Sistemas Clasificadores Genéticos aplicados al juego de Damas

Dr. Felipe Padilla Díaz[1]
Dr. Marcelo Mejía Olvera[2]
Mtro. Francisco Alvarez Rodríguez[3]
Mtro. Alejandro Padilla Díaz[4]

1. INTRODUCCIÓN

Dentro del campo de la Inteligencia Artificial, se han desarrollado multitud de sistemas de aprendizaje. Los Sistemas Clasificadores [HOL75, HOL80, HOL85, HOL86, HOL95, GB89, MIM96, BUT00], objeto de este trabajo, se estudian desde el punto de vista del comportamiento, perspectiva denominada conductista, que considera sólo el cambio de comportamiento de un sistema defendido, entre otros, por Narendra, Thathachar y Simon, [THNA89]. En general, en los sistemas compuestos por diferentes funciones (o reglas) se pueden aplicar sistemas de aprendizaje para obtener el conjunto de funciones (o reglas) que proporciona una solución mejor [LIT88]. Cada función propone una salida para cada posible situación de entrada. El sistema toma una decisión conjunta basada en una votación entre todas las funciones. Cada función está pesada por un factor que valora el número de aciertos y equivocaciones de cada función a lo largo de la experiencia anterior. La decisión final se compara con la salida ideal y se produce el aprendizaje a través del ajuste de los pesos para cada función según si se han equivocado o han acertado. Este tipo de sistemas discriminan entre las distintas funciones (o reglas) definidas *a priori*. Existen distintas variantes de este tipo de sistemas [LW94]. En general, este tipo de sistemas de aprendizaje basan el proceso de aprendizaje en el conocimiento *a priori* del dominio.

Los Sistemas Clasificadores (SC), aúnan las ventajas de estos sistemas con la posibilidad de aplicar un sistema de aprendizaje independiente del dominio como es el caso de los Algoritmos Genéticos. En un SC, el valor relativo de las diferentes reglas es una de las piezas clave de la información que debe ser aprendida. Para facilitar este aprendizaje, los SC fuerzan a las reglas a coexistir en lo que se denomina un servicio de economía basado en la información. Se mantiene una competición entre reglas, donde el derecho a responder a la activación va desde los ofertadores más altos, que pagarán el valor de sus ofertas a aquellas reglas responsables de su activación. En este recorrido se forma una cadena de intermediarios que va desde fabricantes (los detectores) hasta los consumidores (acciones hacia el entorno). La naturaleza competitiva de la economía asegura que las reglas buenas (beneficiosas) sobrevivan y las malas desaparezcan. En un SC los distintos niveles tienen un alto grado de relación y comunicación [GB89].

Las condiciones y mensajes de un SC forman un sistema de reglas que los convierte en una clase especial de sistema de producción. Uno de los principales problemas que plantean los sistemas de producción radica en la complejidad de la sintaxis de las reglas. Los SC se alejan de este problema restringiendo cada regla a una representación de longitud fija. Esta restricción tiene dos ventajas: la primera es que todas las reglas, bajo un alfabeto permitido, tienen significado sintácticamente, y en segundo lugar, una representación en cadenas de longitud fija permite la aplicación de operadores de cadena de tipo genético. Esta última ventaja proporciona una puerta abierta a la búsqueda en el espacio de reglas permitidas, mediante el empleo de Algoritmos Genéticos [HOL75].

1 Profesor Investigador del Depto. de Sistemas Electrónicos. Centro de Ciencias Básicas. Universidad Autónoma de Aguascalientes
E-mail: fpadilla2000@yahoo.com
2 Instituto Tecnológico Autónomo de México
E-mail: marcelo@itam.mx
Universidad Autónoma de Aguascalientes.
3 E-mail: fjalvar@correo.uaa.mx
Universidad Autónoma de Aguascalientes
4 E-mail: apadilla@correo.uaa.mx

Como ya se ha comentado anteriormente, los Sistemas Clasificadores tradicionales combinan la representación del conocimiento mediante reglas con el aprendizaje genético. Existe una clara diferencia entre los sistemas que utilizan Algoritmos Genéticos para el aprendizaje y los Sistemas Clasificadores. En los primeros, la solución del problema se encuentra totalmente codificada en la representación binaria con la que trabaja el Algoritmo Genético, de manera que la evaluación de un individuo es la evaluación de una solución completa [MIM96]. En los Sistemas Clasificadores, sin embargo, la evaluación de una salida representa la evaluación de una regla que contribuye parcialmente a la solución del problema. Esta evaluación es repartida entre todas aquellas reglas que contribuyen a la activación de la regla final, mediante el algoritmo de reasignación de créditos, pero en ningún caso es una evaluación del sistema compuesto por todas las reglas, esto constituye la aproximación propuesta por la Universidad de Michigan [HOL86a]. La generación de nuevas reglas o conjuntos de reglas se realiza a partir de estas evaluaciones. Así, aquellas reglas que han resultado activadas y dan una solución adecuada a una parte del problema serán origen de nuevas reglas. En la figura 1 se aprecia el esquema general del SC.

Uno de los grandes problemas relacionados con los Sistemas Clasificadores es el de la pérdida de reglas. Esta pérdida se produce por la aplicación del Algoritmo Genético sobre toda la población de reglas de manera conjunta. Evidentemente, los operadores genéticos discriminan las reglas por el valor de la fuerza, de manera que la evolución favorece la generación de aquellas reglas cuya fuerza es mayor. Cuando el sistema de aprendizaje trabaja en un entorno donde es posible generar un conjunto de entrenamiento muy completo, la fuerza de las reglas del SC reflejará la relación relativa entre reglas de forma adecuada y, por lo tanto, la aplicación del Algoritmo Genético producirá los efectos deseados. Sin embargo, cuando el proceso de aprendizaje presenta casos particulares y permite que el sistema vaya aprendiendo gradualmente a partir de estos casos, cada intervalo de aprendizaje con un conjunto de casos particulares puede producir que el reparto de la fuerza favorezca a un determinado tipo de reglas que serían a su vez favorecidas por el Algoritmo Genético. Si se extiende este razonamiento a todo el proceso de aprendizaje, puede ocurrir que la diversidad genética, tan necesaria para el aprendizaje, desaparezca debido al crecimiento en la población de un determinado tipo

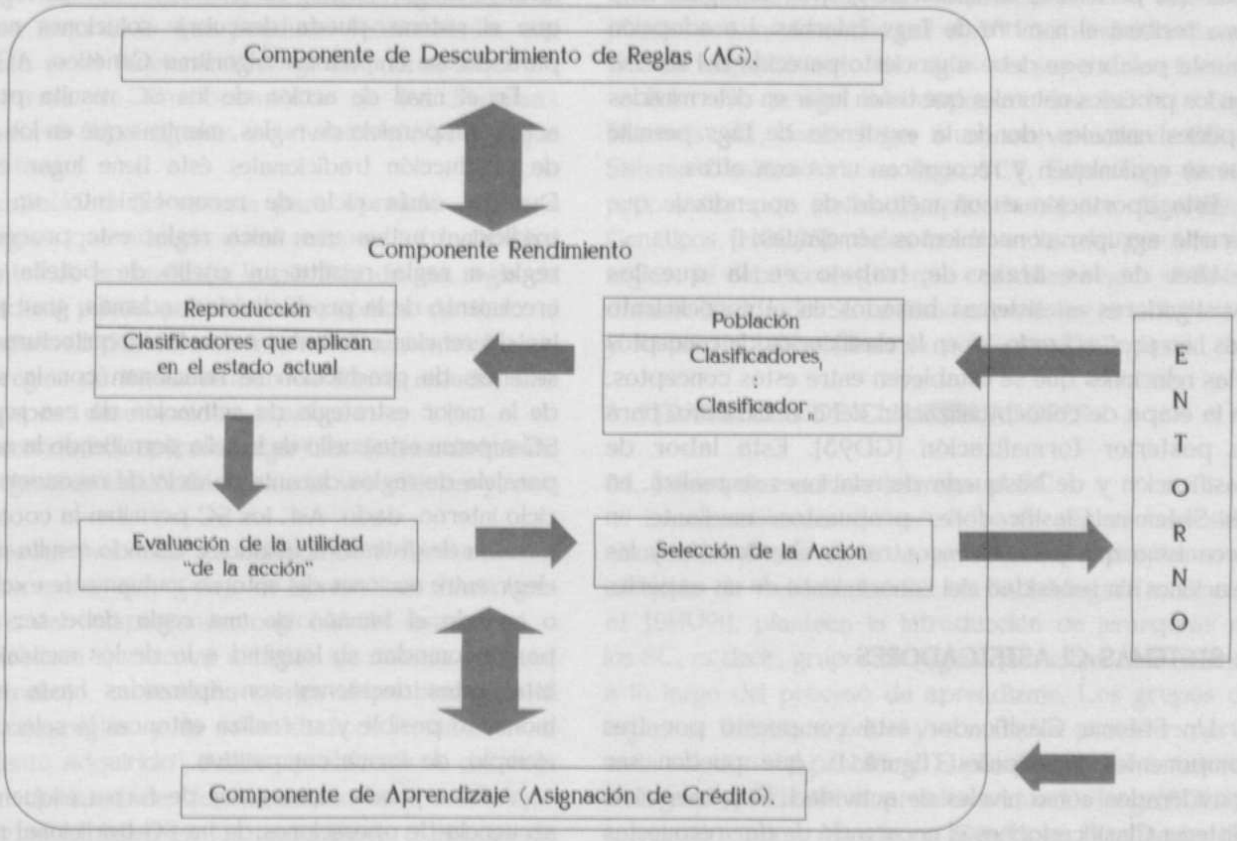


Figura 1: Esquema general del Sistema Clasificador.

de reglas. Además, cuando existen diferentes conjuntos de reglas necesarios para solucionar parcialmente el sistema, éstas pueden desaparecer si en los ejemplos encontrados hasta un determinado momento, parte de la problemática (la correspondiente a las reglas que se pueden perder) no se produce. Sin embargo, dichas reglas pueden ser enormemente necesarias.

Este problema se hace especialmente grave cuando existen en el SC tipos de reglas muy diferenciados. Básicamente, la idea es establecer una división de las reglas en grupos de manera que se fuerce su permanencia en el sistema; esto permite la supervivencia de grupos de reglas al modificar la aplicación de los operadores genéticos clásicos, mutación y sobrecruzamiento, en el nivel de descubrimiento y al modificar la función de pago y las ofertas entre reglas del mismo grupo, haciendo que la recompensa obtenida por una regla del grupo afecte a todo el grupo.

El objetivo de este trabajo ha sido la obtención de una estructura de codificación que permita la evolución genética de estos grupos, de manera que el número y la relación de los mismos sea también aprendida en el proceso de evolución. Para ello se han introducido en la parte de condición y de mensaje de las reglas codificadas, una zona que permita la definición de grupos de reglas. Esta zona recibirá el nombre de Tags Internas. La adopción de esta palabra se debe a un cierto parecido del sistema con los procesos naturales que tienen lugar en determinadas especies animales, donde la existencia de tags permite que se comuniquen y reconozcan unos con otros.

Esta aportación es un método de aprendizaje que permite agrupar conocimientos semejantes.

Una de las áreas de trabajo en la que los investigadores en sistemas basados en el conocimiento más han profundizado, es en la clasificación de conceptos y las relaciones que se establecen entre estos conceptos, en la etapa de conceptualización del conocimiento, para su posterior formalización [GD93]. Esta labor de clasificación y de búsqueda de relaciones se realiza, en los Sistemas Clasificadores propuestos, mediante un mecanismo que permite encontrar la clasificación y las relaciones sin necesidad del conocimiento de un experto.

2. SISTEMAS CLASIFICADORES

Un Sistema Clasificador está compuesto por tres componentes principales (figura 1), que pueden ser considerados como niveles de actividad. El primer nivel (Sistema Clasificador) es el encargado de dar respuestas (adecuadas o inadecuadas) para la resolución del problema planteado. En este nivel se encuentran las reglas del sistema, codificadas mediante cadenas de caracteres de alfabeto restringido. La ejecución de este nivel produce una respuesta a una determinada situación. La adecuación de la respuesta al problema que se desea resolver se mide mediante la recompensa que recibe dicha regla del entorno. El segundo nivel (Asignación de Créditos) evalúa los resultados obtenidos en el nivel inferior, distribuyendo las recompensas recibidas por las reglas que proporcionan la salida entre todas aquellas que han contribuido a la activación de cada una de esas últimas reglas. Como se trata de un método de aprendizaje reforzado, esta evaluación puede ser ajustada aplicando una recompensa o pago por parte del entorno, que tendrá un valor alto si la solución es adecuada y bajo si no lo es. La reasignación se puede llevar a cabo mediante distintos algoritmos [HOL86, LIE91] de los cuales el Bucket Brigade [HOL85] y el Q-Learning han sido los más extendidos y los que se han empleado en este trabajo. En este nivel, no resulta posible modificar, sin embargo, el comportamiento del sistema por medio de cambios en sus reglas, sino que sólo es posible ajustar sus valores y establecer, en cierta medida, una jerarquía de reglas buenas y malas. La tarea del tercer nivel (Descubrimiento) es encontrar nuevas vías para que el sistema pueda descubrir soluciones nuevas y para ello se emplea un Algoritmo Genético, AG.

En el nivel de acción de los SC resulta posible la activación paralela de reglas, mientras que en los sistemas de producción tradicionales ésta tiene lugar en serie. Durante cada ciclo de reconocimiento, un sistema tradicional activa una única regla, este procedimiento regla a regla resulta un cuello de botella para el crecimiento de la productividad; además, gran parte de las diferencias existentes entre las arquitecturas de los sistemas de producción se relacionan con la selección de la mejor estrategia de activación de esa regla. Los SC superan este cuello de botella permitiendo la activación paralela de reglas durante un ciclo de reconocimiento, o ciclo interno, dado. Así, los SC permiten la coordinación paralela de distintas actividades. Cuando resulta necesario elegir entre acciones del entorno mutuamente excluyentes, o cuando el tamaño de una regla debe ser podado para acomodar su longitud a la de los mensajes de la lista, estas decisiones son aplazadas hasta el último momento posible y se realiza entonces la selección, por ejemplo, de forma competitiva.

Así, se puede representar de forma esquemática la secuencia de operaciones de un SC tradicional mediante la Tabla I.

TABLA 1: SECUENCIA DE OPERACIIONES DEL NIVEL DE ACCIÓN DE UN SC TRADICIONAL

Paso	Operación
1	Llega del entorno un mensaje codificado de longitud k, a través de la interfaz de entrada.
2	Limpiar la lista de mensajes.
3	El mensaje del entorno se coloca en la lista de mensajes.
4	Todos los clasificadores cuya parte de condición coincida con el mensaje se activarán. Un mismo mensaje puede activar varios clasificadores.
5	Los clasificadores activados enviarán sus mensajes a la lista.
6	Los pasos 4 y 5 se repetirán para n ciclos internos.
7	Finalmente, se elegirá un mensaje para producir la salida, a través de la correspondiente interfaz.

Por lo tanto, el funcionamiento de los SC tradicionales se basa en tres conceptos fundamentales:

1. La solución del problema global es un conjunto de reglas (un subconjunto de reglas es una solución a situaciones concretas, incluso una sola regla puede ser una solución para una situación muy específica aunque no es habitual).
2. El pago de cada regla se reparte entre aquéllas que la activaron en los ciclos internos.
3. El Algoritmo Genético permite generar reglas a partir de las mejores lo que produce, teóricamente, una mejoría en el funcionamiento global del sistema.

La forma de funcionamiento de los Sistemas Clasificadores tiene algunos inconvenientes, de los que cabe destacar:

1. La capacidad del sistema para aprender cadenas de reglas, que además no se rompan entre distintos instantes de aprendizaje; la pérdida de una regla de la cadena puede provocar la pérdida de todo el conocimiento debido a las interrelaciones entre reglas. Las reglas no tienen sentido de forma aislada, sino en grupos, desconocidos *a priori*.
2. La necesidad de aplicar el algoritmo de descubrimiento para generar clasificadores cada vez mejores y, por último,
3. La secuenciación de los casos que se presentan al sistema para que guíen el aprendizaje hacia una mejora del comportamiento global del sistema.

El problema que se va a tratar de manera especial en este trabajo, es la lucha contra el problema de la pérdida de reglas y la necesidad de "mantener el conocimiento adquirido". Ambos problemas se deben a la acción del nivel de descubrimiento en los SC, que hace que éstos fallen en los mecanismos a la hora de formar y mantener asociaciones entre reglas.

El nivel de descubrimiento actúa sobre el conjunto de clasificadores que acaban de ejecutarse; de esta manera, las nuevas reglas se generan a partir de las mejores reglas anteriores a la acción del nivel de descubrimiento. Este funcionamiento puede llevar a perder aquellas reglas que son necesarias para resolver ciertos aspectos del problema, pero que hayan aparecido al principio del periodo de aprendizaje y que posteriormente no lo hayan hecho. Esto provoca que aquellas reglas que han sido muy buenas al principio de la ejecución puedan ser consideradas por el AG como menos valiosas, debido a que el resto de reglas las superan en fuerza.

Para ello se han introducido en los SC las "Tags Internas", TI, dando lugar a una nueva clase de SC, el Sistema Clasificador con Tags, SCT. Estas Tags fueron propuestas por Holland, aplicadas a los Algoritmos Genéticos [HOL95]. Además de evitar la pérdida de reglas, se debe conseguir que coexistan reglas distintas durante la vida del SC, evitando que éstas se uniformicen y se pierda variedad en la población de reglas.

3. JERARQUÍAS DE CLASIFICADORES

3.1. Jerarquías *ad-hoc* internas al SC

Los problemas de pérdida de reglas han sido abordados desde distintas perspectivas, siempre en la intención de mejorar los SC, en la bibliografía, Shu *et al.* [SHU91], plantean la introducción de jerarquías en los SC, es decir, grupos de reglas que se han de mantener a lo largo del proceso de aprendizaje. Los grupos de reglas se forman *a priori* y son dados por el experto en la solución del problema. El objetivo de estos autores es lograr solucionar el problema que DeJong [BGH89] solucionó en los Algoritmos Genéticos por medio del crowding. Así, por una parte establecen grupos de

reglas, familias, y por otra plantean operadores genéticos que actúan intra-familias e inter-familias. El sistema de pago es, así mismo, modificado y, cuando una regla de un grupo gana, todas las de su grupo reciben también ese premio.

Básicamente, el problema de la acción del nivel de descubrimiento es la consideración de todas las reglas como iguales. Esta idea, que en otras técnicas de Computación Evolutiva es lógica, ya que cada individuo es una solución al problema y por lo tanto todos deben competir con todos, no es extensible de forma directa a los SC. Esto es debido a que en muchos problemas cada una de las reglas no es capaz por sí misma de resolver el problema, por lo que no todas las reglas son iguales.

No es lo mismo una regla que en una determinada situación es disparada y su acción resuelve el problema, que un conjunto de reglas que deben dispararse de forma ordenada para resolver una situación distinta. En este caso, la primera regla puede ver aumentada su fuerza en una cantidad muy superior a la que verán aumentada su fuerza todas las reglas encadenadas del segundo caso.

Para resolver el problema, Shu propone la división del conjunto de reglas del SC en subconjuntos, cada uno de ellos con reglas especializadas en un determinado aspecto del problema, de manera que la competición se establezca entre los miembros de una misma familia de reglas.

Además, el reparto del pago entre miembros de una familia permite que el conocimiento adquirido en instantes anteriores no se pierda rápidamente, ya que una regla que alcanza un determinado valor de fuerza, sigue recibiendo fuerza por la ejecución de reglas de su misma familia.

La pérdida de reglas es especialmente crítica cuando el problema a solucionar necesita encadenamientos de reglas complejos, ya que la pérdida de una regla de la cadena en el nivel de descubrimiento, puede hacer que todo el encadenamiento sea olvidado y se produzca el olvido total de la cadena por lo que será necesario, posteriormente, volver a aprenderla.

3.2. SC's independientes organizados jerárquicamente

En 1995 Dorigo [DOR95] muestra los resultados del diseño de soluciones que hacen a los Sistemas Clasificadores aprender de la forma más rápida. Las herramientas que emplea son: paralelismo, una arquitectura distribuida y el entrenamiento. Respecto al paralelismo y la arquitectura paralela, propone una versión paralela del ICS [DOS93], mediante el diseño de un Sistema Clasificador paralelo, denominado Alecsys, aplicado a lo que se ha denominado el "animat problem", [WIL85]. Este problema se aborda desde la perspectiva de la división del problema en partes más pequeñas, desde una arquitectura jerárquica en la cual un conjunto de ICS aprende a cooperar en la solución de un problema de aprendizaje. Los distintos niveles del ICS se ejecutan de forma paralela en distintas máquinas y, por otra parte, distintos ICS, que se ocupan de distintas tareas, se ejecutan también de forma paralela. El autor, [DOR95], emplea la idea de "reactividad" de Brooks [BRO91], es decir, la existencia de un conjunto de comportamientos, cada uno de ellos implementado mediante un ICS, que son independientes entre ellos y que producen una salida para cada entrada. El sistema completo está compuesto por 3 sistemas: un ICS para evitar obstáculos, otro para alcanzar una meta y, por último, un sistema que decida cuál de las dos posibles salidas es la salida del sistema conjunto.

Para conseguir el encadenamiento entre reglas (que equivale a un encadenamiento entre comportamientos en este caso), el autor propone la inclusión de unas condiciones internas que permitan distinguir entre mensajes procedentes del entorno y mensajes procedentes de ciclos anteriores. El estudio de Dorigo se centra en la utilidad de las condiciones internas sin explicar claramente cómo son utilizadas internamente por el SC. Los resultados de esta parte del trabajo demuestran que el tamaño de estas condiciones internas, tal y como se utilizan, no es muy relevante para el aprendizaje.

En resumen, el trabajo de Dorigo [DOR95] propone una especie de jerarquía al componerse el SC final de tres SC: dos básicos y otro que decide cuál es el SC adecuado para cada situación. En este caso, la evolución de reglas se realiza de forma independiente, de manera que cada comportamiento evoluciona de forma separada. El problema de esta aproximación jerárquica frente a la de Shu, es la imposibilidad de realizar operaciones genéticas que permitan evolucionar al conjunto, ya que cada Sistema Clasificador es evolucionado de forma independiente y no tiene ninguna relación con los otros. Es decir, no pueden evolucionar relaciones entre cada uno de los comportamientos, de manera que una regla de uno de los clasificadores pueda activar a una regla del otro. La pregunta a responderse es si la separación del Sistema Clasificador en varios Sistemas Clasificadores aumenta la eficacia del sistema ante determinadas situaciones, pero, en cualquier caso, impide la generalización del aprendizaje.

La generación automática de categorías dentro de un SC no ha sido abordada en ningún trabajo anterior. Tal vez la idea pueda tomarse de la naturaleza: algunas especies utilizan "tags" para limitar una "llamada o aviso" a un conjunto de individuos, discriminando un subconjunto entre el conjunto total. Del mismo modo, se pueden incluir partes en las reglas que permitan discriminar unas de otras. Esta definición de lo que se va a denominar Tags Internas, TI, puede realizarse *ad-hoc*, creando una determinada cadena de llamadas [SHU91], o puede hacerse de forma que las propias TI evolucionen, determinando qué grupos resultan necesarios. En resumen, puede dotarse a cada una de las reglas de un campo, que evolucionará genéticamente y que identifica a esa regla como perteneciente a un grupo, de una forma similar a la propuesta por Holland con sus "tags" [HOL95].

4. EVOLUCIÓN DE TAGS EN UN SC: EL SCT

Como se ha comentado en la sección anterior, cualquier solución que trate de evitar la pérdida de reglas debe pasar por la creación de subconjuntos dentro del conjunto de clasificadores que componen el SC. Las dos soluciones que se han estudiado son:

- Creación de una jerarquía en la población a priori (descrita en la sección 3.1).
- Composición de varios Sistemas Clasificadores independientes (descrita en la sección 3.2).

Estas soluciones limitan la generalidad del proceso de aprendizaje del SC ya que dan por aprendidas en un caso las jerarquías, o la descomposición en varios SC en el otro, de manera que el marco del aprendizaje se ve limitado a estas restricciones.

La solución propuesta, por lo tanto, debe aunar la capacidad de aprender sin conocimiento *a priori* con la capacidad de generar algún tipo de subdivisión interna dentro del SC para permitir que existan categorías de reglas. Se ha diseñado un SC que recibe el nombre de SCT que permite la evolución de grupos de forma automática. Para plasmar esta solución se deberá modificar la codificación de los clasificadores de manera que se incluya un campo que represente el tipo o grupo, al que pertenece cada clasificador. Así por ejemplo dado un SC como el que se muestra en la tabla 2(a), si se reserva un campo de 1 bit para establecer las clases que componen el SC, el SC anterior quedaría como se muestra en la tabla 2(b).

Mediante este campo, el SC puede estar subdividido en varios grupos de clasificadores, cada uno de los cuales contienen aquellos clasificadores que tienen el mismo valor en ese nuevo campo. Por ello se puede decir que ese campo establece de qué tipo o grupo es el clasificador, como se recoge en la tabla 3(a). Se puede realizar la misma operación en la parte del mensaje, obteniéndose un SC como por ejemplo el de la tabla 3(b).

Tal y como se ha definido el valor del campo que establece las clases (en el ejemplo de la tabla3), existen 2 clases, una definida por aquellos clasificadores que tienen un 1 (los clasificadores 1, 2 y 3) y la otra por aquellos que tienen un *0* (los clasificadores 4, 5 y 6). Obsérvese que la definición de una clase viene determinada por el valor de ese campo en la parte de la condición de la regla, es decir, son del mismo grupo aquellas reglas que para ser activadas deben tener el mismo valor en ese campo. Este campo, que aparece en la codificación, evoluciona de la misma manera que el resto de campos, por lo que el número y tamaño de cada una de las clases de la jerarquía del SC es variable y se debe aprender. Es posible establecer muy diferentes grupos, pudiendo ocurrir que todos

TABLA 2(a): EJEMPLO DE REGLAS EN UN SC.

Clasificador	Regla	Mensaje
1	011*0*	*0*000
2	000*1*	111*0*
3	110*1*	1000
4	0000	000*1*
5	111*0*	110*1*
6	000*1*	011*0*

TABLA 2(b): EJEMPLO DE REGLAS EN UN SC, CON UN BIT RESERVADO PARA TI

Clasificador	Regla	Mensaje
1	011*0*X	*0*000X
2	00111X	1110*0*X
3	11011X	10000X
4	0000*1*X	00011X
5	1110*0*X	11011X
6	000*1*0X	011*0*1X

TABLA 3(a): SC DE LA TABLA 2 CON TI EN LA CONDICIÓN.

Clasificador	Regla	Mensaje
1	*O*11*O*1	*OOOO*X
2	*OOO*11	111*OO*X
3	11*O*11	1*OOO*X
4	*OOOOO*	*OOO*1X
5	111*OO*	11*O*1X
6	*OOO*1*O*	*O*11*O*X

TABLA 3(b): SC DE LA TABLA 2 CON TI EN CONDICIÓN Y MENSAJE

Clasificador	Regla	Mensaje
1	*O*11*O*1	*OOOOO*X
2	*OOO*11	111*OO*X
3	11*O*11	1*OOOO*X
4	*OOOOO*	*OOO*11X
5	111*OO*	11*O*11X
6	*OOO*1*O*	*O*11*O*1X

los clasificadores tengan el mismo valor, con lo que el sistema funcionaría como un SC clásico. Además de establecer el tipo de clasificador según el valor en la parte de la condición, al incluirse en la parte del mensaje y evolucionar esta de la misma manera, en realidad se están evolucionando no sólo los grupos de reglas, sino también cómo se produce la activación entre los grupos. Así, en el ejemplo anterior, podría evolucionar un conjunto de clasificadores como el de la tabla 3(b) y en este caso, los clasificadores del grupo 1 activan a los del grupo *O* y viceversa.

Evidentemente, este tipo de activación debe ser aprendida por el SC y pueden producirse diversas configuraciones.

Por último, debe tenerse en cuenta que la inclusión de un campo en los clasificadores obliga a incluir también un valor en el mensaje del entorno en esa posición. Este valor no viene determinado por el entorno, sino que se define *a priori* con un valor que codifica que ese mensaje es el del entorno. De esta manera, el SC tendrá que aprender qué grupo de reglas se activarán teniendo en su campo de definición de grupo ese mismo valor para responder al mensaje del entorno.

Para la aparición de jerarquías en el SC, es necesario que en cada regla se mantenga la información sobre la categoría a la que ésta pertenece. Esta información debe evolucionar genéticamente; evidentemente, puede ocurrir que si la información sobre la categoría en cada regla es capaz de representar "n" categorías distintas, la solución al problema se componga de m (m<n) categorías y que el resto de categorías sea irrelevante. Si se representa esta información en cada regla y se deja evolucionar, el número de reglas asociadas a una determinada categoría es también variable; en este sentido, la evolución genética de las categorías permitirá no sólo que evolucionen las categorías necesarias sino también que cada una tenga el tamaño necesario para la solución del problema.

4.1. Esquema de Funcionamiento del SCT

La secuencia de operaciones en el nivel de acción es la misma que la de los SC tradicionales (Tabla 1), pero en las condiciones y en los mensajes existen unos campos cuya única misión es la de definir la categoría a la que pertenecen las reglas que han enviado su mensaje a la lista de mensajes en el ciclo anterior. La primera activación se realiza con el mensaje del entorno, tal y como se muestra en la Tabla 1. El mensaje del entorno deberá contener unos valores por defecto que indiquen que se trata de un mensaje del entorno. A partir de este primer mensaje, los clasificadores activados enviarán sus mensajes a la lista de mensajes y, por lo tanto, se tendrá una generación de mensajes de distintas categorías que se habrán especializado en responder al mensaje del entorno. Esta información en los mensajes influirá en los siguientes clasificadores que se activen, es decir, intervendrá en el encadenamiento de las reglas que se van disparando. De esta manera evolucionarán reglas del tipo:

Instante inicial:
SI La_señal_externa es <tipo I>
ENTONCES mensaje <*OO*1...> y grupo <X>

Instantes siguientes:

SI el mensaje es <*OO*1...> y el grupo es <X>
ENTONCES mensaje <1*O*1...> y grupo <Y>

En definitiva, el mecanismo de inclusión de Tags Internas, TI, en las reglas favorece la evolución de soluciones elaboradas dentro de un SC. Como la ejecución del SCT se realiza en paralelo y todas las reglas son activadas al mismo tiempo, se generan, en la lista de mensajes, distintas estrategias elaboradas, encadenando reglas de distintos grupos. Estas estrategias se mantienen durante los ciclos internos de ejecución del SC, y mediante

los procesos de reasignación de créditos y descubrimiento se van aprendiendo las mejores.

Además de la necesidad de diferenciar la codificación para distintos grupos, será necesario realizar un ajuste de otros dos niveles del SC: el algoritmo de reasignación de créditos (BBA o Q-Learning), y el de descubrimiento (AG). Esto se debe a la necesidad de que cada uno de los grupos o jerarquías de reglas evolucione en paralelo y de forma gradual. Por una parte, resulta necesario el reparto del crédito ganado por una regla entre todas las reglas de su grupo para que tiendan a ganar a otros grupos, de manera que la fuerza de cada grupo puede ser considerada como factor a tener en cuenta a la hora de realizar las operaciones genéticas inter≤grupos. En este caso la evolución no es únicamente de un individuo, sino que la evolución se concentra en la generación de grupos compactos, los cuales son muy utilizados y, por lo tanto, deben tener un conjunto de reglas mejor, pero sin perder de vista la necesidad de grupos que pueden tener menos elementos pero ser indispensables para la elaboración de la estrategia final. Obsérvese que, si en un grupo se aumenta la fuerza de todas las reglas, cuando una de las reglas del grupo es valorada como positiva por el BBA o Q-Learning, aquellos grupos de reglas que se encadenan con este grupo verán aumentada también su fuerza debido a que el porcentaje de fuerza, que se entrega por la activación se calculará sobre cantidades mayores.

En resumen, un SCT, que permita la evolución libre de grupos de reglas presentará las siguientes modificaciones en cada uno de sus niveles:

- ***ACCIÓN:*** la codificación incluirá un conjunto de campos donde se almacenará la información sobre el grupo de pertenencia de la regla. Esta codificación afecta tanto a la condición como al mensaje lo que permite la generación de encadenamientos entre reglas, que pueden denominarse estrategias.
- ***BBA o Q-Learning:*** si una regla perteneciente a un grupo recibe un pago por ser parte de una buena solución, o por haber activado a otra que lo ha sido, el pago puede repartirse entre todas las reglas que forman ese grupo, apoyando el crecimiento del grupo y de aquellos que son activados por él, como conjunto.
- ***AG:*** han de existir operadores genéticos que actúen sobre reglas de un mismo grupo, permitiendo la evolución de los grupos, y otros que actúen de la forma habitual, permitiendo la generación de nuevas reglas, a partir de progenitores de distintos grupos.

5. EVALUACIÓN DEL SCT EN EL JUEGO DE LAS DAMAS

En este trabajo se pretende obtener una medida de la aportación de las Tags Internas, TI, al proceso de aprendizaje en un Sistema Clasificador. Para realizar una evaluación clara de la aportación de las TI en la codificación, se debe buscar un problema que se resuelva en un entorno perfectamente definido. El entorno elegido en este caso ha sido el de **finales en el juego de las damas.**

El objetivo de la aplicación del SCT al aprendizaje del juego de las damas no es la obtención de un SC que juegue a las damas, sino la aplicación de los Sistemas Clasificadores en un entorno claro y definido, que permita realizar una comparación entre los Sistemas Clasificadores tradicionales y la modificación del SC que incluye TI. Evidentemente, existen muchos sistemas que juegan a las damas, algunos de ellos con gran éxito [SCH97], pero para realizar un estudio y comparación se va a utilizar un jugador que sigue una estrategia aleatoria, y se van a medir las partidas que gana cada tipo de SC (clásico/con TI) frente al jugador aleatorio en distintas configuraciones.

5.1. Reglas del Juego

Existen muchas formas de jugar a las damas. En este trabajo se emplea las que juegan con tablero de ocho por ocho, con cuadros blancos y negros, con dos jugadores, uno con fichas blancas y otro con negras, que pueden ser, a su vez, damas o peones. Inicialmente, las fichas blancas están en la parte inferior del tablero, las negras en la superior y no hay damas. En este trabajo, los tableros iniciales no se emplean, pues sólo se trabaja con finales de partidas, *donde el número máximo de fichas es 5*, pudiendo ser estas cinco fichas de cualquier tipo y estar situadas en cualquier posición válida del tablero.

En la Figura 2 se muestran los posibles movimientos de un peón y una dama en el tablero. Las damas aparecen cuando un peón alcanza el extremo opuesto del tablero. Los bordes del tablero son los límites de los movimientos. El tablero no es continuo por sus bordes. En este trabajo se han considerado las direcciones de movimiento como absolutas, de la forma que se muestra en la Figura.

Cuando una ficha del jugador contrario se encuentra en una de las direcciones hacia las que se puede dirigir una ficha del jugador que mueve, esta última comerá a la que le obstruye el paso, colocándose en la casilla siguiente a la de la ficha comida, en la misma dirección.

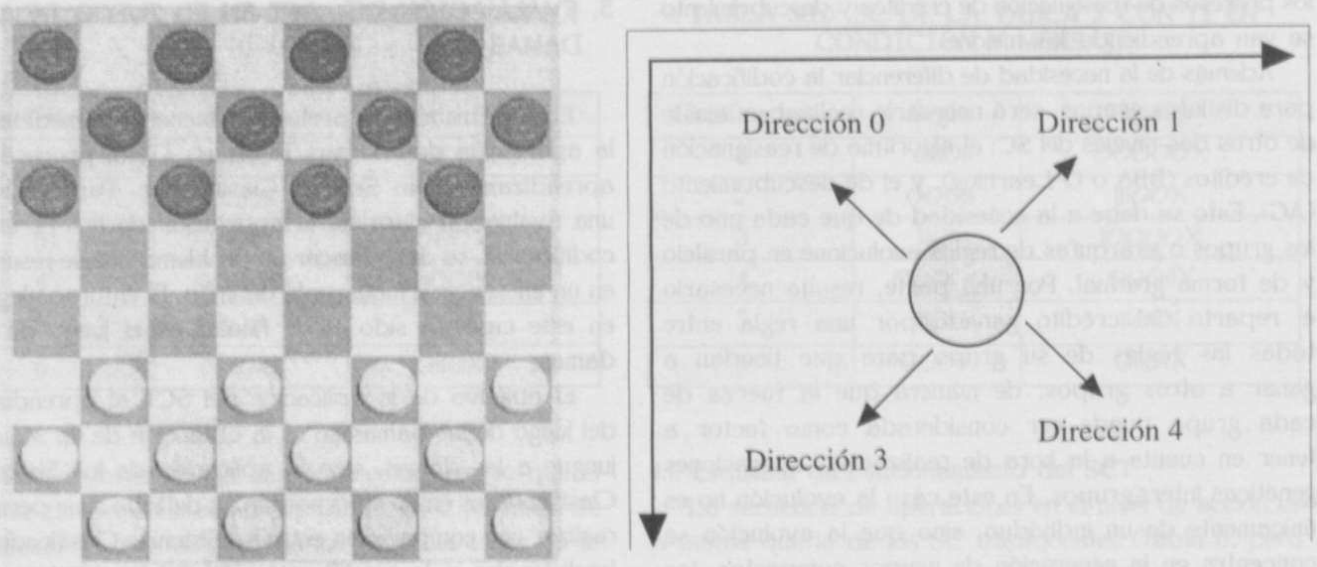


Figura 2: Posibles movimientos de peones y damas en un tablero y definición de las direcciones de movimiento.

La ficha comida desaparecerá del tablero. El proceso de comer se realizará tantas veces como sea posible antes de cambiar de turno. Cuando uno de los jugadores realice un movimiento o coma (y ya no le sea posible comer ninguna ficha más), se cambiará el turno y le tocará jugar al contrincante. El juego terminará cuando sólo queden fichas de un jugador en el tablero o se produzcan tablas. Las tablas tienen lugar cuando el jugador que posee el turno no pueda realizar ningún movimiento.

5.2. Codificación de la información

Se trata de analizar cómo y qué información sobre el tablero, las fichas, los jugadores, los turnos, movimientos, etc. debe ser suministrada al SC en forma de mensaje de entrada. La codificación elegida para el juego de las damas es tal que una salida del SC es siempre interpretada como un movimiento, esto quiere decir que las decisiones del SC son interpretadas en función de la situación en la que se encuentra el sistema. Evidentemente, el sistema debe ser capaz de jugar tanto con fichas negras como con fichas blancas, por lo que se ha elegido una codificación que no tiene en cuenta "el color" de la ficha. Además, las direcciones de movimiento se han tomado de forma absoluta, como se explicó en el apartado anterior.

5.2.1. Mensaje de Entrada

La información disponible en un tablero y susceptible de ser introducida al sistema es: el número de fichas que hay en el tablero, el color de cada una de ellas, coordenadas (x,y) de cada ficha, tomadas como se muestra en la Figura 2, tipo de ficha (peón o dama), direcciones en las que se puede mover o comer, y cuánto puede mover o comer en cada una de ellas. En los mensajes de entrada se recoge el estado del tablero en cada instante: número total de fichas, número de ellas que pertenecen al jugador SC, de qué color, turno, cuantas damas, etc. Esta información se codificará en un mensaje de entrada cuya longitud será de 57 bits, o si se trabaja con un clasificador con TI, el número de bits se incrementa hasta ól al introducir 4 bits para representar las TI.

La codificación del clasificador será:

- En la primera posición del mensaje de entrada se encuentra codificada la información sobre la posibilidad de comer *(con un 1)* o solamente de mover *(con un 0)*.
- Las 4 posiciones siguientes contienen información sobre el número total de fichas que hay en el tablero, y
- Las 12 siguientes sobre las fichas que pertenecen al jugador que tiene el turno, de ellas, cuántas son damas y el número de damas del contrario, todo ello codificado con 4 bits para cada uno de los casos, considerando el porcentaje que representan.

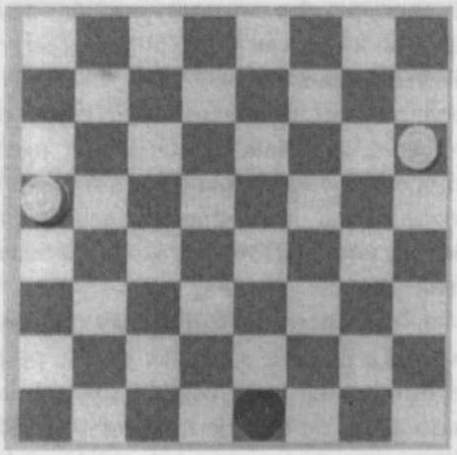
A continuación, se recoge la información referente a la posición ocupada por estas fichas, transformando dicho número decimal en un número binario de hasta 8 bits. Por último, si el número total de fichas es inferior a 5, el resto del mensaje de entrada se completa con

FIGURA 3: TRANSFORMACIÓN DEL TABLERO A SC

	(1)	(2)	(3)	(4)	(5)	Dec.	Unid.	Dec.	Unid.	Dec.	Unid.	Dec.	Unid.	Dec.	Unid.
	Come	Total	Mías	D. mías	D. El	Ficha 1	Ficha 1	Ficha 2	Ficha 2	Ficha 3	Ficha 3	Ficha 4	Ficha 4	Ficha 5	Ficha 5
Número		3	2	1	1	4	0	5	7	1	3				
Porcentaje		6	4	2	2										
Binario	0	0110	0100	0010	0010	0100	0000	0101	0111	0001	0011	****	****	****	****

símbolos "*". Para realizar la codificación se siguen los siguientes pasos (que puede observarse resumidos en la Figura 3), para un ejemplo que se encuentra recogido en la Figura 4.

En la Figura 3 se muestra la transformación que sufren los datos del entorno para ser introducidos en el mensaje de entrada del SC, corresponde a la situación de la Figura 4 y se trata de decidir el movimiento de las fichas blancas.



5.2.2. Mensaje de Salida

El mensaje de salida tiene una longitud igual a la del mensaje de entrada, 61 ó 57 bits según si se consideran o no las TI. De todo el mensaje que el SC envía a través de la interfaz de salida, después de haber realizado el proceso de encadenamiento durante varios ciclos internos, se utilizan como salida únicamente los últimos 16 bits. Un ejemplo de mensaje de salida podría ser el que aparece en la Figura 4.

Respecto a los mensajes de salida, se tomarán las correspondientes posiciones, y se realizará el proceso de decodificación. Para su mejor comprensión, se irá particularizando en el ejemplo de tablero de la Figura 3, y como mensaje de salida el de la Figura 4.

1. Cálculo de la ficha a mover/comer:
   - Con los bits de las posiciones 45 a la 52, se calculará qué ficha es la que se ha de mover/comer. En el ejemplo de la Figura 4 se trata del número binario *00000100*, que corresponde al número decimal 4.
   - Para decidir a qué ficha corresponde, se cuenta ese número decimal sobre las fichas reales existentes, y cuando éstas se terminen, si aún no se ha alcanzado dicho número, se vuelve a comenzar, desde la ficha *0* hasta la 4, ambos inclusive; en el ejemplo correspondería a *0,1,0,1,0*, por lo que la ficha elegida sería la *0*, de coordenadas (4,*0*).
2. Cálculo de la dirección de movimiento:
   - Con los bits de las posiciones 53 a 56, se calcula la dirección de la ficha. En el ejemplo se trata del número binario *0110*, que corresponde a la dirección 6.
   - Para transformar la dirección 6 en una dirección de ficha real, se cuenta ese número de veces la dirección decimal real, sobre las posibles direcciones hacia las que se pueda mover y, cuando no existan direcciones reales, se continuará contando desde la primera dirección real. En el ejemplo, la ficha *0* sólo tiene dos direcciones de movimiento, la 2 y la 3 (ver Figura 3) (pues se trata de una dama que se encuentra en el borde del tablero), por lo que contando desde *0* a 6,

Posiciones	0	...	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
Mensajes	*	...	*	0	0	0	0	0	1	0	0	0	1	1	0	0	1	0	1

Figura 4: Ejemplo de mensaje de salida, que deberá ser traducido en la acción correspondiente por la interfaz de salida.

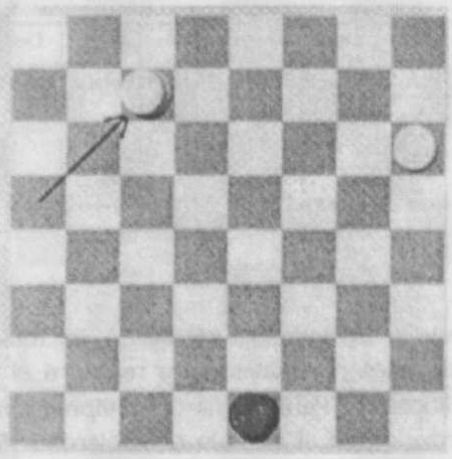


Figura 5: Tablero de la Figura 3 después de realizar los movimientos sugeridos por el SCT.

ambas inclusive, entre las dos direcciones posibles se obtendrá: 2,3,2,3,2,3,2, por lo que la dirección elegida será la 2.

3. Cálculo de las posiciones que ha de avanzar la ficha elegida en la dirección correspondiente:
   - Con los bits de las posiciones de la 57 a *6O*, se calculará el número de celdas a avanzar. En el ejemplo se trata del número binario *OlOl*, que corresponde al número decimal 5.
   - Ahora hay que transformar dicho valor en una cantidad real. Para ello, si el valor corresponde a una situación posible, se toma dicho valor; en caso contrario, se transforma en un valor entre 1 y el máximo. En el ejemplo, se ha obtenido un 5 como movimientos a realizar y, sin embargo, sólo son posibles una, dos o tres celdas de movimiento, por lo que se mueve dos posiciones.

En resumen, con el procedimiento detallado y sobre el ejemplo, la estrategia sugerida por el SCT mediante el mensaje de salida que se recoge en la Figura 4, es: *mover la ficha de coordenadas (5,O) en la dirección 2, 2 posiciones. Lo que daría lugar al tablero de la Figura 5.*

5.3. Función de pago

La función de pago que analiza la bondad de los clasificadores, tiene como objetivo guiar el aprendizaje del SC. El SC aprenderá según la función de pago y que, precisamente ésta, es el objetivo central del desarrollador de SC cuando éstos se aplican a un problema concreto. Para los propósitos de este trabajo interesa que el SC sea capaz de ganar a un jugador aleatorio, es decir un jugador que carece de estrategia y para el cual los movimientos no están determinados por la situación. Este objetivo hace innecesario buscar una función de pago capaz de evaluar distintas situaciones y de encontrar la mínima sutileza en los movimientos que decide el SC; por el contrario se debe suponer una función de pago simple que evalúe objetivamente las decisiones tomadas por el SC y donde cada movimiento sea evaluado por resultados "cuantitativos". De esta manera, el SC será capaz de ganar a un jugador aleatorio, no a un jugador experimentado, y se podrá realizar una comparación "objetiva" entre el SC clásico y el SCT.

A continuación se detalla el tipo de Aprendizaje que usaremos y para ello mencionaremos primeramente en que consiste este tipo de aprendizaje.

6. APRENDIZAJE POR REFUERZO

El Aprendizaje por Refuerzo (en inglés Reinforcement Learning y usaremos el término RL) ataca el problema de aprender a controlar agentes autónomos, mediante interacciones por prueba y error con un ambiente dinámico, el cual le provee señales de refuerzo por cada acción que realiza.

Si los objetivos del agente están definidos por la señal de refuerzo inmediata, la tarea del agente se reduce a aprender una estrategia de control (o política) que permita maximizar la recompensa acumulada a lo largo del tiempo (ver [SUT98] para una formalización de esta tarea).

Si bien en sus orígenes el RL sirvió como una herramienta teórica limitada a problemas con pequeños espacios de estados, en la actualidad sus aplicaciones han alcanzado áreas de considerable complejidad tales como robótica, manufacturación industrial, problemas de búsqueda combinatorial, etc.

La aplicación del RL a problemas del mundo real, trajo aparejado la necesidad de adaptar las técnicas existentes en el área para manejar características complejas propias de este tipo de ambientes (ambientes estocásticos no estacionarios con grandes espacios de estados y/o acciones).

7. BALANCE ENTRE EXPLORACIÓN Y EXPLOTACIÓN

Una de las principales características del RL está dada por el hecho de delegar en el agente que aprende la responsabilidad de determinar la estrategia para explorar el ambiente.

A diferencia del aprendizaje supervisado, en RL es el agente quien controla los ejemplos de

entrenamiento mediante la secuencia de acciones que elige. Esto implica que el agente debe balancear la *exploración* de nuevos estados y acciones para obtener nueva información que le permita evitar óptimos locales, y la *explotación* de estados y acciones ya aprendidos y con un alto reward que le garantice un reward acumulado aceptable.

Dado que es imposible explorar y explotar en forma simultánea con una única selección de acción, esta situación es a menudo denominada como el **"conflicto o dilema"** entre exploración y explotación.

Existen varias propuestas para lograr el balance entre exploración y explotación (ver [THRO1] para un survey) pudiéndose mencionar entre las más conocidas a la estrategia Î-greedy [LIT96], valores iniciales optimistas [6], métodos de selección de acción basados en la distribución de Boltzmann [LIT96, MITO1, SUT98, THRO1], el método de estimación de intervalo [KAE93], el bono de exploración usado en Dyna [SUT91, SUT9O] y los mapas de competencia [THR92].

La necesidad de mantener un mínimo de exploración es un factor fundamental en problemas reales, si se toma en cuenta las características dinámicas de este tipo de ambientes.

8. ACELERACIÓN DEL APRENDIZAJE

Uno de los principales problemas del RL es que el agente requiere un gran número de episodios de entrenamiento para aprender una función de valor aceptable.

Existen actualmente dos enfoques principales para la aceleración del proceso de aprendizaje:

1) permitir la incorporación de información provista por un observador externo [LIN92,MAC92, PEN93], e
2) integrar learning con planning [LIN92, MOO93, PEN93, SUT91, SUT9O].

El primero consiste en posibilitar que un experto u observador externo pueda incorporar "consejos" que le sirvan al agente para aprender ciertos aspectos complejos del ambiente en forma más eficiente.

En el segundo caso, el agente aprende un modelo del ambiente en forma simultánea al aprendizaje de la política, lo que permite hacer un uso más intensivo de una cantidad limitada de experiencia.

9. GENERALIZACIÓN

El RL se basa en la estimación de funciones de valor óptimas definidas sobre el conjunto de estados o el conjunto de acciones. Cuando el espacio de estados o acciones es pequeño, estas funciones son usualmente representadas explícitamente en forma tabular.

Si bien este tipo de representación trabaja relativamente bien en dominios pequeños como el que nos interesa en este trabajo, se torna inadecuado para la mayoría de los dominios complejos del mundo real, donde los espacios de estados o acciones suelen ser excesivamente grandes e incluso continuos. En estos casos, la representación tabular explícita de las funciones de valor no sólo implicará requerimientos de memoria inaceptables, sino además un uso ineficiente de la experiencia adquirida durante el aprendizaje. Cuando

TABLA 5: FUNCIÓN DE PAGO PARA LOS SC Y SCT APLICADOS AL JUEGO DE LAS DAMAS

		SISTEMA CLASIFICADOR (Computadora)			
		Come nsc DAMA	No Come DAMA	Come nsc No DAMA	No come No Dama
Jugador Contrario	Come n_{et} DAMA	$n_{et} \geq n_{et}$: $-100*n_{et}$	$-200*n_{et}$	$n_{et} \geq n_{sc}$: $-100*n_{et}-10$	$-200*n_{et}-10$
		$n_{et} < nsc$: $100*n_{et}$		$n_{et} < n_{sc}$: $100*n_{et}-10$	
	No Come DAMA	$200*n_{sc}$	20	$200*n_{sc}-10$	-20
	Come n_{et} No DAMA	$n_{et} \geq n_{sc}$: $-100n_{et}+10$	$-200*n_{et}+10$	$n_{et} \geq n_{sc}$: $-100*n_{et}$	$-200*n_{et}$
		$n_{et} < n_{sc}$: $100*n_{et}10$		$n_{et} < n_{sc}$: $100*n_{et}$	
	No Come No DAMA	$200*n_{sc}+10$	70	$200*n_{sc}$	0

los espacios de estados o acciones incluyen variables continuas o sensaciones complejas tales como imágenes visuales, la mayoría de los estados encontrados nunca habrán sido experimentados antes. La única manera para aprender algo sobre estas tareas es generalizar desde estados previamente experimentados a aquellos que nunca han sido vistos antes. Esto se puede lograr a través de las técnicas de generalización clásicas en aprendizaje automático las cuales permiten un almacenamiento compacto de la información aprendida y la transferencia de conocimiento entre estados y acciones "similares".

En este sentido, la idea consiste en aproximar las funciones involucradas en las arquitecturas y algoritmos de RL usando cualquiera de la amplia variedad de técnicas de aproximación de funciones para aprendizaje supervisado que soportan ejemplos de entrenamiento con ruido, como por ejemplo métodos de backpropagation, basados en memoria local [BOY95], árboles de decisión, etc.

La función de pago tiene en cuenta los siguientes factores: si se ha comido alguna ficha o no, si se ha hecho alguna dama o no, y el número de fichas que se han comido.

El pago se realiza una vez que el jugador contrario ha realizado un movimiento. Así, la función de pago empleada viene dada por los resultados conseguidos por el jugador contrario y los resultados obtenidos por el movimiento del SC. En este caso el pago puede representarse mediante la Tabla 5, donde n_{sc} es el número de fichas comidas por el SC y n_{ct} las comidas por el contrario.

Cuando termina la partida, el jugador contrario no realiza ningún movimiento sobre el cual poder evaluar el movimiento anterior del SC, por lo que se evalúa directamente el resultado de la partida:

SI (la partida termina en tablas)
ENTONCES (se paga *400*)

SI (ha ganado el SC)
ENTONCES (se paga *700*)

SI (ha ganado el jugador contrario)
ENTONCES (se paga - *700*)

El pago que se ha desarrollado es totalmente objetivo, y función de si se come o no y de si se hace dama o no, esto implica que no se realiza ningún pago cuando la jugada no tiene un resultado cuantificable. De hecho, cuando no existe ninguna cantidad que pueda medirse, el pago es *0*. Este pago *0* define las situaciones que no se van evaluar y por lo tanto limita la capacidad de aprendizaje del Sistema Clasificador. Esta limitación evita introducir cierta subjetividad en el pago que tenga en consideración el funcionamiento del SC y por lo tanto desvirtúe la comparación entre el SC clásico y el SCT.

10. COMPARACIÓN DEL SC TRADICIONAL Y EL SCT

Para efectuar la comparación entre ambos Sistemas emplearemos un nivel de aleatorización para probar cómo irá aumentando esta comparación. Los dos Sistemas comienzan sin ningún conocimiento previo, es decir, toda su población se genera aleatoriamente, por lo que sus reglas y mensajes no se adaptan a ningún caso concreto y sus movimientos serán, en un principio, también aleatorios.

Los tres tipos de experimentos realizados dentro de este punto se han realizado incrementando progresivamente su grado de dificultad, con el fin de examinar el comportamiento de los dos sistemas ante dichos cambios.

En el primer tipo de experimentos el jugador aleatorio va aumentando progresivamente su grado de aleatoriedad. Esto quiere decir, que dentro de un jugador aleatorio existen distintos niveles de aleatoriedad. Este grado de aleatoriedad se introduce en el mensaje de salida que produce el jugador aleatorio. El mensaje de salida del jugador aleatorio está constituido de igual forma que el del SC, pero en este caso únicamente posee dieciséis caracteres que son los que necesita el proceso de decodificación para transformarlos en un determinado movimiento. El mensaje de salida del jugador aleatorio está basado para todas las partidas que se han ejecutado en este apartado, en el siguiente mensaje:

Posiciones:	15															
Mensaje de Salida:	*0*	*0*	*0*	*0*	1	1	1	*0*	*0*	1	*0*	*0*	1	*0*	*0*	1	1

TABLA 6: EJEMPLO DE VARIACIÓN DE LA ALEATORIEDAD EN EL JUGADOR CONTRARIO.

% de caracteres aleatorios	Ejemplo de un mensaje de salida cuyos caracteres marcados son aleatorios.															
0 (fijo)	0	0	0	1	1	1	0	0	1	0	0	1	0	0	1	1
10	0	0	1	1	1	1	0	0	1	0	1	1	0	0	1	1
20	0	0	0	0	1	1	0	1	1	0	0	1	1	0	1	1
30	0	1	0	1	0	1	0	0	0	0	1	1	0	0	1	0
40	1	0	1	0	1	1	0	1	1	1	0	1	1	0	1	1
50	0	1	0	0	1	0	0	0	0	0	1	1	0	1	0	0
60	1	1	0	1	0	1	1	1	1	0	1	0	1	0	0	0
70	1	0	1	0	0	1	1	0	0	1	0	0	1	1	1	0
80	1	0	1	0	0	1	1	0	0	1	1	0	1	1	1	0
90	1	0	1	0	0	1	1	0	0	1	1	0	1	1	0	0
100 (completamente aleatorio)	1	1	1	0	0	0	1	1	0	1	1	0	1	1	0	0

La aleatoriedad se introduce en el mensaje de salida del jugador aleatorio dependiendo de la cantidad de caracteres de dicho mensaje que se generan aleatoriamente. Esta generación se regula proporcionalmente, es decir, existen jugadores aleatorios que poseen por ejemplo un 40% de caracteres aleatorios en el mensaje de salida. Un ejemplo de la aplicación de la aleatoriedad del jugador según los distintos porcentajes se muestra en la Tabla 6, donde aparecen resaltadas un conjunto de posiciones que se corresponden con aquéllas, en este ejemplo, cuyo valor puede variar respecto del original pues son generadas al azar.

Este tanto por ciento de aleatoriedad que posee el mensaje de salida, se aplica a cada movimiento que deba realizar el jugador aleatorio dentro de cada partida, por lo que en cada movimiento se genera un mensaje de salida distinto al creado en el movimiento anterior. Hay que tener en cuenta que, incluso aunque el mensaje de salida tenga un 0% de aleatoriedad, cada movimiento que produzca no ha de ser igual al realizado anteriormente, esto es así, porque el mensaje de salida, al sufrir una transformación en el procedimiento de decodificación, cambia el significado dependiendo del entorno (estado del tablero) en ese preciso momento, en este caso, depende de:

- El número de fichas que posea el jugador aleatorio.
- El tipo (dama o peón) de cada ficha del jugador aleatorio.
- Las direcciones reales hacia las que se pueda desplazar cada ficha del jugador aleatorio. Por ejemplo, una dama sólo podría desplazarse realmente hacia dos de sus cuatro posibles direcciones si se encontrase en uno de los límites del tablero.
- La cantidad de movimientos que pueda desplazarse cada ficha del jugador aleatorio dentro del tablero, en las direcciones que le permita el punto anterior. Por ejemplo, un peón sólo podría desplazarse realmente hacia una de sus dos posibles direcciones si en la otra dirección se encontrase una ficha.
- Si alguna de sus fichas puede comer o no. En este caso el mensaje de salida no serviría, porque el "juez" del juego de las damas actuaría comiendo la(s) ficha(s) del contrario que se pudiesen (elegiría aquella ficha que comiese mayor cantidad de fichas del contrario).

Para la realización de la comparación se han realizado tres grupos de experimentos con una situación inicial diferente. Los experimentos se han definido, de menor a mayor complejidad, en función del tablero inicial con el que comienza cada una de las partidas que se van a jugar: en primer lugar se dejará fijo el tablero inicial para todas las partidas, posteriormente se variarán las posiciones de las fichas que aparecen en el tablero en cada partida y, finalmente, el tablero inicial será generado al azar para cada una de ellas. En el primer experimento se aplicarán distintos grados de aleatoriedad al jugador contrario, comenzando en un 0% de aleatoriedad y se seguirá incrementando este porcentaje hasta concluir con un porcentaje de aleatoriedad de un 100%. En los dos últimos experimentos, el jugador contrario será siempre el más aleatorio y se irán modificando incrementalmente los tableros iniciales, ya sea modificando

la posición de las fichas, ya sea generando un nuevo tablero.

En resumen, puede deducirse de los resultados obtenidos, que los Sistemas Clasificadores son capaces de aprender en entornos de juegos y que, cuando el juego es complicado requiere una solución compleja que los SC Clásicos no pueden proporcionar de forma adecuada, por lo que se hace necesaria la inclusión de Tags.

Los resultados presentados han mostrado cómo los Sistemas Clasificadores propuestos son capaces de mejorar la aproximación clásica de los Sistemas Clasificadores en casos en los que el encadenamiento de reglas es relevante. La importancia de esta aportación es el descubrimiento de un método de aprendizaje que permite agrupar conocimientos semejantes, o que tienen alguna relación. Esta propiedad de las TI, la realización automática de grupos de reglas que comparten un mismo objetivo, resulta de especial interés, por lo que se ha realizado un estudio, analizando cómo afectan y qué resultados se obtienen en cada uno de los Sistemas Clasificadores propuestos.

11. CONCLUSIONES

En este trabajo se ha presentado un Sistema Clasificador modificado que incluye Tags. Este sistema permite, no sólo aprender a partir de prueba y error, sino que además permite la clasificación de las reglas aprendidas en grupos. En estos Sistemas Clasificadores, al emplear una codificación que utiliza conocimiento simbólico del problema y, al mismo tiempo, técnicas genéticas para el aprendizaje, resulta posible evolucionar, como parte de la solución del problema, la clasificación de conceptos.

Respecto a cómo el sistema es capaz de aprender qué conceptos están relacionados, la respuesta es que el Algoritmo Genético no establece relaciones entre conceptos mediante el análisis del significado de los mismos, sino en función de los resultados que se obtienen. Así, el Sistema Clasificador con Tags evoluciona los grupos de reglas de manera que la solución para el problema sea cada vez mejor. La pertenencia de una regla a un grupo no es función de su significado simbólico, sino de lo buena que resulta para que el sistema completo resuelva el problema, evaluada a través de la función de pago.

Estas ideas permiten la generación de estrategias elaboradas que solucionan problemas que necesiten una profundidad de encadenamiento elevada y donde cada una de las reglas de esta cadena sea imprescindible para la consecución de la solución final.

Por ejemplo, si se trata de un SC que juega a las damas, entonces cobra importancia la necesidad de obtener soluciones elaboradas y el que coexistan distintos tipos de reglas, (como reglas para comer y reglas para mover, por ejemplo) y aparece el problema de que si durante un cierto número de ciclos no se emplean reglas de alguno de los tipos, éstas pueden desaparecer por la acción del Algoritmo Genético. Para resolver este problema se ha propuesto el SC con tags, SCT, en el que las reglas cuentan con unas posiciones reservadas para codificar información respecto al grupo al que pertenecen.

12. REFERENCIAS

[BGH89] L. Booker, D.E. Goldberg y J.H. Holland, "Classifier Systems and Genetic Algorithms", Artificial Intelligence, 235-282, (1989).

[BOY95] J.Boyan y A. Moore. "Generalization in reinforcement learning: Safely approximating the value function". In Tesauro, Toretzky y Leen (Eds), Advances in Neural Information Proccesing Systems: Proc. of the 1994 Conference, pags. 369 - 376. Cambridge, MA. The MIT Press, 1995.

[BRO91] R.A. Brooks, "Intelligence Without Representation", Artificial Intelligence, 47, 139-159, (1991).

[CHA91] D.Chapman y L.P. Kaelbling. "Input generalization in delayed reinforcement learning: An algorithm and performance comparisons". In Proceedings of the Twelfth International Conference on Artificial Intelligence, pags. 726 - 731. Morgan Kaufmann, San Mateo, CA, 1991.

[DOR95] M. Dorigo, "ALECSYS and the AutonoMouse: Learning to Control a Real Robot by Distributed Classifier Systems", Machine Learning, 19, 209-240, (1995).

[DOS93] M. Dorigo y U. Schnepf, "Genetics-Based Machine Learning and Behavior Based Robotics: A New Synthesis", IEEE Trans. on Systems, Man and Cybernetics, 23:1, 141-154, (1993).

[ECM93] Proceedings of the European Conference on Machine Learning, Spinger, New York, (1993)

[ERR99] M. Errecalde y G. Aguirre. "Extendiendo interfases de usuario: agentes, aprendizaje y distribución". Trabajos del WICC´99. Workshop de aspectos teóricos de la Inteligencia Artificial, 1999.

[ERR99b4] M.Errecalde, M. Crespo y C. Montoya. "Aprendizaje por Refuerzo: Un estudio comparativo de sus principales métodos". Proc. del II Encuentro Nacional de Computación (ENC´99). Sociedad Mexicana de Ciencia de la Computación. México, 1999.

[ERR99c5] M. Errecalde, y G. Aguirre. "Una propuesta para integrar agentes de interfase, aprendizaje y sistemas multiagentes". Proc. del II Encuentro Nacional de Computación (ENC´99). Sociedad Mexicana de Ciencia de la Computación. México, 1999.

[GOL89] D.E. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning". Addison Wesley, Reading Massachusetts, (1989).

[GON93] A.J. González, D.D. Dankel, "The Engineering of Knowledge-

Based Systems", Prentice Hall, (1993).

[HOL75] J. Holland, "Adaptation in Natural an Artificial Systems". University of Michigan Press, Ann Arbor, (1975).

[HOL80] J. Holland, "Adaptive Algorithms for Discovering and Using General Patterns in Growing Knoledge Bases", International Journal of Policy Analysis and Information Systems, vol 4, 245-268, (1980).

[HOL85] J.Holland, "Properties of the Bucket Brigade". In Proc. of International Conference on Genetic Algorithms and their Applications, vol 1, 1-7, (1985).

[HOL86] J. Holland, "A mathematical Framework for studying Learning in Classifier Systems", Physica D, 22, 307-317, (1986).

[HOL86a] J.H. Holland, "Escaping Brittleness, The Possibilities of General Purpose Learning Algorithms Applied to Rule-Based Systems" en [MCM86], 593-623, (1986).

[HOL95] J.H. Holland, "Hidden order: how adaptation builds complexity". Reading Massachusetts, Addison-Wesley, (1995)

[KAE93] L.P.Kaelbling. Learning in Embedded Systems. MIT Press. Cambridge, MA, 1993.

[LIE91] G.E. Liepins, M.R. Hilliard, M. Palmer y G. Ranjaran, "Credit Assignment and Discovery in Classifier Systems", International Journal of Intelligent Systems, vol 6, 55-69, (1991).

[LIN92] L. - J.Lin, "Self-Improving Reactive Agents Based on Reinforcement Learning, Planning and Teaching". Machine Learning - Volumen 8 - Número 3/4 - Mayo 1992.

[LIT88] N. Littlestone, "Learning Quickly when irrelevant attributes abound: a new linear-threshold algorithm", Machine Learning, 2, 285-318 (1988).

[LIT96] L. P.Kaelbling, M. Littman y A. Moore. "Reinforcement Learning: A Survey". Journal of Artificial Intelligence Research 4 (1996) - 237-285 - Mayo 1996.

[LIT94] N. Littlestone y M.K. Warmuth, "The Weighted Majority Algorithm", Information and Computation, 108 (2), 212-261 (1994).

[MAC96] R.Maclin y J. W. Shavlik. "Creating Advice-Taking Reinforcement Learners". Machine Learning - Volumen 22 - Págs. 251-282. 1996.

[MIT01] T.Mitchell. "Machine Learning". Capítulo 13. (Versión preliminar), mayo 2001.

[MOO93] A.Moore y C. Atkeson. "Prioritized Sweeping: Reinforcement Learning with Less Data and Less Time". Machine Learning - Volumen 13 - Número 1 - Octubre 1993.

[PEN93] J. Peng y R. J. Williams. "Efficient learning and planning within the Dyna framework". Adaptative Behavior, 1(4), Págs. 437-454, 1993.

[RUS95] S.Russell y P. Norvig. "Artificial Intelligence. A modern Approach". Prentice -Hall -1995.

[SCH97] J. Schaeffer, "One Jump Ahead", Springer-Verlag, (1997).

[SHU91] L. Shu y J.Schaeffer, "HCS: Adding Hierarchies to Classifier Systems", Proceedings of the 4th International Conference on

Genetic Algorithms, 339-345, (1991).

[SUT90] R.Sutton. "Integrated Architectures for Learning, Planning, and Reacting Based on Approximating Dynamic Programming". – Proceedings of the Seventh Int. Conf. On Machine Learning, pp. 216-224, Morgan Kaufmann, 1990.

[SUT91] R. Sutton. "Dyna, an Integrated Architecture for Learning, Planning, and Reacting" Working Notes of the AAAI Spring Symposium, pp.151-155, 1991.

[SUT98] R. Sutton y A. Barto. "Reinforcement Learning: an introduction". The MIT Press, 1998.

[TES92] G. Tesauro. "Practical issues in temporal difference learning". Machine Learning - Volumen 8 - Número 3/4 - Pags. 257 - 277. Mayo 1992.

[THN89] M.A. Thathachar, K. Narendra. "Learning Automata, an Intro-duction" Prentice Hall International, Englewood Cliffs, N.J. (1989).

[THR01] S. Thrun. "The role of Exploracion in Learning Control". Handbook in Intelligent Control: Neural, Fuzzy, and Adaptative Approaches. White, D. A., & Sofge, D. A. (Eds.), Junio 2001.

[THR92] S. Thrun y K. Moller. "Active exploration in dynamic environments". Advances in Neural Information Proccesing Systems, 4, pags. 531 - 538. San Mateo, CA, Morgan Kaufmann, 1992.

[WIL85] S. Wilson. "Knowledge growth in an Artificial Animal". Proc. of the First International Conference on Genetic Algorithms and their Applications, 16-23, (1985).

[ZEN97] D. Zeng y K. Sycara. "Benefits of learning in negotiation". In Proceedings of AAAI-97, 1997.

[ZEN98] D.Zeng y K. Sycara. "Bayesian learning in negotiation". International Journal of Human-Computer Studies, 48, 1998.