Universidad Central “Marta Abreu” de Las Villas

Facultad de Matemática, Física y Computación



Trabajo para optar por el Título de

Máster en Ciencia de la Computación

Algoritmo para mejorar la convergencia en Mapas Cognitivos Difusos Sigmoidales

Autor

Lic. Gonzalo R. Nápoles Ruiz

Tutor

Dr. Rafael Bello Pérez

2014

El que subscribe, Gonzalo R. Nápoles Ruiz, hace constar que el trabajo titulado “Algoritmo para mejorar la convergencia en Mapas Cognitivos Difusos Sigmoidales” se realizó en la Universidad Central “Marta Abreu” de Las Villas como parte de la culminación de los estudios de la Maestría en Ciencia de la Computación, autorizando que el mismo sea utilizado por la institución, para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos ni publicado sin la autorización de la Universidad.

_____

Firma del Autor

Los abajo firmantes, certificamos que el presente trabajo ha sido realizado según acuerdos de la dirección de nuestro centro y que el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____

Firma del Tutor
Dr. Rafael Bello Pérez

_____

Firma del Jefe de Seminario
Dr. Carlos Morell Pérez

A mis tres pilares

AGRADECIMIENTOS

Algo que he aprendido a través de los años es que un resultado científico es un conglomerado de esfuerzos y este trabajo no es una excepción. Por eso me gustaría a gradecer a:

- Mi tutor Prof. Dr. Rafael Bello, sin dudas un ejemplo admirable de profesional, de líder y no de jefe, de académico, de ser humano y de amigo.
- La Prof. Dra. María Matilde García por su empeño, dedicación y paciencia.
- Mi abuela, que a sus 90 años se empeñó en aprender Inteligencia Artificial.
- Mi mamá por su fe en mi persistencia, y al resto de mi familia sin excepción.
- La que primero fuera mi ilusión desesperada, luego fuera mi amiga y más recientemente mi brazo derecho, mi tranquilizante natural y mi compañera, Isel Grau.
- Libia, Ricardo, Edilio y Magaly, mucho de lo que soy y seré se los debo a ellos.
- I want to express my gratitude to Prof. Dr. Koen Vanhoof, Prof. Dr. Yiannis Boutalis and Prof. Dr. Elpiniki Papageorgiou for their kind support during this research.
- Todos mis amigos en general, y por supuesto a Ciego Montero.
- A mi gato Timoteo por apartarme unos minutos del trabajo.

"*No one has found a stability result for such hierarchical fuzzy dynamical systems*"

Bart Kosko, 1998

RESUMEN

Los Mapas Cognitivos Difusos (MCD) son estructuras conexionistas recurrentes especialmente útiles para modelar y simular sistemas dinámicos. Desde su introducción en 1984 por Bart Kosko, se han propuesto varios algoritmos de aprendizajes supervisados y no supervisados para ajustar los parámetros que definen el comportamiento de estas estructuras basadas en el conocimiento. Sin embargo, los mapas ajustados que utilizan funciones continuas (o construidos por expertos) regularmente no son capaces de converger a un estado de equilibrio, haciendo imposible la toma de decisiones. Recientemente se han publicado métodos analíticos para determinar condiciones de convergencia en MCD Sigmoidales, pero los resultados no son aplicables en contextos de predicción, toma de decisiones o clasificación de patrones.

Este trabajo propone un algoritmo de aprendizaje heurístico para mejorar las propiedades de convergencia en MCD Sigmoidales, una vez que se han estimado las relaciones de causalidad. La idea general consiste en estimar una función sigmoidal para cada concepto del mapa, en lugar de utilizar la misma función para todas las neuronas. El algoritmo se reduce a un problema de optimización donde se calcula la inclinación adecuada para cada función de transformación, minimizando la variabilidad en las respuestas del sistema para todas las instancias (datos históricos). Para solucionar este problema se adopta un enfoque basado en Inteligencia Colectiva, particularmente la Optimización de Enjambres de Partículas, y además se describe un operador para corregir estados de no progreso, típicos en espacios multimodales.

ABSTRACT

Fuzzy Cognitive Maps (FCM) are recurrent connectionist structures specially designed for modeling and simulating dynamic systems. During the last thirty years several supervised and non-supervised learning algorithms for tuning their parameters have been proposed. However, adjusted maps that use continuous transformation functions (or maps designed by experts) are frequently unable to converge to a fixed-point attractor. As a result, decision-making is practically impossible. It is remarkable that FCM having discrete neurons never exhibit chaotic states, but this premise cannot be guaranteed for maps having continuous concepts. Recently several analytical methods that ensure convergence conditions in Sigmoid FCM have been introduced, but these results cannot be used in pattern classification problems.

This work introduces a heuristic learning algorithm for improving the convergence properties in Sigmoid FCM, once the phase concerning the estimation of the causal weight matrix is complete. The main idea of this procedure is to compute a custom threshold function for each map concept, instead of using the same transformation function for all neurons. It leads to a continuous optimization problem where the adequate inclination of each sigmoid function needs to be estimated. So we can expect that the global variability on the system responses (considering a set of example) decrease during the inference process. For addressing this challenging problem we adopt an approach based on Swarm Intelligence, particularly Particle Swarm Optimization. Moreover, we introduce a novel clearing operator for handling potential non-progress states which are common situations in complex multimodal solution spaces.

TABLA DE CONTENIDOS

INTRODUCCIÓN 1

Objetivo general 3
Objetivos específicos 3
Tareas de investigación 3
Hipótesis de investigación 3

CAPÍTULO 1. TEORÍA DE MAPAS COGNITIVOS DIFUSOS 4

1.1 Mapas Cognitivos Difusos 4
1.2 Proceso de inferencia de los Mapas Cognitivos Difusos 7
1.3 Algoritmos de aprendizaje para Mapas Cognitivos Difusos 9
1.3.1 Algoritmos de aprendizaje Hebbianos 9
1.3.2 Algoritmos de aprendizaje basados en metaheurísticas 12
1.4 Estabilidad en Mapas Cognitivos Difusos 15
1.5 Aplicaciones de los Mapas Cognitivos Difusos 18
1.6 Conclusiones parciales 20

CAPÍTULO 2. ALGORITMO PARA MEJORAR LA CONVERGENCIA EN MAPAS COGNITIVOS DIFUSOS SIGMOIDALES 21

2.1 Preliminares sobre el algoritmo 21
2.2 Descripción de la propuesta 23
2.3 Interpretación del método de aprendizaje 28
2.4 Selección del optimizador 30
2.4.1 Optimización de Enjambres de Partículas 30
2.4.2 PSO con topologías estructuradas 33
2.5 Operador para mejorar la diversidad 35
2.5.1 Detección de estados de no progreso 35
2.5.2 Reorganización de la población 37
2.6 Conclusiones parciales 39

CAPÍTULO 3. EVALUACIÓN DEL ALGORITMO DE APRENDIZAJE 40

3.1 Evaluación del rendimiento del optimizador 40
3.2 Modelación de proteínas del VIH-1 usando MCD Sigmoidales 45
3.3 Evaluación del algoritmo de aprendizaje 48
3.4 Comparación con otros enfoques 53
3.5 Conclusiones parciales 55

CONCLUSIONES 56

RECOMENDACIONES 57

REFERENCIAS BIBLIOGRÁFICAS 58

ANEXO A. PSEUDOCÓDIGOS DE LOS ALGORITMOS ENUNCIADOS 61

ANEXO B. CARACTERÍSTICAS DE LA PLATAFORMA *FCM TOOL* 62

INTRODUCCIÓN

Los Mapas Cognitivos Difusos (MCD) fueron diseñados por (Kosko, 1984) como una extensión de los Mapas Cognitivos (Axelrod, 1976). Desde el punto de vista conexionista, los MCD son modelos recurrentes representados por grafos dirigidos pesados. En este esquema cada concepto denota una variable o estado del sistema que se pretende modelar; mientras que los pesos en las conexiones determinan la causalidad entre los nodos. Debido a su flexibilidad para representar sistemas dinámicos, los MCD se han utilizados en diferentes áreas, por ejemplo: en sistemas de supervisión y control, en problemas de clasificación, en el análisis de riesgo, en la medicina, en problemas de visión artificial, en el análisis de proteínas, en problemas de toma de decisiones, en sistemas autónomos, etc. (Papageorgiou and Salmeron, 2013).

Existen dos estrategias fundamentales para diseñar un MCD: manual o automática. En la primera variante los expertos determinan los conceptos que describen el sistema, así como la dirección e intensidad de las conexiones causales entre las neuronas. En la segunda variante la topología del mapa y la matriz de pesos se construyen a partir de datos históricos, o ejemplos representativos que definen el comportamiento del sistema. Independientemente de la estrategia, los expertos del dominio necesitan definir cuál es la arquitectura que mejor se ajusta al problema que se pretende modelar, así como las restricciones inherentes al modelo. Hasta donde es conocido, no existe una topología única capaz de modelar óptimamente cualquier problema.

Las limitaciones concernientes a la arquitectura de los MCD han reducido la usabilidad de estas estructuras en problemas reales. Por ejemplo, resulta tedioso para un ser humano estimar el valor exacto de los pesos causales, sobre todo en sistemas con docenas de variables. Para corregir estos problemas en la literatura se han propuesto varios algoritmos de aprendizaje, orientados a refinar los parámetros que caracterizan la modelación original del sistema.

En el artículo de revisión (Papageorgiou, 2012) la autora clasifica los enfoques existentes en tres grupo principales, de acuerdo al objetivo final del algoritmo:

- Producción de matrices de pesos utilizando datos históricos.
- Adaptación de las relaciones causa-efecto basada en la intervención de expertos.
- Producción de matrices de pesos para MCD combinado los datos históricos disponibles y la intervención de expertos del dominio de aplicación.

A pesar de su éxito, la formulación de algoritmos de aprendizaje para estimar la causalidad del sistema (la matriz de pesos causales que regulan la interacción entre los conceptos) es todavía un problema abierto para la comunidad científica (Papageorgiou, 2012). Por ejemplo, las funciones de evaluación utilizadas en el aprendizaje son frecuentemente no convexas y caracterizadas por espacios de búsqueda multimodales (Nápoles et al., 2013c), donde los métodos de aprendizajes son propensos a converger prematuramente hacia soluciones locales. En síntesis, el diseño de un MCD requiere procedimientos más escalables y robustos, que sean capaces de detectar posibles soluciones locales y donde no se requiera la intervención del experto.

Una vez que se han estimado los valores causales entre los conceptos (manualmente o de forma automatizada) el experto puede simular la dinámica del sistema investigado. En la primera etapa las neuronas son activadas, y luego el proceso de inferencia actualiza de forma iterativa el valor de activación de cada concepto. Este algoritmo finaliza cuando el mapa converge a un estado de equilibrio (el mapa produce la misma salida a través del tiempo) o se alcanza un número máximo de iteraciones. En el primer caso se dice que el mapa es estable, mientras que el segundo caso el mapa puede tener un comportamiento cíclico o caótico. Los estados inestables son frecuentes en sistemas continuos, limitando la eficiencia de estas estructuras neuronales en problemas de toma de decisiones o clasificación de patrones (Nápoles et al., 2014a).

Hasta donde el autor conoce, no existe un procedimiento que permita producir matrices de pesos estables, sin afectar la exactitud del sistema resultante. Incluso en (Kosko, 1998) el autor afirma que este problema podría permanecer muchos años sin una solución matemática precisa, aunque recientemente se han publicado algunos trabajos que establecen condiciones de convergencia en sistemas con funciones de activación continuas (ej. variantes sigmoidales).

Por ejemplo, en (Boutalis et al., 2009) los autores demostraron que si la matriz de pesos satisface ciertas premisas entonces el mapa siempre convergerá a un único punto fijo, independientemente del estado inicial de los conceptos. De forma similar, (Knight et al., 2014) demostraron que si la inclinación de la función de activación sigmoidal es suficientemente pequeña, entonces el mapa tendrá una única solución linealmente estable. Sin embargo, estos resultados no son aplicables en contextos de clasificación de patrones, sino a MCD usados en tareas de simulación, y las razones serán argumentadas durante el desarrollo del presente trabajo.

Objetivo general

Desarrollar un algoritmo de aprendizaje supervisado basado en técnicas de Inteligencia Colectiva para mejorar la convergencia en los Mapas Cognitivos Difusos Sigmoidales usados en problemas de clasificación de patrones, predicción o de toma de decisiones.

Objetivos específicos

1. Diseñar un método para mejorar la convergencia en Mapas Cognitivos Difusos Sigmoidales, que no reduzca la capacidad del mapa para predecir nuevos patrones.
2. Implementar el método como un problema de optimización continuo, potencialmente capaz de solucionar estados de convergencia prematura y estancamiento.
3. Evaluar el desempeño del método a través de un caso de estudio, concerniente a la predicción de la resistencia a fármacos de la proteína *proteasa* del VIH-1.

Tareas de investigación

1. Realizar una revisión sobre los algoritmos de aprendizaje más representativos (supervisados y no supervisados) para Mapas Cognitivos Difusos.
2. Identificar las principales deficiencias y potencialidades de los enfoques analíticos existentes para estabilizar Mapas Cognitivos Difusos continuos.
3. Modelar el nuevo algoritmo de aprendizaje como un problema de optimización.
4. Seleccionar una metaheurística poblacional adecuada como optimizador.
5. Proponer un operador que permita mejorar el rendimiento de la metaheurística en espacios de búsqueda multimodales, donde son frecuentes los estados de estancamiento.
6. Evaluar el desempeño del algoritmo en problemas de predicción y clasificación de patrones, usando un caso de estudio real concerniente al campo de la Bioinformática.

Hipótesis de investigación

Las propiedades de estabilidad de los Mapas Cognitivos Difusos Sigmoidales utilizados en tareas de clasificación de patrones mejoran notablemente si se usa una función de umbral ajustada para cada neurona, en lugar de utilizar la misma función para todos los conceptos.

CAPÍTULO 1. TEORÍA DE MAPAS COGNITIVOS DIFUSOS

En este capítulo se aborda la teoría de MCD desde su introducción en 1984 por Bart Kosko. Este capítulo además incluye valoraciones sobre el proceso de inferencia y los criterios relacionados con la convergencia. Además, se hace una revisión sobre los principales métodos de aprendizaje reportados en la literatura, así como sus ventajas y deficiencias. Finalmente, con el objetivo de ejemplificar las potencialidades de los MCD, se discuten varios problemas de aplicación que han sido solucionados usando estas estructuras basadas en el conocimiento.

1.1 Mapas Cognitivos Difusos

La teoría de Mapas Cognitivos fue originalmente propuesta por (Axelrod, 1976) en un intento de obtener una estructura basada en el conocimiento más comprensible. En términos generales, un mapa cognitivo puede ser caracterizado por un conjunto de conceptos y relaciones causales entre los nodos. Los conceptos representan variables, entidades o estados que usualmente describen el sistema que se pretende modelar en un dominio determinado. Por su parte las relaciones causales denotan conexiones entre los conceptos. Cada relación tiene un signo asociado para expresar la dirección de la causalidad. Más explícitamente, se cumple que:

- Si el signo de la conexión es positivo, entonces una variación en el concepto *causa* (concepto inicial) provocará una variación en el concepto *efecto* en la misma dirección.
- Si el signo de la relación es negativo, entonces una variación en el concepto *causa* provocará una variación en el concepto *efecto* en la dirección opuesta.

Los Mapas Cognitivos han sido aplicados a varios problemas reales (Eden, 1988) con un notable éxito, aunque la capacidad de representación de las relaciones causales en un sistema es bastante limitada. Es complicado modelar la semántica de un sistema complejo a partir de la dirección de las relaciones de causalidad. Para solucionar este inconveniente (Kosko, 1984) introdujo la teoría de Mapas Cognitivos Difusos, donde a cada conexión del mapa se le asigna un peso que denota el grado de *causalidad*. Esta característica permite implementar conocimientos concernientes a la fortaleza de cada relación causal. Concretamente, las conexiones en un MCD tienen asociadas un valor numérico en el rango $[-1,1]$ que regula la dirección e intensidad de la relación causal entre los conceptos que definen el sistema (Kosko, 1986). Note que una relación se puede representar por un término difuso, por ejemplo: débil, medio, fuerte o muy fuerte.

De forma general la interpretación de las influencias causales entre los conceptos de un MCD es similar a la semántica de los Mapas Cognitivos. Para dos conceptos C_i y C_j se cumple que:

- Si el peso w_{ij} asociado a la conexión entre los conceptos C_i y C_j es mayor que cero, entonces un incremento (decremento) en el concepto C_i provocará un incremento (decremento) en el concepto C_j de forma proporcional al valor absoluto de w_{ij}.
- Si el peso w_{ij} asociado a la conexión entre los conceptos C_i y C_j es menor que cero, entonces un incremento (decremento) en el concepto C_i provocará un decremento (incremento) en el concepto C_j de forma proporcional al valor absoluto de w_{ij}.
- Si el peso w_{ij} asociado a la conexión entre los conceptos C_i y C_j es igual (o muy próximo) a cero, entonces no existe relación causal entre los dos conceptos.

Desde el punto de vista matemático, un MCD puede ser definido usando una 4-tupla (C, W, A, f) donde $C = \{C_1, C_2, C_3, \dots, C_N\}$ representa el conjunto de conceptos del grafo, $W: (C_i, C_j) \rightarrow w_{ij}$ es una función que asocia un valor causal w_{ij} a cada par de neuronas (C_i, C_j). Este valor w_{ij} denota la dirección e intensidad de la arista que conecta al concepto C_i con la neurona C_j donde la matriz de pesos W define el comportamiento del sistema (Kosko, 1988). De forma similar, $A: (C_i) \rightarrow A_i$ es una función que asocia un grado de activación $A_i \in \mathbb{R}$ a cada concepto C_i del sistema, durante el tiempo t $(t = 1,2, \dots, T)$. Además, el mapa utiliza una función $f: \mathbb{R} \rightarrow [0,1]$ para mantener el grado de activación de cada concepto en el rango adecuado.

El grado de activación de una neurona es un indicador del nivel de presencia del concepto en el sistema (Kosko, 1992). Esta característica es un aspecto clave en la interpretabilidad y usabilidad de los MCD, sobre todo si la función de umbral es continua (dada la importancia de esta función para la presente propuesta sus propiedades serán analizadas más adelante). La Figura 1.1 muestra la apariencia de un MCD y la semántica en la activación de una neurona.

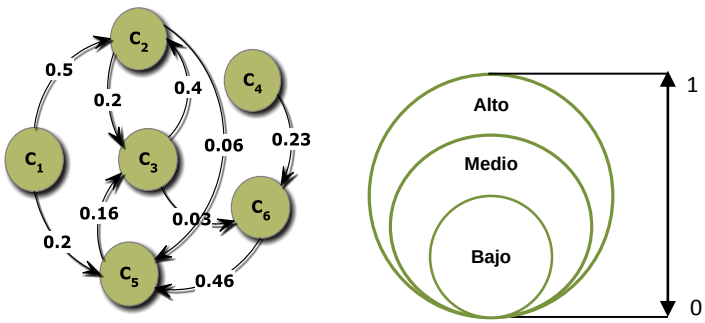


Figura 1.1 Mapa Cognitivo Difuso. Nivel de activación de un concepto.

Otro aspecto importante en un MCD es su carácter difuso. Los MCD no explotan las ventajas de la Lógica Difusa en el proceso de propagación de la información, más bien estos principios son usados durante la construcción del mapa. De esta forma los conceptos y relaciones del mapa son representados por variables difusas expresadas como términos lingüísticos. Más explícitamente, durante la etapa de ingeniería del conocimiento cada experto expresa la relación que existe entre cada par de conceptos C_i y C_j del mapa. Entonces, para cada relación causal se obtienen k reglas con la siguiente estructura: **SI** C_i es A **ENTONCES** C_j es B y el peso w_{ij} es C.

Luego se utiliza el método del centroide y el mecanismo de inferencia Mamdani para agregar las k reglas, y el valor defuzificado es el valor de la relación. Por ejemplo, suponga que los términos lingüísticos asociados a la causalidad son: *muy bajo*, *bajo*, *medio*, *alto* y *muy alto*. La Figura 1.2 muestra las funciones de pertenencia para dos expertos del dominio. El primer experto considera que si el concepto origen C_i se activa totalmente, entonces el concepto efecto C_j tendrá un grado de activación *alto* con certeza 0.6. Por su parte, el segundo experto considera que bajo la misma premisa, el nivel de activación del concepto C_j será *muy alto* con certeza 0.3. Esto implica que el valor defuzificado para el peso w_{ij} será 0.76. De forma similar se estima el resto de la matriz de pesos, para todas las conexiones causales que intervienen en el modelo.

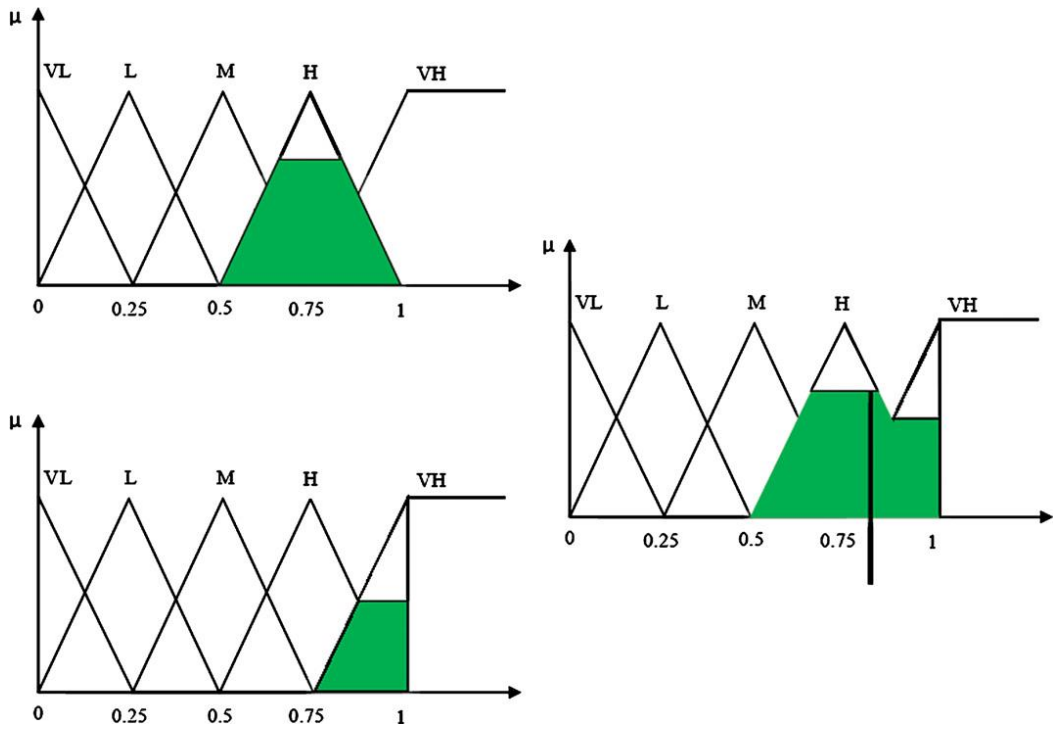


Figura 1.2 Proceso de construcción de un MCD donde intervienen dos expertos del dominio. El primer experto considera que el efecto de C_i sobre C_j es *alto*, mientras que el segundo considera que es *muy alto*.

1.2 Proceso de inferencia de los Mapas Cognitivos Difusos

Sin perder generalidad, el proceso de inferencia en un MCD se puede definir matemáticamente usando dos componentes: un vector de estado $A_{1\times N}$ que representa el grado de activación de los conceptos del mapa, y una matriz de pesos causales $W_{N\times N}$ la cual define la interacción entre las neuronas. La siguiente ecuación (1.1) resume este proceso, el cual consiste en calcular el vector de estado A a través del tiempo, para una condición inicial A^0. De forma análoga a otros sistemas neuronales, la activación de C_i dependerá de la activación las neuronas que inciden directamente sobre el concepto C_i y de los pesos causales asociados a dicho concepto.

$$A_i^{(t+1)} = f\left(\sum_{j=1}^{N} w_{ji} A_j^{(t)}\right), i \neq j \tag{1.1}$$

Desde el punto de vista algebraico la fase de propagación de un estímulo se reduce a sucesivas multiplicaciones del vector de estado por la matriz de pesos, hasta que se alcance un criterio de parada. Por lo tanto, una variación en un concepto puede afectar directa o indirectamente a todo el mapa (Kosko, 1988). Esto es esencial en escenarios de simulación, pues permite a los expertos estudiar el comportamiento del sistema para diferentes condiciones iniciales.

Dada su naturaleza recurrente, el sistema modelado por un MCD evolucionará durante el tiempo, donde la activación de cada neurona dependerá del grado de activación de sus antecedentes en la iteración anterior. Normalmente este proceso se repite hasta que el sistema estabilice o se alcance un número máximo de iteraciones (Kosko, 1992). Los sistemas inestables pueden ser totalmente caóticos o cíclicos, y son frecuentes en modelos continuos. En resumen, el proceso de inferencia en un MCD puede mostrar una de las siguientes características:

- **Estados de estabilidad**: si $\exists t_k \in \mathbb{N} : A_i^{(t+1)} = A_i^{(t)}, \forall t > t_k$. En otras palabras, después de la iteración t_k el MCD producirá el mismo vector de estado. Desde luego, esta configuración es ideal, pues representa la codificación de un patrón oculto en la causalidad.
- **Estados cíclicos** (Wang et al., 1990) si $\exists t_k, P \in \mathbb{N} : A_i^{(t+P)} = A_i^{(t)}, \forall t > t_k$. El mapa tiene un comportamiento cíclico con periodo P. En este caso el sistema producirá el mismo vector de estado cada P-ciclos del proceso de inferencia.
- **Estados caóticos** (Wang et al., 1990): el mapa produce un vector de estado diferente en cada ciclo, o sea, los conceptos siempre varían su valor de activación.

En (Tsadiras, 2008) se concluye que la convergencia de un MCD está estrechamente relacionada con la función de transformación utilizada para mantener el valor de activación de las neuronas en el intervalo adecuado. De acuerdo al tipo de salida, estas funciones pueden ser clasificadas en dos grandes categorías: discretas y continuas. Las funciones más frecuentemente utilizadas son: la función binaria, la función de tres estados y las variantes logísticas o sigmoidales (Bueno and Salmeron, 2009). Las dos primeras funciones son discretas y nunca inducirán estados caóticos, pero esta premisa no se puede generalizar para las variantes sigmoidales.

Más explícitamente, los MCD son sistemas determinísticos y por tanto, si el sistema produce un estado previamente visitado, entonces el mapa entrará en un ciclo cerrado. Por ejemplo, si se usa una función binaria el sistema producirá a lo sumo 2^N estados diferentes (asumiendo que N es el número de nodos del grafo). Esto significa que, en el peor escenario, el mapa alcanzará un estado visitado cada 2^N iteraciones. En el caso de la función de tres estados, el ciclo máximo tiene un periodo de 3^N, y por tanto el sistema nunca será caótico. Claramente esto no ocurre si se adoptan funciones de salida continua, donde el espacio de estados es un hipercubo.

Sin embargo, no es más factible utilizar funciones discretas, pues se ha demostrado que los MCD tienen mayor capacidad de inferencia si se usan funciones de transformación continuas (Tsadiras, 2008). Incluso en (Bueno and Salmeron, 2009) los autores demostraron que la variante sigmoidal tiene mayor capacidad predictiva, si se usa el mismo sistema de decisión. Independientemente de estos resultados, resulta conveniente destacar que la selección de la función frecuentemente está condicionada por los requerimientos del sistema, o sea, por rol que desempeñe cada concepto en el sistema modelado. En (Tsadiras, 2008) se discuten varias reglas para la selección adecuada de este componente. Algunas de estas ideas son resumidas a continuación:

- **Los MCD binarios** (MCD con función de activación binaria) son apropiados para problemas altamente cualitativos, donde solamente se requiere representar el incremento o estabilidad de los conceptos que describen los estados del sistema.
- **Los MCD de tres estados** (MCD con función de activación de tres estados) son apropiados para problemas cualitativos, donde el experto puede representar el incremento, decremento o estabilidad de los conceptos que describen los estados del sistema.
- **Los MCD Sigmoidales** (MCD con función de transformación sigmoidal) son convenientes para modelar sistemas cualitativos y cuantitativos, donde es preciso modelar el decremento, incremento o estabilidad de los conceptos del sistema.

Otra característica que depende del sistema investigado está relacionada con la propagación de la información. Según la ecuación (1.1) un concepto recibe la influencia de las neuronas conectadas directamente con él, pero no considera el hecho de que el propio concepto puede ser antecedente de su propia activación en cada iteración. La ecuación (1.2) soluciona este problema, confiriendo al mapa una capacidad memorística adicional. Sin embargo, esta variante en ocasiones conduce a neuronas sobresaturadas, limitando la capacidad para predecir nuevos patrones.

$$A_i^{(t+1)} = f\left(\sum_{j=1}^{N} w_{ji}A_j^{(t)} + w_{ii}A_i^{(t)}\right), i \neq j \tag{1.2}$$

Independientemente de la variante usada, el impacto que un concepto ejerce sobre otro se aprecia paulatinamente en cada iteración al calcular principalmente los efectos directos, ya que los nodos indirectos tienen un retardo (en número de iteraciones) igual a la cantidad de arcos que integren la ruta más corta. Los MCD tienden a obtener un valor único para el efecto total de los conceptos asociados directa e indirectamente a un concepto particular, a diferencia de los mapas causales que obtienen uno por cada neurona origen que tiene una relación directa o indirecta con él, por lo que se genera un vector de efectos en vez de un escalar.

1.3 Algoritmos de aprendizaje para Mapas Cognitivos Difusos

El objetivo principal de los algoritmos de aprendizajes para MCD es estimar una matriz de pesos que maximice el rendimiento del sistema. Este proceso es equivalente al ajuste de las conexiones causales entre los conceptos (similar a las relaciones sinápticas en una Red Neuronal Artificial). De forma general los algoritmos existentes se agrupan en tres grupos: los algoritmos Hebbianos, los enfoques poblacionales y los métodos de aprendizaje híbridos.

1.3.1 Algoritmos de aprendizaje Hebbianos

Los métodos Hebbianos son procedimientos no supervisados, y por tanto no requieren múltiples datos históricos etiquetados, donde se conoce el valor de los rasgos objetivos. El primer método de aprendizaje para MCD fue presentado por (Dickerson and Kosko, 1994) para ajustar los pesos causales. El algoritmo Hebbiano Diferencial (*Differential Hebbian Learning*, DHL) se basa en el siguiente principio: si el concepto causa C_i y el concepto efecto C_j cambian el valor de activación de forma simultánea, entonces el peso causal w_{ij} se incrementará con un factor constante, en otro caso la relación de causalidad no será modificada en esa iteración.

La diferencia en el valor de activación ΔA_i del concepto i es: $\Delta A_i = A_i^{(t+1)} - A_i^{(t)}$. Este proceso se ejecuta iterativamente hasta que se obtenga una configuración deseada o se alcance un número máximo de iteraciones. La ecuación (1.3) resume el principio de este algoritmo de aprendizaje; donde el coeficiente $\gamma(t) = 0.1[1 - (t/1.1q)]$ denota el incremento que modificará los valores causales en el proceso de ajuste. En este caso el parámetro $q \in \mathbb{N}$ debe seleccionarse de forma tal que el peso w_{ij} se mantenga en el rango $[-1,1]$. Un criterio válido para seleccionar el valor del parámetro q puede ser el número máximo de iteraciones del algoritmo.

$$w_{ij}^{(t+1)} = \begin{cases} w_{ij}^{(t)} + \gamma(t)\left[\Delta A_i \Delta A_j - w_{ij}^{(t)}\right] & si\ \Delta A_i \neq 0 \\ w_{ij}^{(t)} & si\ \ \Delta A_i = 0 \end{cases} \tag{1.3}$$

La principal desventaja de este enfoque local radica en la ausencia de información concerniente al sistema como un todo. En otras palabras, DHL modifica los pesos teniendo en cuanta cambios en el valor de activación entre los pares de conceptos, ignorando la influencia proveniente de las demás neuronas. Para atenuar este problema, (Huerga, 2002) introdujo el Algoritmo Diferencial Balanceado (*Balanced Differential Algorithm*, BDA). La idea central consiste en usar el grado de activación de los nodos que se modifican simultáneamente, durante el proceso de actualización de la matriz de pesos. La siguiente ecuación (1.4) formaliza este método, asumiendo un MCD sin conceptos con auto-conexiones, el cual se ajusta de forma iterativa.

$$w_{ji}^{(t+1)} = \begin{cases} w_{ji}^{(t)} + \gamma(t)\left[\dfrac{\Delta A_i/\Delta A_j}{\sum \Delta A_i/\Delta A_k} - w_{ji}^{(t)}\right] & si\ \ \Delta A_i \Delta A_j > 0,\ \ \forall C_k: \Delta A_i \Delta A_k > 0 \\ w_{ji}^{(t)} + \gamma(t)\left[\dfrac{-\Delta A_i/\Delta A_j}{\sum \Delta A_i/\Delta A_k} - w_{ji}^{(t)}\right] & si\ \ \Delta A_i \Delta A_j < 0,\ \ \forall C_k: \Delta A_i \Delta A_k < 0 \end{cases} \tag{1.4}$$

Si hay un cambio en el valor de activación del nodo C_j en la misma dirección que el concepto C_i ($\Delta A_i \Delta A_j > 0$) entonces w_{ji} se asume como una relación positiva, en otro caso ($\Delta A_i \Delta A_j < 0$) se asume una relación de causalidad negativa. En este algoritmo, el peso w_{ji} se ajusta a través del cociente $\Delta A_i/\Delta A_j$. Este valor representa una medida del factor de cambio del concepto C_i cuando el concepto C_j cambia durante el tiempo. Note que la normalización del factor permite calcular el ajuste de la causalidad teniendo en cuenta el valor relativo del cambio de la neurona C_j, respecto al valor de activación de los demás conceptos que influencian al concepto C_i. El algoritmo BDA es más poderoso que su predecesor, aunque no es muy popular (hasta donde se conoce solo se ha utilizado en problemas de reconocimiento de patrones binarios).

En otro intento por ajustar la matriz de pesos causales (Papageorgiou et al., 2004) introdujeron el Algoritmo Hebbiano Activo (*Active Hebbian Learning*, AHL). Los algoritmos anteriores asumen que en cada iteración las neuronas se actualizan de forma síncrona, sin embargo, AHL considera que los nodos se activan de forma asíncrona, siguiendo una secuencia. A diferencia de los demás algoritmos, AHL modifica todos los pesos casuales en cada iteración (exceptuando los pesos de la diagonal principal), aunque el experto debe especificar tres elementos:

- Definición del conjunto de conceptos del modelo.
- Definición de la secuencia de activación de los conceptos.
- Definición de la estructura inicial de las relaciones causales.

Si estas especificaciones son conocidas, entonces AHL es capaz de estimar la matriz de pesos a través de en un procedimiento iterativo de siete pasos (Papageorgiou et al., 2004). La principal desventaja de este método es su dependencia del criterio de expertos.

Adicionalmente, (Papageorgiou et al., 2003a) presentaron un Aprendizaje Hebbiano No-Lineal (*Nonlinear Hebbian Learning*, NHL), el cual es una extensión no lineal de ley Hebbiana. La idea central del método es modificar los pesos que fueron inicialmente especificados por los expertos, hasta que satisfaga un criterio de parada. En este esquema el signo de cada conexión corresponde con la interpretación física del sistema y constituye una restricción del modelo. De forma general el experto del dominio tiene que especificar los siguientes aspectos:

- Definición del conjunto de conceptos (conceptos de salida y conceptos de entrada).
- Especificación del rango de valores que puede tomar un concepto.
- Especificación del signo de cada conexión.

En otras palabras, el algoritmo debe respetar la dirección de la causalidad así que sólo se ajustará la magnitud de la interacción entre las neuronas. La ecuación (1.5) formaliza cómo se ajustan los pesos en este modelo, donde γ es un factor que denota la disminución del peso, η es la velocidad del aprendizaje y $sgn(.)$ es la función signo. Adicionalmente el algoritmo introduce dos criterios de paradas novedosos. En el primer caso se finaliza cuando se alcance una solución próxima a la respuesta deseada, mientras que el segundo caso el algoritmo finaliza cuando se detecte un punto atractor de equilibrio. NHL es un algoritmo muy eficiente debido a su simplicidad, y su principal desventaja radica en la dependencia al criterio inicial de los expertos.

$$w_{ij}^{(t+1)} = \gamma w_{ij}^{(t)} + \eta A_j^{(t)} \left(A_i^{(t)} - sgn\left(w_{ij}^{(t)} \right) A_j^{(t)} w_{ij}^{(t)} \right) \tag{1.5}$$

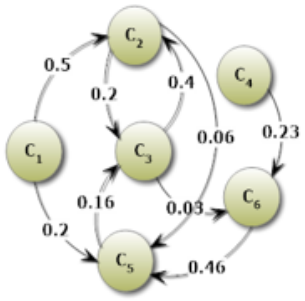
De forma general los algoritmos Hebbianos son procedimientos no supervisados diseñados para ajustar la matriz de pesos causales y que utilizan un único ejemplo en su esquema de aprendizaje. Sin embargo, el objetivo de estos métodos no es maximizar la capacidad del mapa para predecir nuevos patrones (ej. exactitud de la clasificación) sino lograr que mapa converja a un punto fijo y que la salida corresponda con un escenario determinado. Este esquema se enfoca principalmente en problemas de simulación. Recientemente se han publicado algunos trabajos que exploran una nueva arista de estas estructuras neuronales: la clasificación de patrones. Para este contexto estos algoritmos de aprendizaje no resultan adecuados, pues no es suficiente asegurar la convergencia del sistema a un atractor, también debe considerarse la exactitud.

En un intento de aliviar el problema de la precisión del mapa (Stach et al., 2008) propusieron una variante del método NHL. El nuevo modelo, llamado NHL Guiado por Datos (*Data-driven NHL*, DDNHL), utiliza el mismo principio de aprendizaje que su predecesor, y conjuntamente utiliza datos históricos para mejorar la calidad del sistema resultante. En (Stach et al., 2008) los autores demostraron a través de varios experimentos que DDNHL es superior a NHL si existen los datos históricos, pero en un estudio más exhaustivo (Papakostas et al., 2012) se concluyó que incluso esta variante tiene un desempeño pobre en problemas de clasificación.

1.3.2 Algoritmos de aprendizaje basados en metaheurísticas

En literatura se han propuesto varios algoritmos de aprendizaje para computar la matriz de pesos de un MCD utilizando metaheurísticas poblaciones. La idea central de este enfoque consiste en generar soluciones que minimicen una función de error, o sea, las diferencias entre las respuestas conocidas y las salidas inferidas por mapa. Estos métodos son más costosos computacionalmente pues requieren optimizar una función objetivo, y adicionalmente requieren múltiples secuencias de entrada para el proceso de entrenamiento. Una secuencia usualmente codifica la activación de los conceptos de entrada del mapa y la respuesta conocida para ese estímulo.

En este esquema cada individuo denota una matriz de pesos candidata, donde cada solución en el espacio de búsqueda representará un conjunto de pesos causales. Por tanto, un sistema definido por N neuronas se codificará en un vector con dimensión a lo sumo $(N \times N)$, pues las relaciones causales con peso cero no son modificadas. Más concretamente, cada dimensión del individuo se asocia con una conexión causal. La Figura 1.3 ilustra la idea general de esta codificación para un mapa parcialmente conexo que involucra seis conceptos de entrada.



	C_1	C_2	C_3	C_4	C_5	C_6
C_1	0	0.5	0	0	0.2	0
C_2	0	0	0.2	0	0.06	0
C_3	0	0.4	0	0	0	0.03
C_4	0	0	0	0	0	0.23
C_5	0	0	0.16	0	0	0
C_6	0	0	0	0	0.46	0

0.5	0.2	0.2	0.06	0.4	0.03	0.23	0.16	0.46

Figura 1.3 Codificación de un individuo en un algoritmo de apendizaje poblacional.

Inspirados en este enfoque, (Koulouriotis et al., 2001) utilizaron un método basado en Estrategia Evolutiva para ajustar la estructura del mapa. Este algoritmo utiliza datos históricos, donde cada instancia está compuesta por un par de vectores. El primer vector codifica el grado de activación de los conceptos de entrada, mientras que el segundo vector denota la respuesta del sistema para la configuración especificada. La ecuación (1.6) formaliza la función objetivo asociada, donde K es el número total de instancias, A_i denota la respuesta actual del mapa, y $\tilde{A}_i$ es la salida esperada para el i-ésimo concepto. A diferencia de los métodos Hebbianos, este modelo requiere múltiples secuencias de entrada para estimar las relaciones causales del sistema.

$$f(x) = \sum_{k=1}^{K} \sum_{i=1}^{N} \left| A_i - \tilde{A}_i \right| \tag{1.6}$$

Siguiendo un lógica similar, (Parsopoulos et al., 2003) aplicaron Optimización de Enjambres de Partículas (*Particle Swarm Optimization*, PSO) para estimar la estructura adecuada del sistema a partir de una secuencia de múltiples vectores de estado. La peculiaridad de este método radica en su esquema de aprendizaje: la respuesta del sistema la definen nodos decisores. Un experto debe establecer los límites de activación $A_i^{\min}$ y $A_i^{\max}$ permitidos para cada nodo decisión. Durante el progreso del algoritmo se explora el espacio de búsqueda con el objetivo de encontrar una matriz de pesos que satisfaga las restricciones impuestas por el experto. La siguiente ecuación formaliza la función objetivo asociada, donde H representa la función *Heaviside* (Parsopoulos et al., 2003), mientras que A_i^{DC} denota el i-ésimo concepto decisor del sistema.

$$f(x) = \sum_{i=1}^{N} H(A_i^{min} - A_i^{DC}) \left| A_i^{min} - A_i^{DC} \right| + \sum_{i=1}^{N} H(A_i^{DC} - A_i^{max}) \left| A_i^{DC} - A_i^{max} \right| \tag{1.7}$$

En (Stach et al., 2005) los autores propusieron un enfoque automatizado basado en Algoritmos Genéticos con Codificación Real (*Real-Coded Genetic Algorithm*, RCGA). La idea del método consiste en maximizar la función objetivo $f(x) = 1/(1 + \alpha * E(x))$, reduciendo el error global del mapa (véase ecuación 1.8). En esta ecuación α denota el coeficiente de calidad del algoritmo, $A_i(t)$ es la respuesta conocida en el tiempo t, mientras que $\tilde{A}_i(t)$ denota la respuesta del sistema usando la matriz de pesos candidata, para un estímulo inicial. Este método es capaz de computar mapas con alta capacidad de predicción, sin embargo, su rendimiento decrece significativamente cuando el número de variables en el sistema aumenta ligeramente.

$$E(x) = \frac{1}{N(T-1)} \sum_{t=1}^{T-1} \sum_{i=1}^{N} \left| A_i(t) - \tilde{A}_i(t) \right| \tag{1.8}$$

En este procedimiento la intervención de expertos no se requiere, de hecho, este método es capaz de estimar una configuración adecuada utilizando una única instancia, aunque también se pueden utilizar múltiples secuencias de entrada a través del tiempo.

Más recientemente, en (Chen et al., 2012) los autores propusieron un novedoso método inspirado en Optimización de Colonias de Hormigas (*Ant Colony Optimization*, ACO) para ajustar mapas con docenas de conceptos. En este modelo cada peso se codifica como una secuencia de estados discretos, o sea, la parte entera y las cifras de precisión decimal. Durante el proceso de búsqueda, cada hormiga construye una solución. En el próximo paso, cada solución discreta se transforma a su equivalente continua, resultando en una matriz de pesos causales. La ecuación (1.9) formaliza la función de evaluación que se minimiza en este procedimiento.

$$E(x) = \frac{1}{NK(T-1)} \sum_{k=1}^{K} \sum_{i=1}^{N} \sum_{t=1}^{T-1} \left(A_i^k(t) - \tilde{A}_i^k(t) \right)^2 \tag{1.9}$$

Note que estos métodos no pueden garantizar la estabilidad de los mapas utilizados en problemas de clasificación de patrones. Más explícitamente, en estos problemas sólo se conoce la respuesta esperada para un vector de estímulo inicial. Esto significa que el objetivo del algoritmo se reduce a minimizar las diferencias entre las respuestas del mapa y las salidas esperadas, sin considerar la variabilidad del sistema durante el proceso de inferencia. Como consecuencia se obtienen mapas con alta capacidad de predicción, pero frecuentemente inestables.

1.4 Estabilidad en Mapas Cognitivos Difusos

Los algoritmos de aprendizaje ilustrados en la sección anterior no consideran la convergencia del mapa final, y esta observación es extensiva a otras propuestas. En otras palabras, la mayoría de los métodos de aprendizaje para MCD tienen como objetivo la estimación de los pesos causales, ignorando las propiedades de estabilidad del sistema final. En la literatura se han reportado pocos estudios analíticos sobre la convergencia de los MCD, principalmente debido a la complejidad del problema matemático asociado. En esta sección se abordan los resultados más prometedores en esta dirección, y se explica por qué no son aplicables en todos los casos.

En (Kosko, 1998) el autor formula un complejo procedimiento analítico basado en la función de Liapounov para determinar las condiciones necesarias y suficientes que aseguran la convergencia de los *Sistemas Aditivos Difusos Generalizados*. Estos sistemas comparten varios rasgos con los MCD, sin embargo, el autor concluye que los resultados no son generalizables a otros sistemas difusos. Incluso Kosko afirma que nadie ha encontrado un resultado concerniente a la estabilidad en MCD, y que este problema permanecerá insoluble en los próximos cientos de años, aunque el esfuerzo de los matemáticos en esta dirección aún persiste.

Por otra parte, (Boutalis et al., 2009) presentaron dos teoremas (ver Teoremas 1 y 2) sobre la estabilidad de MCD Sigmoidales. El primer teorema está orientado a los MCD Sigmoidales sin entradas[1] y asegura que si la matriz de pesos satisface ciertas condiciones, entonces existirá un único estado de equilibrio, independientemente del estímulo inicial. La demostración analítica de este interesante resultado se basa en las propiedades de las contracciones en un espacio métrico completo (Rudin, 1964). La idea general consiste en determinar condiciones matemáticas sobre la matriz de pesos causales de forma tal que no se perturbe la propiedad de contracción inherente a la función sigmoidal que garantizan la unicidad del punto de equilibrio.

Aparentemente el Teorema 1 soluciona el problema de la inestabilidad en los MCD Sigmoidales sin neuronas de entrada, pues sería suficiente determinar una matriz que satisfaga las condiciones necesarias del teorema. Sin embargo, la aplicabilidad de este resultado se reduce, en el mejor de los casos, a problemas de simulación y representación del conocimiento.

[1] Los autores definen los conceptos de entrada como neuronas que influencian a las demás, pero ellas no son influenciadas por las demás neuronas. En otras palabras, el valor de activación de estos conceptos permanecen invariable durante el proceso de inferencia del mapa definido por la ecuación (1.1).

Teorema 1. Existe una y solo una solución para cualquier valor de activación A_i de cualquier MCD Sigmoidal si es cierta la siguiente expresión:

$$\left(\sum_{i=1}^{N}\|w_i\|\right)^{1/2} < 4 \tag{1.10}$$

Para comprender las limitaciones del teorema anterior, suponga un problema de clasificación con dos posibles clases de decisión, donde 1/5 de las instancias están asociadas a la primera clase, mientras que el resto están asociadas a la segunda clase. De acuerdo a los resultados del teorema, si la matriz de pesos satisface (1.10) entonces el mapa convergerá a un único estado atractor de equilibrio, sin importar el vector de estímulo inicial. Esto significa que los objetos siempre serán clasificados de la misma forma sin considerar la dirección o intensidad de las relaciones casuales que definen el sistema, o las características del nuevo objeto.

La exactitud en la clasificación para este problema particular será 80% o 20%, dependiendo de la clase asociada al atractor encontrado. En cualquier caso es incuestionable que el sistema no tiene capacidad de predicción alguna, pues las instancias siempre serán ubicadas en la misma clase, independientemente de la naturaleza de la instancia. Esto sugiere que el teorema no es aplicable en contextos donde se requiere la simulación de múltiples estímulos.

Por su parte, el Teorema 2 es levemente diferente pues asume un MCD Sigmoidal con neuronas de entrada. En este caso, el estado de equilibrio dependerá de las características de la matriz de pesos y del vector de estímulo. En otras palabras, el teorema asegura que existirá un único punto fijo para cada instancia del problema. Además se presenta un algoritmo de aprendizaje basado en el método del gradiente para calcular una matriz causal capaz de producir una salida acorde a la respuesta conocida, y que no comprometa la estabilidad del sistema. A pesar de ser un resultado muy prometedor, tampoco es aplicable a problemas de predicción.

Teorema 2. Para un MCD Sigmoidal con m conceptos de entrada, existe una y solo una solución para cualquier valor de activación A_i si es cierta la siguiente expresión:

$$\left(\sum_{i=1}^{N}\|w_{m+i}\|\right)^{1/2} < 4 \tag{1.11}$$

En otras palabras, el algoritmo de adaptación de los pesos estima una matriz de pesos estable que satisface la respuesta del sistema esperada, para una única instancia del problema. Si el problema tiene múltiples instancias (como es típico en problemas de aprendizaje automatizado) entonces el algoritmo debe ser aplicado sucesivamente. Como resultado se obtiene un conjunto diferente de pesos para cada instancia del problema, y por lo tanto no hay aprendizaje.

Otro resultado teórico fue publicado en (Knight et al., 2014) donde los autores demuestran que el número de atractores de un MCD Sigmoidal está determinado por la inclinación de la función de umbral (Teorema 3). Desde el punto de vista analítico este teorema es una ligera variación de los Teoremas 1 y 2, pero enfocado a las propiedades que debe satisfacer la función de normalización para que el mapa sea linealmente estable. Según los autores, asegurar la existencia y unicidad del punto fijo en un MCD conduce a sistemas más confiables, incluso cuando el grado de activación de las neuronas no es cuantificado de forma precisa por su carácter difuso.

Teorema 3. El número de soluciones de (1.1) depende de la inclinación λ de la función.

a) Si $\lambda > 0$ es suficientemente pequeño, entonces el mapa tendrá una única solución, y esta solución será linealmente estable.
b) Si $\lambda > 0$ es suficientemente grande, entonces el mapa tendrá múltiples soluciones, y muchas de estas soluciones pueden ser linealmente estable.

Al igual que los otros teoremas, este resultado no es aplicable a MCD utilizados en problemas de clasificación de patrones o toma de decisiones (Nápoles et al., 2014a). Por ejemplo, supongamos el modelo basado en MCD para la toma de decisiones concerniente a la transportación pública de pasajeros ciudades desarrolladas (León et al., 2013). En este modelo el sistema debe predecir el medio de transporte más cercano a las preferencias del experto, conociendo el valor de un grupo de variables contextuales (ej. temperatura, disponibilidad de transporte propio). Pero si el sistema converge al mismo atractor de equilibrio, entonces el mapa no será capaz de distinguir entre dos decisiones, no importa cuán diferentes sean las situaciones iniciales.

En resumen, el principal problema de los teoremas discutidos en esta sección no está relacionado con la unicidad del atractor de equilibrio, sino con los criterios de convergencia utilizados en las demostraciones (ej. radio espectral de la matriz Jacobina). Estos criterios aseguran que el atractor existe y es único, pero resulta el mismo estado de equilibrio para todos los objetos del problema, lo cual es un enfoque incorrecto en problemas de clasificación de patrones.

1.5 Aplicaciones de los Mapas Cognitivos Difusos

Los MCD han sido ampliamente utilizados en varios dominios de aplicación; algunos ejemplos incluyen: embebidos en sistemas de control, en problemas de toma de decisiones, en análisis de riesgo, en categorización de textos, en problemas de predicción, en problemas ingenieriles, para modelar procesos de negocios, en reconocimiento de patrones, etc. De hecho, en los últimos años las investigaciones relacionadas con los MCD ha aumentado considerablemente.

La Tabla 1 resume el número de estudios, revistas/capítulos de libros y conferencias relacionadas con la teoría de los MCD en la última década (2000-2010)[2]. En esta década se desarrollaron 512 estudios relacionados con los MCD. Note que el número de estudios reportados en el año 2010 fue seis veces superior a los estudios reportados en el año 2000. Este breve resumen confirma la creciente aceptación los MCD por la comunidad científica internacional.

Tabla 1.1 Investigaciones relacionadas con los MCD reportadas en SCOPUS (2000-2010).

Año	Estudios	Libros	Revistas
2000	18	8	10
2001	22	2	20
2002	7	4	3
2003	19	9	10
2004	53	15	38
2005	35	16	19
2006	50	11	39
2007	56	19	37
2008	81	21	60
2009	84	26	58
2010	102	61	41

Por ejemplo, (Papakostas et al., 2008) implementó un algoritmo basado en MCD para solucionar problemas de reconocimiento de patrones. En su trabajo, el autor utilizó una variante híbrida que explotaba las principales ventajas de los MCD y de las Redes Neuronales. En este caso la matriz de pesos fue ajustada utilizando un enfoque genético. Para validar el nuevo modelo se utilizaron cuatro problemas de clasificación de patrones. Las simulaciones mostraron que los porcientos de clasificación obtenidos eran superiores a los reportados por otros algoritmos. Además, el nuevo clasificador fue exitosamente aplicado a un problema real de visión artificial, resaltando de esta forma la utilidad de los MCD en este campo de aplicación.

[2] La tabla fue tomada de un artículo de revisión publicado por (Papageorgiou et al, 2011). Los datos corresponden a una consulta a SCOPUS realizada el 20 de febrero de 2011 por la autora.

Otro enfoque interesante es el uso de los MCD en el análisis de escenarios relacionados con la transportación de pasajeros en ciudades modernas. En (León et al., 2013) los autores propusieron un Ingeniero del Conocimiento Automatizado para construir la topología del mapa (conceptos, relaciones y valor de las conexiones); de la misma forma se obtuvieron secuencias de entradas y respuestas (datos históricos) asociados a la preferencia de los expertos. En una segunda etapa, la matriz de pesos inicial fue ajustada utilizando los datos históricos disponibles y un algoritmo de aprendizaje inspirado Optimización de Enjambres de Partículas. En este sentido, la simulación de los escenarios podría ayudar a la formulación de políticas más cercanas a las preferencias de los usuarios, explotando más eficientemente las infraestructuras existentes.

Más recientemente (Nápoles et al., 2013b) propusieron un MCD para estudiar el comportamiento de algunas proteínas del Virus de Inmunodeficiencia Humano (VIH). En este modelo cada nodo del mapa se representa como una posición de la secuencia proteínica, mientras que se utiliza un concepto adicional para medir el grado de resistencia de cada mutación observada a un fármaco determinado. La topología supone que todos los nodos están relacionados y que todos influyen sobre la resistencia. Para estimar las relaciones causales entre las posiciones de la secuencia, se adoptó una variante de Optimización de Enjambres de Partículas.

Como principal contribución de este trabajo, los autores determinaron patrones mutacionales que podrían ayudar a comprender el comportamiento del virus. Por otra parte, un estudio posterior (Grau et al., 2013) demostró que los MCD Sigmoidales son clasificadores potentes para este tipo de problemas, superando a otros clasificadores tradicionales (ej. Redes Neuronales, Máquinas de Soporte Vectorial, Árboles de Decisión, Redes Bayesianas, entre otros).

Los MCD también se han aplicado con éxito en problemas de apoyo a la toma de decisiones y para solucionar problemas de clasificación en la medicina. En (Papageorgiou et al., 2003b) los autores propusieron un MCD para modelar la adecuada selección del tratamiento de radiación en terapias de ionización. Para estimar la matriz de pesos, se utilizó un algoritmo de aprendizaje basado en AHL, reportándose resultados prometedores. Por otra parte, (Kannappan et al., 2011) desarrollaron un modelo capaz de predecir desórdenes de autismo en niños con riesgo, donde las relaciones causales se estimaron utilizando el método NHL. Para relacionar otras aplicaciones de los MCD el lector puede consultar (Papageorgiou, 2011) y (Papageorgiou, 2012).

1.6 Conclusiones parciales

Los MCD son una técnica basada en el conocimiento que permite el razonamiento causal y la simulación de complejos sistemas dinámicos. Usualmente son construidos de forma manual (por expertos del dominio) o de forma automática. Un aspecto central durante la etapa de ingeniería del conocimiento es la estimación de la dirección e intensidad de las relaciones causales entre las variables del mapa, lo cual determina la efectividad y eficiencia de la modelación. Para lidiar con este problema se han desarrollado varios algoritmos de aprendizajes:

- No supervisados (ej. algoritmos Hebbianos).
- Supervisados (ej. inspirados en metaheurísticas).

A pesar de su éxito relativo, los algoritmos de aprendizaje existentes no consideran cuestiones de estabilidad como parte de su esquema, solo incluyen aspectos relacionados con la precisión final del proceso de inferencia. Como consecuencia se obtienen sistemas con capacidad de predicción comparable con los clasificadores tradicionales (ej. Redes Bayesianas, Árboles de Decisión) pero esta precisión es engañosa si el sistema no es estable. Sin embargo, garantizar la convergencia en un MCD no es trivial, sobre todo en mapas que utilizan funciones continuas donde el dominio de excitación de las neuronas está definido sobre un hipercubo difuso.

Por otra parte, los métodos analíticos que garantizan la existencia y unicidad del punto atractor de equilibrio no pueden ser aplicados en contextos donde se requiere la simulación de múltiples secuencias de entrada, sino en problemas donde existe un único estímulo. Estos métodos limitan significativamente la capacidad de inferencia de los MCD pues no consideran los ejemplos en su esquema de convergencia. De forma general, el problema de la estabilidad en MCD Sigmoidales persiste como un problema para los matemáticos y la ausencia de una solución exacta evidencia la necesidad de enfoques alternativos inspirados en técnicas aproximadas.

CAPÍTULO 2. ALGORITMO PARA MEJORAR LA CONVERGENCIA EN MAPAS COGNITIVOS DIFUSOS SIGMOIDALES

En este capítulo se introduce un nuevo algoritmo de aprendizaje para mejorar las propiedades de convergencia en MCD Sigmoidales una vez que se ha estimado la causalidad. El método usa una función sigmoidal para cada neurona del mapa, en lugar de usar la misma función de umbral para todos los conceptos. En la práctica, el procedimiento se reduce a la estimación de la inclinación adecuada para cada función sigmoidal, lo cual conduce a problema de optimización continuo que puede ser solucionado usando diversas estrategias. De esta forma es posible obtener mapas más estables, sin afectar la capacidad para clasificar nuevos patrones.

2.1 Preliminares sobre el algoritmo

En el capítulo anterior se revisaron varios métodos analíticos que permiten encontrar condiciones que garanticen la estabilidad en los MCD Sigmoidales. El principal problema de estos modelos está relacionado con la reducción de la capacidad de predicción del sistema. Más explícitamente, después de aplicar estos algoritmos, el mapa es capaz de converger a un estado de equilibrio que es único para todas las instancias del problema, sin importar la naturaleza del vector de estímulo codificado en los casos de entrada. Esto implica que las nuevas instancias serán clasificadas de la misma forma, por lo que solo se preserva la modelación del sistema.

¿Qué sentido práctico tiene un sistema que siempre converja a la misma solución? En problemas de aprendizaje automatizado estos criterios no son útiles. Sin embargo, la unicidad del punto fijo es un criterio de convergencia poderoso. Por ejemplo, el método de Gauss-Seidel es un algoritmo numérico iterativo para resolver sistemas de ecuaciones lineales. Este método requiere un vector inicial como aproximación de la solución. La incorrecta estimación de este vector regularmente causa la divergencia del algoritmo (Burden and Faires, 2002).

Como alternativa se han propuesto varios criterios analíticos para seleccionar el vector inicial de una forma segura. Por ejemplo, el Teorema 4 establece condiciones de convergencia a partir del radio espectral de la matriz de Gauss-Seidel asociada al procedimiento.

Teorema 4. Para cualquier aproximación inicial $x^{(0)} \in \mathbb{R}$, la sucesión definida por el algoritmo de Gauss-Seidel converge a la solución única del sistema de ecuaciones lineales $Ax = b$ si y solo si el radio espectral de la matriz de Gauss-Seidel es menor que uno.

En otras palabras, si el máximo de los valores propios de esta matriz es estrictamente menor que uno entonces el punto de equilibrio es único. Por lo tanto, el método convergerá a la solución del sistema de ecuaciones, independientemente del vector inicial seleccionado. Este criterio analítico es adoptado en (Knight et al., 2014) para demostrar el Teorema 3, pero su aplicación a los MCD Sigmoidales no es correcta pues no considera la exactitud del mapa y como resultado el sistema siempre convergerá al mismo atractor. En el Capítulo 3 se realizará un estudio experimental más riguroso que ilustra explícitamente las limitaciones de estos teoremas.

Un enfoque adecuado para asegurar la estabilidad en un MCD Sigmoidal sin afectar su habilidad de simulación debe considerar los estímulos iniciales. De esta forma el mapa convergerá al punto de equilibrio (preferiblemente único) correspondiente a cada estímulo. Incluso sería conveniente que el sistema tuviera un único punto fijo por cada clase de decisión, aunque este resultado no es precisamente trivial. Sin embargo, el inciso b) del Teorema 3 es un punto de partida para obtener una solución aproximada referente al problema de la estabilidad.

Ciertamente el teorema permite afirmar que la inclinación de la función sigmoidal tiene relación con la estabilidad del mapa. Basados en esta hipótesis, se diseña un experimento donde se varía aleatoriamente la inclinación de la función de transformación. En este experimento se usaron seis sistemas biológicos descritos en (Nápoles et al., 2013b) los cuales serán detallados en el próximo capítulo, como parte del análisis estadístico. La siguiente ecuación muestra la formulación de la función de transformación asociada a cada concept C_i del mapa. Por lo tanto, estas simulaciones se reducen a variar aleatoriamente la inclinación λ_i de cada función, y luego estudiar el efecto de estos cambios sobre las propiedades de convergencia de cada mapa.

$$f_i(x) = \frac{1}{1 + e^{-\lambda_i(x-0.5)}} \tag{2.1}$$

Los resultados de las simulaciones fueron resumidos en (Nápoles et al., 2014a) y constituyen los antecedentes del método que se presenta en este capítulo. Concretamente este estudio mostró que la variabilidad en las respuestas de los mapas inestables ocasionalmente se reducía notablemente, mientras que en otros casos empeoraba. Esto confirma que la inclinación de la función de umbral está ciertamente relacionada con la estabilidad en un MCD Sigmoidal, y que utilizar una función para cada neurona puede resultar conveniente. Sin embargo, la variación aleatoria de cada factor no es un enfoque adecuado, pues la estabilidad puede empeorar.

2.2 Descripción de la propuesta

Esta sección propone un nuevo método de aprendizaje llamado Estabilidad basada en Funciones Sigmoidales (EFS) para mejorar la convergencia en MCD. El objetivo es encontrar la inclinación adecuada λ_i para cada función f_i de forma tal que se minimice la variabilidad en las salidas del mapa para cada estímulo inicial. Esto conduce a un complejo problema de optimización continuo con tantas variables como conceptos involucre el sistema. En principio, este tipo de problemas se puede solucionar con cualquier técnica de optimización, sin embargo, este trabajo apuesta por un enfoque heurístico, concretamente metaheurísticas poblacionales.

La razón detrás de esta decisión está relacionada con la complejidad del problema de optimación en sí, pues para un MCD con docenas de conceptos, un enfoque exacto (ej. Programación Lineal) podría resultar demasiado costoso. Por otra parte, las metaheurísticas poblacionales son capaces de estimar soluciones próximas al óptimo global en un tiempo de ejecución razonable, ignorando las propiedades analíticas de la función objetivo (ej. continuidad, convexidad, diferenciabilidad o información del gradiente). Independientemente de estas razones, el algoritmo que se introduce a continuación es extensible para otras técnicas de optimización.

El primer aspecto a discutir está relacionado con la codificación de los agentes. En este caso cada individuo se representa como un vector N-dimensional, donde N es el número total de conceptos que describen el sistema investigado. Cada dimensión del vector solución codifica la inclinación de la función correspondiente. Por ejemplo, la i-ésima dimensión del vector codifica el factor de inclinación λ_i asociado a la función de normalización f_i (ver ecuación 2.1). Más explícitamente, la activación de cada nodo C_i se actualizará usando la i-ésima función de normalización, en lugar de adoptar la misma función para todas las neuronas del modelo. La siguiente ecuación formaliza la modificación al proceso de inferencia que respalda el nuevo enfoque.

$$A_i^{(t+1)} = f_i\left(\sum_{j=1}^{N} w_{ji} A_j^{(t)}\right), i \neq j \tag{2.2}$$

Una vez que se ha determinado la codificación de las soluciones que serán generadas durante el proceso de búsqueda y la modificación al proceso de inferencia, es necesario definir la semántica del método de optimización. Este procedimiento tiene como objetivo central reducir al mínimo la variabilidad en las respuestas del mapa ante un vector de estímulo inicial.

Suponga que C_i es un concepto de un MCD, el cual tiene asociado una función de umbral f_i para mantener el valor de activación del nodo en el rango [0,1]. De acuerdo a la semántica del modelo descrito, el optimizador debe encontrar los parámetros λ_i que minimicen las diferencias entre dos respuestas sucesivas del sistema. Si se estiman estos parámetros entonces para cualquier estímulo inicial A^0 el mapa reportaría respuestas similares a través del tiempo, pero estas respuestas serán consecuentes con la instancia estímulo. Incluso es posible que el mapa estabilice, y las respuestas del sistema serán idénticas después de alcanzado el punto de equilibrio.

La ecuación (2.3) muestra la función que debe ser minimizada para reducir la variabilidad en las respuestas del i-ésimo concepto. El argumento de la función ϕ_i se refiere a la función de umbral asociada al concepto C_i y que depende del parámetro λ_i. Note que esta fórmula solo considera la respuesta de una neurona para un vector de estímulo, sin embargo, los problemas de clasificación de patrones normalmente involucran varias instancias. De hecho, la principal insuficiencia de los métodos analíticos descritos está relacionada con la incapacidad de predecir múltiples instancias del problema modelado (Nápoles et al., 2014a). El objetivo es formular un método que sea capaz de estabilizar todas las neuronas asociadas a las clases de decisión.

$$min \rightarrow \phi_i(f_i) = \sum_{t=2}^{T} |A_{it} - A_{it-1}| \tag{2.3}$$

La ecuación (2.4) formaliza la función objetivo que debe minimizar el optimizador seleccionado durante el progreso de la búsqueda. El nuevo algoritmo debe determinar una familia de funciones sigmoidales que minimice la variabilidad en las respuestas de las N neuronas del mapa para dos iteraciones sucesivas, considerando K estímulos iniciales diferentes.

$$min \rightarrow \phi(f_1, f_2, f_3, \dots, f_N) = \sum_{k=1}^{K} \sum_{i=1}^{N} \sum_{t=2}^{T} \left| A_{it}^k - A_{it-1}^k \right| \tag{2.4}$$

Ciertamente el mínimo global de esta función corresponderá con un estado atractor de equilibrio, aunque no es posible garantizar que el algoritmo encontrará esta solución. Por otra parte, hasta el momento el modelo propuesto no soluciona las deficiencias de los enfoques analíticos existentes y como resultado tampoco es aplicable a problemas de clasificación. En otras palabras, un mapa estable no implica un sistema con alta capacidad de predicción, lo cual sugiere que el algoritmo debe considerar esta característica en su esquema de aprendizaje.

Es importante recalcar que el método está orientado a estabilizar MCD Sigmoidales previamente ajustados, una vez que se han estimado las relaciones causales entre los nodos. La estimación de la matriz de pesos causales puede ser de forma manual a través de expertos, usando un algoritmo de aprendizaje, o incluso combinado ambas estrategias. En cualquier caso, el procedimiento para mejorar la estabilidad en mapas sigmoidales asume que los mapas originales tienen una adecuada capacidad para clasificar nuevos patrones. Esto significa que la respuesta final del mapa coincide con la respuesta esperada, pero esta salida es variable a través del tiempo. Por lo tanto, el modelo debe ser capaz de estabilizar el mapa preservando su salida.

Para preservar la exactitud del mapa no es estrictamente necesario modificar la función objetivo, es suficiente con adicionar una restricción al problema de optimización. La ecuación (2.5) es una variante para calcular el error de un mapa usado en problemas de predicción, donde $\mathcal{M}$ denota el mapa candidato, mientras que φ representa un conjunto de instancias (cada instancia codifica la activación de las neuronas y la salida esperada). Por otra parte, la función $R(\mathcal{M}, \varphi_i)$ calcula la respuesta para la i-ésima instancia, mientras que R_i es la salida conocida. En resumen, la función de error $E(\mathcal{M}, \varphi)$ cuantifica la distancia (usando una norma L) entre la respuesta inferida por el mapa y la salida conocida, para cada objeto usado como estado inicial.

$$E(\mathcal{M}, \varphi) = \sum_{k=1}^{|\varphi|} \frac{\|R(\mathcal{M}, \varphi_k) - S_k\|_L}{|\varphi|} \tag{2.5}$$

Suponga que $E_0(\mathcal{M}, \varphi)$ es el error del mapa donde se usa la misma función de umbral para todos los conceptos del mapa, mientras que $E(\mathcal{M}, \varphi)$ representa el error usando la familia de funciones sigmoidales $f_1, f_2, f_3, \dots, f_N$ estimada por el modelo descrito. Si $E(\mathcal{M}, \varphi) - E_0(\mathcal{M}, \varphi)$ es mayor que cero, entonces los parámetros estimados inducen un error adicional y por tanto es necesario penalizar la función objetivo (2.4) con un valor positivo. La ecuación (2.6) muestra la función de penalización $P(\mathcal{M}, \varphi)$ usada en este modelo, donde $\Delta E(\mathcal{M}, \varphi) = E(\mathcal{M}, \varphi) - E_0(\mathcal{M}, \varphi)$ expresa la magnitud del error, mientras que $\Delta\phi(f_1, f_2, f_3, \dots, f_N)$ representa la diferencia entre el máximo valor que puede tomar la función objetivo y su valor actual. De acuerdo a (2.4) el máximo valor que puede tomar la función objetivo corresponde a un mapa cíclico.

$$P(\mathcal{M}, \varphi) = \frac{\Delta E(\mathcal{M}, \varphi)\Delta\phi(f_1, f_2, f_3, \dots, f_N)}{E(\mathcal{M}, \varphi)} \tag{2.6}$$

Más explícitamente, si el rango de activación de una neurona para tiempos sucesivos del proceso de inferencia es estrictamente uno, entonces para T tiempos la máxima diferencia se alcanzará en un sistema cíclico con dos estados con máxima variabilidad. Esto implica que la cota superior de la función objetivo está dada por la expresión (2.7), donde K es el número de estímulos iniciales, mientras que N denota el número de neuronas. La cota ínfima de esta función es cero, aunque es muy improbable que se alcance este estado pues el sistema tendría que estabilizar en la primera iteración para todas las instancias (estímulos iniciales) del problema.

$$\phi_M = KN(T-1) \tag{2.7}$$

La Figura 2.1 muestra el comportamiento de una neurona durante el proceso de inferencia de dos mapas sigmoidales para 50 tiempos. En el primer caso el mapa es cíclico y tiene solo dos estados con variabilidad máxima. De acuerdo a la ecuación (2.3) la variabilidad de la neurona está dada por la expresión $\phi_i(f_i) = (T-1) = 49$. Si todas las neuronas se comportaran de esa forma para todos los estímulos iniciales, entonces la función objetivo alcanzaría su pésimo valor. El segundo ejemplo corresponde con un estado caótico, pero en este caso la variabilidad en las respuestas del sistema no es máxima, por tanto $\phi_i(f_i) = 16.39 < 49$. No obstante, el mapa caótico puede tener hasta T estados diferentes asociados al mismo vector estímulo inicial.

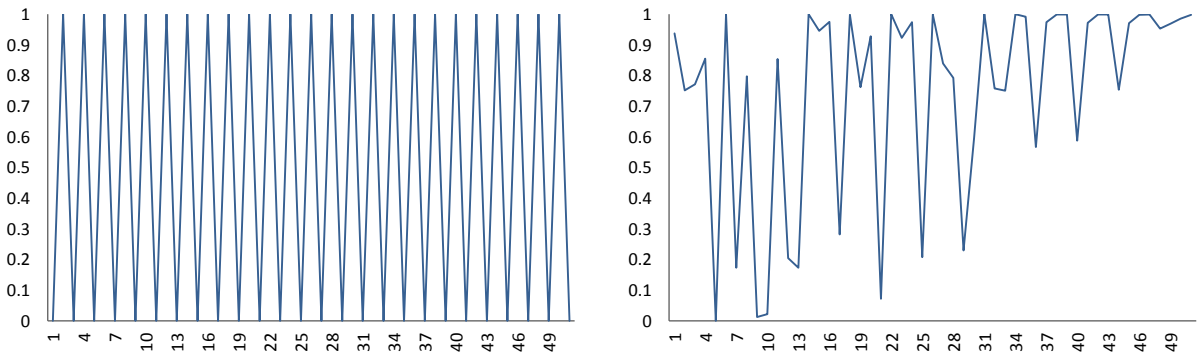


Figura 2.1 Ejemplos de la variabilidad de una neurona en un MCD Sigmoidal. El eje horizontal denota el tiempo durante el proceso de inferencia, mientras que el eje vertical representa el grado de activación del concepto.

Otro aspecto a considerar está relacionado con el dominio asociado a cada factor de inclinación, pues todo lo que asegura el Teorema 3 es que los valores tienen que ser suficientemente grandes para que existan múltiples atractores de equilibrio. En (Knight et al., 2014) los autores presentan un teorema que define el término "suficientemente pequeño" y el complemento de este resultado serán las condiciones que debe garantizar el algoritmo. En otras palabras, resulta necesario violar las condiciones de este teorema para asegurar que el punto fijo no sea único.

Teorema 5. Para $A \in \mathbb{R}^N \times \mathbb{R}^N$ un MCD Sigmoidal puede tener múltiples atractores estables para cualquier $\lambda > \lambda(N)$ si $\lambda(N)$ satisface la siguiente expresión:

$$\left(1 - \frac{\lambda(N)}{4}\right)^N - \sum_{i=1}^{N} b_i \, C_i^N \left(\frac{\lambda(N)}{4}\right)^i = 0 \tag{2.8}$$

En la expresión anterior C_i^N denotan los coeficientes binomiales, mientras que los b_i están dados por la relación de recursión $b_i = ib_{i-1} + (-1)^i$ con $b_0 = 1$. Este sugiere que si la inclinación de la función sigmoidal $\lambda > 2$, entonces el mapa podría tener varios atractores, independientemente del número de neuronas del mapa. Entonces, es razonable suponer que $2 < \lambda_i \leq 2 + \sigma$ donde el parámetro $\sigma > 0$ denota la amplitud del intervalo. Por otra parte, este teorema está diseñado para mapas que usan la misma función de umbral para cada neurona, pero el modelo descrito en este trabajo asume una función f_i diferente para cada concepto.

De forma general es permitido que algún $\lambda_i < 2$, pero es preferible que $2 < \lambda_i \leq 2 + \sigma$ y evitar la situación extrema donde $\lambda_i < 2, \forall i$ lo cual conduciría de forma irremediable al mismo atractor de equilibrio, independientemente del estímulo inicial. Ciertamente este caso no es probable pues como resultado se afectaría la exactitud del mapa y la función sería penalizada. Para favorecer el proceso de búsqueda se asume $\sigma = 10$ en todas los experimentos, pues normalmente los valores recomendados para λ son relativamente pequeños (Miao and Liu, 2000).

¿Por qué utilizar en esta propuesta solo funciones sigmoidales? En principio es posible formular un algoritmo más completo que incluya otras funciones de normalización (ej. binaria, trivalente o basadas en tangentes hiperbólicas). De esta forma no solo se optimizaría la inclinación adecuada de la función sigmoidal, sino que también sería necesario determinar el tipo de función asociada a cada concepto del mapa. De hecho, los experimentos preliminares se basaron en la solución de este complejo problema de optimización con variables mezcladas.

Sin embargo, esta variante no es adecuada para todos los casos. Por ejemplo, hay sistemas donde las neuronas tienen que ser numéricamente cuantificadas, y las funciones discretas solo expresan si el concepto está activo o no. Pero una función sigmoidal puede ser discretizada si se establece un umbral de corte adecuado al problema. Asimismo, en (Bueno and Salmeron, 2009) los autores concluyen que las funciones sigmoidales tienen una capacidad de inferencia superior, respecto a las demás funciones, así que son más adecuadas para estos escenarios.

2.3 Interpretación del método de aprendizaje

Hasta el momento se tiene un algoritmo capaz de mejorar la convergencia en MCD Sigmoidales, una vez que se ha estimado la matriz de pesos. Es importante discutir cuál es el efecto general de estas modificaciones sobre el mapa y cómo debe interpretarse. En primer lugar, es conocido que los MCD son estructuras basadas en el conocimiento que permiten la simulación e interpretación de un sistema dinámico a través de relaciones causales. Sin embargo, el modelo clásico asume la misma función de transformación para todas las neuronas del mapa, de forma tal que el grado de activación depende solamente de la propagación del estímulo inicial.

En el nuevo modelo esta percepción es diferente, pues el grado de activación de las neuronas está condicionado adicionalmente por la inclinación de cada función. Esto es equivalente al efecto de un estímulo externo sobre la neurona. El efecto del estímulo puede ser calculado numéricamente como la diferencia entre el grado de activación que alcanza un concepto C_i utilizando la misma función de normalización ($\lambda_1 = \lambda_2 = \cdots = \lambda_N$) para todos las neuronas y la función estimada por algoritmo, la cual tiene su propio factor de inclinación λ_i. La siguiente ecuación (2.9) formaliza matemáticamente esta idea para cualquier concepto C_i del sistema.

$$\Delta\gamma = f_i\left(\sum_{j=1}^{N} w_{ji}A_j^{(t)}\right) - f\left(\sum_{j=1}^{N} w_{ji}A_j^{(t)}\right), i \neq j \tag{2.9}$$

Si el estímulo $\Delta\gamma > 0$ entonces la neurona está excitada, si $\Delta\gamma < 0$ entonces se inhibe el valor de activación de la neurona, y si $\Delta\gamma = 0$ entonces no hay estímulo. Este modelo confiere a los MCD Sigmoidales nuevas propiedades que permite atenuar las deficiencias inherentes a la modelación y arquitectura de estas estructuras cognitivas (Nápoles et al., 2014a).

Por ejemplo, en (Nápoles et al., 2013b) los autores presentaron un MCD Sigmoidal para modelar el comportamiento de algunas proteínas del VIH-1. Este mapa considera la relación causal entre los aminoácidos para predecir el grado de resistencia a los fármacos existentes, pero es conocido que el comportamiento de las proteínas también depende de otros factores (ej. procesos químicos relacionados con la respuesta catalítica, interacción con el sustrato). Precisamente, esos factores externos son estímulos que inciden sobre las neuronas, y pueden ser modelados usando el factor de inclinación de la función sigmoidal. De esta forma es posible obtener una representación más realista de la dinámica del sistema biológico como un todo.

Otro aspecto notable que debe ser discutido está relacionado con las propiedades analíticas de la función de transformación (2.1) adoptada en este trabajo. Esta función está desplazada respecto a la formulación original (Tsadiras, 2008) pero preserva las asíntotas, garantizando que el grado de activación de las neuronas se mantenga en el rango $[0,1]$. Sin embargo, esta simple modificación le confiere propiedades muy interesantes que pudieran influir positivamente en el desempeño del algoritmo, aunque la evaluación de estos impactos no es objetivo de la tesis.

La Figura 2.2 muestra el comportamiento de la función (2.1) para dos parámetros de inclinación diferentes respecto a la recta $f(x) = x$ para una neurona, la cual representa el valor de activación del concepto sin modificación alguna. En el primer caso ($\lambda_i = 2$) es posible observar que el nodo se estimula si su grado de activación es menor que 0.5, pero si es mayor entonces el concepto se inhibe respecto al valor que sugiere la recta. Esta variante podría ser adecuada en sistemas donde la interacción de las neuronas es muy débil. En el segundo caso ($\lambda_i = 12$) la neurona se inhibe al principio y luego se excita, lo que es muy conveniente en mapas donde los conceptos se saturan rápidamente con insuficiente intercambio de información.

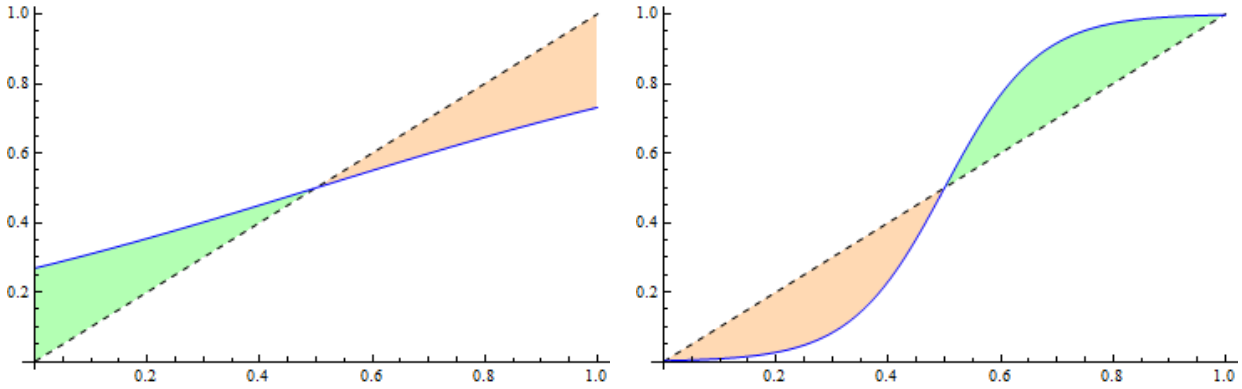


Figura 2.2 Grado de activación de una neurona sigmoidal para diferentes inclinaciones. El eje x representa el valor original de la neurona, mientras que el eje y denota el valor que resulta después de la normalización.

Es justo mencionar que la idea de incorporar los estímulos externos como parte de la modelación en un MCD no es una contribución de la propuesta. En (Stylios and Groumpos, 1999) los autores proponen un modelo donde cada neurona tiene una entrada externa cuyo efecto sobre el concepto se ajusta por un peso causal, lo que deriva en una matriz de pesos aumentada. Sin embargo, esta propuesta nunca fue asociada a la estabilidad del mapa, sino a corregir limitaciones relacionadas con la modelación y arquitectura. Como parte del modelo, en la próxima sección se discute cómo generar soluciones factibles que permitan minimizar la función objetivo, y por tanto mejorar las propiedades globales de convergencia de los MCD Sigmoidales.

2.4 Selección del optimizador

En principio es posible usar cualquier metaheurística continua, pues el objetivo fundamental del algoritmo propuesto es la estabilidad del mapa, no la adecuada selección del método de búsqueda heurístico. Sin embargo, un optimizador poco robusto conducirá irremediablemente a soluciones sub-óptimas, afectando el rendimiento del algoritmo. Por este motivo, este trabajo apuesta por la Optimización de Enjambres de Partículas debido a su alta tasa de convergencia y su capacidad de solucionar problemas complejos de optimización global con diferentes características (Wang et al., 2011). Además, esta sección describe un operador para atenuar con los efectos negativos que inducen los estados de no progresos en las metaheurísticas poblacionales.

2.4.1 Optimización de Enjambres de Partículas

La metaheurística PSO es un procedimiento de búsqueda bioinspirado en el comportamiento de las poblaciones de pájaros. Este método de búsqueda fue introducido por (Kennedy and Eberhart, 1995) para solucionar problemas de optimización continuos. Durante la búsqueda, PSO explota un conjunto de agentes (llamados partículas) que se desplazan a través del espacio de soluciones, tratando de localizar regiones prometedoras. Entonces una partícula se denota como un vector en un espacio N-dimensional, o sea, una posible solución del problema.

En el proceso de búsqueda la posición de una partícula X_i se actualiza en cada ciclo usando una combinación entre la atracción que ejerce el mejor registro personal P_i del agente y la atracción que ejerce la mejor solución global G encontrada por toda población en la vecindad (Bratton and Kennedy, 2007). Estos atractores son la base del tercer elemento: la velocidad V_i asociada a cada partícula del enjambre, la cual controla la dirección y magnitud del desplazamiento del individuo en el espacio de búsqueda. La siguiente figura ilustra gráficamente como se actualiza la posición de cada partícula X_i en el espacio desde el punto de vista algebraico.

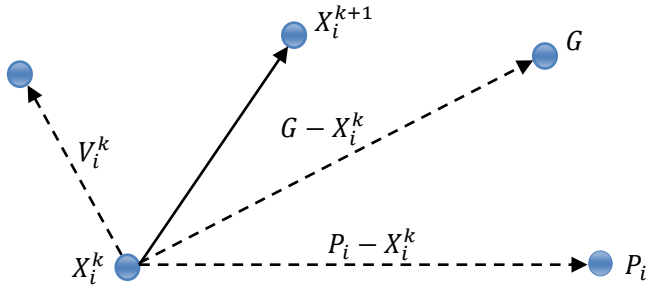


Figura 2.3 Actualización de la posición de una partícula durante la búsqueda.

Las ecuaciones (2.10) y (2.11) formalizan matemáticamente las reglas para actualizar la posición de los individuos, donde k denota las iteraciones, $\delta_1 \sim U(0, c_1)$ y $\delta_2 \sim U(0, c_2)$ son dos números aleatorios con distribución uniforme. Los factores c_1 y c_2 representan el coeficiente cognitivo y social respectivamente. Estos parámetros deben ser especificados por el usuario. Los coeficientes de aceleración c_1 y c_2 determinan la magnitud de las fuerzas en dirección al registro personal de cada individuo y de la mejor solución encontrada por la bandada. La correcta estimación de estos valores resulta central para garantizar el rendimiento de la metaheurística.

$$V_i^{k+1} = V_i^k + \delta_1\left(P_i - X_i^k\right) + \delta_2\left(G - X_i^k\right) \tag{2.10}$$

$$X_i^{k+1} = \mathrm{X}_i^k + V_i^{k+1} \tag{2.11}$$

Otro parámetro del método es la velocidad máxima V_M. Este es un importante factor que se usa para acotar la velocidad de las partículas. La selección incorrecta del valor de V_M puede afectar el rendimiento global del algoritmo (Poli et al., 2007). Como una alternativa, en (Shi and Eberhart, 1998) los autores presentaron el peso de inercia ω para reemplazar V_M como muestra la siguiente ecuación (2.12). Este factor regula la influencia de la velocidad previa de las partículas durante la actualización de la poción de cada individuo en el espacio de búsqueda.

$$V_i^{k+1} = \omega V_i^k + \delta_1\left(P_i - X_i^k\right) + \delta_2\left(G - X_i^k\right) \tag{2.12}$$

La introducción de este parámetro es una alternativa para balancear la habilidad de exploración y explotación del algoritmo. Valores altos ($\omega = 0.9$) facilitarán la exploración de nuevas áreas del espacio de soluciones, mientras que valores pequeños ($\omega = 0.1$) son adecuados para explotar las zonas prometedoras. Incluso varios trabajos sugieren estimar el valor de este parámetro de forma dinámica (Nápoles et al., 2012a). Por ejemplo, resultaría conveniente valores altos al principio de la búsqueda y valores pequeños al final, garantizando mayor explotación.

A pesar de las ventajas de utilizar este peso de inercia, determinar su valor exacto puede resultar muy complejo en algunos problemas. Incluso algunos autores sugieren combinar esta variante con el método de velocidad acotada, para reducir el efecto de la incorrecta selección del valor de este parámetro. Como una alternativa, en (Clerc and Kennedy, 2002) los autores propusieron dos nuevas variantes conocidas como coeficiente de restricción. La siguiente ecuación (2.13) muestra el coeficiente de restricción Tipo 1. Esta implementación es posiblemente la más difundida entre los investigadores debido a la robustez y simplicidad de su implementación.

$$\chi = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|} \tag{2.13}$$

En la anterior ecuación $\varphi = c_1 + c_2$. Esto significa que el coeficiente de restricción χ dependerá del valor de los factores de aceleración del algoritmo. La siguiente ecuación (2.14) muestra cómo actualizar las velocidades de las partículas según este modelo. En resumen, si $\varphi > 4$ entonces es posible prevenir la explosión de la nube de partículas (fenómeno donde los agentes y velocidades tienden al infinito). De esta forma se asegura la convergencia del algoritmo (Clerc and Kennedy, 2002). A pesar de las garantías del resultado, en (Poli et al., 2007) los autores sugieren acotar las velocidades de las partículas como un enfoque complementario. Un criterio bastante aceptado es fijar la velocidad máxima como la mitad del rango del espacio de soluciones.

$$V_i^{k+1} = \chi\left(V_i^k + \delta_1\left(P_i - X_i^k\right) + \delta_2\left(G - X_i^k\right)\right) \tag{2.14}$$

Pero si $\varphi < 4$ entonces no es posible garantizar la convergencia del algoritmo. Incluso es posible que las partículas muestren comportamientos discretos. Por ejemplo, (Clerc and Kennedy, 2002) notaron que si $\varphi = (5 + \sqrt{5})/2$ entonces una partícula no aleatoria en $\mathbb{R}$ se desplazaría de forma cíclica a través de 5 estados, independientemente de la naturaleza de la metaheurística. Si alguno de estos estados no representa un óptimo, entonces esto desencadenará irremediablemente en una configuración de estancamiento, donde no hay progresos en la búsqueda. Otro escenario curioso que viola estas condiciones tiene lugar cuando $\varphi = 5 + \sqrt{3}$. En este caso específico una partícula en un espacio 1-dimensional tendrá un comportamiento cuasi-cíclico.

Normalmente se toma $c_1 = c_2 = 2.05$ lo que garantiza que $\varphi > 4$. Observe que el coeficiente de restricción χ es equivalente al peso de inercia, si $c_1 = c_2 = 1.4961$ y $\omega = 0.7298$. Estos valores serán adoptados en el resto del trabajo, pues en otros estudios han demostrado ser una alternativa robusta para diferentes tipos de problemas (Nápoles et al., 2014b). Por lo tanto, este optimizador solo requiere que el usuario determine el número máximo de evaluaciones de la función objetivo permitidas para la ejecución del algoritmo y el tamaño de la población.

Más simplificadamente, basta garantizar estas condiciones de convergencia para asegurar que la metaheurística tendrá un comportamiento adecuado. Esto reduce elocuentemente el esfuerzo que requiere la configuración de los parámetros sensibles a un modelo. De hecho, el trabajo de (Clerc and Kennedy, 2002) ha motivo al autor para usar este método como optimizador.

2.4.2 PSO con topologías estructuradas

Otro aspecto relevante en la metaheurística PSO es la elección del atractor global. Este atractor denota la mejor solución encontrada por los individuos en su vecindad. En PSO una vecindad se define, para cada partícula de la bandada, como el subconjunto de agentes que pueden interactuar con dicha partícula (Bratton and Kennedy, 2007). Si la vecindad de cada partícula incluye toda la bandada entonces se está en presencia del modelo global, pero si existen algún tipo de restricción en el protocolo de comunicación se está en presencia de un modelo local.

El primer modelo local usaba una vecindad Euclidiana, donde se calculaba la distancia entre las partículas y aquellos individuos suficientemente próximos a la partícula actual conformaban la vecindad de la partícula. Este modelo es una imitación exacta de las bandadas de pájaros, donde un individuo solo puede comunicarse con los pájaros que se encuentran en la vecindad inmediata a su posición espacial (Reynolds, 1987). La vecindad Euclidiana fue desplazada por alternativas menos costosas y que se centran más en la optimización matemática.

Al principio las topologías locales no tuvieron mucha aceptación pues su rendimiento no lograba superar al modelo global. Sin embargo, en (Kennedy and Mendes, 2006) los autores demostraron que el rendimiento de PSO era superior si se usaba una topología estructurada en conjunción con otras mejoras a la metaheurística. La siguiente Figura 2.4 muestra el protocolo de comunicación entre las partículas para las dos topologías más utilizadas: la Topología Global y la Topología de Anillo. La Topología de Anillo es el modelo local más simple; donde el agente X_i interactúa con las partículas X_{i-1} y X_{i+1} de la población. De forma general este protocolo es bastante aceptado debido a su habilidad para preservar la diversidad entre los individuos.

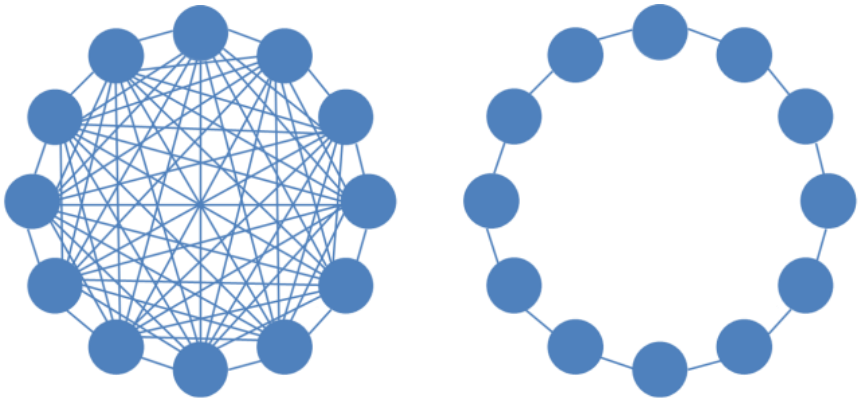


Figura 2.4 Diferentes protocolos de comunicación entre las partículas de la población. El primer ejemplo muestra la Topología Global (*Gbest*), mientras que el segundo ejemplo ilustra la Topología de Anillo (*LBest*).

Supuestamente, la desventaja de los modelos estructurados respecto al modelo PSO global radica en la lenta velocidad convergencia (Bratton and Kennedy, 2007) debido al limitado intercambio de información entre los agentes. Si el problema tiene un único óptimo global entonces tener una velocidad de convergencia alta hacia esa región del espacio de soluciones resultará beneficioso, pero en espacios de búsqueda con muchos óptimos locales y globales aumenta la probabilidad de converger de forma prematura a un óptimo local. Por lo tanto, usar una topología estructurada es una alternativa factible a los estados de convergencia prematura.

Irónicamente la lenta velocidad de convergencia de las topologías estructuradas es la responsable de su pobre aceptación. La velocidad de convergencia superior del modelo global parece ser más adecuada siempre, pero esto realmente es incorrecto. Por ejemplo, el PSO global ciertamente es superior en las primeras iteraciones del proceso de optimización, pero las topologías locales son capaces de encontrar mejores soluciones después de algunas iteraciones adicionales, mayormente en problemas multimodales. Esto significa que la variante adecuada depende en última instancia de la complejidad de la función. Si esta función involucra operaciones muy costosas es preferible la velocidad de convergencia del método global, pero si es conocido que existen óptimos locales, entonces probablemente será más adecuada una topología local.

Por otra parte, en (Li, 2010) se demostró que un PSO con Topología de Anillo podía converger a múltiples óptimos de forma simultánea, sin nuevos operadores. Durante el proceso de búsqueda la bandada implícitamente se descompone en varias subpoblaciones, como resultado de la escasa interacción entre los agentes. En espacios de búsqueda multimodales, cada subpoblación podría converger a un óptimo diferente (local o global). Eventualmente la información se propagará por toda la bandada, sin embargo, en algunos casos esto no es suficiente. Como una modificación, en (Nápoles et al., 2013a) se propuso un operador para potenciar la capacidad de búsqueda de esta topología. El algoritmo fue capaz de encontrar el 100% de los óptimos en las funciones simples, mientras que para las complejas resultó ser superior al método original.

A pesar de las ventajas evidentes de las variantes locales, en problemas de optimización global es preferible la topología con mayor comunicación. No obstante, la velocidad de convergencia de esta variante frecuentemente conduce a estados de convergencia prematura, como resultado de la pérdida de la diversidad en la población artificial. En la próxima sección se describe un operador que permite detectar y remediar estados potenciales de no progreso, incrementando la efectividad de la metaheurística PSO en espacios de búsqueda multimodales.

2.5 Operador para mejorar la diversidad

Un problema común del algoritmo PSO global (y de las metaheurísticas poblacionales de forma general) es la pérdida progresiva de la diversidad entre los agentes, lo cual conduce regularmente a soluciones sub-óptimas. Como tercera contribución de la tesis, en este epígrafe se presenta un operador para fomentar la diversidad en la bandada, una vez que se detectan estados potenciales de no progreso. De la misma forma, se puntualiza la semántica de las estrategias para detectar las configuraciones de estancamiento y los estados de convergencia prematura.

2.5.1 Detección de estados de no progreso

Los estados de convergencia prematura y de estancamiento son frecuentes en las metaheurísticas poblacionales, sobre todo en espacios de búsqueda multimodales. En los estados de convergencia prematura todos los individuos de la población se agrupan en una zona minúscula del espacio de búsqueda (Nápoles et al., 2012a). En este estado los agentes no son capaces de explorar regiones alternativas y por lo tanto no se observan progresos en la búsqueda. Si los individuos convergen prematuramente hacia el óptimo global entonces la situación es favorable, pero en funciones con muchos óptimos locales este escenario es usualmente improbable.

Por su parte, en los estados de estancamiento tampoco se evidencia progresos en la optimización de la función objetivo, pero la diversidad no se pierde necesariamente. El estado de convergencia prematura es en realidad un estado de estancamiento donde los individuos artificiales se agrupan alrededor de un óptimo. En espacios con muchos óptimos locales esta solución a menudo denota una solución sub-óptima, lo cual afecta el desempeño de la metaheurística. La Figura 2.5 muestra un ejemplo donde se ilustra la diversidad de la población en ambas configuraciones, utilizando la distancia promedio entre individuos como medida de diversidad.

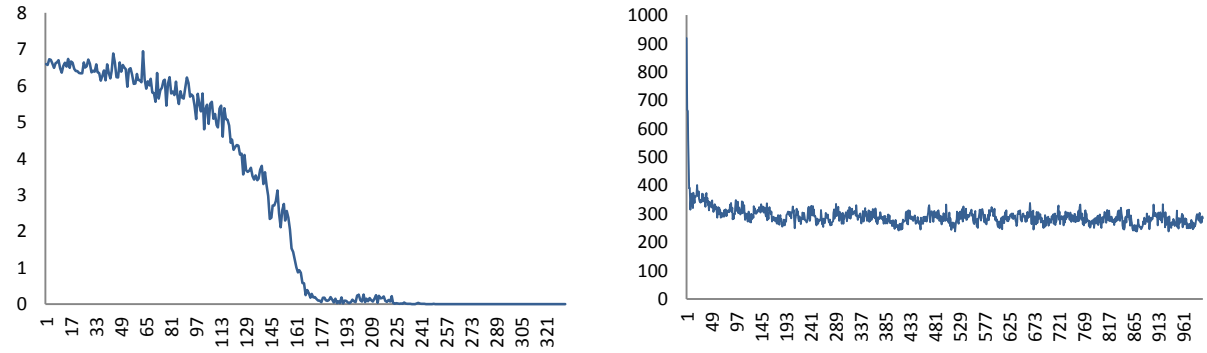


Figura 2.5 Ejemplos de estados de no progreso en una población de individuos. El primer caso ilustra un estado de convergencia prematura, mientras que el segundo caso ilustra un estado de estancamiento. El eje x representa el número de evaluaciones de la función objetivo, mientras que el eje y denota la medida de diversidad.

¿Cómo determinar si la población ha convergido prematuramente o se encuentra en una situación de estancamiento? En el caso de los estados de convergencia prematura es suficiente con adoptar una medida robusta que permita cuantificar la diversidad en la población. En la literatura existen varias medidas que permiten hacer esto (ej. el radio de la bandada, la coherencia de la población, la distancia promedio normalizada alrededor del centro de la bandada). Sin embargo, la selección de la medida apropiada en cada caso es un aspecto complicado de establecer con exactitud, pues frecuentemente depende de las características del espacio de búsqueda.

Por ejemplo, en (Nápoles et al., 2012a) los autores usaron el radio de la bandada como criterio para detectar situaciones de convergencia prematura. La siguiente ecuación (2.15) formaliza esta medida, donde B es una bandada de partículas, X_i denota un agente de la población, G representa la mejor solución global encontrada hasta el momento, mientras que $|X_M - X_m|$ denota el rango del espacio de soluciones. Si el radio de la bandada $\rho(B)$ es menor que un umbral determinado por el experto, entonces se asume un estado de convergencia prematura. Sin embargo, el radio de la bandada, al igual que las demás medidas, involucra algunos problemas que ciertamente limitan su usabilidad y que pueden afectar el rendimiento general del algoritmo.

$$\rho(B) = \frac{\max\limits_{1 \leq i \leq |B|} \|X_i - G\|}{|X_M - X_m|} \tag{2.15}$$

En primer lugar, no es trivial la selección adecuada del umbral pues valores demasiado grandes son propensos a revelar falsos estados de convergencia prematura, mientras que parámetros muy pequeños podrían no detectar configuraciones reales de convergencia prematura. En ambos casos probablemente se tomarán decisiones equivocadas (ej. reorganización de la población cuando los agentes están próximos a encontrar el óptimo global en un espacio unimodal) que pueden afectar de forma irremediable la efectividad y eficacia de la metaheurística.

En segundo lugar, las medidas de diversidad pudieran detectar de forma conveniente potenciales estados de convergencia prematura, pero no se pueden utilizar para detectar posibles situaciones de estancamiento. Por este motivo, en (Nápoles et al., 2012a) se usa otro criterio: se dice que la población se encuentra en un estado de no progreso si no es posible encontrar una mejor solución en np iteraciones. De esta forma se obtiene un criterio más simple (ej. no es necesario calcular la distancia Euclidiana) y que se puede usar para detectar estados de no progreso. Este criterio será adoptado en las simulaciones y experimentos del resto de la tesis.

2.5.2 Reorganización de la población

Una vez que se ha detectado un estado de no progreso (convergencia prematura o estancamiento) es necesario adoptar una estrategia que permita mejorar la búsqueda. Por ejemplo, para el caso de la metaheurística PSO regularmente se utilizan dos enfoques. En el primer enfoque el objetivo es encontrar un atractor global alternativo que permita guiar la búsqueda hacia otras regiones del espacio de soluciones. Sin embargo, encontrar una mejor partícula global no es trivial, e incluso si esto fuera posible, en situaciones de convergencia prematura no se soluciona el problema de la pérdida de la diversidad (Nápoles et al., 2012a). Entonces es muy probable que la bandada quede atrapada en un óptimo local en las próximas iteraciones del algoritmo.

La segunda variante consiste en reorganizar la población con el objetivo de mejorar la diversidad entre los agentes, pero este enfoque también tiene limitaciones. Por ejemplo, si la estrategia para detectar estados de no progreso no es robusta, entonces el operador encargado de reorganizar la bandada puede perjudicar la búsqueda (Nápoles et al., 2012a). Esto significa que la población se alejará indiscriminadamente de las regiones prometedoras encontradas.

En esta sección se describe un operador para reorganizar la población una vez que se detecta una configuración de convergencia prematura o estancamiento. El procedimiento Muestreo Aleatorio en Vecindades Variables (*Random Sampling in Variable Neighborhoods*, RSVN) fue presentado inicialmente en (Nápoles et al., 2012b) como una mejora a la metaheurística PSO, pero luego los autores propusieron un conjunto de modificaciones en (Nápoles et al., 2014b) que incrementaron significativamente su rendimiento. Con el objetivo de ganar en claridad, a continuación se detalla la semántica del operador RSVN considerando la variante mejorada.

El objetivo del operador consiste en fomentar la diversidad de la población, pero preservando el conocimiento aprendido sobre el espacio de búsqueda. Primero se generan conjuntos de muestras aleatorias alrededor del mejor individuo encontrado hasta el momento. Más claramente, en cada vecindad j se genera un conjunto de muestras Ψ_j donde cada partícula $X_i^j \in \Psi_j$ se distribuye tal y como muestra la siguiente ecuación (2.16). En esta fórmula G es el atractor global, ξ_j es el factor de vecindad, R_1 y R_2 son el extremo izquierdo y derecho del intervalo respectivamente, sobre los cuales se generan números aleatorios usando la distribución de Lévy.

$$X_i^j \sim \text{Lévy}(G - \xi_j R_1, G + \xi_j R_2) \tag{2.16}$$

El extremo izquierdo se estima como $R_1 = G - X_m$ y el extremo derecho es $R_2 = X_M - G$ donde las variables X_m y X_M son el valor mínimo y máximo que pueden tomar las variables (asumiendo el mismo dominio para todas las dimensiones). De esta forma nunca se generarán individuos no factibles, o sea, fuera del espacio de búsqueda. El cálculo de los extremos fue una de las mejoras más significativas discutidas en (Nápoles et al., 2014b) pues en la variante anterior era necesario acotar las muestras, y como resultado se generaban muchas agentes en las fronteras del espacio de búsqueda, reduciendo paradójicamente la diversidad en la población.

Otro elemento a tener en cuenta es el parámetro de vecindad, el cual se calcula automáticamente como $\xi_j = j/M$ donde M es el número de vecindades variables. Este factor denota la proporción de la j-ésima vecindad respecto al tamaño del espacio de búsqueda, estableciendo la cercanía de esa vecindad a la solución encontrada. Valores pequeños propiciarán la explotación alrededor del mejor individuo global, mientras que valores altos favorecerán la exploración de nuevas regiones del espacio de búsqueda. Observe que el término "variable" se refiere a la proximidad de cada vecindad al atractor global encontrado, no al tamaño de la vecindad en sí.

El segundo paso del proceso de reorganización de la bandada consiste en seleccionar el conjunto de individuos Φ_j de cada conjunto de muestras Ψ_j que serán incluidos en la siguiente población artificial. Por lo tanto, es lógico suponer que $|\Psi_j| > |\Phi_j|$. Un criterio viable para seleccionar los agentes pudiera estar basado en la evaluación de la función objetivo, o sea, en la calidad relativa de cada solución. Sin embargo, frecuentemente la evaluación de la función objetivo es costosa computacionalmente, y por lo tanto pudiera afectar la simplicidad del algoritmo. Si ese escenario se presenta, entonces es suficiente que $|\Psi_j| = |\Phi_j|$. La siguiente ecuación formaliza la estrategia para conformar la bandada, suponiendo que G pertenece a la población.

$$B = \Phi_1 \cup \Phi_2 \cup \ldots \cup \Phi_M = \bigcup_{j=1}^{M} \Phi_j : \Phi_j \subseteq \Psi_j, \forall j \tag{2.17}$$

Hasta el momento se tiene un algoritmo que no involucra operaciones costosas, capaz de mejorar la diversidad en la población. Por otra parte, este operador solo necesita el número de vecindades como parámetro, sin embargo, en (Nápoles et al., 2014b) los autores concluyen que para $M = 10$ el operador alcanza su máximo rendimiento. Esto conduce a un método libre de parámetros, con un balance adecuado entre la exploración y explotación, características regularmente deseadas en la solución de problemas de optimización en espacios multimodales.

2.6 Conclusiones parciales

La estabilidad en los MCD Sigmoidales es un aspecto esencial que puede afectar la usabilidad de estas estructuras neuronales. Sin las condiciones de convergencia, la interpretabilidad del sistema resulta incierta, y no es posible tomar de decisiones acertadas. Sin embargo, no existen garantías de obtener mapas estables a partir de los sistemas modelados o de los algoritmos de aprendizaje existentes. Es válido resaltar que los métodos analíticos descritos en el Capítulo 1 son capaces de determinar condiciones de convergencia en MCD Sigmoidales pero no consideran la exactitud en contextos de clasificación de patrones, predicción o toma de decisiones.

En este capítulo se discutió un algoritmo que permite mejorar las condiciones de convergencia en MCD Sigmoidales, una vez que se ha estimado la matriz de pesos. El nuevo algoritmo se basa en la hipótesis de la propuesta: es posible mejorar la convergencia en los MCD Sigmoidales si se usa una función de activación adecuada para cada neurona. Este problema se reduce a estimar el factor de inclinación λ_i asociado a cada nodo, de forma tal que se minimice la variabilidad global en las respuestas del sistema. Es importante acentuar que este método no garantiza que el sistema convergerá a un atractor de equilibrio, pero ciertamente resulta una alternativa factible para lidiar con el problema de inestabilidad de los MCD, sin afectar su exactitud.

Para solucionar el problema de optimización relacionado, se seleccionó el algoritmo PSO global con coeficiente restricción. Además se introdujo un nuevo operador para preservar la diversidad de la población, como alternativa a los estados de no progreso. Este procedimiento es aplicable a cualquier metaheurística poblacional, aunque solo se conoce su rendimiento para el método PSO global con coeficiente de restricción. A pesar de que el problema de optimización enunciado sea probablemente multimodal, el autor supone que la velocidad de convergencia del PSO global en conjunción con el operador RSVN resulta más efectiva que una topología local. Justamente este punto es uno de los aspectos que serán evaluados en el próximo capítulo.

CAPÍTULO 3. EVALUACIÓN DEL ALGORITMO DE APRENDIZAJE

En este capítulo se presentan experimentos y simulaciones que permite evaluar el rendimiento de los algoritmos descritos en el capítulo anterior (el operador RSVN para incrementar la diversidad en metaheurísticas poblacionales y el algoritmo para mejorar las propiedades de convergencia en MCD Sigmoidales). En el primer caso, se adoptan 25 funciones *benchmark* que permiten evaluar el rendimiento del operador RSVN en espacios con distintas propiedades matemáticas, mientras que para el segundo caso se adoptan seis mapas sigmoidales relativos a la modelación de algunas proteínas del VIH-1 y su relación con la resistencia a los fármacos existentes.

3.1 Evaluación del rendimiento del optimizador

Para evaluar el rendimiento global del algoritmo PSO-RSVN se propone un estudio comparativo que involucra múltiples variantes de PSO, diseñadas para atenuar los efectos de las situaciones de no progreso. A continuación se resumen las principales características de estos métodos:

- ***Bare Bones Particle Swarm*** (Kennedy, 2003): es conocido que las partículas de la población convergen a un promedio pesado entre los mejores registros personales obtenidos durante el proceso de búsqueda, y la mejor solución global encontrada. Sin embargo, con el objetivo de comprender el comportamiento de las partículas e identificar las similitud de PSO con otros algoritmos poblacionales estocásticos, (Kennedy, 2003) presentó una variante que elimina las velocidades de las partículas. En BBPSO la posición de cada individuo se actualiza siguiendo una distribución Gaussiana, tal y como muestra la siguiente ecuación:

$$X_i \sim N\left(\frac{G + P_i}{2}, |P_i - G|\right) \tag{3.1}$$

  Esta versión de PSO funciona bastante bien para la mayoría de los problemas, pero en otros su rendimiento es inferior. La razón de este comportamiento está en la distribución utilizada, por lo que se sugiere utilizar distribuciones con mayor capacidad de exploración.
- ***Comprehensive Learning Particle Swarm*** (Liang et al., 2006): esta variante fue diseñada especialmente para optimizar funciones multimodales, donde la existencia de óptimos locales aumenta la posibilidad de converger de forma prematura. La idea central consiste en eliminar el atractor global G de la ecuación (2.10) de forma tal que la velocidad de cada partícula, y su posición en el espacio, se calculan utilizando el registro personal de los individuos durante la búsqueda, como una alternativa para preservar la diversidad en la población.

Por otra parte, la velocidad de una partícula puede ser influenciada por el registro personal de cualquier agente de la población. La siguiente ecuación muestra la expresión para estimar la velocidad de una partícula X_i, donde $P_{fi(d)}$ define el registro personal que será adoptado para actualizar la d-ésima dimensión del vector velocidad. Durante el proceso de actualización se selecciona un conjunto de partículas (muestras) y luego se genera un número aleatorio R con distribución uniforme. Si $R < LP_i \in [0.05,0.5]$ entonces el agente X_i usará su registro como atractor, sino se selecciona un atractor del conjunto de muestras.

$$V_i^{k+1} = \omega V_i^k + \delta_1\left(P_{fi(d)} - X_i^k\right) \tag{3.2}$$

Otra variante para determinar el atractor consiste en seleccionar aleatoriamente dos partículas de la población (excluyendo la partícula X_i) y aquella con mejor registro personal será usada como el componente $P_{fi(d)}$ de la ecuación (3.2). De forma general, CLPSO es un algoritmo bastante eficiente para solucionar problemas multimodales complejos, pero su rendimiento es significativamente menor en dominios unimodales o multimodales simples.

- ***Opposition based Particle Swarm*** (Wang, 2007): la idea central de OPSO consiste en usar un método de aprendizaje basado en oposición y un operador de Cauchy dinámico, para evitar la convergencia hacia soluciones locales. El aprendizaje basado en oposición fue originalmente presentado por (Tizhoosh, 2006) para problemas de aprendizaje reforzado. Cuando se evalúa la solución de un problema, es posible suponer que la solución opuesta reduce la distancia al óptimo global. Durante la actualización de los individuos, primero se genera una población opuesta y luego se seleccionan los mejores agentes de ambas poblaciones. Este algoritmo se ejecuta de forma probabilística para preservar la habilidad de convergencia del método PSO clásico. Por otra parte, en cada iteración se muta la mejor partícula de la población usando un operador de Cauchy dinámico (Wang, 2007). Este operador se describe como:

$$G_i = G_i + \frac{N(X_m, X_M)}{D}\sum_{j=1}^{D} V_{ij} \tag{3.3}$$

En la fórmula anterior D representa la dimensión del problema, mientras $N(X_m, X_M)$ denota un número del dominio con distribución normal. Experimentos numéricos mostraron que este algoritmo tiene una alta velocidad de convergencia en espacios unimodales, mientras que en espacios multimodales es superior al algoritmo clásico. Sin embargo, existen casos donde la población converge prematuramente a soluciones sub-óptimas.

- ***Quadratic Interpolation based Particle Swarm*** (Pant and Radha, 2007): QPSO introduce un operador cuadrático con la intención de incrementar la diversidad. El operador consiste en un procedimiento de cruzamiento no lineal que utiliza 3 partículas de la bandada como padres. La ecuación (3.4) muestra cómo calcular el hijo X_i para cada dimensión del problema, donde a es el individuo con mínima evaluación, mientras que b y c son dos partículas seleccionadas aleatoriamente de la población actual. Luego, el descendiente X_i sustituye a la partícula con peor evaluación independientemente de la calidad del hijo. De forma general, QPSO es capaz de preservar la diversidad en la bandada, incluso en espacios de alta dimensionalidad, aunque las funciones de prueba utilizadas no sustentan esta conclusión.

$$X_{ij} = \frac{\left(b^{j^2} - c^{j^2}\right)f(a) + \left(c^{j^2} - a^{j^2}\right)f(b) + \left(a^{j^2} - b^{j^2}\right)f(c)}{2[(b^j - c^j)f(a) + (c^j - a^j)f(b) + (a^j - b^j)f(c)]} \tag{3.4}$$

- ***Attraction-Repulsion Particle Swarm*** (Riget and Vesterstrøm, 2004): este algoritmo intenta controlar la diversidad de la población, a partir del movimiento de las partículas. ARPSO usa dos fases para actualizar la posición de los agentes. La fase de atracción consiste en las reglas originales que utiliza PSO, donde las partículas se mueven en dirección a una región definida por el mejor registro de los agentes (incluyendo el atractor global). En la fase de repulsión, la ecuación para actualizar las velocidades se invierte de la siguiente forma:

$$V_i^{k+1} = \omega V_i^k - \delta_1\left(P_i - X_i^k\right) - \delta_2\left(G - X_i^k\right) \tag{3.5}$$

En otras palabras, en la etapa de repulsión las partículas de la población no son atraídas hacia las mejores regiones detectadas durante la búsqueda, mas se mueven en sentido inverso a los atractores personales y globales. Esta estrategia se activa cuando se ha perdido la diversidad en la bandada, concretamente cuando $div(B) < d\_low$. La siguiente ecuación (3.6) muestra cómo calcular $div(B)$, donde $|L|$ representa el tamaño de la mayor diagonal en el espacio de búsqueda, mientras que $\bar{X}$ es la distancia promedio entre los agentes.

$$div(B) = \frac{1}{|B| * |L|}\sum_{i=1}^{|B|}\sqrt{\sum_{j=1}^{D}\left(X_{ij} - \bar{X}_j\right)^2} \tag{3.6}$$

Por otra parte, cuando $div(B) > d\_high$ entonces se adopta la estrategia de atracción, donde las partículas retornan al estado de convergencia hacia los atractores.

Además de las 5 variantes detalladas anteriormente, las simulaciones incluyen el algoritmo PSO global (GBest), el algoritmo PSO local con Topología de Anillo (LBest) y dos implementaciones del método propuesto para reorganizar la población. En el primer caso se utiliza el operador RSVN original descrito en (Nápoles et al., 2012a), mientras que en el segundo caso se adopta el operador LSVN descrito en la sección 2.5. Las principales diferencias entre ambos métodos se refieren a la distribución de probabilidad que se usa para generar las muestras, y sobre todo a la estrategia para garantizar que las muestras sean soluciones factibles.

Para evaluar el rendimiento de estos métodos se usan 25 funciones de dimensión 10, reportadas en (Suganthan et al., 2005). La colección se compone por 5 funciones unimodales ($F_1 - F_5$) y 20 problemas multimodales, desglosadas en 7 problemas básicos ($F_6 - F_{12}$), 2 funciones expandidas ($F_{13} - F_{14}$) y 11 funciones híbridas ($F_{13} - F_{25}$). Estas funciones están desplazadas para asegurar que el óptimo no se encuentre en el centro del espacio de soluciones, lo cual dificulta el proceso de búsqueda. Esta colección de funciones también problemas rotados, no separables, ruidosos, fuera del dominio, con óptimos en la frontera del espacio de búsqueda, etc.

La configuración de parámetros del experimento se resume como sigue: una bandada artificial de 50 partículas, 100 000 evaluaciones de la función objetivo como criterio de parada, coeficientes de aceleración como proponen (Clerc and Kennedy, 2002), mientras que el número máximo de evaluaciones sin progreso $np = 500$, para el caso de la variante PSO que utiliza el operador para reorganizar la bandada. El resto de los algoritmos son configurados como sugieren los artículos originales, para el mismo conjunto de funciones adoptadas en este trabajo.

Las Tablas 3.1 y 3.2 resumen el promedio de la desviación al óptimo global conocido, obtenido por cada optimizador, después de 30 ejecuciones independientes. Concretamente, en la Tabla 3.1 se resumen los errores promedios computados para las funciones unimodales, las multimodales básicas y expandidas ($F_1 - F_{14}$), mientras que la Tabla 3.2 muestra los errores para las funciones compuestas ($F_{15} - F_{25}$). Resulta notable que los errores de BBPSO y PSO-LSVN son los más pequeños, sin embargo, no es posible afirmar que estas diferencias son significativas respecto a las demás variantes, aunque la significación del test de Friedman (Friedman, 1937) para estos resultados (p-valor = 0.0) sugiere que existen diferencias altamente significativas entre todos los algoritmos involucrados en el estudio. Por tal motivo, es necesario usar pruebas no paramétricas más específicas, que permitan comparar los errores en pares de algoritmos.

Tabla 3.1 Error promedio para las funciones unimodales, multimodales básicas y expandidas ($F_1 - F_{14}$).

ID	GBest	LBest	BBPSO	arPSO	QPSO	OPSO	CLPSO	RSVN	LSVN
F_1	9.701E-1	6.331E-1	3.79E-15	1.031E-1	2.396E+0	2.748E+0	3.611E-3	8.670E-7	1.973E-9
F_2	9.770E+2	3.793E+2	1.12E-13	6.362E+2	7.407E+2	3.685E+2	2.202E+1	5.760E-4	6.456E-7
F_3	2.031E+6	1.248E+6	1.666E+5	8.366E+5	1.570E+6	1.377E+6	2.015E+6	3.213E+5	1.114E+5
F_4	7.867E+3	4.690E+3	1.140E-5	7.719E+3	6.846E+3	5.007E+3	3.265E+1	2.163E-1	3.232E-2
F_5	2.070E+3	3.395E+3	2.50E-11	2.152E+3	2.381E+3	1.975E+3	3.147E+2	1.475E+3	3.837E+2
F_6	2.514E+3	4.347E+3	1.142E+0	1.973E+3	2.784E+3	1.483E+3	4.739E+2	9.790E+0	4.299E-1
F_7	1.330E+3	1.398E+3	1.263E+3	1.330E+3	1.346E+3	1.345E+3	1.286E+3	1.298E+3	4.976E+0
F_8	2.017E+1	2.054E+1	2.036E+1	2.177E+1	2.016E+1	2.042E+1	2.037E+1	2.036E+1	2.039E+1
F_9	39.92E+1	2.916E+1	6.533E+0	4.046E+1	3.712E+1	3.912E+1	3.317E+1	3.980E+0	2.004E-1
F_{10}	7.050E+1	4.972E+1	2.368E+1	6.891E+1	7.030E+1	6.366E+1	3.906E+1	2.666E+1	1.767E+1
F_{11}	7.450E+0	6.471E+0	5.006E+0	7.035E+0	7.020E+0	7.028E+0	8.306E+0	5.285E+0	5.109E+0
F_{12}	9.613E+3	3.539E+2	5.225E+2	1.279E+4	1.533E+4	1.145E+4	3.148E+3	6.442E+3	4.507E+2
F_{13}	1.751E+0	1.246E+0	6.607E-1	1.937E+0	1.560E+0	1.550E+0	3.790E+0	7.421E-1	4.892E-1
F_{14}	3.719E+0	3.265E+0	3.148E+0	3.538E+0	3.442E+0	3.276E+0	3.577E+0	2.952E+0	2.616E+0

Tabla 3.2 Error promedio para las funciones compuestas ($F_{15} - F_{25}$).

ID	GBest	LBest	BBPSO	arPSO	QPSO	OPSO	CLPSO	RSVN	LSVN
F_{15}	4.210E+2	3.838E+2	3.104E+2	4.397E+2	4.726E+2	4.240E+2	5.556E+2	2.301E+2	2.350E+2
F_{16}	1.970E+2	1.564E+2	1.428E+2	1.722E+2	1.741E+2	2.029E+2	1.869E+2	2.663E+2	1.286E+2
F_{17}	1.800E+2	2.142E+2	1.457E+2	2.014E+2	2.026E+2	1.888E+2	2.008E+2	1.355E+2	1.422E+2
F_{18}	7.210E+2	3.897E+2	7.128E+2	6.875E+2	8.945E+2	7.831E+2	3.532E+2	7.518E+2	6.664E+2
F_{19}	6.470E+2	4.203E+2	8.662E+2	7.715E+2	9.592E+2	7.571E+2	3.551E+2	5.248E+2	7.176E+2
F_{20}	8.345E+2	5.331E+2	6.634E+2	6.557E+2	7.034E+2	7.827E+2	3.546E+2	5.934E+2	6.149E+2
F_{21}	1.201E+3	1.072E+3	8.933E+2	1.106E+3	1.043E+3	1.001E+3	5.227E+2	7.765E+2	4.955E+2
F_{22}	8.446E+2	7.726E+2	7.882E+2	8.012E+2	8.398E+2	7.033E+2	8.156E+2	7.123E+2	7.891E+2
F_{23}	9.629E+2	8.257E+2	1.100E+3	9.935E+2	1.060E+3	9.983E+2	6.070E+2	8.951E+2	7.982E+2
F_{24}	7.466E+2	2.681E+2	4.069E+2	5.399E+2	6.750E+2	9.066E+2	2.301E+2	3.767E+2	2.000E+2
F_{25}	1.810E+3	1.822E+3	1.735E+3	1.811E+3	1.809E+3	1.681E+3	1.774E+3	1.623E+3	4.255E+2

Para comparar el comportamiento de dos algoritmos concretos (García et al., 2009) recomiendan utilizar la prueba de Wilcoxon (Wilcoxon, 1945). Esta prueba no paramétrica determina si dos muestras representan la misma población, en cuyo caso se acepta la hipótesis H_0. La Tabla 3.3 muestra los p-valores asociados, usando un nivel de significación de 0.05 referente al 95% del intervalo de confianza. En este experimento sólo considera los pares de algoritmos que implican a los métodos PSO-RSVN o PSO-LSVN, el resto de las combinaciones no aportan información relevante al estudio[3]. Por otra parte, los algoritmos tomados como referencia (los listados en las columnas 2 y 6) están ordenados de acuerdo a los rangos medios.

[3] En caso de existir diferencias significativas (p-valor < 0.05) entre dos algoritmo específicos, el resto de los pares que contengan el algoritmo con mayor rango también reportarán diferencias significativas, reduciendo el número de comparaciones necesarias para justificar el estudio.

Tabla 3.3 Resultados del test de Wilcoxon para los pares que involucran el operador RSVN.

Algoritmo1	Algoritmo2	p-valor	Hipótesis H_0	Algoritmo1	Algoritmo2	p-valor	Hipótesis H_0
PSO-LSVN	PSO-RSVN	0.011	Rechazada	PSO-RSVN	PSO-LSVN	0.011	Rechazada
	BBPSO	0.002	Rechazada		BBPSO	0.904	Aceptada
	CLPSO	0.045	Rechazada		CLPSO	0.840	Aceptada
	LBest	0.009	Rechazada		LBest	0.115	Aceptada
	OPSO	0.000	Rechazada		OPSO	0.000	Rechazada
	arPSO	0.000	Rechazada		arPSO	0.000	Rechazada
	GBest	0.000	Rechazada		GBest	0.000	Rechazada
	QPSO	0.000	Rechazada		QPSO	0.000	Rechazada

De los resultados mostrados en la tabla anterior en posible concluir que:

i. El optimizador PSO-LSVN (Nápoles et al., 2014b) es notablemente superior a los demás, incluyendo a la variante original PSO-RSVN (Nápoles et al., 2012a).
ii. No existen razones suficientes para rechazar la hipótesis nula para los siguientes pares de algoritmos: PSO-RSVN vs BBPSO, PSO-RSVN vs CLPSO y PSO-RSVN vs Lbest. Esto sugiere que su desempeño es similar para las 25 funciones estudiadas.

De forma general, las simulaciones y pruebas estadísticas descritas en esta sección indican que la variante PSO-LSVN es una estrategia factible para aliviar los efectos negativos inducidos por los óptimos locales. Incluso, a diferencia de la mayoría de los métodos descritos, PSO-LSVN resulta factible para resolver problemas unimodales que inducen estados de no progreso.

Por otra parte, es presumible suponer que el procedimiento de aprendizaje detallado en la sección anterior involucre estados potenciales de estancamiento o convergencia prematura, debido a las características multimodales del problema de optimización (ej. existen varias soluciones estables que conducen a errores de clasificación diferentes). Este problema resulta peor si se considera la inestabilidad del mapa original que se desea procesar. Por lo tanto, resulta evidente las ventajas que reporta adoptar el algoritmo PSO-LSVN como optimizador.

3.2 Modelación de proteínas del VIH-1 usando MCD Sigmoidales

En esta sección se introduce un caso de estudio concerniente al campo de la Bioinformática, más específicamente a la modelación del mecanismo de resistencia del Virus de Inmunodeficiencia Humano tipo 1 (VIH-1). Este modelo cognitivo fue originalmente publicado por (Nápoles et al., 2013b) para la proteína *proteasa* y luego generalizado por (Grau et al., 2013) para otras proteínas del virus. De estos estudios resultaron 6 mapas con alta habilidad de predicción, sin embargo, los autores no garantizan la convergencia de los sistemas resultantes.

El proceso de replicación del VIH involucra tres proteínas fundamentales: la *proteasa*, la *reverso transcriptasa* y la *integrasa*. Por ejemplo, la proteína *proteasa* es un enzima que el virus necesita para completar el proceso de autocopia (replicación) que produce nuevas mutaciones, capaces de infectar otras células. Durante este proceso, el virus produce largas cadenas de proteínas que la *proteasa* fragmenta en fracciones más pequeñas para formar las proteínas y enzimas que ayudan a construir las nuevas copias del virus. La secuencia de la proteína *proteasa* está definida por 99 posiciones, donde cada posición codifica un aminoácido específico.

Para modelar este complejo sistema biológico se asume que cada posición de la secuencia denota un concepto, obteniéndose un MCD con 99 neuronas de entrada. Por otra parte, es conocido que algunos sitios de la secuencia no están relacionados con la resistencia a los fármacos existentes (característica que se desea modelar) y su inclusión en el modelo aumentaría la complejidad del sistema final. Basados en este conocimiento los autores utilizaron una selección de posiciones reportada en (Cane, 2007) que han sido previamente vinculadas al mecanismo de resistencia del virus, utilizando procedimientos experimentales (ej. ensayos clínicos).

Además de los conceptos que codifican las posiciones relevantes, el modelo incluye una neurona de respuesta (*resistencia*) que permite calcular el grado de resistencia de la secuencia de entrada. Una vez que se determinan las neuronas, es necesario definir las conexiones que existen entre los conceptos del mapa. La Figura 3.1 muestra la topología general del mapa, donde se asume que el grado de resistencia de una mutación está condicionado por la interacción entre los aminoácidos y por el aporte a la resistencia de cada posición de la secuencia. En (Nápoles et al., 2013b) se presentó un método de aprendizaje poblacional que usaba datos históricos para estimar de forma automática la relación causal entre las posiciones de la secuencia.

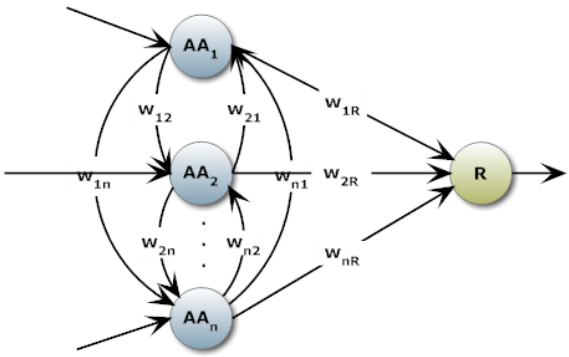


Figura 3.1 Mapa Cognitivo Difuso para modelar el mecanismo de resistencia de la proteína *proteasa* del VIH. En esta topología los conceptos AA_i denotan las posiciones de la secuencia, mientras que la neurona R cuantifica el grado de resistencia.

Otro aspecto fundamental es la codificación de cada mutación como un estímulo de las neuronas de entrada. En este modelo computacional cada aminoácido se codifica utilizando su energía de contacto (Miyazawa and Jernigan, 1999) normalizada. La energía de contacto de un aminoácido es un descriptor biológico que se corresponde con la estructura tridimensional de la proteína y se calcula estadísticamente a partir de diversas secuencias. Concretamente, el nivel de activación de un concepto de entrada, para un aminoácido específico, se calcula como la división de la energía de contacto asociada al aminoácido, por la máxima energía de contacto.

Por ejemplo, suponga que en una posición de una mutación se observa el aminoácido V (Valina, 7.182885) entonces el valor de activación del segundo concepto será: 7.182885/8.783541, donde 8.783541 denota la mayor energía de contacto (Lisina, 8.783541). De esta forma se generaliza el esquema de activación para los demás conceptos del modelo y luego se propaga esta información a través del mapa, hasta que se alcance un número máximo de ciclos o el sistema converja a un estado atractor de equilibrio. Si el grado de activación del concepto *resistencia* es menor que un punto de corte biológico establecido, entonces la mutación será susceptible al inhibidor, en otro caso se considera que la mutación ha desarrollado resistencia al fármaco.

La Tabla 3.4 resume la exactitud de la clasificación para seis inhibidores: Amprenavir (APV), Atazanavir (ATV), Indinavir (IDV), Nelfinavir (NFV), Ritonavir (RTV) y Saquinavir (SQV). El experimento incluye 10 clasificadores: Máquinas de Soporte Vectorial usando diferentes núcleos (SVM1, SVM2, SVM3, SVM4, SVM5), una Red Neuronal Artificial (ANN), una Red Neuronal MLP, una Red Neuronal Recurrente (BRNN), un multi-clasificador (MEHI) y el modelo descrito (FCM, por sus siglas en inglés), donde se adoptó un algoritmo de aprendizaje poblacional basado en PSO-RSVN tal como describe (Grau et al., 2013). Observe que el FCM alcanza una exactitud superior al 95%, superando a los demás métodos. Sin embargo, el esquema de aprendizaje que se utilizó para estimar la causalidad no garantiza la convergencia del mapa.

Tabla 3.4 Exactitud de la clasificación para los inhibidores de la *proteasa* considerando dos clases (Grau, 2012).

Inhibidor	SVM1	SVM2	SVM3	SVM4	SVM5	ANN	MLP	BRNN	MEHI	FCM
ATV	0.78	0.71	0.68	0.72	0.70	-	0.80	0.81	0.84	0.96
SQV	0.87	0.80	0.69	0.85	0.82	0.91	0.85	0.91	0.85	0.95
LPV	0.88	0.85	0.85	0.88	0.85	0.92	0.92	0.94	0.93	0.98
RTV	0.91	0.84	0.79	0.92	0.86	0.96	0.90	0.94	0.99	0.97
IDV	0.91	0.86	0.83	0.92	0.88	0.95	0.86	0.92	0.97	0.99
NFV	0.84	0.75	0.70	0.84	0.80	0.95	0.86	0.93	0.96	0.95

3.3 Evaluación del algoritmo de aprendizaje

A continuación se formalizan varios experimentos que permitirán evaluar el comportamiento del algoritmo de aprendizaje[4]. La primera de estas simulaciones consiste en explorar la convergencia de los 6 mapas descritos en el capítulo anterior. Las Figuras 3.2, 3.3 y 3.4 muestran el espacio de activación del concepto resistencia durante el tiempo para una mutación aleatoria tomada de cada base de conocimiento. En estas figuras, la línea discontinua denota la respuesta del mapa usando la misma función sigmoidal con $\lambda = 9$ para todos los conceptos, mientras que la línea continua representa la respuesta del mapa utilizando una función de transformación específica para cada concepto del mapa. Como fue discutido, la familia de funciones sigmoidales se construye a partir del genoma estimado por el optimizador según la función objetivo diseñada.

De estas simulaciones es posible concluir que el método mejora las características de estabilidad del sistema final, aunque el sistema final no sea totalmente estable. De hecho, estas gráficas son ejemplos representativos del comportamiento del algoritmo. Es importante resaltar que en mapas utilizados en toma de decisiones es suficiente monitorear la convergencia del concepto decisión, pues este nodo abarca el comportamiento de todo el sistema. En otras palabras, resulta altamente improbable que un concepto decisión sea estable si el resto del sistema es caótico o cíclico, pues eventualmente este comportamiento se reflejará en la salida final del mapa.

De forma general, luego de aplicar el algoritmo, se observan cuatro escenarios:

i. La velocidad de convergencia del mapa mejora en mapas estables (Figuras 3.2a y b).
ii. Se eliminan los patrones cíclicos, obteniéndose mapas estables (Figuras 3.3a y b).
iii. Se reduce significativamente la variabilidad en las respuestas del sistema, aunque el mapa final permanece *ligeramente caótico* (Figura 3.4a).
iv. Se elimina el comportamiento caótico del mapa, obteniéndose un sistema perfectamente estable, sin comprometer la respuesta esperada (Figura 3.4a).

El lector puede percibir que el resultado menos favorable concierne al tercer escenario, donde no es posible garantizar la estabilidad absoluta del mapa (el sistema produce exactamente la misma salida). No obstante, en problemas de toma de decisiones donde la decisión está definida por una partición del espacio de activación, este resultado es positivo. Para comprender esta afirmación es necesario definir que se considera un mapa *ligeramente caótico*.

[4] Los Anexos A y B de la propuesta detallan aspectos de la implementación del algoritmo y su incorporación en la herramienta de simulación y experimentación computacional FCM TOOL (León, et al., 2011).

Definición 3.1: Suponga que $\mathcal{M}$ un MCD Sigmoidal con un concepto decisión C_i. Una clase de decisión $d_n \in D$ es una partición cerrada del espacio de activación de C_i acotada inferiormente por L_n y superiormente por R_n. Esta clase se determina por la función $\psi: [0,1] \rightarrow D$, suponiendo que D representa el conjunto de todas las posibles clases de decisión.

Definición 3.2: Sea $\mathcal{M}$ un MCD Sigmoidal con un nodo decisión C_i. El mapa $\mathcal{M}$ es *ligeramente caótico* si $\nexists t_k \in \mathbb{N} : A_i^{(t)} = A_i^{(t+1)}$ pero $\exists t_k \in \mathbb{N} : \psi\left(A_i^{(t)}\right) = \psi\left(A_i^{(t+1)}\right), \forall t > t_k$.

Concretamente, un mapa *ligeramente caótico* es un sistema que produce vectores numéricamente diferentes durante el tiempo, pero que genera la misma clase de decisión. Por ejemplo, el caso de estudio de la resistencia antiviral de la proteína protesta del VIH-1 tiene dos clases: susceptible y resistente. Estas clases para el fármaco ATV se determinan como sigue:

$$\psi(x) = \begin{cases} Susceptible, x \leq 3.5/500 \\ Resistente, x > 3.5/500 \end{cases} \tag{3.7}$$

En el caso que describe la Figura 3.3 para la mutación "FKLDVFMIIVVSVTVNML" el sistema oscilará de forma cíclica entre un estado resistente y susceptible, si se considera el mapa original. Sin embargo, un mapa ligeramente caótico donde $L_n = 0$ y $R_n = 3.5/500$ siempre producirá un estado discreto susceptible, no importa las variaciones en el intervalo. Por tanto, incluso el peor escenario observado podría producir una mejora en la convergencia del mapa.

Por otra parte, si la salida del sistema es continua (ej. un problema de toma de decisiones donde se desea ordenar las decisiones) las definiciones anteriores no son válidas, pues para este tipo de situaciones es necesario considerar $|D|$ conceptos decisión. En cualquier caso, los escenarios que ilustran las Figuras 3.2, 3.3 y 3.4 siguen siendo positivos, ya que reducen la variabilidad global en las respuestas del sistema y por tanto el error absoluto al valor deseado.

Sin embargo, el autor prefiere centrarse en los problemas de clasificación, pues incluso los MCD con funciones sigmoidales son inexactos en problemas de regresión (Song et al., 2010). Además, el método de aprendizaje descrito asume que el sistema original tiene una adecuada capacidad de predicción, aunque sus respuestas sean variables durante el tiempo.

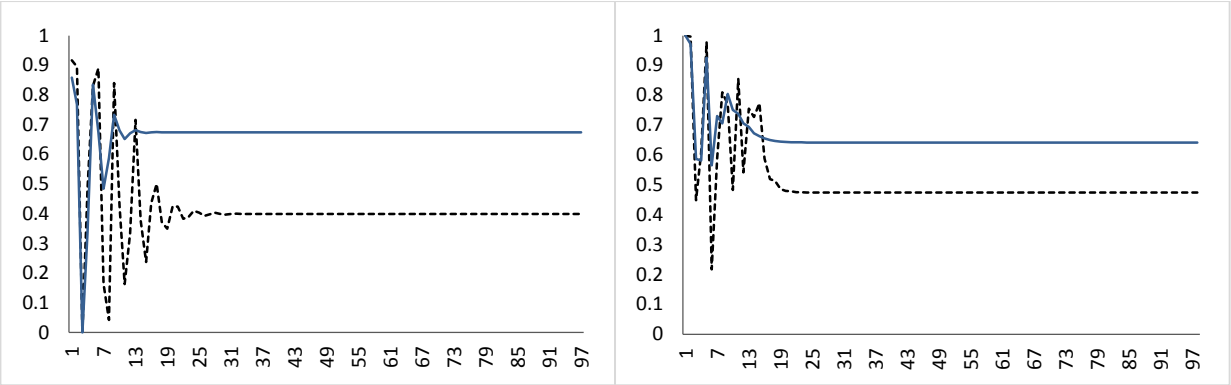


Figura 3.2 Nivel de activación de la neurona decisión para a) fármaco IDV b) fármaco RTV. En ambos ejemplos se mejora la velocidad de convergencia del sistema modelado. Observe que el mapa converge a atractores de equilibrio alternativos.

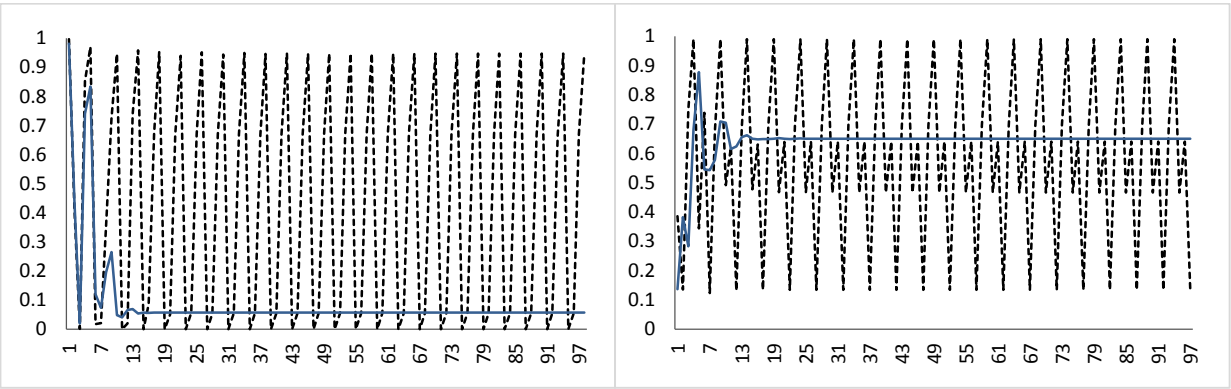


Figura 3.3 Nivel de activación de la neurona decisión para a) fármaco ATV b) fármaco APV. En ambos casos el mapa muestra comportamientos diferentes para la misma entrada, sin embargo, el algoritmo es capaz de eliminar los patrones cíclicos.

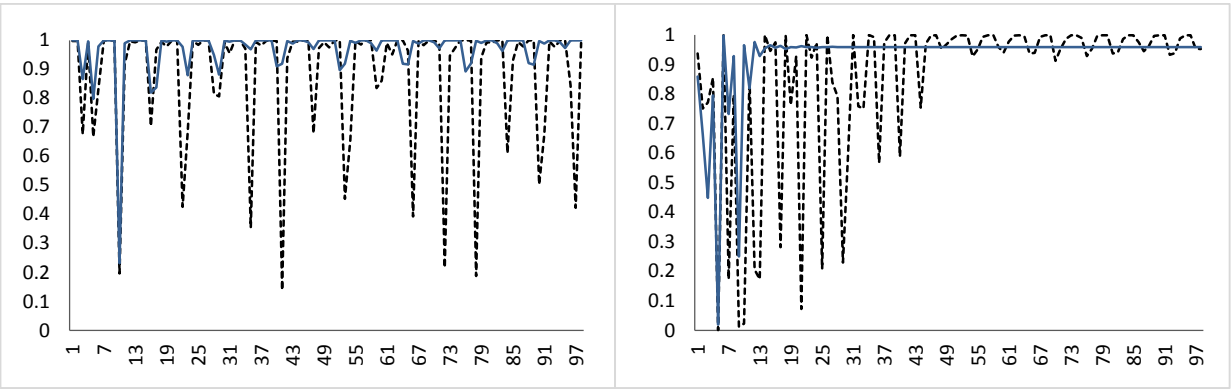


Figura 3.4 Nivel de activación de la neurona decisión para a) fármaco SQV b) fármaco NFV. En el primer caso la variabilidad del sistema se reduce significativamente, mientras que en el segundo caso se obtiene un mapa perfectamente estable.

Otro aspecto positivo observado está relacionado con el incremento en la exactitud de predicción del mapa optimizado. Como fue mencionado antes, el objetivo del algoritmo de aprendizaje es la reducción de la variabilidad en las respuestas del sistema, sin modificar la matriz de pesos y sin alterar la exactitud de la clasificación. No obstante, en algunas situaciones particulares el sistema optimizado converge a un "mejor atractor" independientemente de que las mejoras detectadas en los porcientos de clasificación no se consideran suficientemente significativas.

Por ejemplo, es conocido que la mutación FKLDVFMIIVVSVTVNML es altamente resistente al fármaco IDV como sugiere la Figura 3.2 (en este caso el corte biológico para determinar la clase mutación es susceptible o resistente coincide con el fármaco ATV). Este mapa era originalmente convergente, sin embargo, luego de aplicar el algoritmo de aprendizaje, el sistema converge a un estado atractor de equilibrio alternativo "más resistente". En otras palabras, el atractor alternativo tiene asociado un mayor grado de resistencia. Por otra parte, es destacable que la nueva solución requiere un menor número de iteraciones para converger al punto de equilibrio.

Las Figuras 3.5, 3.6 y 3.7 ilustran el comportamiento de la diversidad de la población durante el progreso de la optimización, donde el eje x denota las generaciones, mientras que el eje y denota la medida de diversidad formalizada a través de la ecuación (3.6). Observe que para los fármacos IDV y RTV el algoritmo PSO-LSVN tiene un comportamiento idéntico al algoritmo PSO, pues el operador RSVN permanece inactivo durante el ciclo generacional (no se detectaron estados de no progreso). Sin embargo, para los fármacos restantes el procedimiento de reorganización de la bandada se activa al menos una vez (como ilustran los picos en la diversidad).

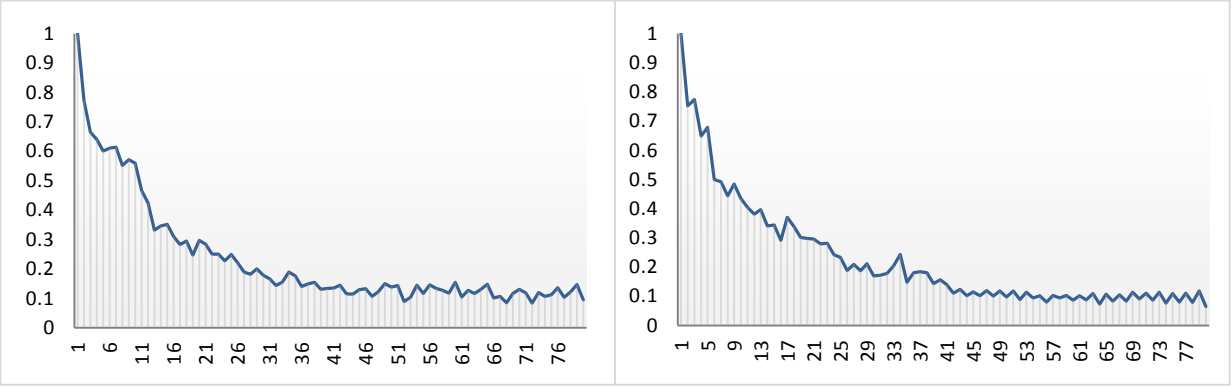


Figura 3.5 Comportamiento de la diversidad de la bandada para a) fármaco IDV b) fármaco RTV. En ambos casos el algoritmo PSO-RSVN funciona igual que el algoritmo PSO clásico pues el operador RSVN permanece inactivo durante la búsqueda.

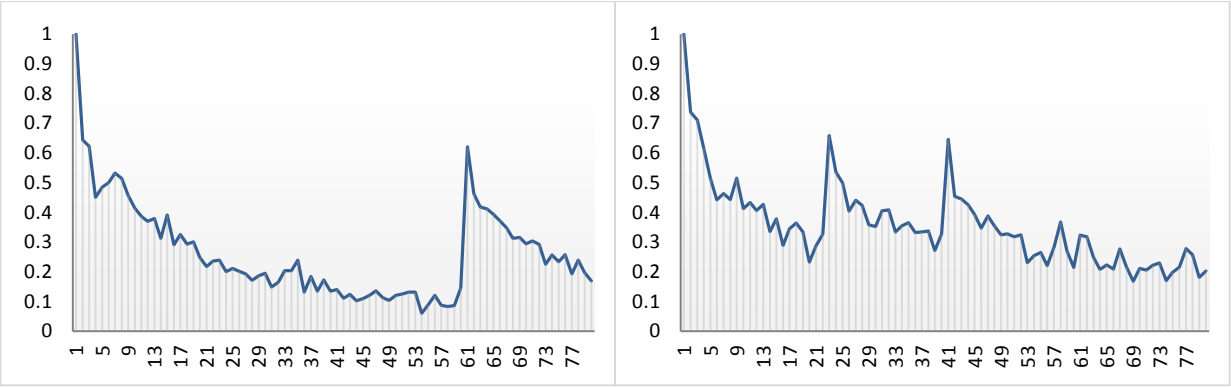


Figura 3.6 Comportamiento de la diversidad de la bandada para a) fármaco ATV b) fármaco APV. En el primer caso se activa el operador RSVN en la iteración 61, mientras que en el segundo escenario la bandada se reorganiza en dos ocasiones.

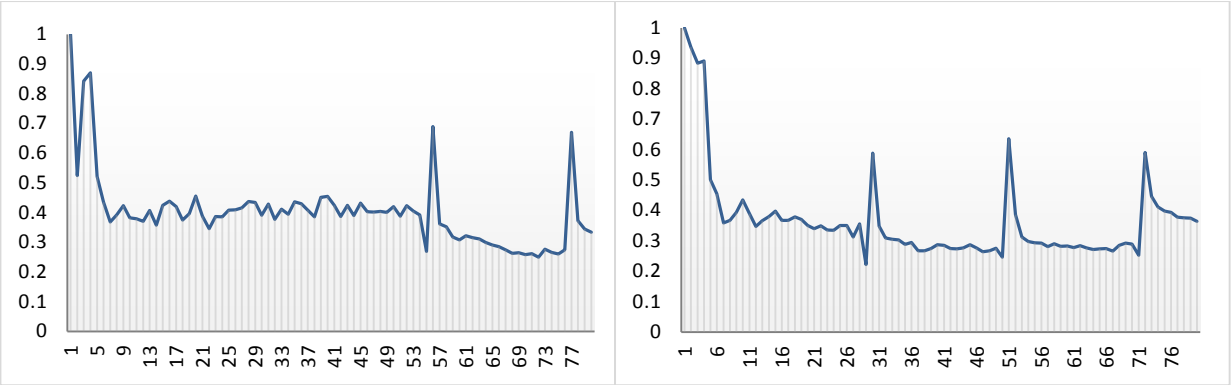


Figura 3.7 Comportamiento de la diversidad de la bandada para a) fármaco SQV b) fármaco NFV. En el primer caso el operador se activa en dos ocasiones, mientras que en el segundo escenario la población se reorganiza en las iteraciones 31, 51 y 72.

Con el objetivo de generalizar los escenarios observados en las Figuras 3.2-3.4, a continuación se describe un tercer experimento. El objetivo de esta simulación es determinar si las mejoras en la estabilidad del sistema son realmente significativas. Como primer paso, se computa la diferencia absoluta entre la variabilidad del mapa original y el promedio de las variabilidades del mapa que se obtiene luego de aplicar el método de aprendizaje 10 veces, para cada mutación de la base de conocimiento que se está analizando. En la práctica esto es equivalente a calcular el valor de que obtiene la función objetivo (2.4) para cada mutación individual (k=1). Estos valores se calculan para cada fármaco, resultando 6 bases diferentes. La Tabla 3.5 resume los resultados de la prueba de Wilcoxon (Wilcoxon, 1945) para cada una de las bases conformadas.

Tabla 3.5 Resultados del test de Wilcoxon (***p*-valor**) para cada inhibidor, donde ***ϕ*-valor** representa la variabilidad en las respuestas del mapa original ($\mathcal{M}_0$) y para el sistema ajustado ($\mathcal{M}_F$) considerando todas las instancias.

Inhibidor	Mutaciones	p-valor	Hipótesis H_0	ϕ-valor ($\mathcal{M}_0$)	ϕ-valor ($\mathcal{M}_F$)
APV	96	0.000	Rechazada	39.09962	1.98303
ATV	69	0.000	Rechazada	13.84345	2.43406
IDV	137	0.038	Rechazada	10.58505	3.78721
RTV	151	0.024	Rechazada	19.19583	7.52529
NFV	204	0.000	Rechazada	19.59037	2.83523
SQV	139	0.010	Rechazada	13.92619	6.15873

Estos resultados confirman la existencia de diferencias altamente significativas, para un nivel de significación de 0.05 correspondiente al 95% del intervalo de confianza. Por lo tanto, en todos los casos se rechaza la hipótesis conservativa. Además de los p-valores asociados a la prueba de Wilcoxon, la tabla anterior muestra el valor de la función objetivo (2.4) para todas las instancias de cada base de conocimiento, considerando el sistema original $\mathcal{M}_0$ y el mapa $\mathcal{M}_F$ resultante del algoritmo de aprendizaje, para la mejor partícula estimada por el optimizador.

3.4 Comparación con otros enfoques

El último experimento de este capítulo está orientado a la comparación del algoritmo EFS con los métodos analíticos discutidos en la Sección 1.4. Esta simulación es bastante simple y tiene un carácter preferentemente cualitativo. Las siguientes tablas ilustran las seis matrices de confusión asociadas al MCD descrito en la sección 3.2, usando el Teorema 1 (Boutalis et al., 2009), usando el Teorema 3 (Knight et al., 2014) y usando el algoritmo propuesto. Note que el Teorema 2 no es aplicable es este mapa, pues la topología no considera *conceptos de entrada*.

En todos los casos el mapa que resulta del Teorema 1 converge a un mismo punto fijo, que puede ser susceptible (S) o resistente (R). Esto significa que el MCD siempre clasificará a las nuevas mutaciones de la misma forma, no importa la naturaleza de la secuencia. En el caso del sistema que resulta del Teorema 3 este comportamiento es incluso peor, pues todas las instancias siempre son etiquetadas como resistentes. Sin embargo, los mapas optimizados que resultan del algoritmo preservan la distribución de los objetos según las clases de decisión. Esto muestra que el método EFS estima al menos un punto de equilibrio por cada clase de decisión.

Tabla 3.6 Matriz de confusión obtenida para el inhibidor APV.

Clase	**Usando el Teorema 1**		**Usando el Teorema 3**		**Usando el método EFS**	
	S	R	S	R	S	R
S	0	22	0	22	20	2
R	0	74	0	74	2	72

Tabla 3.7 Matriz de confusión obtenida para el inhibidor ATV.

Clase	**Usando el Teorema 1**		**Usando el Teorema 3**		**Usando el método EFS**	
	S	R	S	R	S	R
S	11	0	0	11	10	1
R	58	0	0	58	2	56

Tabla 3.8 Matriz de confusión obtenida para el inhibidor IDV.

Clase	**Usando el Teorema 1**		**Usando el Teorema 3**		**Usando el método EFS**	
	S	R	S	R	S	R
S	0	26	0	26	25	1
R	0	111	0	111	0	111

Tabla 3.9 Matriz de confusión obtenida para el inhibidor RTV.

Clase	**Usando el Teorema 1**		**Usando el Teorema 3**		**Usando el método EFS**	
	S	R	S	R	S	R
S	15	0	0	15	15	0
R	136	0	0	136	4	132

Tabla 3.10 Matriz de confusión obtenida para el inhibidor NFV.

Clase	Usando el Teorema 1		Usando el Teorema 3		Usando el método EFS	
	S	R	S	R	S	R
S	17	0	0	17	8	9
R	187	0	0	187	3	184

Tabla 3.11 Matriz de confusión obtenida para el inhibidor SQV.

Clase	Usando el Teorema 1		Usando el Teorema 3		Usando el método EFS	
	S	R	S	R	S	R
S	24	0	0	24	19	5
R	115	0	0	115	2	113

Resulta justo mencionar que los resultados teóricos que involucran los Teoremas 1, 2 y 3 pueden ser muy útiles en contextos de simulación. Por ejemplo, en (Knight et al., 2014) se utilizaron los resultados del Teorema 3 para modelar la región de Huber[5]. Este problema no involucra ningún problema toma de decisiones (al menos con el enfoque original). Aquí las simulaciones están orientadas a determinar aquellas entidades más relevantes para la región, de forma tal que no se altere el equilibrio del mapa. Dado que el punto atractor de equilibrio es único como garantiza el Teorema 3, el ordenamiento de las neuronas del mapa será invariable, independientemente de los cambios en las condiciones iniciales del sistema modelado.

No obstante, esta aplicación no cambia el hecho que los resultados teóricos que se conocen hasta el momento no son aplicables en contextos donde se requiere simular nuevos escenarios. Incluso los métodos de aprendizaje Hebbianos sufren este inconveniente. De hecho, si la matriz de pesos de un MCD para toma de decisiones fuera estimada utilizando un algoritmo Hebbiano, entonces probablemente todos los casos serían etiquetados de la misma forma, aunque la matriz de pesos causales estimada desencadenara la inferencia en un punto de equilibrio. Esto claramente ratifica la novedad del método para mejorar la estabilidad en MCD Sigmoidales que ha sido presentado, discutido y validado durante el desarrollo de esta propuesta.

[5] La región de Hubert es una amplia zona industrial en el Reino Unido. Esta zona abarca diversas industrias que van desde el procesamiento de pescado hasta la refinería de petróleo. De hecho, la región de Hubert proporciona una parte sustancial del gas nacional que consume el Reino Unido. Actualmente la región está inmersa en un proceso de perfeccionamiento que brindará nuevas oportunidades para toda la Unión Europea.

3.5 Conclusiones parciales

Durante el capítulo los experimentos fueron dirigidos hacia tres direcciones. Primero, se estudió el rendimiento del operador RSVN en situaciones de no progreso (estancamiento o convergencia prematura) que requieren una estrategia para reorganizar la bandada. Los resultados numéricos mostraron que esta alternativa es efectiva en espacios unimodales y multimodales, superando a los algoritmos reportados en la literatura para atenuar este problema. Esto incluye variantes que usan una organización topológica de la población. Asimismo, resulta claro que la distribución de Lévy es más adecuada para generar las muestras aleatorias.

Segundo, las simulaciones referentes al método de aprendizaje mostraron que el nuevo algoritmo es capaz de mejorar la convergencia en MCD Sigmoidales. Concretamente se observaron cuatro escenarios: (i) es posible acelerar la convergencia de mapas estables, (ii) es posible encontrar una configuración estable a partir de un mapa cíclico, (iii) es posible encontrar un atractor a partir de un mapa totalmente caótico y (iv) se reduce la variabilidad en las respuestas del mapa, aunque no se logre encontrar un atractor de equilibrio. Sin embargo, no resulta disparatado suponer que este último escenario pudiera ser el resultado de un óptimo local. De cualquier forma, el sistema final es más estable, sin sacrificar la capacidad de predicción del mapa original.

Finalmente, los experimentos realizados sobre los seis mapas sigmoidales adoptados como caso de estudio demostraron que los métodos analíticos existentes no son aplicables en mapas usados en problemas de clasificación de patrones. Esto, en última instancia, ratifica el campo de acción de las técnicas de Inteligencia Artificial, o sea, en problemas muy complejos donde no se conoce la solución exacta o en situaciones donde el costo de encontrar la solución es excesivo, incluso para los medios de cómputo actuales. La solución aproximada discutida en esta propuesta es una alternativa a la no existencia de un procedimiento analítico que permita garantizar la estabilidad de los MCD Sigmoidales usados en tareas de clasificación.

CONCLUSIONES

A pesar de algunos esfuerzos de la comunidad científica, el problema de la estabilidad en MCD Sigmoidales complejos resulta un problema de investigación abierto. Los enfoques adoptados en esta dirección son mayormente analíticos. Sin embargo, en este trabajo se presentó un método de aprendizaje supervisado basado en Inteligencia Colectiva que permite mejorar las propiedades de convergencia de MCD Sigmoidales usados en tareas de clasificación de patrones.

Básicamente el algoritmo consiste en estimar una función sigmoidal para cada neurona del mapa, de lugar de usar la misma función de umbral para todas las neuronas, con el objetivo de reducir la variabilidad en las respuestas del mapa. Esto permite mejorar las propiedades de convergencia del sistema, aunque no hay garantías que se encuentre un punto atractor de equilibrio. A pesar de esto, el sistema final es ciertamente más estable. Por otra parte, este algoritmo no compromete la capacidad del mapa optimizado para clasificar nuevos patrones. Resulta relevante resaltar que el nuevo método ha sido diseñado para mapas usados en tareas de predicción, y según su naturaleza supervisada necesita ejemplos que le permitan preservar esta característica.

En principio es posible estimar la inclinación adecuada de cada función sigmoidal con cualquier metaheurística poblacional, incluso usando técnicas de programación matemática. Sin embargo, los métodos de búsqueda poblaciones son capaces de encontrar soluciones cercanas al óptimo en un tiempo de ejecución razonable. A pesar de esta ventaja, estos algoritmos son muy sensible a las soluciones locales, por lo que el operador RSVN resultar una alternativa factible para escapar de estados de no progreso. Esta hipótesis fue demostrada estadísticamente para 25 problemas con distintas propiedades matemáticas, usando PSO como metaheurística.

Las simulaciones y experimentos numéricos mostraron que, para el caso de estudio seleccionado, el algoritmo es capaz de mejorar la convergencia del mapa, sin afectar su capacidad para predecir nuevos patrones. Esto no ocurre si se utilizan los métodos analíticos existentes, pues en ese caso todas las instancias serán clasificadas de la misma forma, no importa la naturaleza de los objetos que se desean clasificar. Por otra parte, tal vez se podría modificar el algoritmo para garantizar la existencia y unicidad de un estado de equilibrio para cada clase de decisión, usando resultados de los métodos analíticos existentes. Pero esto es solo una conjetura.

RECOMENDACIONES

1. Evaluar el rendimiento del algoritmo de aprendizaje discutido para otros casos de estudio, principalmente para mapas utilizados en problemas de regresión.
2. Proponer un nuevo algoritmo, inspirados en las ideas discutidas, para estimar matrices de pesos estables que garanticen la predicción de nuevos patrones.

REFERENCIAS BIBLIOGRÁFICAS

AXELROD, R. 1976. *Structure of Decision: The Cognitive Maps of Political Elites,* Princenton University Press.

BOUTALIS, Y., KOTTAS, T. L. & CHRISTODOULOU, M. 2009. Adaptive estimation of fuzzy cognitive maps with proven stability and parameter convergence. *IEEE Transactions on Fuzzy Systems,* 17**,** 874–889.

BRATTON, D. & KENNEDY, J. Defining a Standard for Particle Swarm Optimization. IEEE Swarm Intelligence Symposium, 2007.

BUENO, S. & SALMERON, J. L. 2009. Benchmarking main activation functions in Fuzzy cognitive maps. *Expert Systems with Applications,* 36**,** 5221–5229.

BURDEN, R. & FAIRES, D. 2002. *Numerical Analysis,* Thomson Learning.

CANE, P. 2007. Identification of accessory mutations associated with high-level resistance in HIV-1 reverse transcriptase. *AIDS,* 21**,** 447–455

CHEN, Y., MAZLACK, L. & LU, L. Learning fuzzy cognitive maps from data by ant colony optimization. Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference, 2012. ACM, 9–16.

CLERC, M. & KENNEDY, J. 2002. The Particle Swarm–Explosion, Stability, and Convergence in a Multidimensional Complex Space. *IEEE Transactions on Evolutionary Computation,* 6, 58–73.

DICKERSON, J. A. & KOSKO, B. 1994. Virtual worlds as fuzzy cognitive maps. *Presence,* 3**,** 173–189.

EDEN, C. 1988. Cognitive Mapping: a review. *European Journal of Operational Research,* 36**,** 1–13.

FRIEDMAN, M. 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association,* 32**,** 674–701.

GARCÍA, S., MOLINA, D., LOZANO, M. & HERRERA, F. 2009. A study on the use of nonparametric tests for analyzing the evolutionary algorithms' behavior: A case study on the CEC'2005 special session on real parameter optimization. *Journal of Heurisitcs,* 15**,** 617–644.

GRAU, I., NÁPOLES, G. & GARCÍA, M. M. Predicting HIV-1 Protease and Reverse Transcriptase Drug Resistance Using Fuzzy Cognitive Maps. CIARP, 2013. 190–197.

HUERGA, A. V. A balanced differential learning algorithm in fuzzy cognitive maps. 16th International Workshop on Qualitative Reasoning, 2002.

KANNAPPAN, A., TAMILARASI, A. A. & PAPAGEORGIOU, E. I. 2011. Analyzing the performance of fuzzy cognitive maps with non-linear Hebbian learning algorithm in predicting autistic disorder. *Expert Systems with Applications,* 38.

KENNEDY, J. Bare bones particle swarms. Swarm Intelligence Symposium, 2003. 80–87.

KENNEDY, J. & EBERHART, R. Particle Swarm Optimization. Proceedings of the 1995 IEEE International Conference on Neural Networks, 1995. 1942–1948.

KENNEDY, J. & MENDES, R. 2006. Neighborhood topologies in fully informed and best-of-neighborhood particle swarms. *IEEE Transactions on Systems, Man, and Cybernetics,* 36**,** 515–519.

KNIGHT, C. J. K., LLOYD, D. J. B. & PENN, A. S. 2014. Linear and sigmoidal fuzzy cognitive maps: An analysis of fixed points. *Applied Soft Computing,* 15**,** 193–202.

KOSKO, B. 1984. Fuzzy Cognitive Maps. *International Journal of Approximate Reasoning***,** pp. 377–393.

KOSKO, B. 1986. Fuzzy Cognitive Maps. *International Journal of Man-Machine Studies,* 24**,** 65–75.

KOSKO, B. 1988. Hidden patterns in combined and adaptive knowledge networks. *International Journal of Aproximate Reasoning,* 2**,** 377–393.

KOSKO, B. 1992. *Neural Networks and Fuzzy systems, a dynamic system approach to machine intelligence*, Prentice-Hall.

KOSKO, B. 1998. Global stability of generalized additive fuzzy systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews,* 28**,** 441–452.

KOULOURIOTIS, D. E., DIAKOULAKIS, I. E. & EMIRIS, D. M. Learning fuzzy cognitive maps using evolution strategies: a novel schema for modeling and simulating high-level behavior. IEEE Congress on Evolutionary Computation, 2001. 364–371.

LEÓN, M., NÁPOLES, G., BELLO, R., MKRTCHYAN, L., DEPAIRE, B. & VANHOOF, K. 2013. Tackling Travel Behaviour: An approach based on Fuzzy Cognitive Maps. *International Journal of Computational Intelligence Systems,* 6**,** 1012–1039.

LI, X. 2010. Niching Without Niching Parameters: Particle Swarm Optimization Using a Ring Topology. *IEEE Transactions on Evolutionary Computation,* 14**,** 150–169.

LIANG, J. J., QIN, A. K., SUGANTHAN, P. N. & BASKAR, S. 2006. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Transactions on Evolutionary Computation,* 10**,** 281–295.

MIAO, Y. & LIU, Z. Q. 2000. On causal inference in fuzzy cognitive maps. *IEEE Transactions on Fuzzy Systems,* 8**,** 107–119.

MIYAZAWA, S. & JERNIGAN, R. L. 1999. Contacts energies Self-Consistent Estimation of Inter-Residue Protein Contact Energies Based on an Equilibrium Mixture Approximation of Residues. *PROTEINS: Structure, Function, and Genetics,* 34**,** 49–68.

NÁPOLES, G., BELLO, R. & VANHOOF, K. 2014a. Global stability improvements on adjusted Sigmoid Fuzzy Cognitive Maps. *Intelligent Data Analysis* (aceptado).

NÁPOLES, G., GRAU, I., BELLO, M. & BELLO, R. 2014b. Towards swarm diversity: Random Sampling in Variable Neighborhoods procedure using a Lévy distribution *Computación y Sistemas,* 18**,** 79–95.

NÁPOLES, G., GRAU, I. & BELLO, R. 2012a. Constricted Particle Swarm Optimization based Algorithm for Global Optimization. *Polibits,* 46**,** 5–11.

NÁPOLES, G., GRAU, I. & BELLO, R. Particle Swarm Optimization with Random Sampling in Variable Neighbourhoods for solving Global Minimization Problems. *In:* DORIGO, M., ed. ANTS 2012, 2012b. 352–353.

NÁPOLES, G., GRAU, I., BELLO, R., FALCON, R. & ABRAHAM, A. Self-adaptive Differential Particle Swarm using a Ring Topology for Multimodal Optimization. ISDA, 2013a.

NÁPOLES, G., GRAU, I., LEÓN, M. & GRAU, R. Modelling, aggregation and simulation of a dynamic biological system through Fuzzy Cognitive Maps. *In:* BATYRSHIN, I. & MENDOZA, M., eds. MICAI 2012, 2013b. Springer, 188–199.

NÁPOLES, G., GRAU, I., PÉREZ-GARCÍA, R. & BELLO, R. Learning of Fuzzy Cognitive Maps for simulation and knowledge discovery. EUREKA, 2013c.

PANT, M. & RADHA, T. A new Particle Swarm Optimization with quadratic crossover. International Conference on Advanced Computing and Communications, 2007. 81–86.

PAPAGEORGIOU, E. & SALMERON, J. L. 2013. A Review of Fuzzy Cognitive Maps Research During the Last Decade. *IEEE Transactions on Fuzzy Systems,* 21**,** 66–79.

PAPAGEORGIOU, E. I. 2011. Review study on Fuzzy Cognitive Maps and their applications during the last decade. *IEEE International Conference on Fuzzy Systems.*

PAPAGEORGIOU, E. I. 2012. Learning Algorithms for Fuzzy Cognitive Maps – A Review Study. *IEEE Transactions on Systems, Man, and Cybernetics,* 42**,** 150–163.

PAPAGEORGIOU, E. I., STYLIOS, C. D. & GROUMPOS, P. P. Fuzzy cognitive map learning based on nonlinear Hebbian rule. Australian Conference on Artificial Intelligence, 2003a. 256–268.

PAPAGEORGIOU, E. I., STYLIOS, C. D. & GROUMPOS, P. P. 2003b. An integrated two-level hierarchical decision making system based on fuzzy cognitive maps. *IEEE Transactions on Biomedical Engineering,* 50**,** 1326–1339.

PAPAGEORGIOU, E. I., STYLIOS, C. D. & GROUMPOS, P. P. 2004. Active Hebbian learning algorithm to train fuzzy cognitive maps. *International Journal of Approximate Reasoning,* 37**,** 219–249.

PAPAKOSTAS, G. A., BOUTALIS, Y. S., KOULOURIOTIS, D. E. & MERTZIOS, B. G. 2008. Fuzzy cognitive maps for pattern recognition applications. *International Journal on Pattern Recogignation on Artificial Intelligence,* 22**,** 1461–1486.

PAPAKOSTAS, G. A., KOULOURIOTIS, D. E., POLYDOROS, A. S. & TOURASSIS, V. D. 2012. Towards Hebbian learning of Fuzzy Cognitive Maps in pattern classification problems. *Expert Systems with Applications,* 39**,** 10620–10629.

PARSOPOULOS, K. E., PAPAGEORGIOU, E. I., GROUMPOS, P. P. & VRAHATIS, M. N. A first study of fuzzy cognitive maps learning using particle swarm optimization. IEEE Congress on Evolutionary Computation, 2003. 1440–1447.

POLI, R., KENNEDY, J. & BLACKWELL, T. 2007. Particle swarm optimization – An overview. *Swarm Intelligence,* 1**,** 37–57.

REYNOLDS, C. W. 1987. Flocks, Herds, and Schools: A Distributed Behavioral Model. *Computer Graphics,* 21**,** 25–34.

RIGET, J. & VESTERSTRØM, J. S. 2004. A Diversity-Guided Particle Swarm Optimizer – the ARPSO. *EVALife Technical Report no. 2002-02*.

RUDIN, W. 1964. *Principles of Mathematical Analysis,* New York: McGraw-Hill.

SHI, Y. & EBERHART, R. A Modified Particle Swarm Optimizer. IEEE Congress on Evolutionary Computation, 1998. 69–73.

SONG, H., MIAO, C., SHEN, Z., ROEL, W., MAJA, D. & FRANCKY, C. 2010. Design of fuzzy cognitive maps using neural networks for predicting chaotic time series. *Neural Networks,* 23**,** 1264–1275.

STACH, W., KURGAN, L. & PEDRYCZ, W. Data driven nonlinear Hebbian learning method for fuzzy cognitive maps. IEEE World Congress Computation Intelligence, 2008.

STACH, W., KURGAN, L., PEDRYCZ, W. & REFORMAT, M. 2005. Genetic learning of fuzzy cognitive maps. *Fuzzy Sets and Systems,* 153**,** 371–401.

STYLIOS, C. D. & GROUMPOS, P. P. Mathematical formulation of fuzzy cognitive maps. Proceedings of the 7th Mediterranean Conference on Control and Automation, 1999. 2251–2261.

SUGANTHAN, P. N., HANSEN, N., LIANG, J. J., DEB, K., CHEN, Y. P., AUGER, A. & TIWARI, S. 2005. Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization. *Technical Report***,** 1–50.

TIZHOOSH, H. R. 2006. Opposition-based reinforcement learning. *Journal of Advanced Computational Intelligence and Intelligent Informatics,* 10**,** 578–585.

TSADIRAS, A. K. 2008. Comparing the inference capabilities of binary, trivalent and sigmoid fuzzy cognitive maps. *Information Sciences,* 178**,** 3880–3894.

WANG, H. Opposition-based Particle Swarm Algorithm with Cauchy mutation. IEEE Congress on Evolutionary Computation, 2007. 4750–4756.

WANG, L., PICHLER, E. E. & ROSS, J. 1990. Oscillations and chaos in neural networks: An exactly solvable model. *PNAS,* 87**,** 9467-9471.

WANG, Y., LI, B., WEISE, T., WANG, J., YUAN, B. & QIONGJIE, T. 2011. Self-adaptive learning based particle swarm optimization. *Information Sceinces,* 181**,** 4515–4538.

WILCOXON, F. 1945. Individual comparisons by ranking methods. *Biometrics,* 80–93.

ANEXO A. PSEUDOCÓDIGOS DE LOS ALGORITMOS ENUNCIADOS

Algoritmo 1. Estabilidad basada en Funciones Sigmoidales

Inicializar la población B de partículas tal que $X_i \sim U(2,12)$
Inicializar los registros personales P_i y el registro global G
FOR $k = 1$ **TO** N_{MAX} **DO**
  FOR $X_i \in B$ **DO**
    Actualizar la posición de X_i usando las ecuaciones (2.11) y (2.14)
    IF $H(X_i) < H(P_i)$ **THEN**
      $P_i \leftarrow X_i$
      IF $H(P_i) < H(G)$ **THEN**
        $count \leftarrow 0$
        $G \leftarrow P_i$
      ELSE
        $count \leftarrow count + 1$
      END
    END
  IF $count = np$ **THEN**
    Reorganizar la población usando el operador RSVN
  END
 END
END
Construir la familia de funciones sigmoidales $f_1, f_2, f_3, \dots, f_N$ a partir de la partícula G
Asignar a cada neurona C_i del mapa $\mathcal{M}$ la función de transformación f_i

Algoritmo 2. Cálculo de la función de evaluación temporal $H(X)$

Calcular el error inicial $E_0 = E_0(\mathcal{M}, \varphi)$ usando el conjunto de instancias φ (datos históricos)
Construir la familia de funciones sigmoidales $f_1, f_2, f_3, \dots, f_N$ a partir de la partícula X
Asignar a cada neurona C_i del mapa $\mathcal{M}$ la función de transformación f_i
Calcular el valor $EVAL = \phi(f_1, f_2, f_3, \dots, f_N)$ según la expresión (2.4)
Calcular el error $E = E(\mathcal{M}, \varphi)$ según la expresión (2.5)
IF $|E - E_0| > 0$ **THEN**
  $EVAL \leftarrow EVAL + P(\mathcal{M}, \varphi)$ según la expresión (2.6)
END
RETURN $EVAL$

Algoritmo 3. Operador RSVN para mejorar la diversidad de la población

FOR $j = 1$ **TO** M **DO**
  Calcular $\xi_j = j/M$
  FOR $i = 1$ **TO** $|B|/M$ **DO**
    Generar una muestra X_i^j de acuerdo a la expresión (2.16)
    $\Psi_j \leftarrow \Psi_j \cup \{X_i^j\}$
  END
  Seleccionar un subconjunto Φ_j del conjunto de muestras Ψ_j
  Incluir el subconjunto Φ_j como miembros de la nueva población B
END

ANEXO B. CARACTERÍSTICAS DE LA PLATAFORMA *FCM TOOL*

La plataforma *FCM Tool* es un software de experimentación para modelar y simular complejos sistemas dinámicos usando la teoría de Mapas Cognitivos Difusos. Originalmente este software permitía ajustar los pesos causales, pero únicamente para mapas concernientes a comportamiento de viajes, o sea, para un campo muy aplicación muy específico. La primera contribución de la tesis respecto a la implementación del algoritmo propuesto consiste en generalizar la herramienta para otros casos de estudio más generales, sobre todo para solucionar problemas de clasificación de patrones, predicción o toma de decisiones. De forma general, la nueva implementación (véase Figura B.1) incorpora las siguientes funcionalidades y algoritmos:

- Algoritmos para ajustar los pesos usando enfoques supervisados y no supervisados.
- Un algoritmo para mejorar las propiedades de convergencia en MCD Sigmoidales.
- Una función para clasificar nuevos patrones usando los mapas ajustados.
- Un algoritmo para optimizar la topología de un MCD con alta densidad.
- Un algoritmo para agregar múltiples MCD en una única estructura.
- Una función para visualizar la convergencia del concepto decisión.

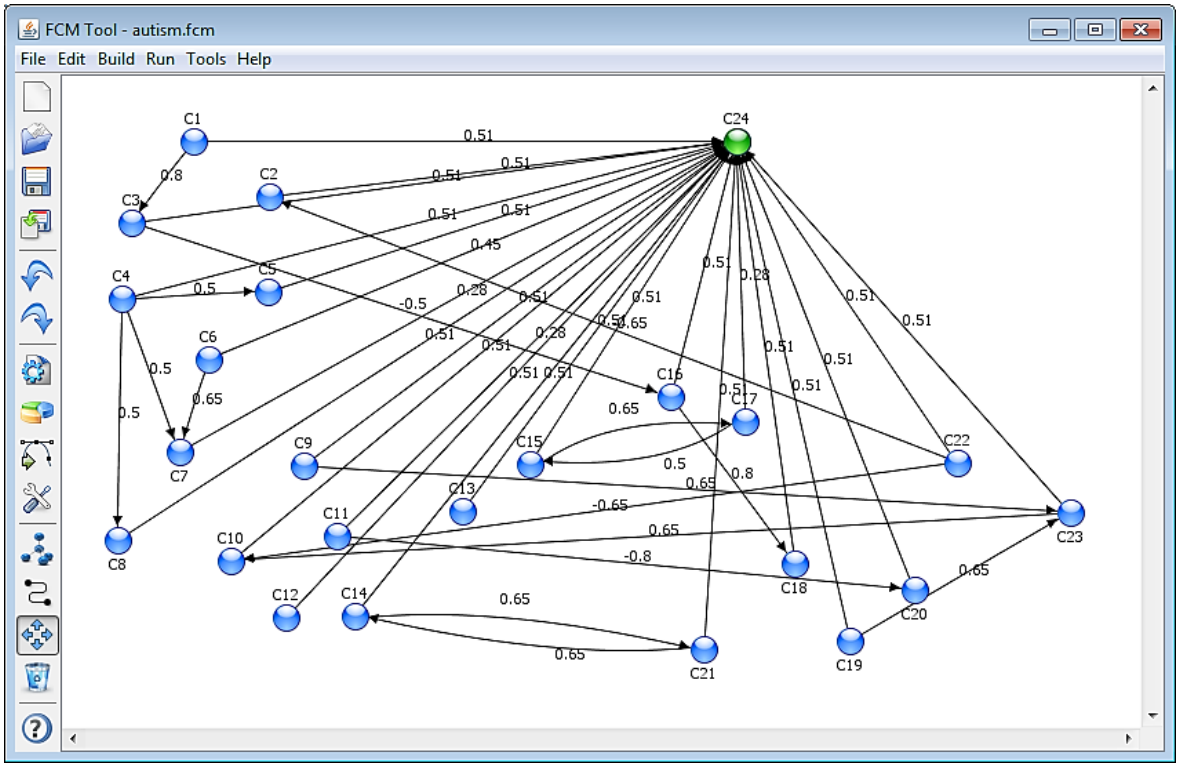


Figura B.1 Interfaz principal de la plataforma de experimentación FCM Tool para Mapas Cognitivos Difusos.

Particularmente, el algoritmo EFS (Estabilidad basada en Funciones Sigmoidales) se ejecuta en el menú *Run\Learning methods\Improve system convergence*. La Figura B.2 muestra el asistente donde el usuario puede configurar los parámetros, aunque en la práctica únicamente se necesita configurar el optimizador, pues el algoritmo EFS no necesita parámetros. En la segunda ventana del asistente el experto debe seleccionar la base de ejemplos que será utilizada como conjunto de aprendizaje para garantizar que el método no afecta la exactitud del sistema. Para este problema la base de ejemplos se organiza según el formato *Attribute Relation File Format* y debe cumplir con algunos requisitos que respeten el diseño de la plataforma, por ejemplo:

- Los nombres de los atributos deben coincidir con los nombres de los conceptos.
- Los atributos deben ser numéricos, normalizados en el intervalo de activación.
- Las clases de decisión se definen por particiones del concepto decisión.
- Cada instancia tiene asociada una única clase de decisión discreta.

En la tercera ventana del asistente (véase Figura B.4) para el método EFS se visualiza el proceso de aprendizaje, así como algunas estadísticas útiles para el experto. Por ejemplo, las estadísticas incluyen el número de ejemplos de la base, el número de atributos de una instancia, el número de clases de decisión (particiones del espacio de activación del concepto decisión), el error inducido en cada paso del método propuesto, la dimensionalidad del problema de optimización y el estado de la estabilidad del sistema, considerando todos los tiempos del proceso.

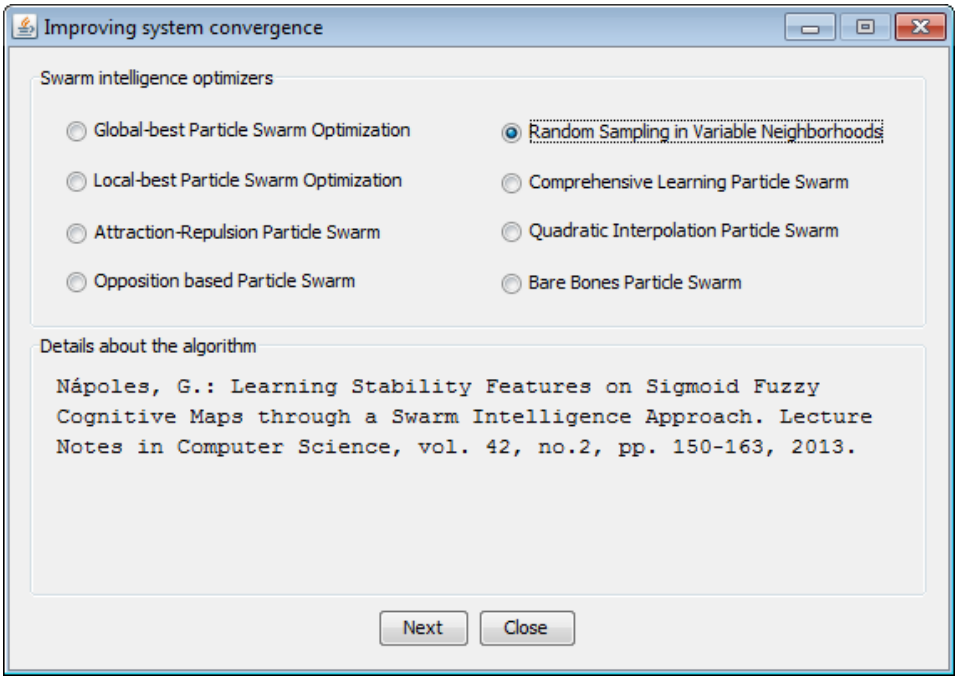


Figura B.2 Primera ventana del asistente para el algoritmo EFS donde el usuario selecciona el optimizador.

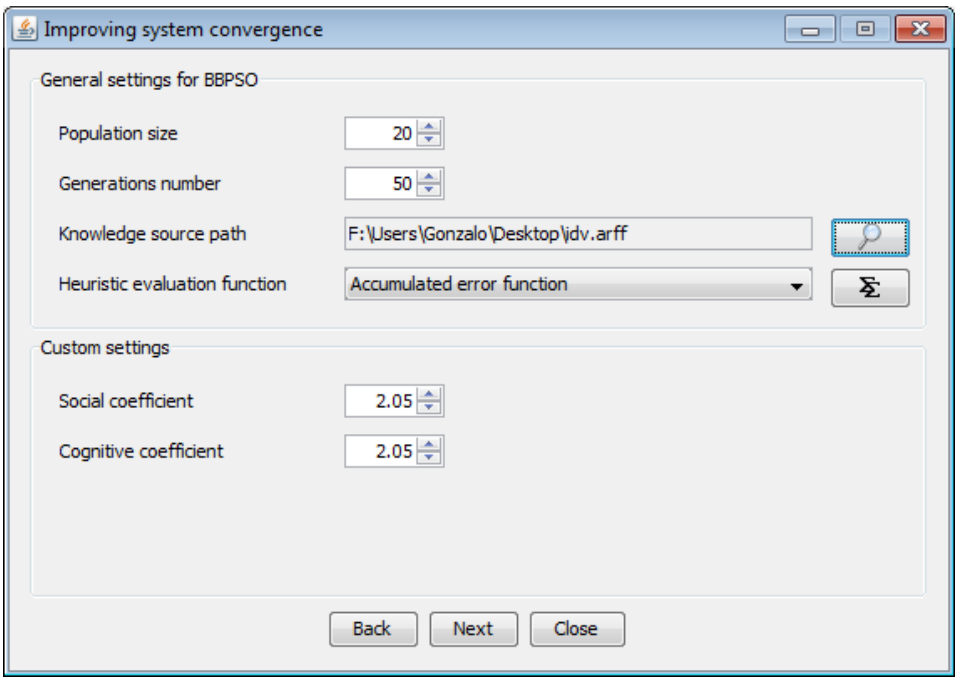


Figura B.3 Segunda ventana del asistente para el algoritmo EFS donde el usuario configura los parámetros.

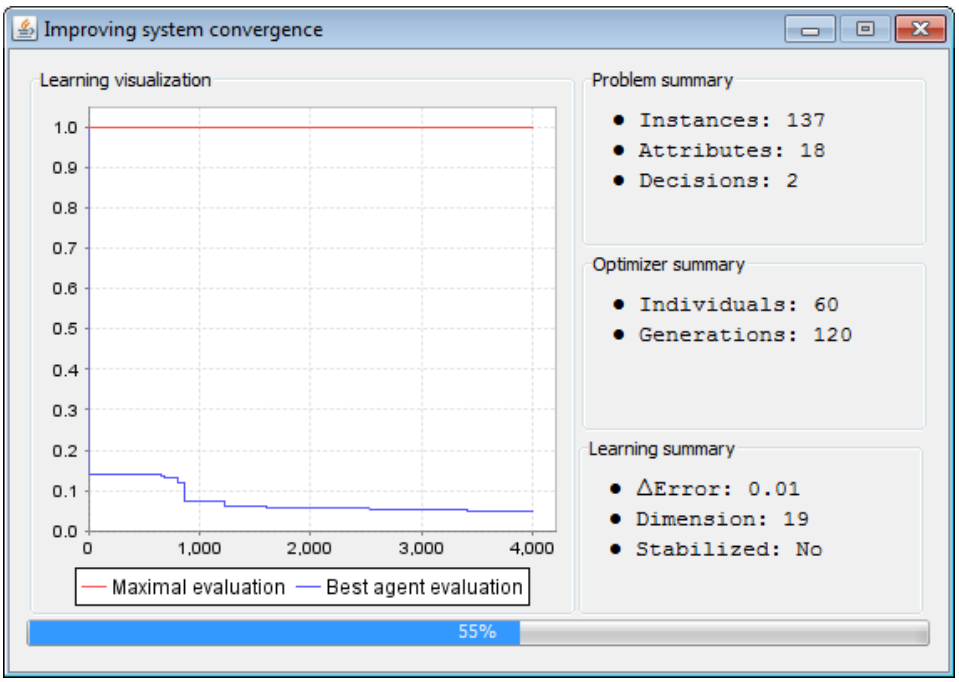


Figura B.4 Tercera ventana del asistente para el algoritmo EFS donde se visualiza el proceso de aprendizaje.

Después de finalizado el algoritmo de aprendizaje cada concepto del mapa se le asigna de forma automática la inclinación correspondiente, y ese factor será utilizado para transformar el valor de activación de la neurona. Si no se encuentra una solución que reduzca la variabilidad del sistema, entonces los conceptos preservarán su inclinación inicial ($\lambda_i = 9$). Por otra parte, si la familia de funciones sigmoidales no satisface las expectativas del experto, entonces es posible deshacer las modificaciones y volver a ejecutar el procedimiento de aprendizaje.