), por lo tanto, la probabilidad de que los dos conjuntos sean significativamente diferentes al 95% es de 99,99%. De acuerdo con estos resultados, RNGA obtiene resultados significativamente mejores que RNA para este problema.

Prueba t para medias de dos muestras emparejadas		
	Variable 1	*Variable 2*
Media	0,54906665	0,59866662
Varianza	0,01702493	0,02090773
Observaciones	100	100
Grados de libertad	99	
	-	
Estadístico t	5,44896911	
P(T<=t) una cola	1,855E-07	
Valor crítico de t (una cola)	1,66039116	
P(T<=t) dos colas	3,7104E-07	
Valor crítico de t (dos colas)	1,9842169	

Tabla 4: Análisis Excel Test t de Student

En la anterior tabla, obtenida automáticamente por MS Excel, se muestran la media y la varianza de ambas variables (resultados obtenidos mediante RNGA y RNA), así como el número de observaciones y los grados de libertad. El *Valor crítico de t (una cola)* corresponde a un test unilateral donde se plantea la hipótesis de que la media de los errores con RNGA son mayores o iguales que con RNA. Este *Valor crítico* es de 1,66, por tanto, el intervalo comprendido entre 1,66 y $-\infty$ es la región en la que se define el criterio de aceptación de la hipótesis nula. Debido a que el valor del estadístico t (5,44) se encuentra dentro de la región de rechazo de la hipótesis Ho, se acepta que las diferencias en los resultados entre RNGA y RNA son significativas (la media de los errores con RNGA es menor que con RNA). Los t-test realizados en los siguientes problemas se han llevado a cabo de este mismo modo, por lo que, al tratarse de procesos idénticos, no se volverá a mostrar esta tabla en cada caso.

Para la realización del test de Wilcoxon de los rangos signados se empleó el lenguaje R (ver figura 34). En dicha figura se puede observar tanto la hipótesis alternativa planteada (la mediana de los resultados obtenidos mediante RNGA es menor que la correspondiente a RNA), como la consideración de las muestras como apareadas, obteniendo como resultado un intervalo de confianza y un p-valor para dicho contraste.

```
> rnga_iris = c(0.546667, 0.88, 0.56, 0.546667, 0.546667, 0.533333, 0.52, 0.466667, 0.466667, 0.
> rna_iris = c(0.626667, 0.826667, 0.613333, 0.533333, 0.52, 0.453333, 0.613333, 0.493333, 0.413
> wilcox.test(rnga_iris, rna_iris, alternative="less", paired=TRUE, conf.int=TRUE, conf.level=0.

        Wilcoxon signed rank test with continuity correction

data:  rnga_iris and rna_iris
V = 1115.5, p-value = 1.045e-06
alternative hypothesis: true location shift is less than 0
95 percent confidence interval:
        -Inf -0.03337547
sample estimates:
(pseudo)median
   -0.05330119
```

Figura 34: Análisis R del test de Wilcoxon de los rangos signados

Por otra parte y de cara a comprobar la evolución del error medio en test de los 100 casos con respecto al tiempo, se ha empleado el Modo 3 Gráficas explicado en el capítulo *Desarrollo del Entorno de Simulación*. Esta herramienta almacena, para cada uno de los 100 casos población-conjunto, los valores cada X minutos, tanto de Error de Test como Entrenamiento para RNGA y RNA. Se puede observar, por ejemplo un caso concreto, conjunto 7 - población 2, tomando los datos cada minuto. En dicho caso se comprueba la superioridad en Test y Entrenamiento de RNGA con respecto a RNA desde el primer instante de evaluación.

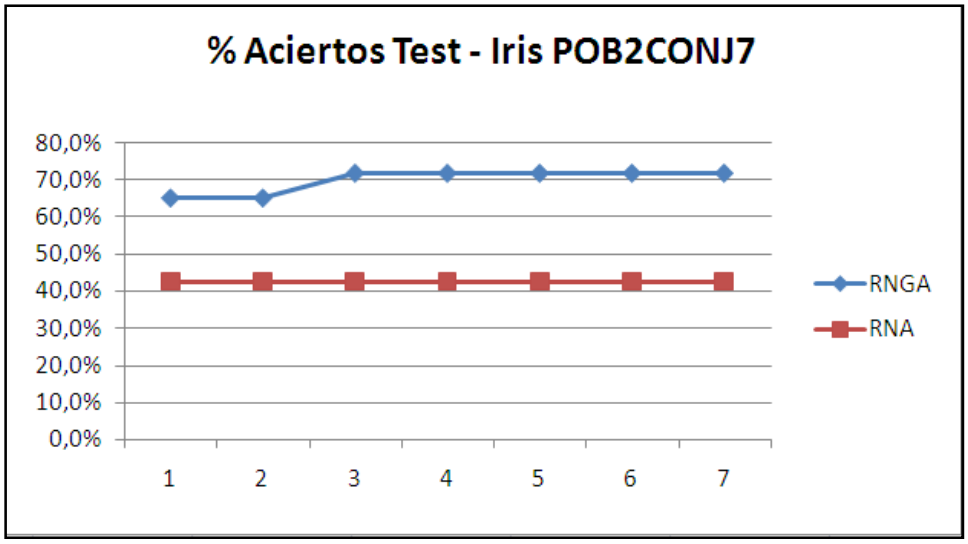


Figura 35: Porcentaje de aciertos en test (Población 2 Conjunto 7 - Flor de Iris)

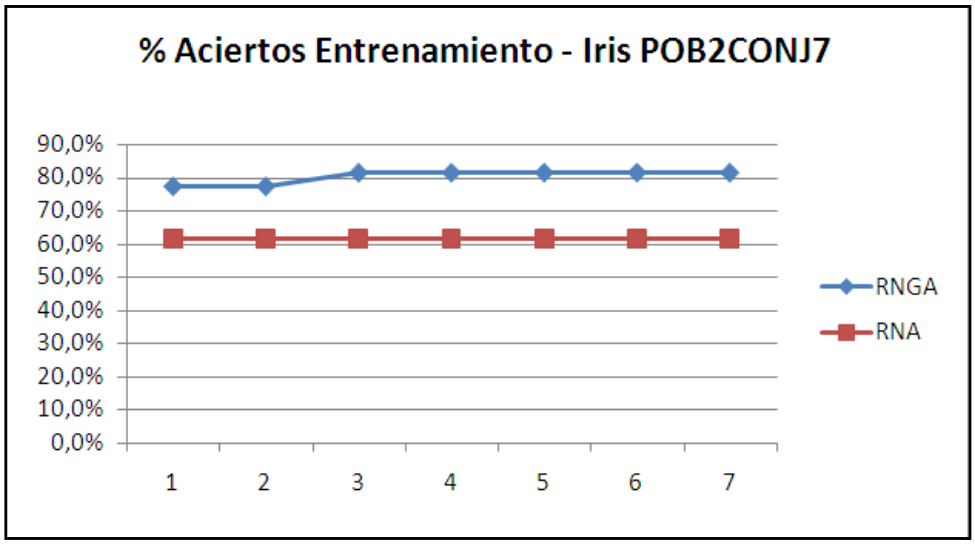


Figura 36: Porcentaje de aciertos en entrenamiento (Población 2 Conjunto 7 - Flor de Iris)

En las siguientes figuras se muestra la evolución de la media del porcentaje de aciertos en test y entrenamiento para el problema Flor de Iris en los 100 casos (10 poblaciones x 10 conjuntos).

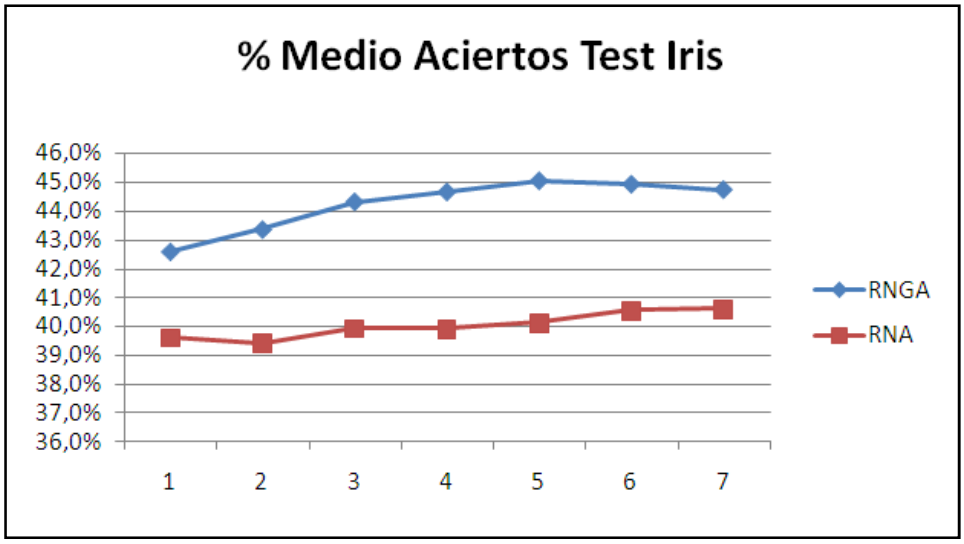


Figura 37: Evolución de Error Medio de Test con respecto al tiempo (Flor de Iris)

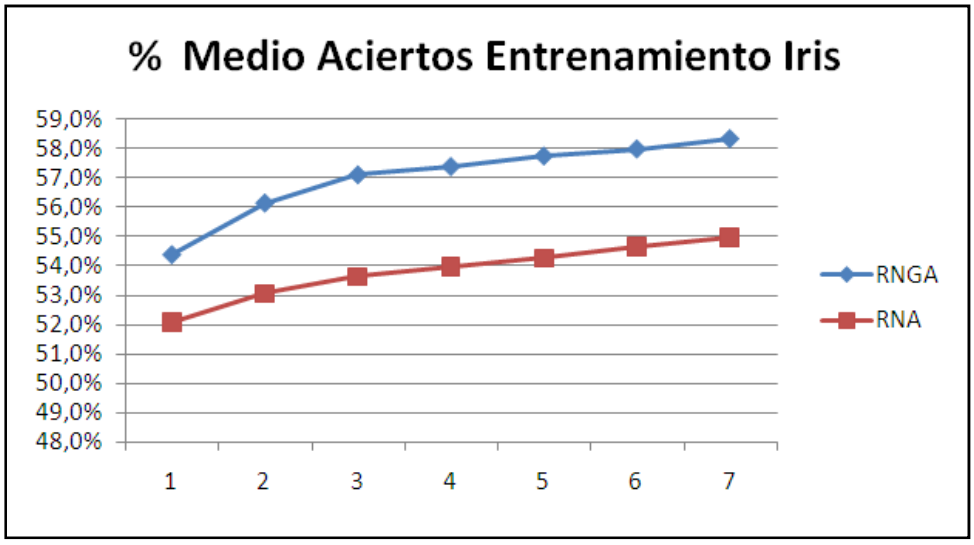


Figura 38: Evolución de Error Medio de Entrenamiento con respecto al tiempo (Flor de Iris)

En este problema, a la vista de las figuras 37 y 38, tanto el proceso de aprendizaje como la capacidad de generalización de las RNGA es más rápido y efectivo que en las RNA, tal y como se puede comprobar al observar la evolución del error de entrenamiento y del porcentaje de aciertos en el test.

Indicar que, aunque se ha realizado la comparación que se buscaba y afortunadamente se han obtenido buenos resultados para RNGA, los errores tan elevados obtenidos, de entrenamiento y test, se creen debidos a la partición que se ha

hecho del conjunto de datos disponibles. El conjunto de datos es muy pequeño, cuenta sólo con 150 patrones. Aplicar la técnica de *cross-validation* ha exigido que este conjunto se particione en dos mitades, es decir, 75 patrones para entrenamiento y 75 para test, lo cual se considera insuficiente para que las redes extraigan relaciones y generalicen de forma adecuada. En trabajos anteriores [ALVA 07; PORT 05a], donde en lugar de dividir a la mitad el conjunto de entrenamiento, éste se particionó de forma que se utilizaron 100 patrones para entrenamiento y 50 para test, los errores de test fueron inferiores y los resultados obtenidos fueron también claramente mejores para las RNGA que para las RNA, alcanzándose un porcentaje de aciertos medio de 72.4% en las RNGA frente al 56% de las RNA. Se considera por tanto también necesario, estudiar los efectos de las RNGA en otro problema que posea una cantidad de datos en los conjuntos de entrenamiento y test en los que estén representadas todas las posibilidades.

Finalmente, mencionar que de los dos Algoritmos Gliales empleados, en el 65% de los casos los mejores resultados se obtuvieron con el Algoritmo Glía Atenuación, mientras que Glía No Atenuación Sin Límite de Pesos fue más eficiente en el 35% restante.

5.1.2. Problema de Predicción de Enfermedades Coronarias

En este problema el objetivo es detectar presencia o no de enfermedad de corazón. Estos datos, tomados igualmente del UCI, corresponden a medidas tomadas a 303 pacientes del centro médico V.A. de Cleveland. En este caso se tienen 13 entradas, y su descripción es la siguiente:

(X1) Age: years

(X2) Sex: 1 = male; 2 = female

(X3) Cp: chest pain type

1: typical angina

2: atypical angina

3: non-anginal pain

4: asymptomatic

(X4) Trestbps: resting blood pressure (in mm Hg on admission to the hospital)

(X5) Chol: serum cholestoral in mg/dl

(X6) Fbs: (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)

(X7) Restecg: resting electrocardiographic results

0: normal

1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)

2: showing probable or definite left ventricular hypertrophy by Estes' criteria

(X8) Thalach: maximum heart rate achieved

(X9) Exang: exercise induced angina (1 = yes; 0 = no)

(X10) Oldpeak = ST depression induced by exercise relative to rest

(X11) Slope: the slope of the peak exercise ST segment

1: upsloping

2: flat

3: downsloping

(X12) Ca: number of major vessels (0-3) colored by flourosopy

(X13) Thal: 3 = normal; 6 = fixed defect; 7 = reversable defect

Toda la experimentación desarrollada con el problema de Enfermedades Coronarias se ha llevado a cabo con una única capa oculta con 4 neuronas, y con los mismos Algoritmos Gliales empleados en el problema anterior (Flor de Iris): *Glía Atenuación* y *Glía No Atenuación SLP (sin límite de pesos*).

5.1.2.1. Resultados y discusión

A continuación se exponen los resultados de las simulaciones realizadas.

Se muestran en la tabla 5 para cada población, los errores de entrenamiento recogidos en todas las pruebas trascurrido un tiempo de 15 minutos desde el inicio del entrenamiento, ya que se observó en el Modo 1 que tras ese período los resultados no variaban:

- El error que se muestra para cada población es la media de los errores de entrenamiento de los 10 conjuntos de patrones tratados, teniendo en cuenta que en el caso de las RNGA el error de entrenamiento considerado es el menor obtenido entre las 4 combinaciones de iteración-activación analizadas para cada uno de los dos algoritmos gliales utilizados para el entrenamiento.

- Se muestra además en esta misma tabla el porcentaje medio de aciertos en el test. Dicho test se realiza para cada conjunto con el individuo de pesos obtenido en la fase de entrenamiento transcurridos 15 minutos del mismo.
- En dicha tabla se muestra también el mínimo de los errores de entrenamiento y máximo porcentaje de aciertos en el test entre los obtenidos con cada uno de los 10 conjuntos de patrones utilizado.

	MEDIA ERROR ENTRENAMIENTO		MEDIA % ACIERTOS TEST		MÍNIMO ERROR ENTRENAMIENTO		MEJOR % ACIERTOS TEST	
	RNGA	RNA	RNGA	RNA	RNGA	RNA	RNGA	RNA
POB1	0,11	0,12	81,61%	78,66%	0,08	0,08	85,91%	82,43%
POB2	0,10	0,12	81,41%	80,94%	0,07	0,09	83,89%	82,55%
POB3	0,11	0,12	81,34%	79,59%	0,08	0,09	83,89%	83,89%
POB4	0,13	0,14	78,25%	78,11%	0,11	0,11	81,21%	83,78%
POB5	0,11	0,11	81,95%	80,54%	0,05	0,08	85,14%	83,89%
POB6	0,08	0,11	81,88%	81,27%	0,05	0,08	86,58%	86,58%
POB7	0,11	0,12	81,14%	81,21%	0,07	0,09	85,14%	84,56%
POB8	0,09	0,12	81,68%	80,87%	0,07	0,07	85,91%	84,56%
POB9	0,11	0,13	81,08%	79,93%	0,07	0,09	83,89%	84,56%
POB10	0,10	0,12	81,75%	79,53%	0,07	0,09	84,56%	83,78%

Tabla 5: Análisis resultados enfermedades coronarias por población transcurridos 15 minutos del entrenamiento.

En forma de resumen, se exponen las medias globales de los resultados obtenidos en los 900 test realizados para este problema.

	RNGA	RNA	DIFERENCIAS RNGA-RNA
MEDIA ERROR ENTRENAMIENTO	0,10	0,12	
MEDIA % ACIERTOS TEST	81,21%	80,07%	1,14%
MÍNIMO ERROR ENTRENAMIENTO	0,05	0,07	
MEJOR % ACIERTOS TEST	86,58%	86,58%	0,00%

Tabla 6: Análisis medias resultados enfermedades coronarias por población transcurridos 15 minutos del entrenamiento.

Puede observarse con este modo de análisis que el error de entrenamiento tras 15 minutos es menor en las RNGA que en las RNA (0,1 frente a 0,12). Si usamos como indicador de eficacia la media del error de test obtenida tras 15 minutos de entrenamiento, los resultados para las RNGA son también claramente mejores que para

las RNA, siendo el porcentaje medio de aciertos en las RNGA un 1,14% mayor al de las RNA.

A pesar de que la diferencia en cuanto a porcentaje de aciertos en test entre RNGA y RNA es solo algo superior al 1%, las diferencias entre los resultados de los 100 casos simulados para este problema son muy significativas, tal y como demuestran los estadísticos t de Student y test de Wilcoxon. El contraste de normalidad de Kolmogorov-Smirnov Lilliefors rechazó nuevamente la hipótesis de normalidad; a pesar de ello, según el Teorema Central del Límite, los resultados del t-test siguen resultando válidos. El p-valor obtenido tras aplicar la prueba t de Student fue de 0,000009, lo que indica que las diferencias son muy significativas. Por otro lado, aplicando el test de Wilcoxon el p-valor obtenido es 0.00039, por lo que la probabilidad de que los dos conjuntos sean significativamente diferentes al 95% es de 99,96%. Por lo tanto, resulta pertinente afirmar que la introducción de la glía reporta claros beneficios también en este problema.

De cara a comprobar la evolución del error en test con respecto al tiempo, para lo cual se ha empleado el Modo 3 Gráficas, se puede observar el caso del conjunto 10 - población 7 (figuras 39 y 40), tomando los datos cada minuto. En dicho caso se comprueba la superioridad en Test y Entrenamiento de RNGA con respecto a RNA desde el primer instante de evaluación.

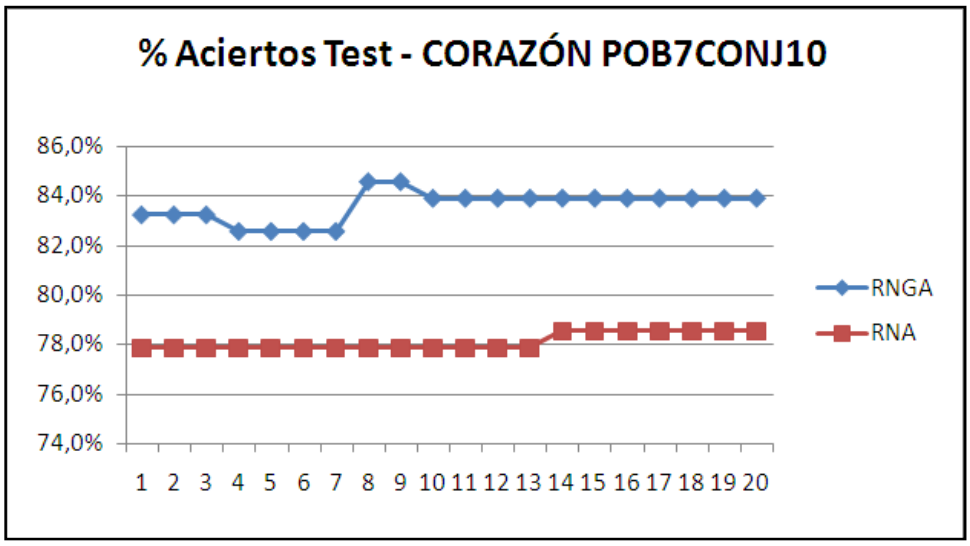


Figura 39: Porcentaje de aciertos en test (Población 7 Conjunto 10 – Enfermedades Coronarias)

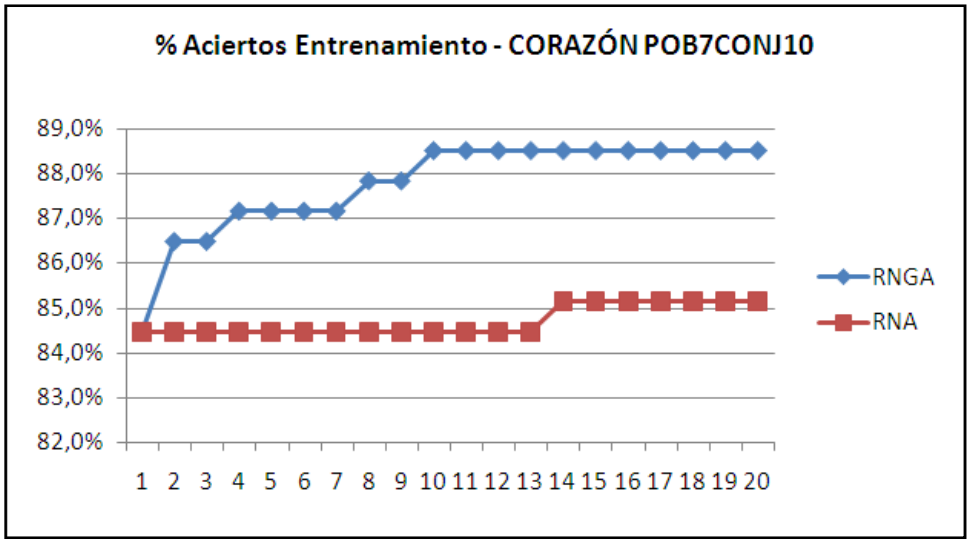


Figura 40: Porcentaje de aciertos en entrenamiento (Población 7 Conjunto 10 – Enfermedades Coronarias)

En las siguientes figuras (41 y 42) se muestra la evolución de la media del porcentaje de aciertos en test y entrenamiento para el problema Enfermedades Coronarias en los 100 casos (10 poblaciones x 10 conjuntos).

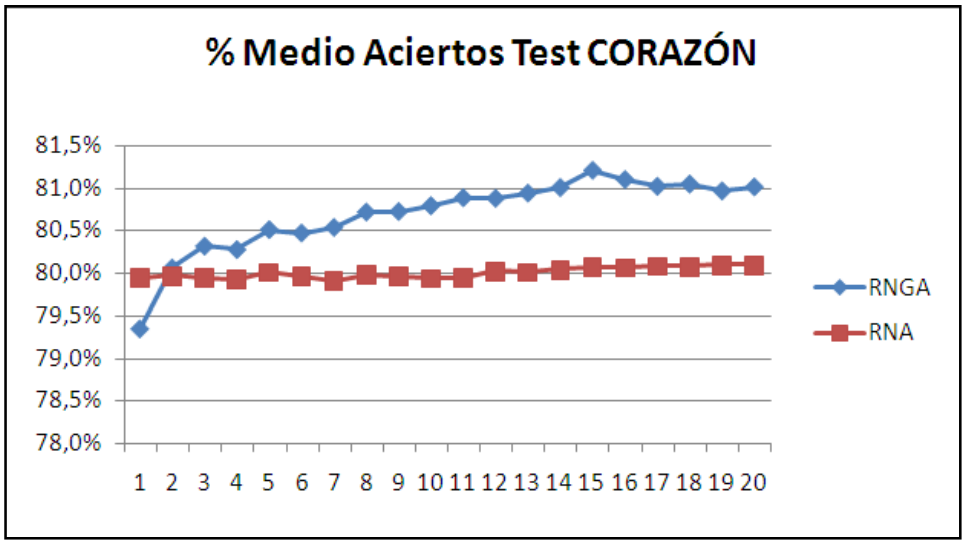


Figura 41: Evolución de Error Medio de Test con respecto al tiempo (Enfermedades Coronarias)

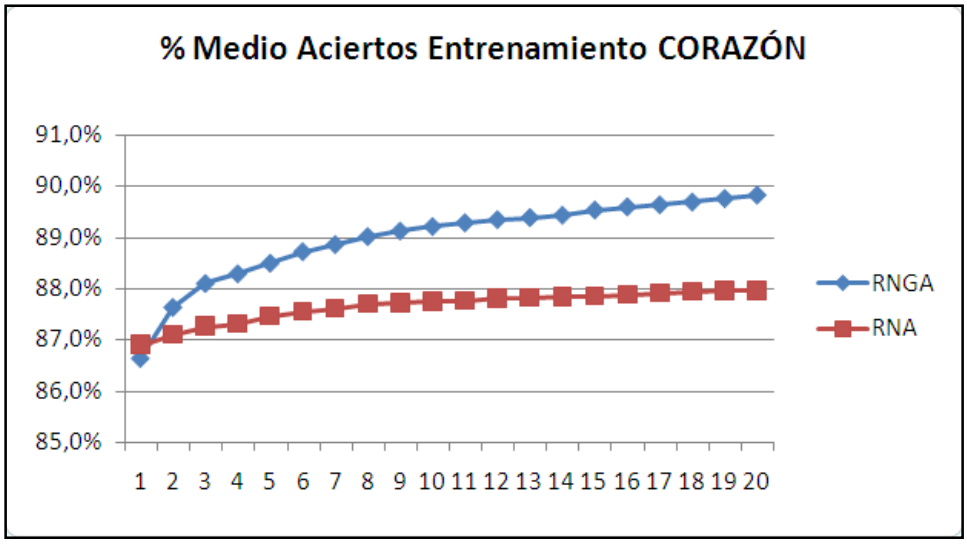


Figura 42: Evolución de Error Medio de Entrenamiento con respecto al tiempo (Enfermedades Coronarias)

Se puede observar cómo en RNGA el proceso de aprendizaje es más profundo, aprende más y en menos tiempo, y la capacidad de generalización es mayor, lo que las convierte en redes más efectivas también en la resolución de este problema.

Con respecto a la eficiencia de los algoritmos gliales empleados en las pruebas, mencionar que de las 100 pruebas realizadas, correspondientes a los 10 conjuntos y 10 poblaciones del método cross-validation, en 49 ocasiones se obtuvieron mejores resultados con el Algoritmo Glía Atenuación y en 51 con el Algoritmo Glía No Atenuación Sin Límite de Pesos. En esta ocasión, a diferencia del problema de clasificación de Flor de Iris, no destaca especialmente uno de los dos algoritmos gliales sobre el otro.

5.1.3. Problema de Predicción de Cáncer de Mama. Estudio de Complejidad.

Este problema consiste en la predicción de Cáncer de Mama. Para ello se cuenta con una base de datos tomada del UCI con 699 datos, donde 458 son benignos (65.5%) y 241 malignos (34.5%). Cada dato se caracteriza por 9 atributos que pueden tomar valores entre 1 y 10, aunque se han normalizado entre 0 y 1. Son los siguientes:

(X1) *Clump thickness*

(X2) *Uniformity of cell size*

(X3) *Uniformity of cell shape*

(X4) *Marginal adhesion*

(X5) *Single epithelial cell size*

(X6) *Bare nuclei*

(X7) *Bland chromatin*

(X8) *Normal nucleoli*

(X9) *Mitoses*

Se ha estudiado este problema con un total de 5 arquitecturas de red diferentes (ver tabla 7).

Número de capas ocultas	1	2	3
Número de neuronas por capa oculta	7	7-3 7-5	9-5-3 12-8-4

Tabla 7: Arquitecturas empleadas en Cáncer

Una de las hipótesis formuladas sobre la influencia de los astrocitos en el cerebro es que actúan mayoritariamente en redes cerebrales más complejas y participan activamente en procesos cognitivos complejos [PERE 07]. Por tanto, se quiere comprobar si esto también ocurre con las RNGA y observar si a mayor complejidad de los problemas a resolver o a mayor complejidad de las redes a utilizar, mayor es la diferencia entre los resultados de RNGA y RNA. Se demuestra a continuación que en problemas más complejos o con arquitecturas con un mayor número de capas ocultas, las RNGA obtienen un rendimiento superior a las RNA.

Se disponía, para la iniciar la comparación de una estructura de red óptima, a la cual el grupo RNASA [RABU 02] había llegado en trabajos previos: siete neuronas en una única capa oculta, función de activación hiperbólica tangente en todas las capas a excepción de la capa de salida donde las neuronas tendrán función de activación umbral (0.5).

Como se pretende llevar a cabo un estudio de la influencia de los astrocitos artificiales al aumentar la complejidad de las redes y no existe ninguna regla que determine a priori la mejor o mejores arquitecturas de una red, el método a seguir ha

sido ensayo-error. Por tanto, se han ido incrementando progresivamente tanto el número de capas como el número de neuronas por capa oculta para observar la evolución de los resultados.

Para cada arquitectura, se han realizado pruebas correspondientes a 10 conjuntos de datos, 10 poblaciones de redes, 4 combinaciones iteración-activación y 1 combinación correspondiente a RNA (iteración-activación: 00), lo que se materializa en 500 test por arquitectura. Considerando esta cantidad multiplicada por las 5 arquitecturas probadas, origina una cifra definitiva de 2500 simulaciones realizadas para este problema.

5.1.3.1. Resultados y discusión

5.1.3.1.1. Una capa oculta

La tabla 8, que se muestra a continuación, resume las medias globales para los 500 test realizados. Se presentan aquí los resultados transcurridos 1h 45' de entrenamiento para una única capa oculta. Este tiempo se ha establecido a partir del Modo 1, observando que transcurrido dicho tiempo, los resultados no mejoraron. Se muestran:

- La media y el mínimo error de entrenamiento, para obtener una visión global del proceso de aprendizaje del sistema.
- La media y el mejor porcentaje de aciertos en test, para comprobar la capacidad de generalización de cada red.

	1 CAPA OCULTA	
	RNGA	RNA
MEDIA ERROR ENTRENAMIENTO	0,07	0,10
MEDIA % ACIERTOS TEST	86,96%	84,98%
MÍNIMO ERROR ENTRENAMIENTO	0,01	0,05
MEJOR % ACIERTOS TEST	97,37%	93,27%

Tabla 8: Análisis resultados Cáncer de Mama transcurridos 1h 45' (1 capa oculta)

El contraste de normalidad de Kolmogorov-Smirnov Lilliefors resultó estadísticamente significativo, al igual que en los casos de dos y tres capas ocultas que se observarán posteriormente: se rechazó la hipótesis nula por lo que no se cumple el supuesto de normalidad. No obstante, por el Teorema Central del Límite, el empleo del

t-test continúa resultando pertinente. Los resultados obtenidos para el caso de una capa oculta tras la aplicación de los estadísticos t de Student y test de Wilcoxon fueron los siguientes. El p-valor obtenido tras aplicar la prueba t de Student fue de 1,33e-11, lo que indica que las diferencias son significativas. Aplicando el test de Wilcoxon el p-valor obtenido fue de 5.642-11, por lo que la probabilidad de que los dos conjuntos sean significativamente diferentes al 95% es del 99,99%, con lo que los resultados de RNGA serían significativamente mejores que con RNA.

Tomando como ejemplo particular, pero representativo y sintomático, los resultados obtenidos con los datos del conjunto 9 de la población 8 y una única capa oculta podemos observar nuevamente la superioridad de RNGA sobre RNA en entrenamiento (mayor capacidad de aprendizaje) y test (mayor capacidad de generalización). Los datos se han obtenido aplicando el Modo 3 con un intervalo de 15 minutos.

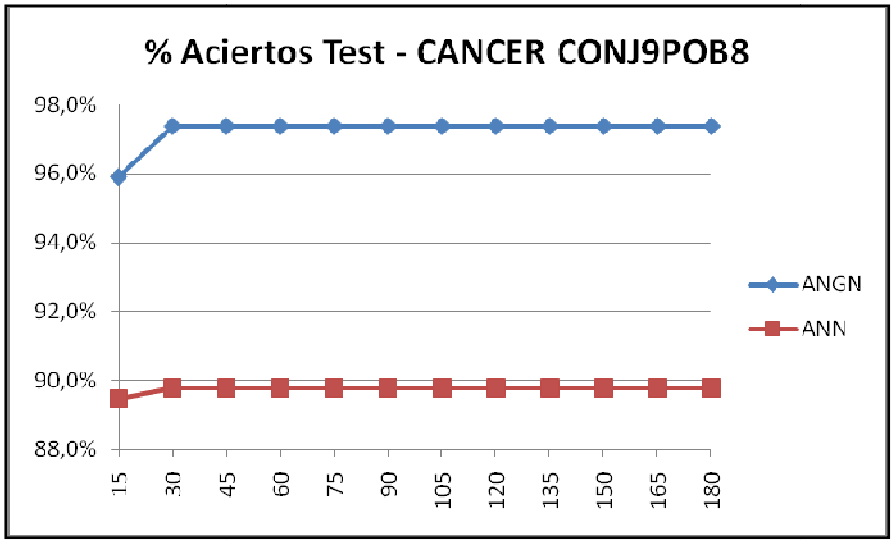


Figura 43: Evolución del Error de Test con respecto al tiempo (Cáncer 1 capa oculta)

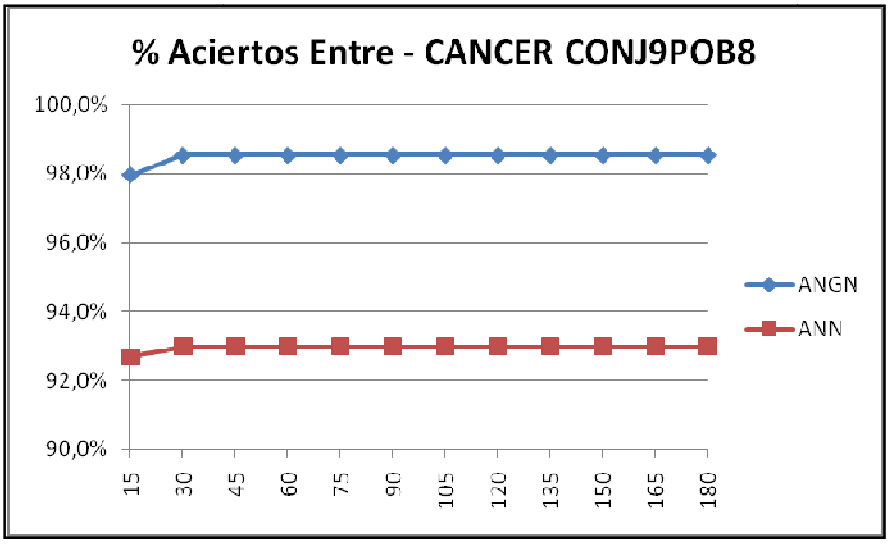


Figura 44: Evolución del Error de Entrenamiento con respecto al tiempo (Cáncer 1 capa oculta)

En las siguientes figuras (42 y 43) se muestra la evolución de la media del porcentaje de aciertos en test y entrenamiento para los 100 casos y una única capa oculta.

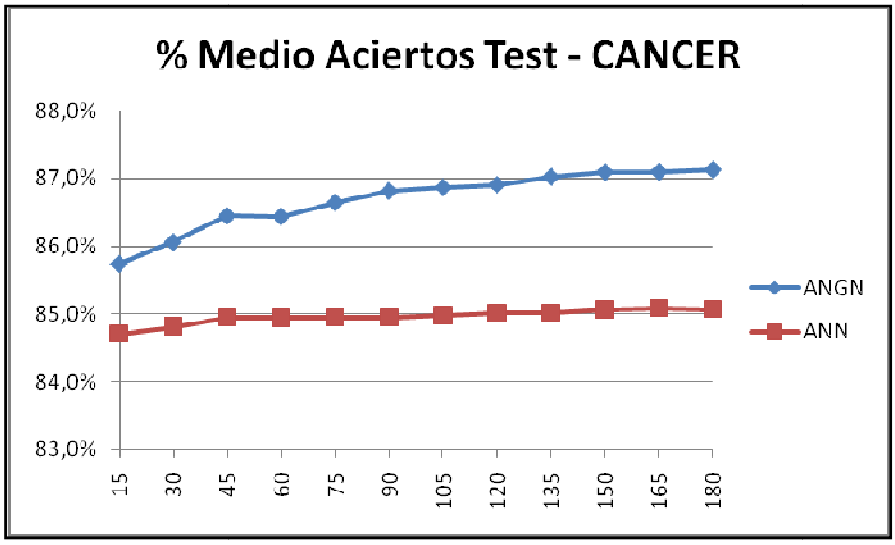


Figura 45: Evolución de Error Medio de Test con respecto al tiempo (Cáncer 1 capa oculta)

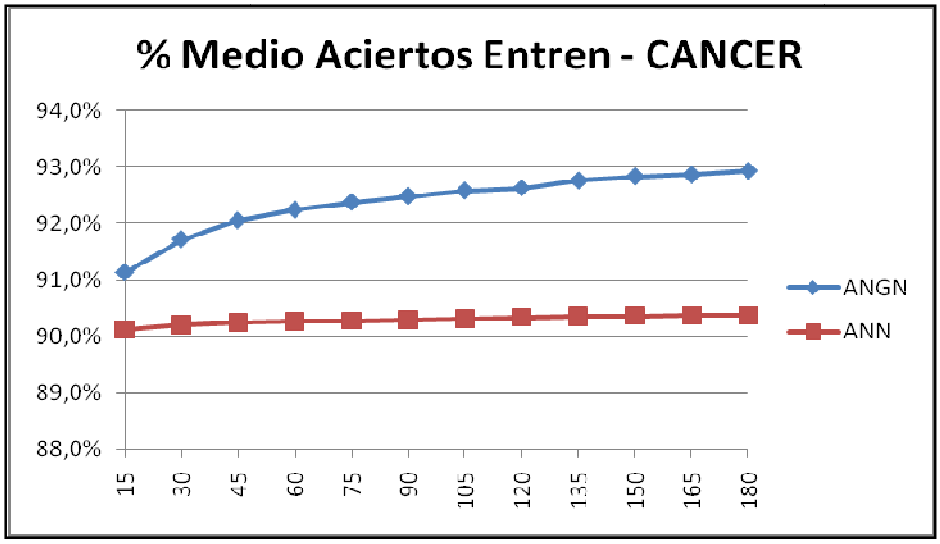


Figura 46: Evolución de Error Medio de Entrenamiento con respecto al tiempo (Cáncer 1 capa oculta)

Para este problema, el proceso de aprendizaje y la capacidad de generalización de las RNGA es también más rápido y efectivo, tal y como se puede comprobar al observar la evolución del porcentaje medio de aciertos.

5.1.3.1.2. Comparación con arquitecturas de dos y tres capas ocultas

El tiempo considerado gracias al Modo 1 para el problema de Cáncer de Mama con arquitecturas de dos y tres capas ocultas ha sido de 7 horas. Esta parte experimental certifica nuevamente un mejor rendimiento de RNGA con respecto a RNA: el gráfico comparativo de la figura 47 muestra que para las diferentes arquitecturas estudiadas la media global de los porcentajes de acierto en test es entre un 2% y un 5% mejor en RNGA que en RNA. También las redes a las que se ha incorporado la glía artificial consiguen en el mismo tiempo un menor error de entrenamiento, es decir, aprenden mejor y lo que es más importante sin sobreentrenar, pues han conseguido una mejor proporción de aciertos en el test en cada una de las poblaciones y globalmente en la media de todas las simulaciones. Generalizan mejor que las redes entrenadas durante el mismo tiempo sólo con AG.

Por tanto, las figuras 47 y 48 muestran claramente el aumento de la influencia de los astrocitos artificiales con el incremento de la complejidad de la arquitectura tanto en entrenamiento como en test.

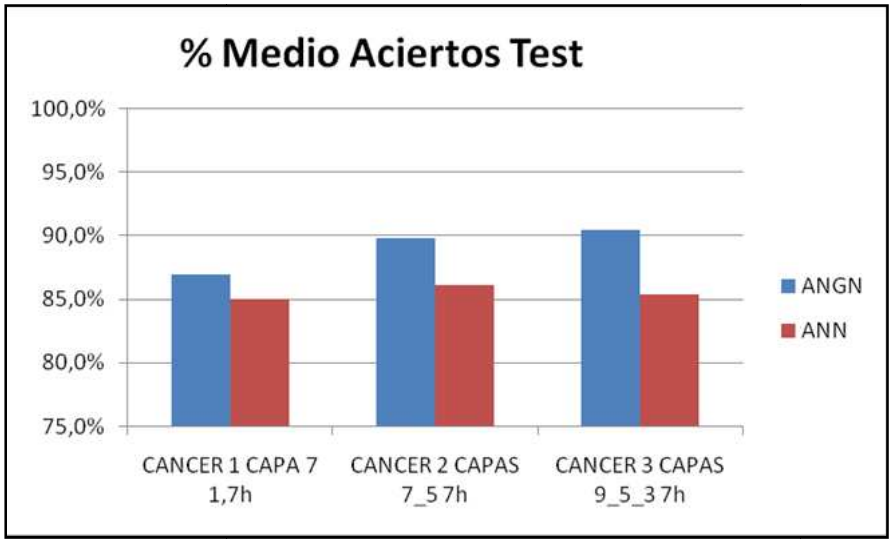


Figura 47: Relación entre la complejidad de la arquitectura y el porcentaje de aciertos en test en Cáncer

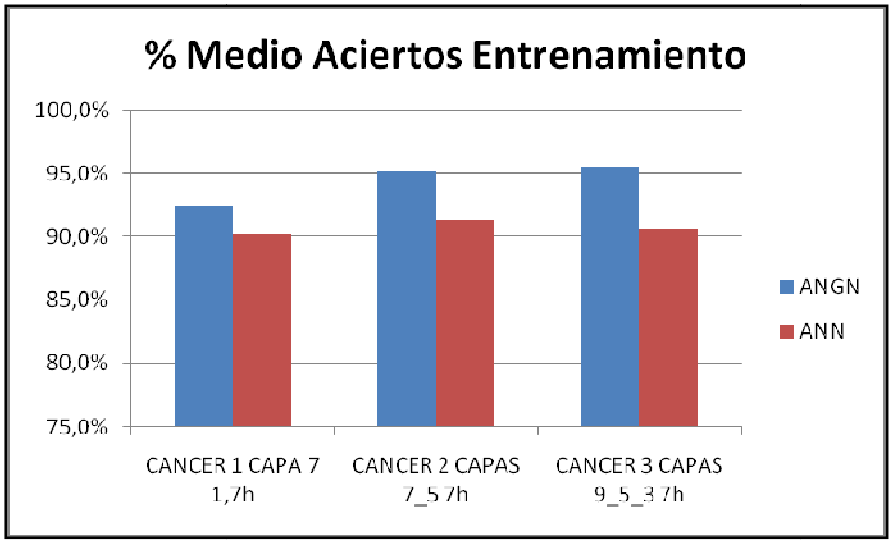


Figura 48: Relación entre la complejidad de la arquitectura y porcentaje de aciertos en entrenamiento en Cáncer

Tras la aplicación de los estadísticos t de Student y test de Wilcoxon a las arquitecturas de dos y tres capas ocultas se obtuvieron los siguientes resultados. El p-valor obtenido tras aplicar la prueba t de Student para dos capas ocultas fue de 3,11e-26, y para tres capas ocultas 2,05e-36, lo que indica que las diferencias son significativas en ambos casos. Por otro lado, aplicando el test de Wilcoxon, el p-valor obtenido para dos

capas ocultas es 2.258e-09, y para tres capas ocultas es 2.2e-016, por lo que la probabilidad, en ambos casos, de que los dos conjuntos sean significativamente diferentes al 95% es del 100%. De nuevo, las diferencias entre los resultados obtenidos con los dos métodos son muy significativas (las medias y las medianas de los resultados obtenidos con RNGA son menores que las obtenidas con RNA). Cabe remarcar el siguiente aspecto de los resultados obtenidos: las diferencias son más significativas cuánto más compleja es la arquitectura a analizar, esto es, estadísticamente las diferencias entre los resultados de RNGA y RNA son más significativas a medida que aumenta la complejidad del sistema.

Por otra parte, al estudiar el tiempo empleado para alcanzar el mismo error de test con RNA y RNGA (ver figura 49, en la que se muestran las arquitecturas en que los resultados son más significativos), puede observarse una clara evolución del tiempo relacionada con la complejidad de la arquitectura: a mayor complejidad, mejor rendimiento temporal por parte de RNGA con respecto a RNA. Se reduce significativamente la diferencia entre los tiempos que tardan ambos tipos de redes. Mientras que con una única capa oculta, RNGA tarda 67 minutos más que RNA en alcanzar el mismo error, para la arquitectura de 3 capas ocultas, RNGA consigue el mismo error en el mismo tiempo (en el caso de la arquitectura 12-8-4) o en menor tiempo (18 minutos menos en el caso de la arquitectura 9-5-3, que es la mostrada en las figuras).

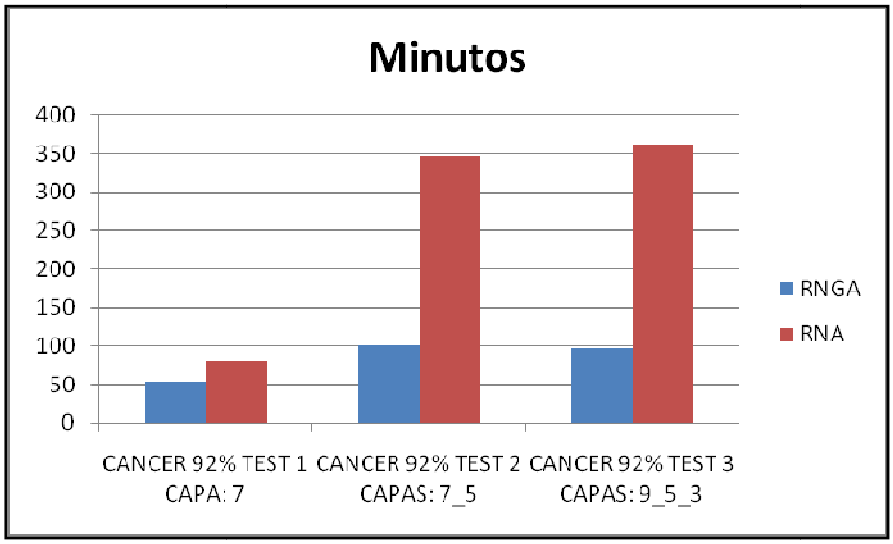


Figura 49: Relación entre la complejidad de la arquitectura y el tiempo en Cáncer

Por lo tanto, se puede afirmar que las RNGA obtienen un mayor porcentaje de aciertos en test, y además, esta mejora va acompañada de una disminución en el tiempo de ejecución a medida que aumenta la complejidad de la arquitectura.

El error de test considerado en el caso de las RNGA para cada simulación de población-conjunto *ij*, donde *i=1..10* y *j=1..10* es el mínimo entre los obtenidos por las 4 posibles combinaciones de iteración-activación simuladas (4-2, 6-2, 6-3 y 8-3). Es decir, para cada simulación *ij* se han realizado 5 pruebas, una con RNA y 4 con RNGA, lo que da un total de 500 resultados que han sido analizados para este problema. En la Tabla 9 puede observarse la proporción de veces en el que cada una de estas combinaciones ha obtenido el menor error de test en los test realizados.

	7	**7-3**	**7-5**	**9-5-3**	**12-8-4**	**Total**
24	45	19	24	29	25	142
26	14	21	28	24	25	112
36	23	16	24	13	22	98
38	18	44	24	34	28	148

Tabla 9: Comparativa de Combinaciones activación-iteración que han resultado ser las mejores en Cáncer

La combinación activación-iteración que ha resultado ser la mejor en el mayor número de casos ha sido la 38 (8 iteraciones, 3 activaciones); de 500 casos, en 148 ocasiones se obtuvo el mínimo error de test con esta combinación. Además, se debe hacer notar que para casi todas las arquitecturas, excepto para 7 y 7_5, la combinación 38 ha sido la mejor. Este hecho es significativo en el sentido siguiente: a mayor actuación de la glía, mejores soluciones.

5.1.4. Problema de Clasificación de Señales de Ionosfera. Estudio de complejidad.

Este problema, también tomado del UCI, consiste en clasificar mediciones de radar de la capa de la ionosfera. Estos datos de radar fueron tomados por un sistema en Goose Bay, Labrador, Canadá. Este sistema consiste en un conjunto de 16 antenas de alta frecuencia con un poder de transmisión total del orden de 6.4 kilowatios. Los

objetivos fueron los electrones libres en la ionosfera. Las mediciones "buenas" son aquellas que muestran evidencia de algún tipo de estructura en la ionosfera. Por su parte, las "malas" son aquellas que no muestran tal evidencia, es decir, las señales pasan a través de la ionosfera. A partir de 34 atributos, se quiere predecir este tipo de presencia o no de estructuras. En este caso se cuenta con 351 instancias.

Se ha estudiado este problema con un total de 5 arquitecturas diferentes (tabla 10).

Número de capas ocultas	1	2	3
Número de neuronas por capa oculta	9	9-4 9-8	9-6-3 12-8-4

Tabla 10: Arquitecturas empleadas en Ionosfera

Al considerar 5 arquitecturas, al igual que en el problema de Cáncer de Mama, se han realizado 2500 pruebas.

5.1.4.1. Resultados y discusión

5.1.4.1.1. Una capa oculta

La tabla 11 resume las medias globales para los 500 test realizados con una arquitectura de 9 neuronas en la capa oculta transcurridas 2 horas de entrenamiento (como se observa más adelante, al igual que en el problema anterior, se han empleado 7 horas para las restantes arquitecturas de red analizadas). Estos tiempos se han establecido, como en los demás problemas, a partir del Modo 1, observando que tras ellos los resultados no sufrían apenas modificación alguna. Se muestran:

- La media y el mínimo error de entrenamiento, para obtener una visión global del proceso de entrenamiento.
- La media y el mejor porcentaje de aciertos en test, para comprobar la capacidad de generalización.

	1 CAPA OCULTA	
	RNGA	RNA
MEDIA ERROR ENTRENAMIENTO	0,07	0,15
MEDIA % ACIERTOS TEST	83,28%	78,11%
MÍNIMO ERROR ENTRENAMIENTO	0,02	0,06
MEJOR % ACIERTOS TEST	89,14%	85,23%

Tabla 11: Análisis resultados Ionosfera transcurridas 2 horas de entrenamiento (1 capa oculta)

Las diferencias entre los resultados de los 100 casos simulados para esta arquitectura son muy significativas, tal y como demuestran los estadísticos t de Student y test de Wilcoxon. El contraste de normalidad de Kolmogorov-Smirnov Lilliefors resultó estadísticamente significativo, al igual que en el caso de la arquitectura con 3 capas ocultas: se rechazó la hipótesis nula por lo que no se cumple el supuesto de normalidad. El p-valor obtenido tras aplicar la prueba t de Student, válida igualmente según el Teorema Central del Límite, fue de 2,37e-21, lo que indica que las diferencias son significativas. Por otro lado, aplicando el test de Wilcoxon el p-valor obtenido es 1.553e-015, por lo que la probabilidad de que los dos conjuntos sean significativamente diferentes al 95% es del 100%. Ambos estadísticos prueban matemáticamente la importante y positiva influencia de la introducción de los astrocitos artificiales en la resolución de este problema.

Tomando como ejemplo representativo el conjunto 9 de la población 4 y una única capa oculta podemos observar nuevamente la superioridad de RNGA sobre RNA en entrenamiento (mayor capacidad de aprendizaje) y test (mayor capacidad de generalización). En concreto, en el proceso de test RNGA muestra un rendimiento muy superior a RNA. Los datos se han obtenido aplicando el Modo 3 con un intervalo de 15’.

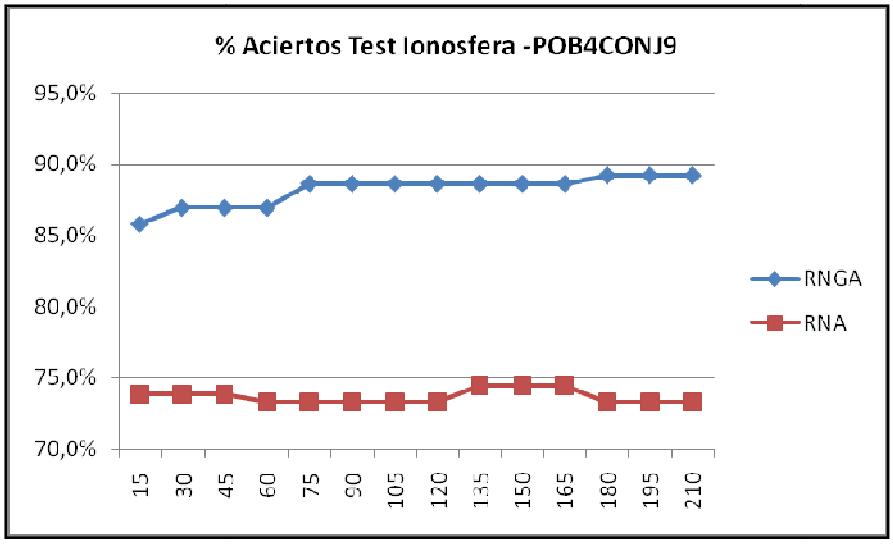


Figura 50: Evolución del Error de Test con respecto al tiempo (Ionosfera 1 capa oculta)

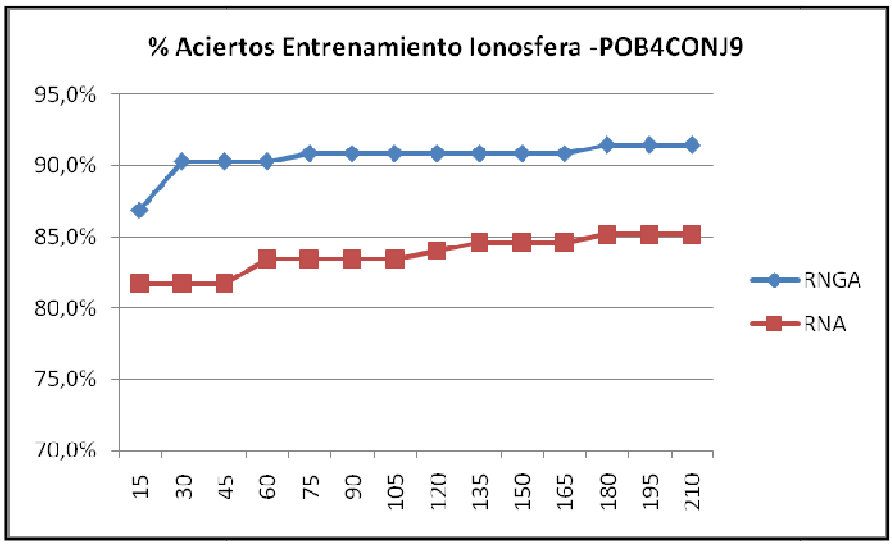


Figura 51: Evolución del Error de Entrenamiento con respecto al tiempo (Ionosfera 1 capa oculta)

En las siguientes figuras (52 y 53) se muestra la evolución de la media del porcentaje de aciertos en test y entrenamiento para los 100 casos de esta arquitectura con una única capa oculta.

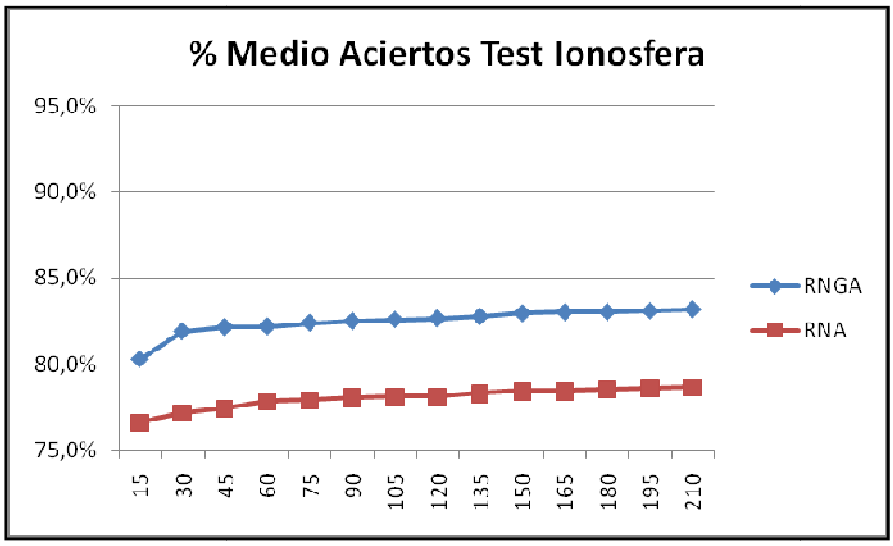


Figura 52: Evolución de Error Medio de Test con respecto al tiempo (Ionosfera 1 capa oculta)

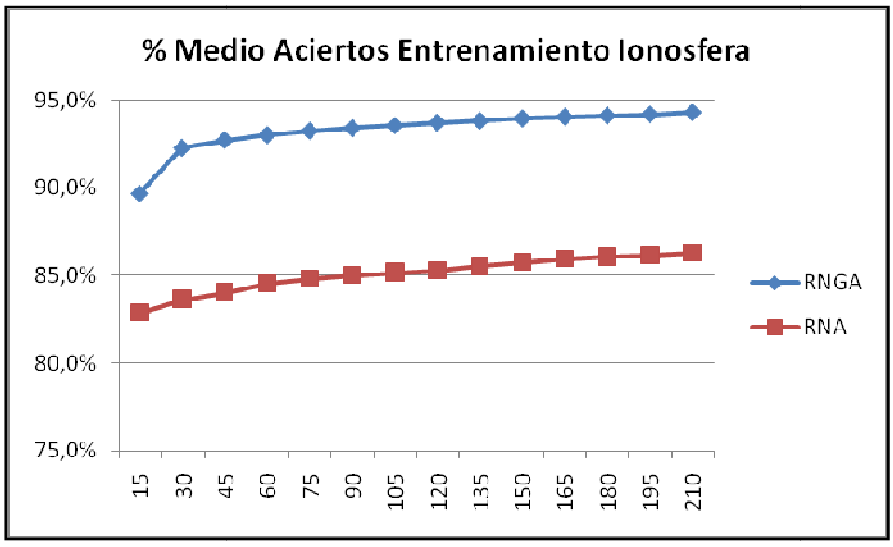


Figura 53: Evolución de Error Medio de Entrenamiento con respecto al tiempo (Ionosfera 1 capa oculta)

Para este problema, el proceso de aprendizaje y la capacidad de generalización de las RNGA es también más rápido y efectivo que en las RNA, tal y como se puede comprobar al observar la evolución del porcentaje medio de aciertos.

5.1.4.1.2. Comparación con arquitecturas de dos y tres capas ocultas

Para las 5 arquitecturas estudiadas, analizando el Modo 2, todos los porcentajes medios de aciertos en test son entre un 1,55% y un 10,73% mejores en RNGA que en RNA. En las figuras 54 y 55 se puede observar el aumento de la influencia de los astrocitos artificiales con el incremento de la complejidad de la arquitectura de las redes:

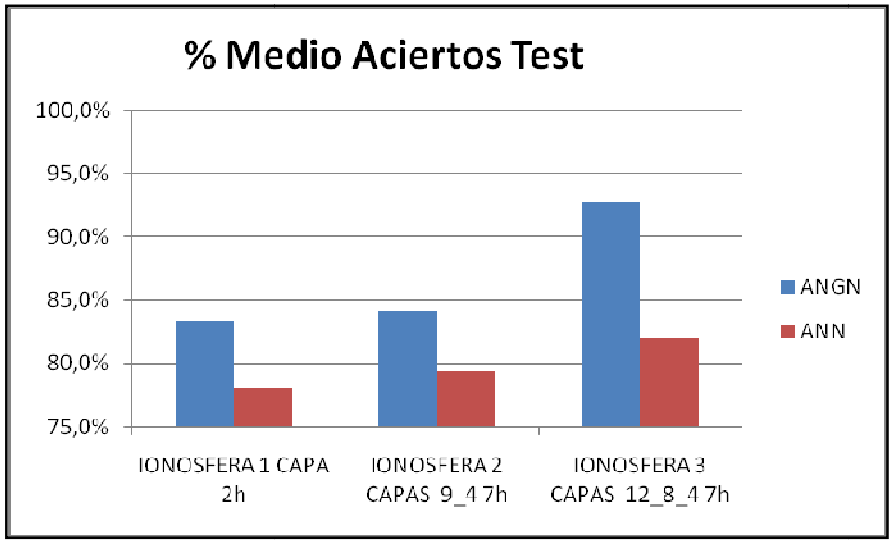


Figura 54: Relación entre la complejidad y el porcentaje de aciertos en test en Ionosfera

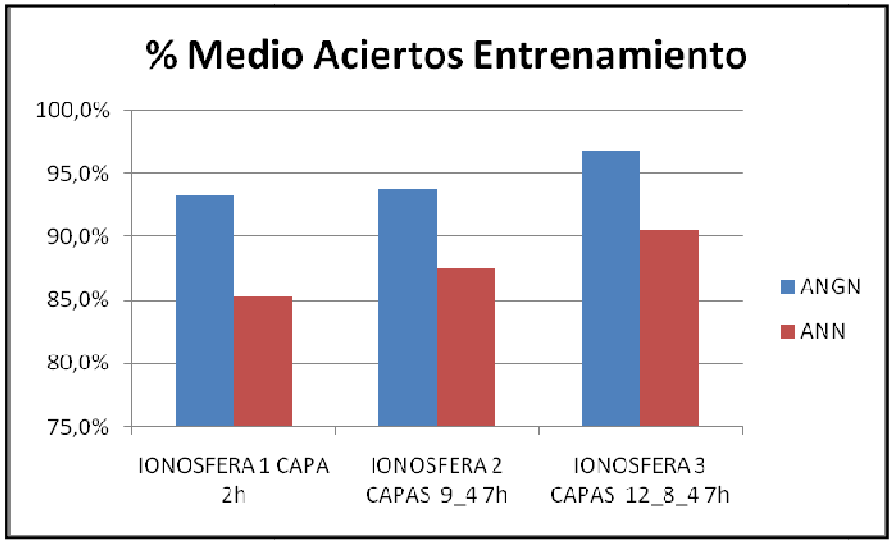


Figura 55: Relación entre la complejidad y porcentaje de aciertos en entrenamiento en Ionosfera

Puede observarse con claridad cómo la glía artificial actúa con mayor efectividad en aquellas arquitecturas de red más complejas.

Tras la aplicación de los estadísticos t de Student y test de Wilcoxon a las arquitecturas de dos y tres capas ocultas se obtuvieron los resultados que se presentan a continuación. El contraste de normalidad de Kolmogorov-Smirnov Lilliefors resultó de nuevo estadísticamente significativo para la arquitectura de 3 capas ocultas: no se cumple el supuesto de normalidad. A pesar de ello, tomando como referencia el Teorema Central del Límite, el empleo del t-test continúa resultando válido. No así con la arquitectura de 2 capas ocultas, cuyo test de Lilliefors resultó estadísticamente no significativo, corroborando la hipótesis de normalidad, tal y como, intuitivamente se podría suponer observando la figura 56. El p-valor obtenido tras aplicar la prueba t de Student para dos capas ocultas fue de 1,81e-15, y para tres capas ocultas 4,35e-29, lo que indica que las diferencias entre los resultados obtenidos con ambos sistemas son muy significativas. Por otro lado, aplicando el test de Wilcoxon el p-valor obtenido para dos capas ocultas es 1.307e-012, y para tres capas ocultas es 2.2e-016, por lo que la probabilidad, en ambos casos, de que los dos conjuntos sean significativamente diferentes al 95% es del 100%. De nuevo, las diferencias entre los resultados obtenidos con los dos métodos son muy significativas.

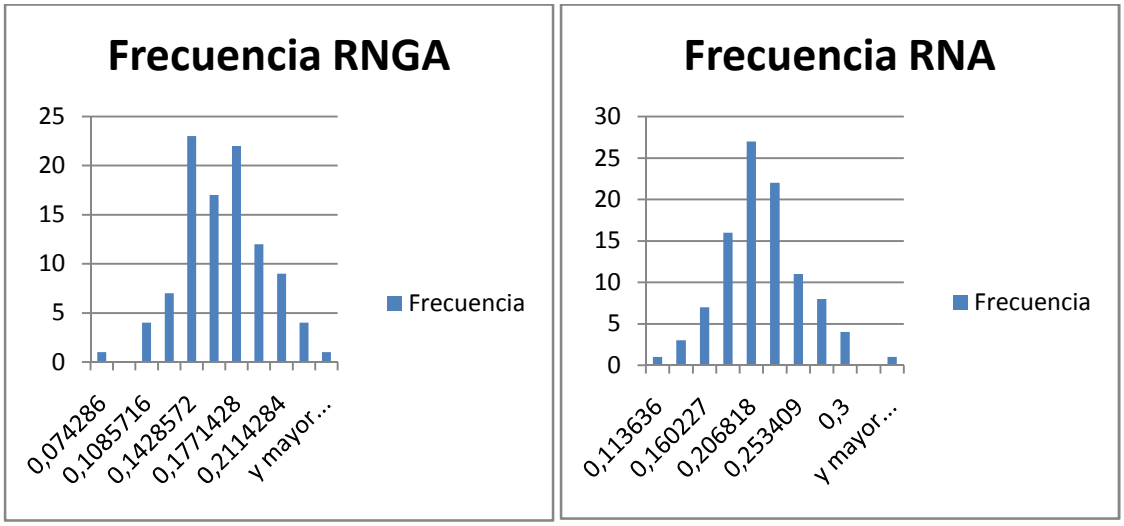


Figura 56: Histogramas correspondientes a las observaciones del problema de clasificación de señales de la Ionosfera con 2 capas ocultas (RNGA y RNA)

Al estudiar el tiempo empleado para alcanzar el mismo error de test con RNA y RNGA, puede observarse cómo, a medida que aumenta la complejidad de la arquitectura, se va reduciendo la diferencia de tiempo entre RNGA y RNA para alcanzar el mismo error. Para una única capa oculta la red con glía obtiene el mínimo 84 minutos antes, mientras que para tres capas ocultas lo alcanza con una anterioridad de 178 minutos (2 horas 58 minutos). Este hecho se muestra en la figura 57, donde se ilustra en una gráfica, la diferencia de tiempos para RNA y RNGA de cara a alcanzar un porcentaje de aciertos del 81%:

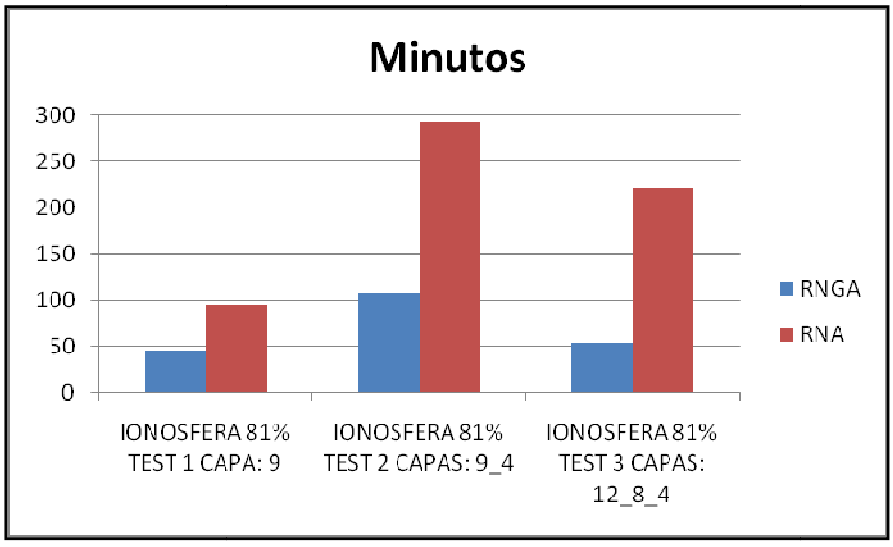


Figura 57: Relación entre la complejidad y el tiempo en Ionosfera

Por lo tanto, para el Problema de Ionosfera se puede afirmar que en todos los casos RNGA obtiene un mayor porcentaje de aciertos en test que RNA y en menos tiempo.

	9	**9-4**	**9-8**	**9-6-3**	**12-8-4**	**Total**
24	42	21	37	27	13	140
26	16	23	16	28	7	90
36	24	24	26	24	43	141
38	18	32	21	21	37	129

Tabla 12: Comparativa de Combinaciones activación-iteración que han resultado ser las mejores en Ionosfera

En la Tabla 12 puede observarse el número de veces que cada una de las combinaciones de iteraciones y activaciones ha obtenido el menor error de test en las simulaciones realizadas. Se puede observar cómo, a medida que va aumentando la complejidad de la red, va disminuyendo el número de veces que las combinaciones con menor actividad de la glía son mejores que las demás: con una única capa oculta la combinación 4-2 es mejor en el 42% de los casos, para pasar a serlo en un 13% con tres capas ocultas. La evolución de la combinación 8-3 es la inversa, se pasa de un 18% a un 37% a medida que aumentan el número de capas ocultas y el número de neuronas por capa. Para la arquitectura más compleja de todas (12-8-4) la mejor combinación es 6-3, seguida directamente por 8-3, lo que indica que es necesaria más actividad para activar el astrocito y, de este modo, la red actúa mejor.

5.1.5. Comparativa Global.

Observando el porcentaje medio de aciertos en test y entrenamiento para los cuatro problemas estudiados en este proyecto (figuras 58 y 59), se puede concluir que la introducción de la glía artificial no sólo resulta siempre beneficiosa en test y entrenamiento, sino que a medida que aumenta la complejidad de los problemas y de las arquitecturas empleadas aumenta su eficacia.

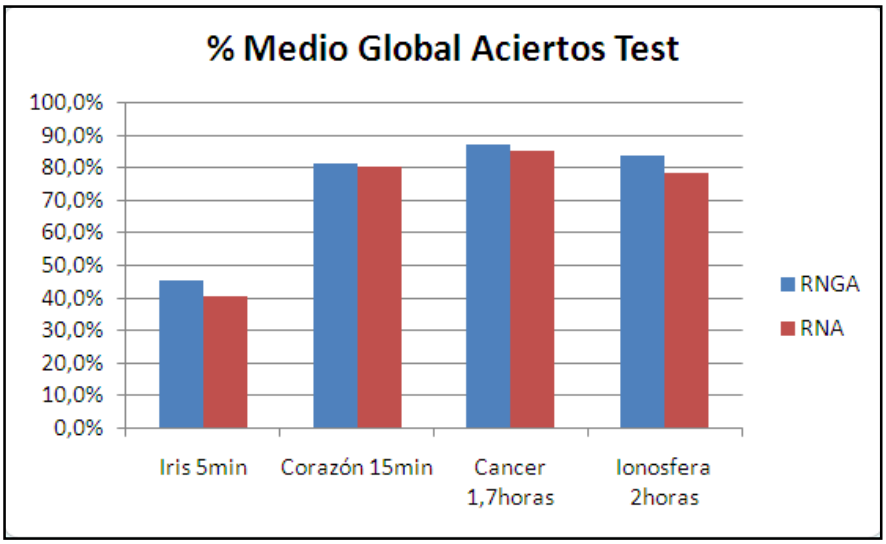


Figura 58: Porcentaje medio de aciertos en el test por problema en arquitecturas de una capa oculta

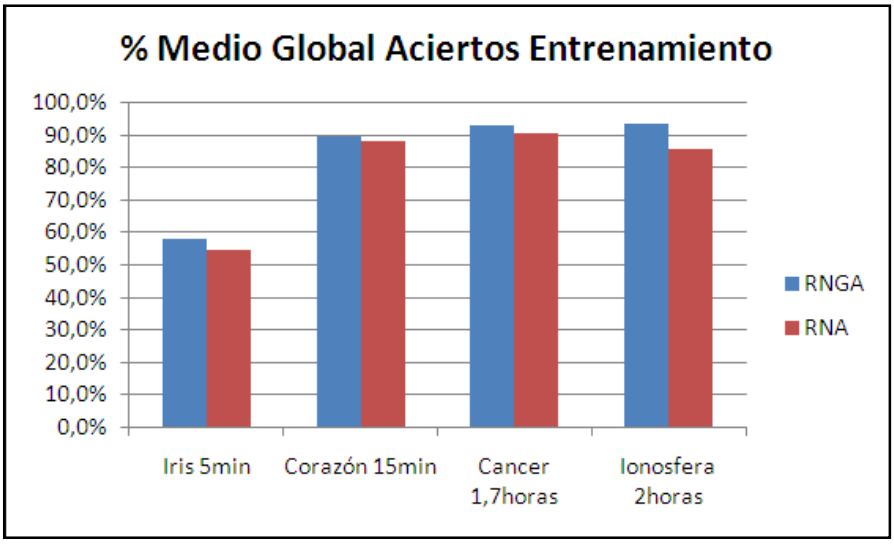


Figura 59: Porcentaje medio de aciertos en el Entrenamiento

En la figura 60 puede observarse que el tiempo transcurrido para alcanzar un mismo porcentaje de aciertos es menor en las RNGA para todos los problemas. Aún a pesar de la lentitud que introduce la glía artificial (al exigir que cada patrón se presente **X** iteraciones a la red, con lo que el tiempo de evaluación se incrementa respecto a las RNA) la RNGA obtiene mejores valores, necesitando menos tiempo para ello que RNA.

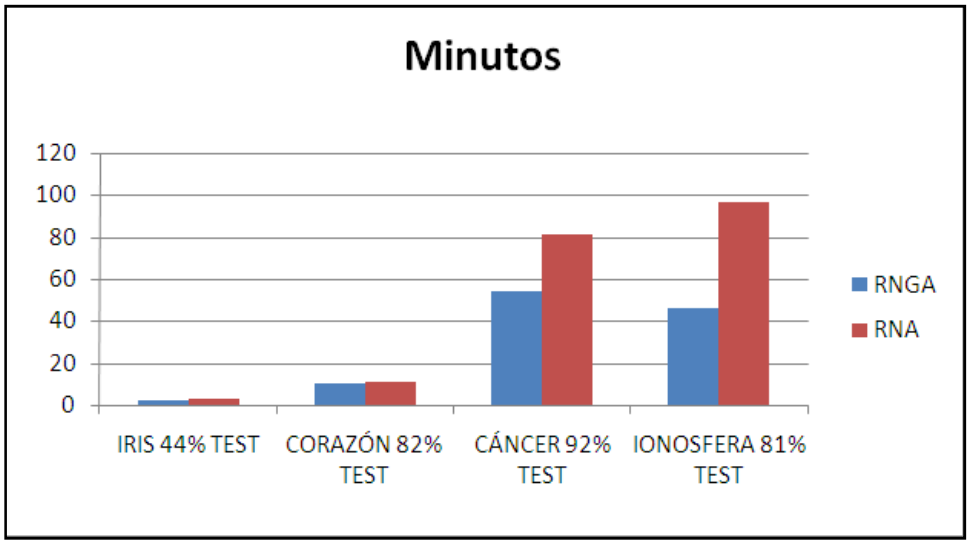


Figura 60: Tiempo transcurrido para alcanzar un determinado porcentaje de aciertos en test

5.2. Simulaciones para la optimización de RNGA mediante Algoritmos Genéticos

5.2.1. Optimización mediante el operador SBX (Simulated Binary Crossover)

En la segunda fase de este proyecto se han realizado pruebas con el cruce sbx (Simulated Binary Crossover) [DEB 95; DEB 98], operador para cromosomas de codificación real y sin punto de cruce, que combina los genes de los padres en una determinada proporción para obtener los genes de los hijos.

Este operador pone énfasis en la generación de hijos próximos a los padres. Esencialmente, hay dos propiedades que dotan al operador SBX de sus capacidades de búsqueda: los valores de los hijos son proporcionales a los de sus padres y, hay más posibilidades de que las soluciones próximas a los padres sean escogidas como soluciones hijas que aquellas otras más lejanas. Gracias a estas propiedades la diversidad en las soluciones de los hijos está directamente controlada por la diversidad en las soluciones paternas.

El principal objetivo es obtener un operador de cruce que evite la pérdida de la diversidad de la población de los individuos, y, al mismo tiempo, favorezca la velocidad de convergencia del algoritmo. Estos dos objetivos son, en un principio, antagónicos; su adecuado equilibrio es controlado por dos de las características básicas del operador de corte: i) el equilibrio entre la exploración y explotación y, ii) el componente auto-adaptativo del mismo.

El procedimiento de computación de soluciones hijas es el que sigue [DEB 07]. En primer lugar, hay que seleccionar un valor aleatorio *u* en el intervalo (0, 1), a partir del cual se calcula β_q como:

$$\beta_q = \begin{cases} (2u)^{\frac{1}{\eta+1}} & si\ u \leq 0.5 \\ \left(1 \Big/ (2(1-u))\right)^{\frac{1}{\eta+1}} & en\ otro\ caso \end{cases}$$

Siendo η cualquier número real no negativo. A mayores valores de η más parecidas serán las soluciones hijas a los padres. A menores valores mayor variabilidad y diversidad.

Finalmente, para obtener los nuevos valores de los hijos se procedería del siguiente modo:

gen_hijo_1 = 0.5 [(1 + β_q) gen_padre_1 + (1 - β_q) gen_padre_2]

gen_hijo_2 = 0.5 [(1 - β_q) gen_padre_1 + (1 + β_q) gen_padre_2]

Con el objetivo de comprobar la eficacia de este operador y los beneficios de su inserción en el sistema ya diseñado e implementado, se realizaron una serie de pruebas con dos de los problemas tratados en el capítulo 5: Ionosfera y Cáncer de Mama. Para el problema de Ionosfera se realizó la experimentación con la arquitectura de dos capas ocultas (9 neuronas en la primera y 4 en la segunda), y para el problema de Cáncer se empleó una arquitectura de 3 capas ocultas (9 en la primera, 5 en la segunda y 3 neuronas en la tercera capa oculta). Se eligieron estas dos arquitecturas, y no otras, por los buenos resultados previamente obtenidos con ellas, así como para observar hasta qué punto estos resultados podían ser superados con este nuevo operador. Se aplicaron sobre 10 poblaciones y 10 conjuntos de patrones siguiendo la técnica 5x2 cv ya explicada pretéritamente, y se experimentó con distintos valores de η para estudiar la influencia de este parámetro. Las pruebas realizadas en torno al operador SBX se llevaron a cabo con 5 valores de η: 0.1, 0.3, 0.5, 0.6 y 1; y las cuatro combinaciones activación-iteración mostradas en el apartado 2.2.3 dedicado al *Funcionamiento de RNGA*, junto con la combinación 00 correspondiente a RNA. Los mejores resultados se obtuvieron con valores de η de 0.3 y 0.5 para Ionosfera y de 0.3 y 0.1 para Cáncer. En total se realizaron 5000 pruebas en el CESGA para verificar la viabilidad e interés de este operador; estas 5000 simulaciones se obtienen como resultado de realizar 500 pruebas con RNA (100*5), 2000 con RNGA (4*100*5), y todo esto con los 2 problemas mencionados.

Los resultados obtenidos sin el cruce SBX para estos dos problemas, y ya comentados en el apartado 5.1, son los que se muestran en la tabla 13:

RESULTADOS PARA IONOSFERA 9_4		**SIN GLÍA**	**CON GLÍA**
MODO 1	% aciertos test	80,60%	85,12%
	media tiempo para alcanzar el mínimo (en minutos)	226,2	166,3
MODO 2 (7 horas)	% aciertos test	79,32%	83,91%
RESULTADOS PARA CÁNCER 9_5_3		**SIN GLÍA**	**CON GLÍA**
MODO 1	% aciertos test	86,27%	91,26%
	media tiempo para alcanzar el mínimo (en minutos)	114,1	96,5
MODO 2 (7 horas)	% aciertos test	85,39%	90,46%

Tabla 13: Resultados para Ionosfera 9-4 y Cáncer 9-5-3 sin el operador SBX

En la tabla siguiente se pueden observar los resultados de los casos más representativos de Ionosfera y Cáncer con SBX, y que demuestran el interés de incluir este nuevo operador debido a la mejora de resultados que conlleva.

RESULTADOS PARA IONOSFERA 9_4		η = 0,3		η = 0,5	
		SIN GLÍA	CON GLÍA	SIN GLÍA	CON GLÍA
MODO 1	% aciertos test	85,24%	86,44%	84,37%	86,33%
	media tiempo para alcanzar el mínimo (en minutos)	132,5	101,2	120,9	119,8
MODO 2 (7 horas)	% aciertos test	83,77%	85,03%	82,90%	85,03%
RESULTADOS PARA CÁNCER 9_5_3		**η = 0,3**		**η =0,1**	
		SIN GLÍA	**CON GLÍA**	**SIN GLÍA**	**CON GLÍA**
MODO 1	% aciertos test	90,21%	91,31%	90,39%	91,51%
	media tiempo para alcanzar el mínimo (en minutos)	101,5	109,8	88,4	99,8
MODO 2 (7 horas)	% aciertos test	89,73%	90,60%	89,67%	90,48%

Tabla 14: Resultados para Ionosfera 9-4 y Cáncer 9-5-3 con el operador SBX

Al analizar estos datos (ver tabla 15) se puede comprobar que el operador SBX mejora los resultados obtenidos tanto para RNA como RNGA, por lo que se incluye en las implementaciones, y se empleará para RNA y para los individuos de la población de pesos de la red en RNGA; en los individuos de la población de parámetros (en el caso de AGCC) no puede usarse debido a la codificación entera del correspondiente cromosoma.

La mejora obtenida con este nuevo operador afecta en mayor medida a RNA que a RNGA, debido presumiblemente a que este tipo de red tenía un mayor margen de mejora que la red con astrocitos artificiales. Otra conclusión que se podría extraer de los datos sería la siguiente: con nuevos métodos de optimización que mejoran a los AG tradicionales se consiguen mejores resultados que continúan sin superar a la RNGA.

DIFERENCIAS RESULTADOS PARA IONOSFERA 9_4		η = 0,3		η = 0,5	
		SIN GLÍA	CON GLÍA	SIN GLÍA	CON GLÍA
MODO 1	% aciertos test	4,64%	1,32%	3,77%	1,21%
	media tiempo para alcanzar el mínimo (en minutos)	-93,6	-65,1	-105,3	-46,5
MODO 2 (7 horas)	% aciertos test	4,45%	1,12%	3,58%	1,12%
DIFERENCIAS RESULTADOS PARA CÁNCER 9_5_3		**η = 0,3**		**η =0,1**	
		SIN GLÍA	**CON GLÍA**	**SIN GLÍA**	**CON GLÍA**
MODO 1	% aciertos test	3,94%	0,05%	4,12%	0,25%
	media tiempo para alcanzar el mínimo (en minutos)	-12,6	13,2	-25,8	3,3
MODO 2 (7 horas)	% aciertos test	4,34%	0,14%	4,28%	0,02%

Tabla 15: Diferencias en los resultados al emplear el operador SBX

Los resultados de la aplicación del test de Wilcoxon se muestran en las tablas 16 y 17. Se demuestra que la capacidad de aprendizaje y generalización de RNGA continúa siendo significativamente mejor y que la inclusión del operador SBX es beneficiosa. Las diferencias resultan menores en el caso de RNGA con y sin operador SBX (debido a que la mejora obtenida con RNGA sin este operador ya era sobresaliente), y máximas en el de RNA y RNGA sin SBX (debido a la clara superioridad de las redes a las que se les incluye glía artificial).

	p-valor	**Probabilidad de que los dos conjuntos sean significativamente diferentes al 95%**
RNA sin SBX – RNGA sin SBX	2.580e-012	1.000 (Significativo)
RNA sin SBX – RNA con SBX	2.043e-007	0.999 (Significativo)
RNGA sin SBX – RNGA con SBX	2.344e-002	0.976 (Significativo)
RNA con SBX – RNGA con SBX	3.570e-005	0.999 (Significativo)

Tabla 16: Resultados del test de Wilcoxon aplicado al problema de clasificación de señales de la Ionosfera con dos capas ocultas (9_4)

	p-valor	Probabilidad de que los dos conjuntos sean significativamente diferentes al 95%
RNA sin SBX – RNGA sin SBX	1.980e-016	1.000 (Significativo)
RNA sin SBX – RNA con SBX	6.229e-005	0.999 (Significativo)
RNGA sin SBX – RNGA con SBX	9.689e-001	0.031 (No significativo)
RNA con SBX – RNGA con SBX	2.857e-002	0.971 (Significativo)

Tabla 17: Resultados del test de Wilcoxon aplicado al problema de predicción de Cáncer de Mama con 3 capas ocultas (9_5_3)

5.2.2. Simulaciones Algoritmo Genético Coevolutivo

5.2.2.1. Ajuste del Algoritmo Genético Coevolutivo

En este proyecto se abordan por primera vez estudios de optimización de RNGA mediante esta técnica de Computación Evolutiva. Indicar por tanto, que se trata de un estudio preliminar, ya que dadas las diferentes y numerosas posibilidades en cuanto a parámetros de ajuste, se requiere una experimentación muy amplia para la generalización de resultados.

La experimentación con el AGCC implementado se llevó a cabo en dos etapas. En la primera de ellas, se realizó una batería de pruebas preliminares para ajustar las variables del AGCC: tamaño de la población de parámetros, número de individuos cooperantes aleatorios y mejores, tipo de fitness y porcentajes de mutación y cruce. Estas pruebas se llevaron a cabo con el problema de clasificación de señales de la Ionosfera y la arquitectura de una única capa oculta con 9 neuronas, ya empleada y estudiada en el apartado 5.1.4, por tratarse de un problema y arquitectura adecuados para la experimentación, dado que se puede observar con gran claridad la influencia de la glía artificial. En la segunda etapa de esta experimentación con AGCC, se aplicaron

los valores de ajuste obtenidos a la resolución de dos problemas de clasificación y predicción: Flor de Iris y Cáncer de mama.

En la tabla 18 se pueden observar todas las combinaciones empleadas para ajustar los parámetros del AGCC (en total, 54), que suponen un total de 540 ejecuciones en la máquina SVG del CESGA, teniendo en cuenta que se emplearon 10 conjuntos de datos según el método 5x2cv y una única población de individuos. A la hora de realizar las simulaciones se estableció un tiempo máximo de ejecución en el CESGA de 20 horas; mismo tiempo límite de ejecución de trabajos en el CESGA que el empleado en las simulaciones anteriores. Como se indica a continuación este tiempo resultó insuficiente.

Cruce Coevolutivo	**Mutación Coevolutivo**	**Tamaño población pesos**	**Tamaño población parámetros**	**Número cooperantes aleatorios**	**Número cooperantes mejores**	**Tipo fitness**
90	1	50	10	1	1	Mínimo
	10	100	25	2	2	Media
		150	30	3	3	máximo

Tabla 18: Combinaciones de parámetros del AGCC empleadas en la experimentación

Las conclusiones extraídas, a partir de los resultados obtenidos en este proceso de ajuste, fueron las siguientes:

- Las 20 horas de ejecución prefijadas resultan insuficientes en 400 de 540 casos: no resulta tiempo suficiente para llegar a las 5000 generaciones. De media, la última generación escrita en el log alcanzadas las 20 horas ronda la 2500. Con 150 individuos en la población de redes y 30 individuos en la de parámetros no acaba ninguna ejecución.
- El espacio de búsqueda de soluciones es mucho más amplio, por lo que la variabilidad en los errores de test en RNGA aumenta mucho más debido a la no convergencia en muchos de los casos.
- Los mejores resultados se obtienen con los fitness media y mínimo.
- Con poblaciones de 50 individuos de pesos y 10 individuos de parámetros se obtienen los peores resultados: un 4,73% peor en test que los resultados obtenidos previamente sin AGCC. Resultan necesarias poblaciones de individuos más amplias que favorezcan una mayor variabilidad genética.

En la siguiente tabla (tabla 19) se presentan las 5 combinaciones de parámetros con las que se obtuvieron los mejores resultados:

Cruce Coevolutivo	Mutación Coevolutivo	Tamaño población redes	Tamaño población parámetros	Número cooperantes aleatorios	Número cooperantes mejores	Tipo fitness	Mejora obtenida
90	10	150	30	1	1	Media	0,79%
90	10	150	30	3	3	Mínimo	-0,41%
90	10	150	25	1	1	Media	-0,23%
90	10	100	25	3	3	Media	-0,40%
90	1	100	25	3	3	Media	-0,28%

Tabla 19: Combinaciones de parámetros del AGCC con los que se obtuvieron mejores resultados

En base a toda esta experimentación, así como de las conclusiones previamente mencionadas, se decidió emplear en las futuras simulaciones 7500 generaciones con un límite de 70 horas de ejecución en las máquinas del CESGA, poblaciones de pesos y parámetros de 150 y 30 individuos respectivamente, 3 cooperantes aleatorios y 3 cooperantes mejores, medida de ajuste "mínimo", y porcentajes de cruce y mutación 90% y 10%. A pesar de que en cuatro de las cinco mejores combinaciones mostradas en la tabla 19 la medida de ajuste es la media, se decidió emplear el mínimo por ser el ajuste más común en la bibliografía [CASI 01; ALCA 03].

5.2.2.2. Problema de Clasificación de Flor de Iris

Se realizó un experimento completo con el problema de clasificación de Flor de Iris (arquitectura con una única capa oculta de 5 neuronas) con el objetivo de comenzar observando el funcionamiento y rendimiento del AGCC para un problema de menor complejidad. Se realizaron las 100 simulaciones correspondientes a las 10 poblaciones y los 10 conjuntos del cross-validation.

A continuación, en la tabla 20, se puede observar una comparativa entre los resultados previamente obtenidos mediante RNGA y los recientes de RNGA optimizada

mediante AGCC. Se muestran los resultados obtenidos tras aplicar el Modo 2 a los 10 minutos.

	RNGA	RNGA AGCC
MEDIA ERROR ENTRENAMIENTO	0,41	0,37
MEDIA % ACIERTOS TEST	45,15%	29,76%
MÍNIMO ERROR ENTRENAMIENTO	0,19	0,03
MEJOR % ACIERTOS TEST	72,00%	86,67%
DESVIACIÓN TÍPICA ERRORES ENTRENAMIENTO	0,17	0,20
DESVIACIÓN TÍPICA ERRORES TEST	0,14	0,32

Tabla 20: Comparativa resultados RNGA y RNGA con AGCC para el problema de clasificación de Flor de Iris (Modo 2)

En estos datos se puede observar que desde el punto de vista de optimización el AGCC mejora mucho el rendimiento: los mínimos alcanzados en el entrenamiento, así como la media del error de entrenamiento, resultan muy superiores a los obtenidos pretéritamente mediante la optimización con AG. No obstante, comparando los resultados del AGCC con los del AG, la media del porcentaje de aciertos en test es casi un 16% peor. Inicialmente, se pensó que esto podría ser debido a un sobreentrenamiento de la red (posiblemente porque el conjunto de patrones disponible es muy pequeño y resulta insuficiente para que las RNGA con AGCC extraigan relaciones adecuadas de los datos), ya que a un error medio de entrenamiento menor le corresponde un error medio de test mayor. Es importante remarcar también que el mejor porcentaje de aciertos en test es claramente superior con AGCC, casi un 15% mejor, lo que implica que ciertas combinaciones concretas de parámetros gliales proporcionan unos aciertos en test máximos muy superiores a los conseguidos con los valores de parámetros probados con anterioridad.

Mediante el Modo 1, que permite observar la evolución completa de los errores durante todas las generaciones definidas, se obtienen los resultados que se presentan en la siguiente tabla.

	RNGA	RNGA AGCC
MEDIA ERROR TEST	0,51	0,50
PORCENTAJE MEDIO ACIERTOS TEST	48,88%	50,32%
DESVIACIÓN TÍPICA	0,11	0,19
MENOR ERROR TEST	0,28	0,13
TIEMPO MEDIO EMPLEADO PARA ALCANZAR EL MÍNIMO (en minutos)	2,8	35,9

Tabla 21: Comparativa resultados RNGA y RNGA con AGCC para el problema de clasificación de Flor de Iris (Modo 1)

Estos datos muestran un porcentaje de aciertos en test un 1,44% superior con AGCC que el obtenido sin emplear este nuevo paradigma de aprendizaje. La principal conclusión que se puede extraer, tras observar los resultados del Modo 1 (tabla 21), es que el AGCC requiere un mayor tiempo de convergencia: no resultan suficientes los 10 minutos empleados con RNGA en el Modo 2.

Resaltar con respecto al tiempo medio necesario para alcanzar ese porcentaje de test en cada caso, que los resultados presentados para el problema de Flor de Iris sin optimizar mediante el AGCC se obtuvieron tras realizar simulaciones con 4 combinaciones de iteración-activación y dos algoritmos gliales (en total 8 ejecuciones), quedándose en cada caso con el mejor resultado obtenido. Mientras que los resultados presentados tras emplear la optimización con AGCC requieren una única simulación (1 única ejecución), en la que se realizan combinaciones de algoritmos gliales, porcentajes de incremento y decremento diferentes, distintos valores de iteraciones y activaciones, etc. El espacio de búsqueda, en este último caso, aumenta considerablemente, de modo que el proceso de entrenamiento requiere mucho más tiempo y los resultados obtenidos presentan una mayor variabilidad, como se puede comprobar observando la dispersión de los mismos reflejada en unas desviaciones típicas muy superiores. Es preciso tener en consideración que pequeños cambios en los valores de los individuos de parámetros implican grandes cambios en el espacio de búsqueda.

Indicar que entre los 100 casos considerados (10 conjuntos x 10 poblaciones) no se encontró ningún patrón común que destacase especialmente en las combinaciones de parámetros con los que se obtuvieron los mejores resultados. No obstante, sí se repitieron en varios casos combinaciones como: 8 iteraciones, 3 activaciones, algoritmo glial No Consecutivo Con Límite de Pesos, porcentaje de incremento 10% y porcentaje de decremento 30%; 7 iteraciones, 5 activaciones, algoritmo glial No Consecutivo Sin

Límite de Pesos, porcentaje de incremento 20% y porcentaje de decremento 30%; o, 3 iteraciones, 2 activaciones, algoritmo glial Consecutivo Con Límite de Pesos, porcentaje de incremento 20% y porcentaje de decremento 80%. Esta variabilidad se supone relacionada con el funcionamiento del cerebro, en donde la glía actúa de distinto modo según la información recibida en cada momento por el ser vivo; del mismo modo que ocurre en esta experimentación al presentar 10 conjuntos y 10 poblaciones diferentes. Por lo tanto, junto con las claras ventajas computacionales que conlleva la optimización, destacar que el funcionamiento del AGCC emula todavía con mayor fidelidad el funcionamiento del cerebro, lo que supone un claro beneficio para la neurociencia.

5.2.2.3. Problema de Predicción de Cáncer de Mama

Para realizar el siguiente experimento, se empleó un problema de mayor complejidad que el anterior y de diferente finalidad: el problema de predicción de Cáncer de Mama (usando una arquitectura con una única capa oculta de 7 neuronas). Se realizaron 100 simulaciones correspondientes a las 10 poblaciones y los 10 conjuntos del cross-validation.

A continuación, en la siguiente tabla, se puede observar una comparativa entre los resultados previamente obtenidos mediante RNGA, y los recientes de RNGA optimizado mediante AGCC, transcurridos 1h 45' del entrenamiento.

	RNGA	RNGA AGCC
MEDIA ERROR ENTRENAMIENTO	0,07	0,04
MEDIA % ACIERTOS TEST	86,96%	87,76%
MÍNIMO ERROR ENTRENAMIENTO	0,01	0,01
MEJOR % ACIERTOS TEST	97,37%	97,08%
DESVIACIÓN TÍPICA ERRORES ENTRENAMIENTO	0,08	0,09
DESVIACIÓN TÍPICA ERRORES TEST	0,12	0,12

Tabla 22: Comparativa resultados RNGA y RNGA con AGCC para el problema de predicción de Cáncer de Mama (Modo 2)

Para este problema las RNGA optimizadas con el AGCC no sólo entrenan mejor, sino que el porcentaje de aciertos en test que alcanzan es también superior.

Revisando el Modo 1 para este problema, con el fin de conocer el tiempo necesario en alcanzar los porcentajes conseguidos, se puede observar la eficacia de la aplicación del AGCC (ver tabla 23).

	RNGA	RNGA AGCC
MEDIA ERROR TEST	0,12	0,09
PORCENTAJE MEDIO ACIERTOS TEST	87,85%	91,00%
DESVIACIÓN TÍPICA	0,11	0,11
MENOR ERROR TEST	0,0263	0,0205
TIEMPO MEDIO EMPLEADO PARA ALCANZAR EL MÍNIMO (en minutos)	90,6	82,8

Tabla 23: Comparativa resultados RNGA y RNGA con AGCC para el problema de predicción de Cáncer de Mama (Modo 1)

El porcentaje de aciertos en test es un 3,15% mejor empleando el AGCC. Nuevamente se puede observar la lentitud de convergencia de este nuevo paradigma de aprendizaje: al superar 1h 45' el AGCC todavía mejora en un 3,24% mientras la RNGA sin optimización sólo lo hace en un 0,89%. Las 2 horas empleadas en el Modo 2, tiempo inspirado en las observaciones previas a la introducción del AGCC, muestran en este punto su carácter obsoleto de cara a su aplicación en este nuevo paradigma de aprendizaje pues, con posterioridad a este instante, el Coevolutivo alcanza todavía mucho mejores resultados.

Al igual que en el problema de clasificación de flor de Iris, no se encontró ningún patrón común en las combinaciones de parámetros con las que se obtuvieron los mejores resultados, exceptuando el algoritmo glial, que en la mayor parte de los casos resultó ser el algoritmo glial Glía Atenuación. Algunas combinaciones con las que se obtuvieron algunos de los resultados más positivos: 8 iteraciones, 4 activaciones, algoritmo glial Glía Atenuación, porcentaje de incremento 20% y porcentaje de decremento 30%; 9 iteraciones, 6 activaciones, algoritmo glial Glía Atenuación, porcentaje de incremento 20% y porcentaje de decremento 40%; o, 5 iteraciones, 4 activaciones, algoritmo glial Glía Atenuación, porcentaje de incremento 10% y porcentaje de decremento 90%.

De nuevo se puede observar que la duración de estas pruebas es mayor que la de las realizadas hasta ahora, debido tanto al mayor coste computacional de la función de evaluación de los individuos, como a la posibilidad de trabajar con combinaciones de

parámetros en las que se incluya un valor elevado del parámetro iteraciones (indicador del número de veces que se muestra cada patrón a la red). De todos modos, dado que para obtener los resultados de RNGA hasta ahora se consideraban 4 combinaciones iteración-activación de glía (correspondientes con 4 ejecuciones independientes), esta lentitud mencionada no resulta tan remarcable.

Por otro lado, destacar especialmente que, para este problema, el AGCC incluso obtiene en un menor tiempo para el Modo 2 un mejor porcentaje medio de aciertos en test que la RNGA sin optimización, lo que podría estar relacionado con la hipótesis, sostenida en este trabajo, de mejor actuación de la glía en casos de mayor complejidad.

6. CONCLUSIONES

A continuación, se presentan las conclusiones a las que se ha llegado en cada una de las dos etapas globales de trabajo que constituyen el presente proyecto de investigación y desarrollo que, como se ha mostrado, relaciona estrechamente las áreas de Inteligencia Artificial y Neurociencia.

6.1. Conclusiones de la primera fase del proyecto

En la primera fase de este proyecto se ha llevado a cabo un estudio de la eficacia y eficiencia de las RNGA frente a las RNA multicapa entrenadas mediante AG, siendo las RNGA redes que incluyen además de neuronas artificiales, elementos de control de procesamiento que simulan el comportamiento los astrocitos del Sistema Glial. Para efectuar este análisis se ha realizado una comparativa de los resultados obtenidos por ambos tipos de redes al resolver, aplicando la técnica de *cross-validation 5x2cv,* cuatro problemas de diferente complejidad: dos de clasificación (Flor de Iris y Señales de la Ionosfera) y dos de predicción (Cáncer de Mama y Enfermedades Coronarias). En dos de estos problemas (Cáncer e Ionosfera) se ha analizado también la influencia de la glía a medida que se incrementaba la complejidad de las arquitecturas de las redes a comparar.

Las técnicas y análisis estadísticos empleados han permitido asegurar que para estos cuatro problemas, las diferencias entre los resultados obtenidos por RNGA y RNA son muy significativas, siendo menores tanto las medias como las medianas de los errores cometidos por las RNGA en todos los problemas y que dichos resultados no dependen del orden de entrada de los datos, ni de la aleatoriedad de los individuos generados.

Se ha comprobado que a pesar de que los astrocitos artificiales son de actuación más lenta que las neuronas artificiales, las RNGA consiguen en un menor tiempo un mayor porcentaje de aciertos en el test que las RNA y por tanto, poseen una mayor capacidad de generalización para resolver todos estos problemas.

Además, se ha observado que a medida que aumenta la complejidad de los problemas, así como las arquitecturas para resolverlos, los resultados para las RNGA son significativamente mejores, no sólo en porcentaje de aciertos en test, sino también

en cuanto al error de entrenamiento. El hecho de que la superioridad de las RNGA sea más significativa en problemas más complejos (como Cáncer e Ionosfera), que precisan de redes más complejas para su resolución, parece denotar que la glía artificial es mucho más efectiva en redes complejas que en redes simples. Esto es un hallazgo muy interesante también para la Neurociencia porque establecería un paralelismo con lo que ocurre en los seres vivos, ya que se sabe que la proporción de glía es mayor en las zonas complejas del cerebro en las cuales tienen lugar procesos cognitivos de más alto nivel, además del hecho de que cuánto más complejo es un ser vivo en la escala evolutiva, más glía posee en su sistema nervioso.

Destacar que se llevó a cabo el desarrollo y optimización tanto de la herramienta de simulación como de la plataforma para realizar las simulaciones, ejecutadas en el Centro de Supercomputación de Galicia, así como la implementación de herramientas de análisis automático de los resultados para su posterior estudio. Todos estos desarrollos agilizaron en gran medida los exhaustivos procesos de análisis y comparación realizados.

6.2. Conclusiones de la segunda fase del proyecto

Conjuntamente con toda esta experimentación, se llevó a cabo una aproximación preliminar a la optimización automática de las características inherentes a las redes neurogliales artificiales, de modo que para cada problema se buscase automáticamente los valores más idóneos, tratando de imitar lo que, según los últimos descubrimientos en Neurociencia sobre los astrocitos, se cree que ocurre en el cerebro.

Primeramente, se introdujo una modificación en el AG a utilizar para el entrenamiento supervisado de las redes artificiales, el cruce SBX. Esta modificación conllevó una mejora en los porcentajes de acierto en test y en los tiempos para alcanzarlos, tanto en RNA como en RNGA, en la resolución de problemas de clasificación y predicción.

A continuación, se planteó una nueva metodología, diseño e implementación del software necesario para conseguir la optimización automática de las RNGA. Para esta optimización se emplearon Algoritmos Genéticos Coevolutivos Cooperativos por su extraordinaria adecuación a las necesidades planteadas, pues este nuevo paradigma de aprendizaje emplea la cooperación evolutiva entre poblaciones (de pesos y parámetros) para obtener una configuración óptima de la glía artificial introducida en estos sistemas

conexionistas estudiados. Se realizaron las primeras pruebas con esta nueva metodología, con las que se obtuvieron resultados positivos con dos de los problemas empleados con anterioridad (Flor de Iris y Cáncer de Mama) resueltos mediante RNGA de arquitectura sencilla (una única capa oculta). El sistema desarrollado obtiene, para cada conjunto y población empleados, la mejor combinación de parámetros gliales para la resolución de un problema. Junto con las claras ventajas computacionales que conlleva la optimización, cabe destacar que el funcionamiento del AGCC emula todavía con mayor fidelidad el funcionamiento del cerebro, pues la glía actúa biológicamente de distinto modo según la información recibida del exterior en cada momento por un ser vivo, lo que supone un claro beneficio para la Neurociencia.

Los resultados obtenidos gracias a este proyecto son parte de un trabajo de investigación que podría proporcionar importantes beneficios tanto a la Inteligencia Artificial, al mejorar el rendimiento de Sistemas Conexionistas orientados a clasificación y predicción, como a la Neurociencia, ya que permitirían afianzar los descubrimientos sobre la importante participación de los astrocitos del sistema Glial en el procesamiento de la información en el sistema nervioso.

7. TRABAJOS FUTUROS

Seguidamente se indican los trabajos con los que se continuará a corto plazo y que derivan de los resultados de este proyecto:

- Un posible trabajo futuro consistiría en considerar, de cara a la obtención de los parámetros inherentes a la glía artificial en cada problema, combinaciones que no produjesen ningún efecto, como por ejemplo *activación=4, iteración=2, incremento=decremento=0,* lo que se traduciría en que la glía no actuase. De cara a implementar esta cuestión, caben varias posibilidades. Dado que existen múltiples combinaciones de este tipo (todas aquellas en las que sí participe la glía pero el incremento-decremento sea 0), podría considerarse el controlar su aparición y aunarlas en una única categoría denominada *"NULA",* o también podría permitirse el que estas combinaciones se produjesen aunque no causasen ningún efecto. A la hora de la implementación hay que analizar cuál es el coste en cuanto a tiempo de estas dos aproximaciones para decantarse por una de ellas.

- Emplear un cromosoma por cada elemento de procesado, de forma que cada neurona artificial tuviese su elemento de control (astrocito artificial) con valores concretos de los parámetros para cada caso.

- Debido a la inclusión en el proceso evolutivo de los parámetros codificados en el cromosoma 2, cuyas modificaciones conllevan grandes saltos en el espacio de búsqueda (a diferencia de la modificación de un peso de una conexión), podemos suponer que el espacio de búsqueda relativo a la nueva función de evaluación es fuertemente multimodal. Una de las aproximaciones evolutivas que mayor éxito han demostrado en la resolución de problemas multimodales son las aproximaciones basadas en nichos [DEB 89; SARE 98; PETR 96]. Por lo tanto, un posible trabajo futuro sería la implementación de una aproximación de nichos, manteniendo subpoblaciones según el valor de los genes del cromosoma 2 (o teniendo en cuenta sólo parte de esos genes, por ejemplo, excluyendo los que codifican el porcentaje de incremento/decremento). Con esta aproximación se trata de mantener variabilidad en la población, porque si la selección se

hiciese sólo en función del valor de *fitness*, podrían perderse rápidamente individuos que en principio no tienen un buen *fitness*, pero que cuando se modifiquen los pesos pueden llegar a ser buenos**.**

- Realizar Data Mining de los datos obtenidos por el AGCC con el objetivo de encontrar patrones comunes en las configuraciones de parámetros resultantes. Con ello se conseguiría la configuración óptima que mejor resuelva cada problema concreto. Un posible método para obtener los mejores parámetros de la glía artificial podría consistir en fijar los pesos de la red y evolucionar únicamente los individuos de parámetros. Hasta ahora se ha realizado una coevolución en la que los mejores individuos de parámetros están íntimamente relacionados con (y son dependientes de) los pesos con los que han cooperado.

8. BIBLIOGRAFÍA

[AGUI 98] Aguiar, H.C. & Maciel R. *Modeling and optimization of pulp and paper process usingneural networks*. Comp. Chem. Eng, Vol. 22, 981-984. 1998.

[ALCA 03] Alcalá, R., Casillas, J., Cordón, O. & Herrera, F. *Linguistic modeling with weighted doubled-consequent fuzzy rules based on cooperative Coevolutionary learning*, Integrated Computer-Aided Engineering, 10, 343-355. 2003.

[ALDR 95] Aldrich, C. & Deventer J.S. *Comparison of different artificial neural nets for the detection and location of gross errors in process systems*, Ind. Eng. Chem. Res., Vol. 34, 216-224. 1995.

[ALPA 99] Alpaydin, E. *Combined 5x2cv F test for comparing supervised classification learning algorithms*, Neural Computation, vol. 11, pp. 1885-1892. 1999.

[ALTI 98] Altissimi, R. *Optimal operation of a separation plant using artificial neural networks*. Comp. Chem. Eng., Vol. 22, 939-942. 1998.

[ALVA 07] Alvarellos, A. *Desarrollo y experimentación con nuevos Algoritmos de Entrenamiento y Test en Redes NeuroGliales Artificiales*. Proyecto Fin de Carrera de Ingeniería Técnica en Informática de Sistemas. Universidade da Coruña. 2007.

[ARAQ 99] Araque, A., Púrpura, V., Sanzgiri, R. & Haydon, P. G. *Tripartite synapses: glia, the unacknowledged partner*. Trends in Neuroscience, 22(5). 1999.

[ARAQ 01] Araque, A., Carmignoto, G. & Haydon, P. G. *Dynamic Signaling Between Astrocytes and Neurons*. Annu. Rev. Physiol, 63, 795-813. 2001.

[BLOC 97] Bloch G. *Neural intelligent control for a steel plant*. IEEE Trans. Neural Networks, Vol. 8(4), 910-917. 1997.

[CAJA 04] Ramón y Cajal, S. *Textura del Sistema Nervioso del Hombre y los Vertebrados*. Tomo II. Madrid. 1904.

[CAJA 11] Ramón y Cajal, S. *Histologie du sisteme nerveux de I`homme et des vertebres*. Maloine, Paris. 1911.

[CANT 05] Cantú-Paz E. & Kamath C. *An Empirical Comparison of Combinations of Evolutionary Algorithms and Neural Networks for Classification Problems*, IEEE Transactions on systems, Man and Cybernetics – Part B: Cybernetics, 915-927. 2005.

[CAO 98] Cao, R., Francisco, M., Naya, S., Presedo, M. A., Vázquez, M., Vilar, J. A. & Vilar, J.M. *Estadística Básica Aplicada*. Tórculo Edicións. pp 396 y 434. 1998.

[CASI 01] Casillas Branquero, J. *Aprendizaje Cooperativo para Modelado Lingüístico Flexible Basado en Reglas Difusas: Interpretabilidad y Precisión.* Tesis Doctoral. Universidad de Granada. 2001.

[CESG 09] "*II Curso de Acceso y Utilización del Superordenador Finis Terrae*" impartido con José Carlos Mouriño el 5.03.2009 en el Centro de Supercomputación de Galicia (CESGA).

[CORD 01a] Cordón, O., Herrera, F. & Villar, P. *Generating the Knowledge Base of a Fuzzy Rule-Based System by the Genetic Learning of Data Base.* IEEE Transactions on Fuzzy Systems 9:4, 667-674. 2001.

[CORD 01b] Cordón, O. & Herrera, F. *Hybridizing Genetic Algorithms with Sharing Scheme and Evolution Strategies for Designing Approximate Fuzzy Rule-Based Systems.* Fuzzy Sets and Systems 118:2, 235-255. 2001.

[CORT 95] Cortes, C. & Vapnik, V. *Support-Vector Networks*. Machine Learning, 20. 1995.

[DARW 59] Darwin, C. *On the Origin of Species by Means of Natural Selection*. 1859.

[DEB 89] Deb, K., *Genetic Algorithms in multimodal function optimization*, Master's Thesis, University of Alabama. 1989.

[DEB 95] Deb, K., Agrawal, R.B., *Simulated Binary Crossover for continuous search space*, Complex Systems, 9, 115-148. 1995.

[DEB 98] Deb, K. & Goyal, M. *A robust optimization procedure for mechanical component design based on genetic adaptative search*, Transactions of the ASME: Journal of Mechanical Design, 120 (2), 162-164. 1998.

[DEB 07] Deb, K., Sindhya, K. & Okabe, T. *Self-adaptive simulated binary crossover for real-parameter optimization*. GECCO 2007: 1187-1194. 2007.

[DEJO 75] DeJong, A. K. *An Analysis of the Behaviour of a Class of Genetic Adaptative Systems.* PhD thesis. University of Michigan. 1975.

[MOYA 98] Moya, F. de, Herrero, V. & Guerrero, V. P. *La aplicación de las Redes de Neuronas Artificiales (RNA): a la recuperación de la información*. Anuario SOCADI de Documentación e Información. Barcelona: Sociedad Catalana de Documentación e Información. 1998.

[DIET 98] Dietterich, T.G. *Approximate statistical tests for comparing supervised classification learning algorithms*. Neural Computation, Vol. 10, No. 7, 1895-1924. 1998.

[DOPI 04] Dopico, J.R., Dorado, J., Pazos, A., Gestal, M., Rivero, D. & Pedreira, N. *Search the Optimal RANN Architecture, Reduce the Training Set and Make the Training Process by a Distribute Genetic Algorithm*. Artificial Intelligence and Applications, 1, 415-420. 2004.

[DORA 99] Dorado, J. *Modelo de un Sistema para la Selección Automática en Dominios Complejos, con una Estrategia Cooperativa, de Conjuntos de Entrenamiento y Arquitecturas Ideales de Redes de Neuronas Artificiales Utilizando Algoritmos Genéticos*. Tesis Doctoral. Universidade da Coruña. 1999.

[DORA 00] Dorado, J., Santos, A., Pazos, A., Rabuñal, J.R. & Pedreira, N. *Automatic selection of the training set with Genetic Algorithm for training Artificial Neural Networks*, in: Proc. Genetic and Evolutionary Computation Conference GECCO'2000, Las Vegas, USA, 64-67. 2000.

[ERKM 97] Erkmen, I. & Ozdogan, A. *Short term load forecasting using genetically optimized neural network cascaded with a modified Kohonen clustering process*, in: Proceedings of the IEEE International Symposium on Intelligent Control, pp. 107–112. 1997.

[FREE 93] Freeman J. A. & Skapura D. M., *Neural Networks*. Ed. Addison-Wesley. 1993.

[GOLD 89] Goldberg, D.E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley. 1989.

[GOME 88] Gómez, A., Juristo, N., Montes, C. & Pazos, J. *Ingeniería del Conocimiento*. Editorial Centro de Estudios Ramón Areces. 1988.

[GONZ 98] González-García R., Rico-Martínez R. & Kevrekidis G. *Identification of distribuid parameter systems: A neural net based approach*, Comp. Chem. Eng, Vol. 22, 965-968. 1998.

[GOSS 08] Gosset, W. S. *The probable error of a mean*, Biometrika 6(1): 1-25. 1908

[GRID 09] http://wiki.gridengine.info/wiki/index.php/Simple-Job-Array-Howto. Marzo 2009.

[HAYD 01] Haydon, P. G. *Glia: listening and talking to the synapse*. Nat. Rev. Neurosci., 2, 185-93. 2001.

[HAYD 02] Haydon, P.G. & Araque, A. *Astrocytes as modulators of synaptic transmission.* In Volterra A, Magistretti P, Haydon (eds.) Tripartite Synapses: Synaptic transmission with glia. PG. Oxford University Press. pp: 185-198. 2002.

[HAYK 99] Haykin, S. *Neural Networks* (2nd ed.). Englewood Cliffs, NJ: Prentice Hall. 1999.

[HEBB 49] Hebb, D. O. *The Organization of Behaviour.* J. Wiley. N.Y. 1949.

[HERR 04] Herrera F., Hervás C., Otero J. & Sánchez L. *Un estudio empírico preliminar sobre los tests estadísticos más habituales en el aprendizaje automático.* En R. Giraldez, J.C. Riquelme, J.S. Aguilar, eds., Tendencias de la Minería de Datos en España, Red Española de Minería de Datos y Aprendizaje (TIC2002-11124-E), 403-412. 2004.

[HILE 95] Hilera, J.R. & Martínez, V.J. *Redes neuronales artificiales. Fundamentos, modelos y aplicaciones.* Addison-Wesley Iberoamericana S.A., España. 1995.

[HOLL 75] Holland, J.H. *Adaptation in Natural and Artificial Systems*. University of Michigan Press. Ann Arbor, MI, USA. 1975.

[HOPF 82] Hopfield, J. J. *Neural networks and physical systems with emergent collective computational properties.* Proceedings of the National Academy of Sciences of the USA, Vol. 79, No. 8, 2554-2558. 1982.

[HOPF 89] Hopfield, J.J. & Tank, D. *Neural architecture and biophysics for sequence recognition in Neural models of plasticity.* Academic Press, 363-377. 1989.

[IBÁÑ 08] Ibáñez, O. *Sistemas Conexionistas y Modelos Computacionales Neurobiológicos.* Trabajo Tutelado DEA. Dep. TIC. Facultad de Informática. Universidade da Coruña. 2008.

[ISAS 04] Isasi, P. & Galván I.M. *Redes de Neuronas Artificiales. Un Enfoque Práctico.* Pearson Prentice-Hall. España. 2004.

[KINN 94] Kinnebrock, W. *Accelerating the standard backpropagation method using a genetic approach*, Neurocomputing 6 pp. 583–588. 1994.

[KOHO 88] Kohonen, T. *Self-Organization and Associative Memory*, Springer-Verlag, New York, second edition. 1988.

[KOZA 92] Koza, J. R. *Genetic Programming: On the Programming of Computers by Jeans of Natural Selection*. Cambridge, MA: MIT Press. 1992.

[KUWA 86] Kuwada, J.Y. *Cell recognition by neuronal growth cones in a simple vertebrate embryo.* Science, 233. 740-746. 1986.

[LAPE 88] Lapedes, A. & Farber, R. *How Neural Nets Works*. Neural Information Processing Systems. Eds. D.Z. Anderson. American Institute of Physics. Pp. 442-456. 1988.

[LARG 96] Largo, C., Cuevas, P., Somjen, G.G., Martin del Rio, R. & Herreras, O. *The effect of depressing glial function in rat brain in situ on ion homeostasis, synaptic transmission, and neuron survival*. J. Neurosci., 16, 1219-1229. 1996.

[LEE 96] Lee, S.-K. & Jang, D. *Translation, rotation and scale invariant pattern recognition using spectral analysis and hybrid genetic neural-fuzzy networks*, Comput. Ind. Eng., vol. 30, no. 3, pp. 511–522. 1996.

[LERO 94] Le Roux P.D. & Reh T.A. *Regional differences in glial-derived factors that promote dendritic outgrowth from mouse cortical neurons in vitro*. Journal of Neuroscience, 14, 4639-4655. 1994.

[LEVI 93] Levin A.U. & Narendra K.S. *Control of nonlinear dynamical systems using neural networks: Controllability and Stabilization*, IEEE Trans. Neural Networks, Vol. 4, No. 2, 192-206. 1993.

[MART 06] Martín del Brío, B. & Sanz Molina, A. *Redes Neuronales y Sistemas Borrosos*, 3ª edición, editorial RA-MA, pág. 19. 2006.

[MAUC 01] Mauch, D.H., Nagler, K., Schumacher, S., Goritz, C., Muller, E.C., Otto, A. & Pfrieger, F.W. *CNS synaptogenesis promoted by gliaderived cholesterol*. Science, 294, 1354-1357. 2001.

[MCCU 43] McCulloch, W.S. & Pitts, W. *A logical Calculus of Ideas Immanent in Nervous Activity*. Bulletin of Mathematical Biophysics, No. 5, 115-133. 1943.

[MENN 94] Mennerick, S. & Zorumski, C.F. *Glial contribution to excitatory neurotransmission in cultured hippocampal cells*. Nature, *368* 59-62. 1994.

[MEER 98] Meert, K. & Rijckaert, M. *Intelligent modelling in the chemical process industry with neural networks: A case study*, Comp.Chem.Eng., Vol. 22, 587-593. 1998.

[MICH 96] Michalewicz, Z. *Genetic algorithms + data structures = evolution programs*. Springer-Verlag, Heidelberg, Alemania, tercera edición. 1996.

[MOOR 02] Moore, D. S. & McCabe, G. P. *Introduction to the Practice of Statistics*, W. H. Freeman, 4th edition. Cap. 14. 2002.

[MORI 08] Morillo, A. *Contrastes No Paramétricos (II)* Universidad de Málaga. http://campusvirtual.uma.es/morillas/CNOPARAII.pdf. 2008.

[NADK 07] Nadkarni, S. & Jung, P. *Modeling synaptic transmission of the tripartite synapse*. Physical Biology, 4, 1-9. 2007.

[OHM. 06] http://ohm.utp.edu.co/neuronales/main.htm. Noviembre 2006.

[OLLE 98] Oller C.A. & Guidici R. *Neural Network based approach for optimisation applied to an industrial nylon-6,6 polymerisation process*, Comp. Chem. Eng., Vol. 22, 595- 600. 1998.

[ORR 98] Orr, G.B. & Müller, K.R. *Neural networks: tricks of the trade.* Springer-Verlag. 1998.

[PARE 95] Paredis J. *Coevolutionary computation*. Artificial Life 2: 355–375. 1995

[PAZO 96] Pazos A. *Redes de Neuronas Artificiales y Algoritmos Genéticos*. Ed. Tórculo. 1996.

[PAZO 99] Pazos, A., Dorado, J., Santos, A., Rabuñal, J.R. & Pedreira, N. *Training of Recurrent ANN with Time Decreased Activation by GA to the Forecast in Dynamic Problems*, in: Proc. 3rd World Multiconference on

Systemics, Cybernetics and Informatics SCI'99 and the 5th Internat. Conf. on Information Systems Analysis and Synthesis ISAS 99. IEEE Computer Society. Orlando, USA 463-469. 1999.

[PEÑA 01] Peña-Reyes C. A. & Sipper M. *Fuzzy CoCo: a cooperative coevolutionary approach to fuzzy modeling*. Por aparecer en IEEE Transactions on Fuzzy Systems. Versión preliminar disponible en http://lslwww.ep_.ch/»penha/. 2001.

[PERE 02] Perea, G. & Araque, A. *Communication betwen astrocytes and neurons: a complex language*. Journal of Physiology. Paris: Elsevier Science. 2002.

[PERE 04] Perea, G. & Araque, A. *Properties of synaptically evoked astrocyte calcium signal reveal synaptic information processing by astrocytes*. Journal of Neuroscience. 2004.

[PERE 07] Perea, G., & Araque, A. *Astrocytes Potentiate Transmitter Release at Single Hippocampal Synapses*. Science, 317(5841), 1083 – 1086. 2007.

[PERT 01] Pértega Díaz, S. & Pita Fernández, S. *Métodos paramétricos para la comparación de dos medias. t de Student.* Unidad de Epidemiología Clínica y Bioestadística. Complexo Hospitalario Juan Canalejo. A Coruña. Cad Aten Primaria. 8: 37-41. 2001.

[PETR 96] Petrowski, A. *A clearing procedure as a niching method for genetic algorithms*. Proceedings of IEEE International Conference on Evolutionary Computation. Nagoya, Japan, page(s): 798-803. 1996.

[PFRI 97] Pfrieger, F.W. & Barres, B.A. Synaptic efficacy enhanced by glial cells in vitro. *Science*,277, 1684-1687. 1997.

[PORT 04] Porto, A. *Modelos Computacionales para optimizar el Aprendizaje y el Procesamiento de la Información en Sistemas Adaptativos: Redes Neurogliales Artificiales (RR.NG.AA.)*. Tesis Doctoral. Universidade da Coruña. A Coruña. 2004.

[PORT 05a] Porto, A., Pazos, A. & Araque, A. *Artificial Neural Networks based on Brain Circuits Behaviour and Genetic Algorithms*. Lecture notes in computer science. Berlín (Alemania). LNCS 3512 pp. 99-106. 2005.

[PORT 05b] Porto, A. & Pazos, A. *NeuroGlial Behaviour in Computer Science. Artificial Neural Networks in Real-Life Applications*. Ed. Idea Group Inc. Hershey PA, USA 1-30. 2005.

[PORT 07] Porto, A., Araque, A., Rabuñal, J., Dorado, J. & Pazos, A. *A New Hybrid Evolutionary Mechanism Based on Unsupervised Learning for Connectionist Systems*. Neurocomputing. ISSN '0925-2312'. 13 págs. Ed. Elsevier. 2007.

[POTT 00] Potter M. A. & Jong K. A. D. *Cooperative coevolution: an architecture for evolving coadapted subcomponents*. Evolutionary Computation 8(1): 1–29. 2000.

[PREC 98] Prechelt, L. *Early Stopping – But when?* En [ORR 98] pp 55-70. 1998.

[PRES 97] Pressman, R. S. *Ingeniería del Software. Un enfoque práctico*. McGraw Hill. 1997.

[RABU 02] Rabuñal, J. *Metodología para el Desarrollo de Sistemas de Extracción de Conocimiento en Rna*. Tesis de Licenciatura. Dep. Computación. Facultad de Informática. Universidade da Coruña. 2002.

[RABU 98] Rabuñal, J. *Entrenamiento de Redes de Neuronas Artificiales con Algoritmos Genéticos*. Tesis de Licenciatura. Dep. Computación. Facultad de Informática. Universidade da Coruña. 1998.

[RABU 04] Rabuñal, J.R., Dorado, J., Pazos, A., Pereira, J. & Rivero, D. *A New Approach to the Extraction of Rules from ANN and their Generalization Capacity through GP*, Neural Computation, 16 nº 7. 1483-1523. 2004.

[RAFF 93] Raff, M.C., Barres, B.A., Burne, J.F. & Coles, H.S. Ishizaki & Jacobson, M.D. *Programmed cell death and the control of cell survival: lessons from the nervous system*. *Science, 262*, 695-700. 1993.

[RAKI 90] Rakic, P. *Principles of neural cell migration*, Experientia, 46, 882-891. 1990.

[RIVE 07a] Rivero D., Rabuñal J., Dorado J. & Pazos A. *Automatically Design of ANNs by means of GP for Data Mining tasks: Iris Flower Classification Problem, Adaptive and Natural Computing Algorithms*, 8th International Conference, ICANNGA 2007, Warsaw, Poland, April 2007, Proceedings, 276-285. 2007.

[RIVE 07b] Rivero, D., *Desarrollo y Simplificación de Redes de Neuronas Artificiales mediante Técnicas de Computación Evolutiva.* Tesis de doctorado de Daniel Rivero Cebrián. Universidade da Coruña. 2007.

[ROSI 97] Rosin C. D. & Belew R. K. *New methods for competitive coevolution.* Evolutionary Computation, vol 5 (1), pp. 1–29. 1997.

[RUME 86] Rumelhart D.E., Hinton G.E. & Williams R.J. *Learning internal representations by error propagation*. En D. E. Rumelhart, J.L. McClelland, eds., Parallel Distributed Processing: Explorations in the Microstructures of Cognition, Cambridge, MA: MIT Press, Vol. 1, 318-362. 1986.

[SARE 98] Sareni, B. & Krahenbuhl, L. *Fitness sharing and niching methods revisited*. IEEE Transactions on Evolutionary Computation, Volume 2, Issue 3, page(s): 97-106. 1998.

[SCHA 92] Schaffer, J.D., Whitley, D. & Eshelman, L. *Combination of Genetic Algorithms and Neural Networks: the state of the art*, Combination of Genetic Algorithms and Neural Networks, IEEE Computer Society. 1992.

[SCHA 90] Schaffer, J.D., Caruana, R.A. & Eshelman, L.J. *Using genetic search to exploit the emergent behavior of neural networks*, in S. Forrest (Ed.), *Emergent computation*, pp. 244-248. 1990.

[SHEN 06] Shen, X. & De Wilde, F. *Long-term neuronal behavior caused by two synaptic modification mechanisms*. Department of Computer Science, Heriot-Watt University, United Kingdom. 2006.

[STON 78] Stone M. *Cross-validation: A review Matemastische Operationsforschung Statischen*, Serie Statistics, Vol. 9, 127-139. 1978.

[SUN 09] http://wikis.sun.com/display/gridengine62u2/Submitting+Array+Jobs. Marzo 2009.

[TAHA 95] Taha, M.A. & Hanna, A.S. *Evolutionary neural network model for the selection of pavement maintenance strategy*, Transportation Res. Rec. 1497 pp70–76. 1995.

[T500 08] http://www.top500.org/list/2008/11/100. Noviembre 2008.

[VEIG 08] Veiguela, N. *Modelización Computacional de Sistemas Conexionistas y el Sistema Glial*. Trabajo Tutelado DEA. Dep. TIC. Facultad de Informática. Universidade da Coruña. 2008.

[VILA 03] Vilar, J.M. *Modelos Estadísticos Aplicados*, Servicio de Publicacións da Universidade da Coruña. pp 15-16. 2003.

[WALL 55] Wallace, A.R. *On the Law Which Has Regulated the Introduction of New Species*. 1855.

[WALL 58] Wallace, A.R. & Darwin, C. *On the Tendency of Species to form Varieties; and on the Perpetuation of Varieties and Species by Natural Means of Selection*. 1858.

[WALP 98] Walpole, R. E., Myers, R. H. & Myers, S. L. *Probabilidad y Estadística para Ingenieros*, Ed. Prentica Hall, 6ª edición. 614-617. 1998.

[WANG 98] Wang H., Oh Y. & Yoon E.S. *Strategies for modeling and control of nonlinear chemical process using neural networks*, Comp. Chem. Eng., Vol. 22, 823-826. 1998.

[WASS 89] Wasserman, P. D. *Neural Computing*. Ed. Van Nostrand Reinhold, New York. 1989.

[WETZ 83] Wetzel, A. *Evaluation of the Effectiveness of Genetic Algorithms in Combinational Optimization.* University of Pittsburg. 1983

[WHIT 95] Whitley, D. *Genetic Algorithms and Neural Networks*, in Genetic Algorithms in Engineering and Computer Science. 1995.

[WILC 45] Wilcoxon, F. *Individual Comparisons by Ranking Methods*. Biometrics 1. pp 80-83. 1945

[YAN 97] Yan, W., Zhu, Z. & Hu, R. *Hybrid genetic/BP algorithm and its application for radar target classification*, en: Proc. IEEE National

Aerospace and Electronics Conf. NAECON Part 2 of 2, pp. 981-984. 1997.

[YAO 99] Yao, X. *Evolving Artificial Neural Networks*, en: Proc. IEEE, Vol. 87, nº 9, pp. 1423-1447. 1999.

[ZHAN 95] Zhang, P., Sankai, Y. & Ohta, M. *Hybrid adaptative learning control of nonlinear system*, en: Proc. American Control Conf. Part 4 of 6, pp. 2744-2748. 1995.