Sistemas Clasificadores en Sistemas de Control: Prueba del Mecanismo Adaptativo

Glenda González, Jose Aguilar

CEMISID, Departamento de Computación, Facultad de Ingeniería,
Universidad de Los Andes, Núcleo la Hechicera, Mérida 5101-Venezuela
(e-*mail: glendag@ula.ve, aguilar@ula.ve*)

Resumen: El objetivo de este trabajo es diseñar e implementar un Mecanismo Adaptativo (MA) para Sistemas Clasificadores (SCs). Para implementar dicho mecanismo utilizamos dos técnicas inteligentes: la Lógica Difusa (LD) y la Computación Evolutiva (CE). La LD fue utilizada para representar el conocimiento en los Sistemas Clasificadores Difusos (SCDs). De la CE utilizamos dos paradigmas para desarrollar los esquemas evolutivos del MA: los Programas Evolutivos (PES) y la Programación Genética (PG). El funcionamiento de la herramienta completa fue probado en un sistema de control para un sistema de un tanque. La herramienta inteligente desarrollada puede ser usada en ambientes de Control Industrial.

1. INTRODUCCIÓN

Los SCs son Sistemas Basados en Conocimiento con características tomadas del razonamiento humano, tales como, autoaprendizaje y adaptación al medio ambiente, logradas estas a través del uso de técnicas inteligentes, como lo son las redes neuronales, la CE, la LD, entre otras. Entre estos sistemas se encuentran extensiones de los Sistemas Expertos (SEs) y los SCDs. Los SCs son un tipo particular de Máquinas de Aprendizaje (MDA) que tienen la virtud de aprender comportamientos a partir de la activación de reglas de producción, debido a mensajes provenientes del exterior. Este aprendizaje se basa en dos subsistemas que lo integran, como lo son el subsistema de asignación de crédito y el subsistema descubridor. El subsistema de asignación de crédito actualiza los pesos de activación de las reglas según su uso, mientras que el subsistema descubridor realiza la tarea de descubrimiento de nuevas reglas. Dentro de los SCs se encuentran los llamados SCDs, estos son SCs con reglas que presentan una connotación difusa.

Por otro lado, la inteligencia artificial ofrece diversas técnicas para tratar los problemas de aprendizaje, entre las que se encuentra la CE. La CE es una técnica que trata los enfoques alternativos de búsqueda y aprendizaje basados en modelos computacionales de procesos evolutivos. La idea principal es la de una búsqueda estocástica, evolucionando a un conjunto de estructuras y seleccionando de modo iterativo a las más aptas con la finalidad de que los mejores individuos sobrevivan mediante su adaptación al entorno. La CE contempla cinco (5) paradigmas Algoritmos Genéticos (AGs), Programas Evolutivos (PEs), Estrategias Evolutivas (EEs), Programación Evolutiva (PE) y Programación Genética (PG).

En el marco de desarrollo de una herramienta de gestión para SCs y SCDs, surge la necesidad de brindarle la capacidad de aprender, es por ello que este trabajo de investigación diseña e implementa un Mecanismo Adaptativo (MA) para el ``Sistema Automatizado de Gestión para Sistemas Expertos y Clasificadores Difusos" SAGSECD (Aguilar *et. al*., 2004; Aguilar *et. al*, 2005). En nuestro trabajo usamos el paradigma de la PG para descubrir reglas, ya que esta técnica ofrece un poder descriptor de la gramática que caracteriza a las reglas, admitiendo la evolución de poblaciones con menos restricciones que las impuestas por los AGs (técnica evolutiva generalmente usada). La evolución no se limita a las reglas, en el caso de los SCDs también se contemplan las funciones de pertenencia que describen los conjuntos difusos de las variables, para captar los cambios del entorno. Para esto último el paradigma que usamos es el de los PEs, ya que este permite una representación de los individuos más cercana al dominio del problema.

2. MARCO TEÓRICO

2.1 Sistemas Clasificadores

Las ideas básicas de los SCs fueron presentadas por Holland en (Holland, 1975). Desde entonces, muchos tipos de SCs se han propuesto en la literatura (Goldberg, 1989; Lashon *et a*l., 1989; Parodi *et al.*, 1993; Valenzuela, 1991 ;Wilson *et al.* , 1989; Wilson, 2000). Un Sistema Clasificador (SC) es un tipo particular de máquina de aprendizaje que aprende reglas (Golberg, 1989).

Las reglas utilizadas por los SCs son generalmente de la forma Si <condición> entonces <acción>, las cuales tienen gran capacidad de representación del conocimiento. Un SC está compuesto por tres componentes principales que coexisten dentro de él, como lo son: un sistema de reglas y mensajes, un sistema de asignación de crédito, y un sistema de descubrimiento.

Depósito Legal LFX23720086205059.
ISBN 978-980-11-1224-2. ©ULA.

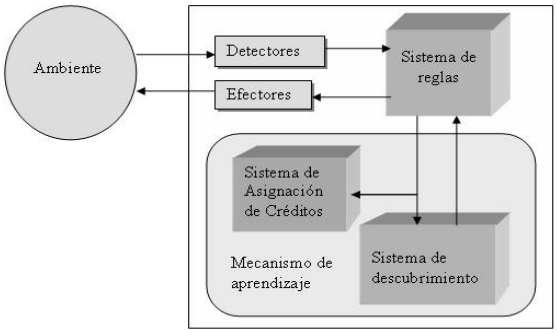

Fig. 1 : Arquitectura de un SC

A. Sistema de reglas y mensajes. Coordina el flujo de información dentro del SC a partir de la información recibida del ambiente, para luego procesar tal información y generar una acción.

B. Sistema de asignación de crédito (SAC). Es el encargado de evaluar la actuación de una regla en función de las veces que ha sido activada y las veces que esta activará otras reglas en un período de tiempo dado. Tradicionalmente se vienen usando dos diferentes esquemas de asignación de crédito: el Algoritmo de Bucket Brigade y el Plan de Profit-Sharing (Aguilar *et al*., 2000; Aguilar *et al*., 2001).

C. Sistema de descubrimiento (SD). El SD crea nuevos, y posiblemente, más útiles clasificadores usando la experiencia pasada, es decir, los antiguos clasificadores.

Un SCD es un sistema cuyos clasificadores están basados en la teoría de LD, el cual integra los mismos elementos de un SC pero trabajando en un marco difuso (Aguilar *et al*., 2000]. Es decir, es un sistema de inferencia difuso con la capacidad de aprendizaje de los SCs.

2.2 Computación Evolutiva

La CE es el campo dedicado a simular la evolución natural en un computador. Bajo este término se engloba un grupo de técnicas de resolución de problemas complejos basadas en la emulación de los procesos naturales de evolución. A continuación presentamos los dos paradigmas utilizados en este trabajo.

A. Programación Genética. La Programación Genética (PG) surge como una propuesta de John Koza que consiste en utilizar la metodología de CE, no para encontrar soluciones a problemas, sino para encontrar el mejor procedimiento para resolverlos. Para aplicar la PG es necesario especificar los siguientes elementos:

a. Conjunto de terminales y funciones. El conjunto de terminales está compuesto por átomos, que son las constantes o acciones específicas que son ejecutadas en el programa, ejemplo: números, operadores matemáticos, identificadores, etc. El conjunto de funciones pueden ser operaciones aritméticas, lógicas, operadores condicionales, instrucciones de repetición (por ejemplo: if, then, else, while, for), entre otros.

b. Individuos. Los individuos son estructuras arborescentes. En general, estas estructuras están formadas por nodos funciones y nodos terminales, los cuales son específicos para cada problema. Por ejemplo, la función f(x,y) =sin(x) * (9+ y) queda representada por el árbol de la Figura 2.

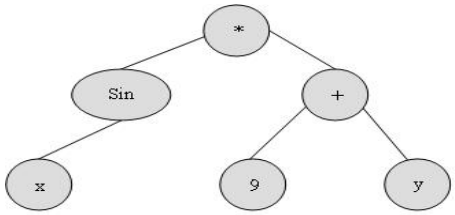

Fig. 2: Árbol de análisis de la función f(x,y) = sin(x) * (9 + y)

c. Población inicial de programas. La PG comienza con una población inicial de programas generados de forma aleatoria. Los árboles aleatorios generados tienen diferentes tamaños y formas.

d. Tamaño de la población. El tamaño de la población se ve principalmente afectado por el tiempo que se tarda en calcular la aptitud de un individuo, que es la tarea que más tiempo consume. Así pues, dependiendo de la complejidad del problema, del tiempo disponible, de la tecnología de los recursos, podremos elegir un tamaño de población que nos permita obtener rápido una solución.

e. Número de generaciones. La evolución se lleva a cabo modificando los individuos que componen la población a través de un cierto número de generaciones.

f. Función de aptitud. Es una expresión matemática que debe ser capaz de evaluar la calidad de cualquier individuo de la población.

g. Operadores genéticos. Los operadores genéticos toman individuos de la población actual y producen nuevos individuos para la generación siguiente, aplicando las transformaciones que impongan los operadores. En el caso de la PG, los operadores de cruce y mutación funcionan de la siguiente manera:

- Cruce. En general, consiste básicamente en escoger al azar un subárbol de cada uno de los individuos, y en intercambiar dichos subárboles entre los individuos.
- Mutación. La mutación es un operador mediante el cual se regenera en forma aleatoria parte de un individuo. Para ello, se escoge un nodo en el cual se va a aplicar la mutación. El subárbol original que contiene ese nodo como raíz es eliminado, y un nuevo subárbol es creado en ese lugar usando el mismo procedimiento que el usado para generar la población inicial.

h. Métodos de Selección. Los métodos de selección son aquellos utilizados para escoger a un individuo, de entre todos los de la población, para ser utilizado por los operadores genéticos.

i. Criterio para terminar la ejecución. Al igual que en la naturaleza, el paradigma de la PG es un proceso sin fin. No

obstante, se establece un criterio de culminación que sirve para finalizar el proceso evolutivo, el cual normalmente es el número máximo de generaciones, y en algunos casos, el criterio de homogeneidad entre generaciones.

B. Programas Evolutivos. La propuesta de los Programas Evolutivos (PEs) fue hecha por Michalewicz en 1944, cuando propuso incorporar conocimiento específico del problema a resolver en las estructuras de datos (Aguilar *et al*, 2001). Con la incorporación de conocimiento específico se pierde un poco de generalización y el paralelismo implícito, sin embargo, esta pérdida es compensada con el procesamiento de información más útil. Desde el punto de vista práctico, podemos ver a los PEs como una modificación de los algoritmos genéticos que comparten exactamente su estructura básica, con las distinciones siguientes:

- La representación es más natural, por lo cual está más cercana al dominio del problema.
- Los operadores genéticos son específicos al contexto del problema. Por lo tanto, pueden estar adaptados al dominio con el fin de generar resultados que mantengan la estructura de los individuos.

En cuanto a los elementos a especificar, se requieren los mismos que para los AGs, a saber: el tamaño de la población, la población inicial, los operadores genéticos a usar, el mecanismo de selección, entre otros.

3. DISEÑO DEL MECANISMO ADAPTATIVO

3.1 SAGSECD

El SAGSECD está conformado por cuatro subsistemas. Cada uno de estos subsistemas lleva a cabo una serie de funciones para permitir la gestión de SCs y SCDs en forma eficiente.

El **subsistema de entrada de reglas** está conformado por un compilador y un editor de reglas. Éste es el encargado de realizar el análisis léxico, sintáctico y semántico de las reglas que se desean almacenar (Aguilar *et a*l., 2005).

El **subsistema de inferencia** es el responsable de realizar todo el proceso de razonamiento asociado a los SCs y SCDs. En el caso de los SCDs, además del proceso de inferencia, se encarga de la fusificación, y de la defusificación de las variables [Aguilar *et al*., 2004].

El **subsistema de adaptación** es el encargado de adaptar el conocimiento almacenado en el sistema a los cambios del entorno. Está formado por un SAC y un SD. En el caso de los SCs, el SAC evalúa las reglas y el SD descubre y/o genera nuevas reglas. En el caso de SCDs, el SAC evalúa las reglas y las funciones de pertenencia de los conjuntos difusos, y el SD se encarga de descubrir y/o generar nuevas reglas, así como funciones de pertenencias.

El **subsistema de almacenamiento de información** está conformado por una base de conocimiento y una base de hechos. La base de conocimiento posee toda la información concerniente a las variables y reglas que describen el sistema, mientras que la base de hechos posee un historial referido a los valores tomados por las variables del sistema.

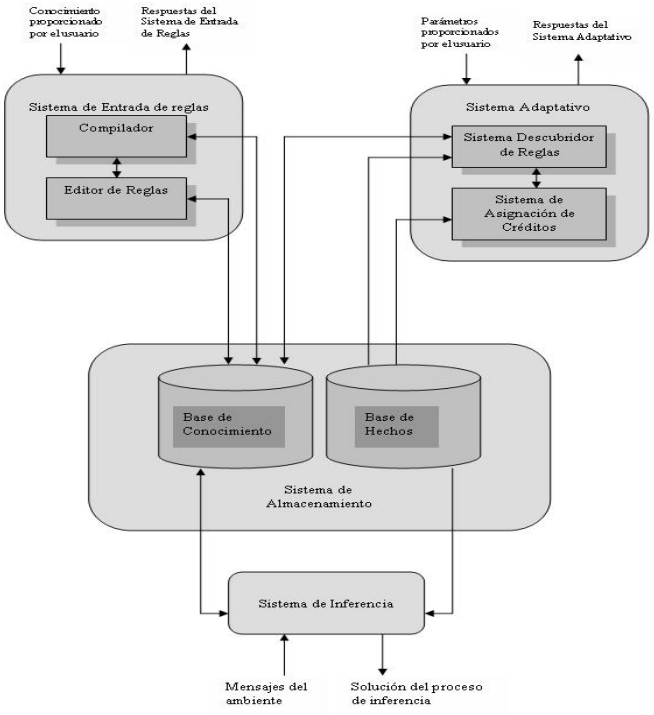

Fig. 3: Arquitectura del SAGSECD

2.2 Descripción del Mecanismo Adaptativo (MA)

Para ser capaz de evolucionar y aprender dinámicamente, el MA debe considerar:

- Realimentarse con la información que recibe del exterior, tal que si considera que las decisiones que se tomaron fueron acertadas premia las reglas incrementándoles el crédito.
- Utilizar una técnica inteligente para crear nuevas reglas a partir de las mejores reglas existentes. Las nuevas reglas sustituirán a las reglas existentes con créditos inferiores.
- Observar como se comporta a lo largo del tiempo una regla para saber si es útil, ya que no se deben generar nuevas reglas constantemente, esto se debe hacer cada cierto intervalo de tiempo. De esta forma, todas las reglas tienen la oportunidad de demostrar su utilidad.

Nuestro sistema establece como individuos para la evolución, las reglas del sistema en estudio. En el caso de SCDs, también presenta un algoritmo de evolución para las funciones de pertenencia correspondientes a los conjuntos difusos de las variables del sistema.

En cuanto al SAC, el crédito de las reglas (para cualquier sistema, sea difuso o no) se basa en el grado de activación de las mismas; mientras que el crédito de una función de pertenencia de una variable difusa es asignado según la participación que esa variable en particular, con dicha función de pertenencia, tenga en el peso de una regla que ha sido activada.

A. Descripción del Sistema de Asignación de Crédito

El SAC es llamado en la arquitectura global del SAGSECD cada vez que se llama al MA. Como se mención antes, es el encargado de calcular los valores de crédito de las reglas y, en el caso de SCD, también del crédito de las funciones de pertenencia. Cuando el Sistema de Inferencia produce una salida, asigna a las reglas el grado de participación que tuvieron en la misma, de modo que las reglas que más contribuyeron en la salida reciben un mayor valor. El valor asignado por el proceso de inferencia corresponde al grado de activación de la regla, y es a lo que hemos denominado peso. Así, el crédito de las reglas se calcula mediante la siguiente ecuación:

$$C_i(t) = C_i(t-1) + Act_i(t) + n_i(t) \quad (1)$$

donde:

$C_i(t)$ representa el valor del crédito de la regla i en el tiempo t, $Act_i(t)$ representa el grado de activación de la regla i en el tiempo t y $n_i(t)$ representa el número de reglas que la regla i ayudó a activar en el tiempo t.

Para el caso de la función de pertenencia, cuando el Sistema de Inferencia difusa produce una salida, se debe actualizar el crédito de la función de pertenencia del conjunto difuso F de la variable difusa V que forma parte de la condición del antecedente de la regla i activada (esto se repite para todos los conjuntos difusos de las variables difusas que están en el antecedente de las reglas activadas). Para eso se usa el valor específico de entrada X en el dominio de V, la función de pertenencia $\mu_{F(x)}$ y las siguientes ecuaciones [Aguilar, et al., 2000]:

Si el operador lógico es O:

$$C_{\mu_F}(t) = C_{\mu_F}(t-1) + Act_i(t)\mu_{F(x)} \quad (2)$$

Si el operador de la condición es Y:

$$C_{\mu_F}(t) = C_{\mu_F}(t-1) + \frac{1}{Act_i(t) * \mu_F(x)} \quad (3)$$

Donde $C_{\mu_F}(t)$ es el valor del crédito de la función de pertenencia μ_F en el tiempo t, $Act_i(t)$ es el grado de activación de la regla i en el tiempo t y $\mu_{F(x)}$ es el grado de pertenencia del elemento x al conjunto difuso F en el tiempo t, de la variable difusa V presente en la condición .

B. Descripción del Sistema Descubridor

El SD hace uso de la PG y de los PEs como técnicas evolutivas. Estas técnicas adaptan los elementos involucrados en el proceso de inferencia, tales como:

- La estructura de las reglas, introduciendo nuevas variables en la condición y/o en la acción (para SC y SCD).
- Las instancias de las reglas, introduciendo nuevos conjuntos difusos o valores para las variables que están en la condición o la acción (para SC y SCD).
- Las funciones de pertenencia}, estableciendo nuevos rangos para los conjuntos difusos definidos, o generando un tipo de función de pertenencia diferente al que presenta la función de pertenencia actualmente (para SCD).

C. Elementos del Sistema Descubridor

Los algoritmos evolutivos que conforman al SD requieren diversos elementos para su implementación. En primera instancia describiremos como se establecieron los elementos necesarios para aplicar la PG, y en segunda para implementar los PEs.

a. Elementos requeridos por la PG. Para aplicar la PG a la evolución de reglas se establecieron los elementos requeridos por esta técnica de la siguiente manera:

- **Conjunto de terminales y funciones**. El conjunto de terminales está conformado por las variables del sistema y sus valores posibles. El conjunto de funciones depende del tipo de sistema. En el caso de los SCs, este conjunto está representado por: Si, Entonces, los operadores aritméticos, relacionales y lógicos, y las funciones trigonométricas. En el caso de los SCDs, las funciones del conjunto son: Si, Entonces, Y, O, es, No.
- **Individuos**. Un individuo está representado por un árbol de análisis. El hijo izquierdo del árbol corresponde al antecedente de la regla y el hijo derecho al consecuente. En la Figura 4 podemos observar el árbol de análisis para la regla Si TIME > 4 Entonces ESTADO = CERRAR.

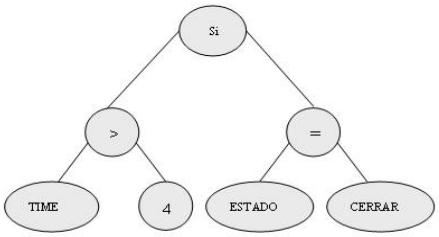

Fig. 4: Ejemplo de un árbol de análisis

- **Población inicial de programas.** La población está conformada por las N reglas que se encuentran en la base de conocimiento.
- **Tamaño de la población**. El tamaño de la población viene dado por el número de reglas en la base de conocimiento.
- **Función de aptitud**. Sobre la función de aptitud, podemos decir que esta coincide con el crédito asignado por el SAC basándose en la ecuación 1. Para evaluar las

reglas generadas en cada generación del algoritmo evolutivo, el SAC se basa en el histórico de datos que se encuentran en la base de hechos.

- **Operadores genéticos**. Se consideran los operadores genéticos: cruce, mutación (con una pequeña probabilidad de uso) y copia. Para implementar el operador genético de cruce, se selecciona un punto aleatorio al azar en cada uno de los árboles a cruzar, y se intercambian las ramas con origen en dichos puntos en los árboles. En cuanto al operador de mutación, se selecciona un punto de mute (un nodo) del árbol a mutar al azar, luego se genera un subárbol que será agregado al árbol actual en el nodo punto de mute.
- **Método de Selección**. Todas las reglas de la población son considerados padres en el proceso de evolución. De esta manera, aseguramos que cada regla tenga un descendiente en la siguiente generación para poder dar paso al mecanismo de reemplazo.
- **Mecanismo de reemplazo**. Un padre puede ser reemplazado únicamente por su hijo. Una vez que el proceso de reproducción concluye los individuos hijos son evaluados usando, como se dijo, los datos presentes en la base de hechos. Posteriormente cada hijo se compara con su padre, si el hijo tiene mayor crédito que el padre entonces este se incluye en el lugar del padre en la siguiente generación, en caso contrario el padre permanece en la población.
- **Número de generaciones**. El número de generaciones n corresponde al número de veces que el algoritmo se ejecutará. Este número es suministrado por el usuario $n \geq 1$.
- **Criterio para terminar la ejecución**. El criterio de convergencia del algoritmo obedece al número máximo de generaciones n.También puede obedecer a la homogeneidad de las soluciones que se van obteniendo. El término homogeneidad se refiere a la no variación de la población durante dos generaciones consecutivas.

b. Elementos requeridos por los PEs. Los elementos requeridos por los PEs para evolucionar las funciones de pertenencia se establecieron de la siguiente manera:

- **Individuos**. Un individuo está representado por una cadena de caracteres. Esta cadena está dividida en dos partes por el signo de puntuación ";". La primera subcadena de caracteres contiene todos los puntos del eje de las abscisas necesarios para definir la representación gráfica de la función de pertenencia, separados por comas; estos valores están en el universo de discurso del conjunto. La segunda subcadena contiene los grados de pertenencia de los elementos de la primera subcadena al conjunto difuso. Cada valor en la primera subcadena tiene su correspondiente en la segunda subcadena, para conformar un par.
- **Población inicial**. La población está conformada por *N* funciones de pertenencia, escogidas en forma aleatoria del total perteneciente a las variables que se encuentran en la base de conocimiento.
- **Tamaño de la población**. El tamaño de la población está dado por un porcentaje del número de variables difusas que tiene el sistema.
- **Número de generaciones**. El número de generaciones n corresponde al número de veces que el algoritmo se ejecutará, este número es suministrado por el usuario y corresponde a $n \geq 1$.
- **Función de aptitud**. Esta coincide con el crédito asignado por el SAC a la función de pertenencia utilizando las ecuaciones 2 y 3. Para esto, hace uso de la información almacenada en la base de hechos.
- **Operadores genéticos**. Se consideran el operador genético de mutación y el operador genético copia. El operador mutación podrá efectuar cambios en el tipo de función de pertenencia y en el intervalo que la define. Las diferentes formas que pueden tener las funciones de pertenencia son: triangular, trapezoidal y gaussiana. También podemos definir la forma gráfica de una función de pertenencia a través de un grupo de rectas. Para aplicar el operador mutación primero se escoge el tipo de mutación al azar. Si se refiere a la forma, los puntos del eje de las abscisas y ordenadas que definen la función de pertenencia se actualizan. Si la mutación será en el rango de valores tomados por la variable, este puede aumentarse o disminuirse agregando nuevos pares en los extremos del individuo.
- **Método de Selección**. En este caso, los individuos de la población son seleccionados uno a uno, como padres para el proceso de reproducción. Así, todos los miembros de la población conforman la población de padres, ya que estos se escogen uno a uno para ser mutados.
- **Mecanismo de reemplazo**. Un padre puede ser reemplazado únicamente por su hijo. Una vez que el proceso de reproducción concluye, a los individuos hijos se les calcula la función de aptitud. Posteriormente se compara cada hijo con su padre, si la aptitud del hijo es mayor entonces este se incluye en el lugar de su padre en la siguiente generación.
- **Criterio para terminar la ejecución**. El criterio de convergencia obedece al número máximo de generaciones $n \geq 1$ suministrado por el usuario al comienzo de la ejecución. También puede obedecer a la homogeneidad de las soluciones que se van obteniendo. El término homogeneidad se refiere a la no variación de la población durante dos generaciones consecutivas.

4. IMPLEMENTACIÓN DEL MA

La funcionalidad de los SCs se puede observar en la Figura 5. Los bloques necesarios se describen a continuación:

- **Bloque de entrada y procesamiento de datos iniciales**. Este bloque es el encargado de organizar todos los datos necesarios para el funcionamiento de los SCs. Captura los datos suministrados por el usuario y los datos presentes en la base de conocimiento, con el fin de procesarlos y convertirlos en la información de entrada al bloque ejecutor.
- **Bloque ejecutor**. Coordina el ciclo de activación de reglas y generación de mensajes.

- **Bloque de asignación de crédito**. Es el encargado de ajustar el crédito de las reglas y de las funciones de pertenencia. Corresponde al SAC del MA.
- **Bloque de adaptación**. Es el que ejecuta el SD del MA. Sus actividades son evolucionar las reglas, y en el caso de SCDs también las funciones de pertenencia.

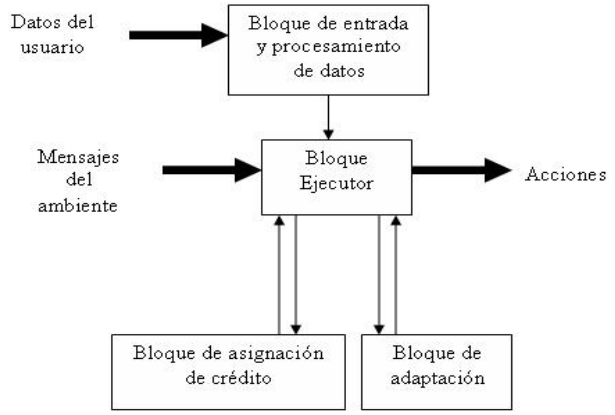

Fig. 5: Diagrama del flujo funcional de la herramienta

El prototipo del MA para el SAGSECD fue diseñado de tal manera que respondiese a las características funcionales de los SCs en dos formas: manual y automática.

1. Ejecución Manual. En este tipo de funcionamiento los datos son ingresados en forma manual, y la activación de cualquier bloque del algoritmo del SC también. Debería permitir:

- Suministrar la información requerida por el sistema, es decir, en forma manual ingresar a SAGSECD los valores tomados por las variables del sistema en gestión.
- Accionar el bloque ejecutor para obtener una solución. Este procedimiento el usuario lo podrá realizar las veces que le sea necesario.
- Ejecutar el mecanismo de aprendizaje (bloque adaptativo) en el momento en que el usuario lo considere pertinente.
- Debe permitirle al usuario escoger cual estructura desea evolucionar (reglas y/o funciones de pertenencia). El usuario también podría ejecutar el bloque adaptativo las veces que lo desee.

Con esta forma de funcionamiento se permite que la herramienta pueda trabajar fuera de línea.

2. Ejecución Automática. Esta forma de ejecución permite al usuario obtener soluciones a través de un ingreso de información y aprendizaje en forma automática. Esta forma de uso requiere la especificación de un repositorio donde se encuentra la información proveniente del ambiente y establecer a cuantos mensajes se activará el MA. El algoritmo de ejecución sigue los pasos siguientes:

1. Lee cada evento en el repositorio y codifica el mensaje.
2. Acciona el mecanismo de inferencia y proporciona la solución al usuario, actualizando la base de hechos y de conocimiento.
3. Si el número de mensajes recibido alcanza un valor preestablecido activa el MA.
4. Al terminar la ejecución del MA vuelve a leer el repositorio de datos hasta que no encuentre más eventos.

Esta forma de uso supone que existe un mecanismo de recolección de datos desde el ambiente del sistema gestionado, que mediante una interfaz apropiada escribe la información proveniente del mismo en el repositorio. Se pretende que de esta forma la herramienta tenga características de una máquina de aprendizaje que trabaja en línea. En el caso manual el usuario puede ejecutar el esquema de funcionamiento de los SCs accediendo a cada módulo en forma directa.

5. CASO DE ESTUDIO: SISTEMA CLASIFICADOR PARA UN TANQUE

Este caso de estudio se refiere a las acciones de control sobre un tanque para mantener el nivel de presión en un rango de operación óptimo. Se desea diseñar un SC para el sistema de la Figura 6 que tenga como entradas los estados de las válvulas y del sensor de presión, y que genere como salida la acción de control a tomar sobre las válvulas del sistema, para mantener la presión en el tanque entre 5 y 10 bar. Se asume que siempre fluye líquido a través de la tubería que alimenta el tanque. Los estados que pueden tomar las válvulas son abierto o cerrado.

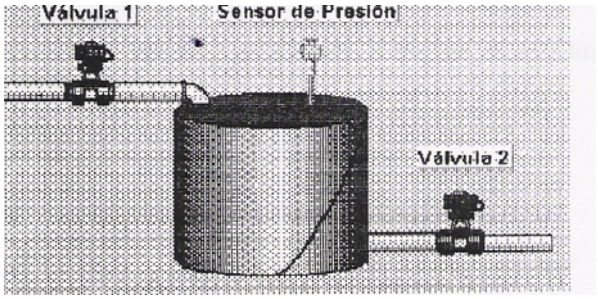

Fig. 6: Sistema de un Tanque

Las variables en el sistema son:
- Variables de entrada: presión, válvula 1 y válvula 2.
- Variables de salida: válvula 1 y válvula 2.

Los tipos de las variables y valores posibles son:
- Válvula 1 (Valvula1): tipo Enumerado, con valores posibles: abierta, cerrada.
- Válvula 2 (Valvula2): tipo Enumerado, con valores posibles: abierta, cerrada.
- Presión: tipo entero, con un rango de variación entre 0 y 20.

Posteriormente, establecimos las siguientes reglas, las cuales definen la acción de control que se debe tomar sobre el sistema en función del estado del mismo:

- Si Valvula1== abierta Y Valvula2==abierta Y Presion < 5 Entonces Valvula1= abierta Y Valvula2 = cerrada.
- Si Valvula1 == abierta Y Valvula2 == abierta Y Presion > 10 Entonces Valvula1 = cerrada Y Valvula2 = abierta.
- Si Valvula1 == abierta Y Valvula2 == abierta Y Presion >= 5 Y Presion <= 10 Entonces Valvula1 = abierta Y Valvula2 = abierta.
- Si Valvula1 == abierta Y Valvula2 == cerrada Y Presion < 5 Entonces Valvula1 = abierta Y Valvula2 = cerrada.
- Si Valvula1 == abierta Y Valvula2 == cerrada Y Presion > 10 Entonces Valvula1 = cerrada Y Valvula2 = abierta.
- Si Valvula1 == abierta Y Valvula2 == cerrada Y Presion >= 5 Y Presion <= 10 Entonces Valvula1 = abierta Y Valvula2 = cerrada.
- Si Valvula1 == cerrada Y Valvula2 == abierta Y Presion < 5 Entonces Valvula1 = abierta Y Valvula2 = cerrada.
- Si Valvula1 == cerrada Y Valvula2 == abierta Y Presion > 10 Entonces Valvula1 = cerrada Y Valvula2 = abierta.
- Si Valvula1 == cerrada Y Valvula2 == abierta Y Presion >=5 Y Presion <= 10 Entonces Valvula1 = cerrada Y Valvula2 = abierta.
- Si Valvula1 == cerrada Y Valvula2 == cerrada Y Presion < 5 Entonces Valvula1 = abierta Y Valvula2 = cerrada.
- Si Valvula1 == cerrada Y Valvula2 == cerrada Y Presion > 10 Entonces Valvula1 = cerrada Y Valvula2 = abierta.
- Si Valvula1 == cerrada Y Valvula2 == cerrada Y Presion >=5 Y Presion <=10 Entonces Valvula1 = cerrada Y Valvula2 = cerrada.

En la Figura 7 se muestra el Panel Principal de SAGSECD. En el lado izquierdo, se listan todas las bases de conocimiento, tanto para los SCs como para los SCDs.

Fig. 7: Interfaz del Panel Principal de SAGSECD

Al pulsar el botón Entrar en El Sistema aparece el Panel del Diseñador, el cual se muestra en la Figura 8. En el lado izquierdo del panel aparecen todas las operaciones que se pueden realizar sobre la base de conocimiento. Las opciones son las siguientes: Mostrar Base de Conocimiento, Borrar Base de Conocimiento, Añadir Variable, Editar Variable y Borrar Variable. Detalles del uso de cada una de estas operaciones se pueden ver en (Aguilar *et al*., 2004].

Fig. 8: Panel del Diseñador

El Panel del Diseñador en el lado superior derecho muestra un botón llamado Base de hechos, esta ventana es necesaria para la modalidad ejecución manual de la herramienta. El panel tiene en el lado superior derecho un botón llamado "Editor de Regla" para construir las reglas del sistema. Detalles de uso de este subsistema en [Aguilar, et al., 2004]. En la parte inferior derecha del Panel de Diseñador se ofrecen tres botones: Diseñar Sistema Adaptativo, SA Manual y SA Automático. Para especificar el tipo de ejecución, lo cual define el tipo de entrada de mensajes, se pulsa el botón Diseñar Sistema. Inmediatamente aparece una ventana en la cual se introducen todos los parámetros requeridos por los algoritmos evolutivos para generar nuevas reglas.

Los parámetros requeridos por el mecanismo de evolución de reglas se pueden apreciar en la Figura . Allí se especifica que la ejecución del MA, y por lo tanto los mensajes que vienen del ambiente, se hacen de manera Automática. También se pueden observar el resto de los parámetros introducidos como: Tasa de mutación con un valor de 0.1, Número de Generaciones igual a 12. Además, se seleccionó el uso de los operadores de cruce y mutación, se establecieron los máximos niveles para los árboles involucrados en la evolución, los nuevos individuos, y los subárboles generados en el proceso de mutación como 5. Finalmente, se especificó el número de mensajes recibidos del exterior para hacer el llamado al MA como 3, y la ruta de acceso al archivo de texto que contiene toda la información del ambiente.

Luego de especificar el tipo de ejecución y de terminar el diseño del MA, se presionó el botón SA Automático de la Figura 9 para que aparezca la ventana de la Figura 11, en la cual al presionar el botón aceptar comenzó la ejecución del funcionamiento automático de los SCs expuesto en la sección anterior.

Los resultados de dicha ejecución se visualizaron en el área de texto de esta ventana. En esta figura se observa la llegada de un mensaje del exterior, presentando los valores que toman cada una de las variables de entrada. Luego se despliegan las reglas que se activaron al recibir dicho mensaje. Finalmente, se muestran las reglas que se reemplazaron seguidas de la regla hijo que la reemplazó.

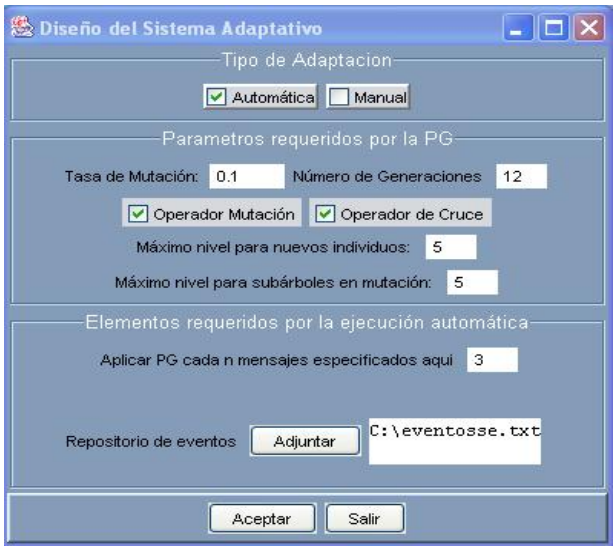

Fig. 9: Parámetros requeridos por el MA

Luego de evolucionar las reglas, observamos que el mecanismo es capaz de construir reglas que aportan soluciones al sistema en estudio. También, la herramienta es capaz de regenerar reglas que en evoluciones anteriores han sido eliminadas de la base de conocimiento por su inactividad, pero que a medida que el sistema en estudio va cambiando se hacen útiles.

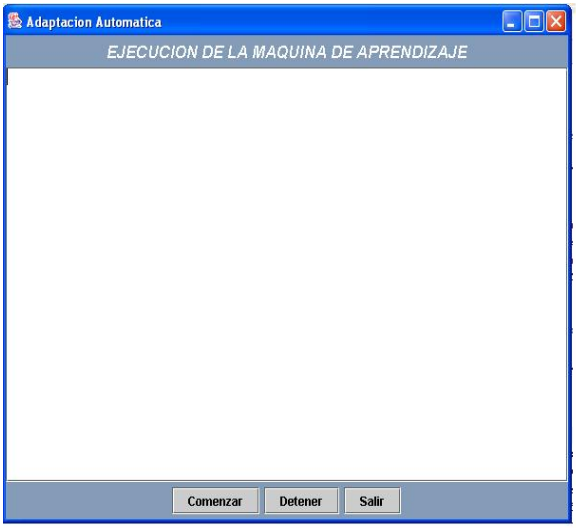

Fig. 10: Ventana para la ejecución automática

6. ANÁLISIS DE RESULTADOS

Aunque los SCs contemplan obligatoriamente un procesamiento paralelo de los mensajes generados a través del sistema ejecutor, el diseño del prototipo fue realizado según un procesamiento secuencial, en consecuencia, el tiempo de ejecución se hace lento. Este tiempo de ejecución depende de las variables presentes en las condiciones y de las acciones de las reglas consideradas. Mientras más condiciones en los antecedentes y consecuentes de las reglas existan, mayor será el tiempo de procesamiento de cada regla activada. Esto también conlleva a una gran cantidad de mensajes manipulados por la lista interna del mecanismo de inferencia al efectuar el encadenamiento dinámico, lo cual genera más tiempo de ejecución de este mecanismo, y en consecuencia del sistema en general. Ahora bien, una forma para mejorar el tiempo de ejecución es la paralelización del sistema.

En cuanto al MA, este permite introducir nuevas relaciones entre las variables de la condición y de la acción que desconocíamos. En este sentido, es importante resaltar que las reglas generadas son afectadas por el uso de los operadores relacionales “Y” y “O” en la generación de las mismas. Cuando el MA hace uso únicamente del operador Y las reglas generadas aportan mejores resultados que cuando el mecanismo hace uso de ambos operadores en el proceso de evolución. En ocasiones, las reglas que hacen uso del operador relacional O pueden no ser útiles en la solución de los problemas. Por ejemplo, puede darse el caso de que una regla contenga en el antecedente condiciones relacionadas con el operador O que al unir comprendan todo el dominio de una variable; en este caso la regla en particular siempre será activada por el mecanismo de inferencia ante cualquier entrada, sin ser realmente útil ya que no permite realizar tareas de clasificación. Es posible para el MA detectar esta situación y eliminar dicha regla. Este problema no ocurre con el uso del operador Y.

Asimismo, el MA introduce nuevas instancias de reglas al generar nuevas relaciones entre los conjuntos difusos de la condición y de la acción, que brindan soluciones a problemas que las reglas que se tenían no permitían resolver.

Respecto a los reemplazos generacionales, en nuestro MA un padre solo puede ser reemplazado por su hijo. Establecimos que si el hijo tiene mejor crédito que el padre el reemplazo se haría efectivo.

Partiendo del hecho de que las reglas introducidas inicialmente por el experto son efectivas, y de que no queremos perder la información proporcionada por el experto, una forma de evitar su ocurrencia es hacer uso de la siguiente variante del mecanismo de reemplazo: nuestro MA reemplaza reglas que tienen menor crédito que sus hijos, si y sólo si las reglas a reemplazar tienen crédito menor al valor promedio de crédito de todas las reglas, es decir, hace uso del promedio de crédito de las reglas existentes. Las pruebas realizadas según este criterio fueron mejores que los basados en nuestra primera propuesta para el mecanismo de reemplazo.

En el caso de los SCDs, el MA establece nuevos rangos para los conjuntos difusos definidos, y genera funciones de pertenencia de tipo diferente al que presentaban las funciones de pertenencia originalmente. Esto permite corregir errores de diseño entorno al sistema y/o actualizar la información del sistema a los cambios que estén ocurriendo en el entorno.

Finalmente, se realizaron pruebas en las cuales se crearon sistemas con reglas que no aportaban soluciones ante ciertos eventos, con el objetivo de observar si el MA era capaz de generar reglas que ofrecieran soluciones acertadas a esos eventos [González *et al.*, 2005]. En estas pruebas el MA

logró descubrir reglas que aportaban soluciones a los eventos producidos en el ambiente que en un principio no podían ser resueltos.

7. CONCLUSIONES

Los SCs permiten la manipulación de una cantidad de información que es imposible manejar en forma manual cuando la cantidad de variables involucradas en el sistema es numerosa; en consecuencia, el uso de herramientas de este tipo en la gestión de sistemas complejos es un elemento importante que apunta a la optimización de los mismos.

Los sistemas basados en conocimiento clásicos tienen la capacidad de resolver problemas lo suficientemente complicados, para lo cual requieren una experticia humana significante. El deterioro de su comportamiento a lo largo del tiempo, al cambiar las condiciones del sistema para el cual fueron diseñados, y su deficiencia para poder generar técnicas de razonamiento adaptativas, pueden ser solventadas gracias a las MDA. Las MDA ofrecen un marco de trabajo para solventar tales situaciones, a través de mecanismos de aprendizaje basados en dos subsistemas: el subsistema de asignación de crédito y el subsistema descubridor.

En este trabajo se desarrolló un mecanismo de aprendizaje para una herramienta de gestión de SCs y SCDs, denominada SAGSECD. Se pretende que el mecanismo de aprendizaje se adapte a las características de cualquier sistema gestionado por el SAGSECD. El mecanismo reforzará el conocimiento que se tenga, para poder contar con reglas de razonamiento adaptativas que ayudarán a tomar decisiones, clasificar y/o reconocer comportamientos, entre otras cosas.

Se consideraron dos módulos de desarrollo, uno para los SCs y otro para los SCDs. Ambos módulos utilizaron la PG como técnica evolutiva para desarrollar el subsistema descubridor, esto facilitó la representación de las reglas involucradas en el proceso. El módulo correspondiente a los SCDs hizo uso también de los PEs, como técnica para la evolución de las funciones de pertenencia de los conjuntos difusos de las variables difusas. El mecanismo de asignación de crédito también fue tratado en dos módulos, el primero fue dedicado a las reglas y se basó en el mecanismo conocido como Bucket Brigade, el segundo se refirió a las funciones de pertenencia de los conjuntos de las variables de los SCDs, y fue basado en el grado de participación que dichos conjuntos tenían en la solución de los problemas.

Una vez incorporado el MA a la arquitectura de SAGSECD, este fue probado para desarrollar un SC para el control de un tanque. En ese ejemplo se ejecutó el algoritmo básico de los SCs utilizando mensajes provenientes del ambiente en gestión, permitiendo observar el proceso de descubrimiento de reglas, y en el caso de los SCDs también de funciones de pertenencia. Los resultados obtenidos demuestran que el MA cumple con el objetivo propuesto. Las reglas descubiertas por nuestro mecanismo permiten dar soluciona problemas que no se podían resolver. Es importante decir que el MA proporcionó, en algunos casos, reglas no fáciles de inferir. Las reglas establecieron relaciones entre variables, o entre conjuntos difusos de las variables difusas.

Debido a la naturaleza del funcionamiento de los SCs, para futuros trabajos se debería considerar la paralelización de SAGSECD. Particularmente, esa es una forma de mejorar los tiempos de ejecución del sistema, lo cual es necesario en sistemas con muchas variables.

AGRADECIMIENTO

Este trabajo ha sido financiado por el FONACIT (proyecto 2005000170) y el CDCHT-ULA (proyecto I-820-05-02-AA).

REFERENCIAS

Aguilar J., and M. Cerrada (2000) Un Sistema Clasificador Difuso para Manejo de Fallas, *Revista Técnica de la facultad de Ingeniería*, Universidad del Zulia, **Vol. 23**, pp. 98-108.

Aguilar J., and F. Rivas (ed.). (2001) *Introducción a la Computación Inteligente*. Meritec, Mérida - Venezuela.

Aguilar J., F. Rivas, and J. Sánchez (2004), Knowledge Based and Inference Motor for an Automated Management System for developing Expert Systems and Fuzzy Classifiers, *WSEAS Transactions on Systems Journal* , **Vol. 3**, Nº 2, pp. 682-687.

Aguilar J., Y. Menolascina and F. Rivas (2005). Compiler Design for Fuzzy Classifier Systems, *WSEAS Transactions on Systems Journal*, **Vol. 4**, Nº 4, pp. 262-267.

Goldberg D. (1989) *Genetic Agorithms in Search, Optimization, and Machine Learning*, Adison – Wesley Publishing Company Inc. New York.

González G. and J. Aguilar (2005), Diseño e Implementación de un Mecanismo Adaptativo para Sistemas Clasificadores, Informe Técnico No. 12-2005, CEMISID, Universidad de Los Andes, Venezuela.

Holland J. (1975) *Adaptation in Natural and Artificial Systems*, Ann arbor: The University of Michigan Press.

Lashon B., D. Goldberg, and J. Holland (1989) Classifier Systems and Genetic Algorithms, *Artificial Intelligence*, **Vol. 40**, 1989, pp. 235-282.

Parodi A. and P. Bonelli (1993) A New Approach to Fuzzy Classifier System, *Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA'93)*, San Mateo,, p.p. 223-230.

Valenzuela R. (1991) The Fuzzy Classifier System: Motivations and First Results", In H.-P. Schwefel and R. Mnner editors, *Parallel Problem Solving from Nature II*, Springer, Berlin, pp. 330-334.

Wilson S. and D. Goldberg (1989) A critical review of classifier systems, In J. D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kauffmann*, pp. 244-255.

Wilson S. (2000) Get Real! XCS with Continuous-Valued Inputs, *Lecture Notes in Computer Science*, **Vol. 1813**, pp. 209-219.