Resolución de Juegos de Información Perfecta empleando Algoritmos Genéticos y Redes Neuronales

Raul Bouroncle Cuba
Escuela Profesional de Ingeniería de Sistemas
Universidad Nacional de San Agustín (UNSA) Arequipa, Perú
raulbouroncle@hotmail.com

Waldo Cancino Ticona
Instituto de Ciencias Matemáticas y de Computación de São Carlos (ICMC)
Universidad de São Paulo
wcancino@icmc.sc.usp.br

Juan Carlos Gutiérrez Cáceres
ICMC, Universidad de São Paulo
juan@icmc.sc.usp.br

Resumen

El principal problema en juegos complejos radica en la infinidad de alternativas para su solución. Sin embargo, considerando una de esas alternativas como una sucesión de estrategias puras, a través de generaciones de jugadas (como un árbol de juego), se puede resolver el problema con un Algoritmo Genético. Por otro lado, la función de utilidad que evalúa a los individuos, necesita saber el estado del juego, para aplicar la estrategia mas adecuada. Representando discretamente el estado del juego, podemos entrenar una Red Neuronal Backpropagation para reconocerlo.

Este trabajo de investigación propone una forma de resolver los juegos de información perfecta desde la perspectiva de la inteligencia artificial. Se mostrara el empleo de estas técnicas con la elaboración de un sistema híbrido que pueda sostener con éxito una partida de Ajedrez. La ventaja de esta propuesta frente al método extensivo es que las generaciones mantienen siempre el mismo número de individuos (sobreviven las mejores estrategias).

Palabras Clave: Juegos de Información Perfecta, Algoritmo Genético, Backpropagation,

Abstract

The problem with complex games is that there are many ways to play them. In other words the group of possible strategies is large. However, if we consider a strategy as a succession of pure strategies through generations in the game (as a game tree), we can solve the problem with a Genetic Algorithm (GA). Furthermore, fitness function, which evaluates the individuals, needs to know the state of the game, in order to apply the best strategy. Representing discreetly the state of the game (pattern) we can train a Backpropagation ´s Neural Network (BPN) to recognize it.

This research work proposes a method to solve games with perfect information from Artificial Intelligence's perspective. The use of this method will be shown on the development of a hybrid system, which can play chess departures successfully. The advantage between this method and the extensive method is that the number of individuals in each generation is always the same (only the best strategies survive).

Keywords: Games with perfect information, Genetic Algorithm, Backpropagation.

1 Introducción.

Para la mayoría de las personas, el termino juego sugiere un tema de poca amplitud y frívolo, lo que en verdad dista mucho de la realidad. John Von Neumann y Oskar Morgenstern propusieron la teoría de juegos para proporcionar un nuevo acceso a los problemas económicos, ellos intuyeron que "*los problemas típicos del comportamiento económico son estrictamente idénticos a las nociones matemáticas de juegos de estrategia adecuados*" [11]. Esta teoría tiene aplicación en la economía, ciencias políticas, sociales, sicología, marketing, finanzas, estrategias militares, etc.

La teoría de juegos esta formada en el fondo por varias teorías que describen los distintos tipos de juegos, por lo que no se puede esperar que un único juego o modelo describa todas las posibilidades que se dan en el mundo real. Pero básicamente esta teoría busca responder 2 preguntas: *¿Cómo deberían comportarse los jugadores?* y *¿Cuál debería ser el resultado final del juego?* , la respuesta a estas preguntas es lo que generalmente se denomina solución del juego [4].

Los métodos comúnmente empleados para la resolución de juegos básicamente consisten en un análisis de prácticamente todas las situaciones posibles que se pueden dar en el juego, es decir un método extensivo que implica el desarrollo del árbol de juego. Cuando se trata de un modelo complejo este tipo de métodos no son eficientes dado que las posibilidades de juego son enormes.

El enfoque propuesto aquí es utilizar un Algoritmo Genético (AG) para evaluar las posibilidades de juego ya que en esencia el AG tomara en cuenta las mejores opciones de juego y sus consecuencias en el desarrollo del juego a través de generaciones de jugadas dejando de lado las opciones inútiles o poco valiosas con el fin de simplificar el análisis. Además, debemos considerar que las condiciones de juego varían conforme se va desarrollando el mismo, por lo que sería enormemente útil poder conocer el estado del juego para determinar la forma de evaluar las posibilidades de juego en ese momento. Si hablamos de juegos de información perfecta, donde el estado del juego esta completamente determinado en cualquier momento del juego, podemos utilizar una Red Neuronal Backpropagation para este reconocimiento.

El juego que quizás ha despertado mas interés en los investigadores del tema es el Ajedrez, debido a su complejidad y además porque es un juego de información perfecta. Ernst Zermelo demostró en base a un teorema que se le puede atribuir un valor, es decir, que tiene una única forma de jugarse lo que implica que alguno de los jugadores puede forzar un resultado (aunque nadie ha encontrado dicho valor hasta ahora).

Este articulo está organizado de la siguiente manera: luego de esta primera introducción una descripción de la teoría existente sobre juegos de información perfecta que servirá para hacer en la siguiente parte el planteamiento teórico y práctico del método de solución propuesto. Luego un análisis comparativo con las otras técnicas comúnmente empleadas en la solución de este tipo de problemas y finalmente se muestran diez partidas de ajedrez realizadas con la aplicación desarrollada. Están en notación ajedrecística estándar.

2. Juegos de información perfecta (JIP).

La teoría de juegos es lo bastante amplia como para pretender hacer un análisis general de todos los juegos. Por ello el presente trabajo esta restringido únicamente a los juegos de información perfecta.

Los juegos de información perfecta son aquellos donde los jugadores en cada momento del juego están completamente informados de lo que pueda suceder, dicho de otra forma, no hay elementos "sorpresa" como el azar que si están en los juegos de información imperfecta, como por ejemplo el póquer. La resolución de estos juegos consiste en describir una (o varias) estrategias s que le garantice a un participante del juego obtener el mejor resultado posible teniendo conocimiento de todas las estrategias disponibles por los otros jugadores (información perfecta) y suponiendo que los demás participantes están igualmente informados, actúan de manera racional y buscan de igual manera obtener el mejor resultado. Es obvio que el análisis de los juegos de información perfecta es mas exacto y simple que el análisis de los juegos de información imperfecta, pero en el fondo la complejidad depende del juego que se este analizando, por ejemplo el ajedrez y es mucho mas complejo que el póquer.

El problema en la vida real es mas complicado porque como todos sabemos los jugadores pueden equivocarse, lo que significa que pueden actuar de manera irracional. En el fondo pueden suceder 2 cosas: Que exista una estrategia s que nos garantiza que obtendremos un resultado v que no se puede mejorar sin importar lo que hagan los oponentes, caso en el cual tenemos una estrategia ganadora s a priori y el valor del juego será v (el juego perderá todo interés porque todos sabrían como jugarlo como en el caso del tres en raya o *tic tac toe*).

El otro caso es que exista una estrategia *s* que nos garantiza un resultado *v* que no se puede mejorar suponiendo que el oponente actuara racionalmente e impedirá mejorar nuestro resultado, en este caso estamos ante una *estrategia de equilibrio* (en la misma situación están todos los jugadores). Aquí la complicación adicional es que ante la posibilidad de una acción irracional del o los oponentes, algún jugador podrá emplear una estrategia *s'* que le permita obtener un resultado $v' > v$. En conclusión, debemos poder *adaptar* nuestra estrategia conforme se vaya desenvolviendo el juego con el fin de obtener el mejor resultado de acuerdo con la *habilidad* demostrada por los oponentes.

En el análisis de los juegos de información perfecta podemos encontrar una clasificación que ha sido empleada tradicionalmente: juegos cooperativos y no cooperativos. Para la investigación aquí propuesta el análisis de los juegos no cooperativos es mas relevante ya que la demostración practica se hará con un juego no cooperativo y porque en la realidad el equilibrio cooperativo es relativo, ya que los jugadores no siempre actúan racionalmente.

Los juegos no cooperativos y los juegos cooperativos se formalizan de distinta manera. Los juegos no cooperativos, se presentan en forma estratégica, en forma extensiva o en forma grafica. En cambio, los juegos cooperativos se presentan en forma característica o coalicional, que consisten en la descripción de los pagos que reciben cada una de las coaliciones posibles. En la teoría de juegos además existen otras clasificaciones las cuales pueden ser [8]:

- Por el numero de jugadores pueden ser unipersonales, bipersonales o pluripersonales.
- Finitos o infinitos, dependiendo del numero de alternativas que tenga cada jugador en su turno y si el juego acabara en numero finito de jugadas.
- De suma cero o de suma no nula. El primero quiere decir que cuando acaba el juego, la suma de las ganancias es siempre cero (lo que un jugador gana, es lo que otro ha perdido), en el otro caso pueden perder o ganar ambos.

Para tener esperanza de hallar la estrategia adecuada para cada juego debemos tener la seguridad de poder forzar algún resultado en el juego. De hecho la primera conclusión importante que debemos tomar en cuenta con respecto a los juegos de información perfecta es la que obtuvo E. Zermelo en 1912 con este teorema: *"Sea T un conjunto cualquiera de resultados de un juego finito de dos jugadores, de información perfecta y sin jugadas al azar. Entonces o bien el jugador I puede forzar un resultado en T o el jugador II puede forzar una resultado en ~ T"* [10].

Juegos no cooperativos.

Los juegos no cooperativos son aquellos donde se requiere una descripción completa de las reglas del juego, de manera que las estrategias disponibles a los jugadores puedan ser estudiadas al detalle [2]. Estos juegos pueden describirse de 3 formas [1]: extensiva (árbol), estratégica (normal) o gráfica (esta ultima no es relevante). La forma estratégica o normal se refiere al juego que se define de forma que cada jugador escoge una estrategia, y el conjunto de estrategias escogidas entre todos los jugadores, simultánea e independientemente, determinan los resultados de cada jugador. La forma extensiva de un juego describe con gran detalle la secuencia de movimientos de los jugadores, la información que tienen los jugadores en cada momento del juego y otros detalles, como situaciones de azar [8].

Forma estratégica (normal) de un juego.

Formalmente, podemos definir un juego como un conjunto de los siguientes elementos:

$N = \{1,2,...n\}$: Conjunto de jugadores.

A: Conjunto de resultados posibles.

$P^k(A) = \{A^k\}, k \in N$: Preferencias sobre A para cada jugador (relaciones binarias, completas y transitivas).

$\{ S^k \}\ k \in$ N: Conjunto de estrategias del jugador k.

$a : S \to A$: Regla de selección: A cada n-upla de estrategias s, siendo s de la siguiente forma

$$s = (s^1, s^2, ..., s^n) \in S = \prod_{k \in N} S^k$$, le asigna un resultado en A.

π^k : S $\to \Re$: Funciones de utilidad ($\pi^k(s) \geq \pi^k(t) \Leftrightarrow s \geq t$).

Así pues, la forma estratégica de un juego finito es el conjunto formado por,

$$\{N, \{S^k\}_{k\in N}, \{\pi^k\}_{k\in N}\}. \tag{1}$$

Existen dos tipos de estrategias en un juego no cooperativo:

- Estrategias puras: asignar una acción a cada jugador.
- Estrategias mixtas: conjunto de distribuciones de probabilidad definidas sobre el conjunto de estrategias puras.

3. Resolución de juegos de información perfecta con un AG y una RBP.

Los Algoritmos Genéticos tienen aplicación en muchas áreas debido a que fueron concebidos precisamente como un método adaptativo, para encontrar una solución en cualquier ambiente (con algunas restricciones evidentes). La ventaja principal de los AG es la robustez, que es el equilibrio entre eficiencia y eficacia necesaria para sobrevivir en ambientes diferentes entre sí [6]. En ese sentido, la resolución de JIP puede ser propuesta bajo la perspectiva de los AG, mas aún considerando la resolución de JIP como un problema de optimización, problemas donde los AG han demostrado mejor su eficiencia.

Por otro lado, al desarrollar una partida de algún JIP, nos daremos cuenta que ante las numerosas posibilidades de juego, la memoria nos podría ser de mucha ayuda. Es decir, si nos encontramos en una situación similar que ya se analizó en el pasado, recordaríamos directamente la acción tomada como solución a la situación presente, o al menos, saber que aspectos son relevantes para la evaluación de las posibles jugadas. Necesitamos entonces codificar cada situación y sus aspectos relevantes en patrones (entrada-salida), con el objeto de formar un conjunto de entrenamiento que será usado por una red neuronal.

Teniendo en cuenta la estructura entrada-salida de los patrones, tomaremos una Red Backpropagation considerando que para su aprendizaje emplea un conjunto predefinido de entradas-salidas (aprendizaje supervisado) dados como ejemplo, empleando un ciclo propagación-adaptación de dos fases: aplicación del patrón de entrada y modificación de los valores internos de acuerdo al error obtenido de la comparación de la salida que produjo la red con la salida deseada [7].

Como el AG deben tener algún criterio para evaluar las alternativas de solución, que depende de las características del ambiente (escenario o estado del juego), utilizaremos la RBP con cada escenario como entrada, para obtener como salida contiene los valores de los criterios para la evaluación de alternativas en ese escenario u otro parecido. Veremos ahora un planteamiento teórico y otro práctico (ajedrez) del método propuesto.

3.1 Planteamiento Teórico.

Para poder resolver el juego con Algoritmos Genéticos y Redes Neuronales describiremos el juego en su forma estratégica. De tratarse de un juego complejo, que son los que al final son los mas importantes porque describen mejor los modelos reales, el conjunto de estrategias $\{S^k\}$ $k \in$ N es muy grande, aun cuando los resultados posibles sean pocos: ganar, perder o empatar. Entonces lo ideal es considerar solamente estrategias puras en S^k, es decir considerar las acciones inmediatas (jugadas) que puede realizar el jugador *k* en un momento determinado, de tal forma que A ya no será el conjunto de resultados posibles (finales) del juego sino que seria el conjunto de escenarios posibles de aplicar cada estrategia pura (de ser la ultima jugada, A será efectivamente el conjunto de resultados posibles).

De esta manera podemos aplicar un Algoritmo Genético considerando como generación inicial A, es decir, el conjunto de escenarios posibles de aplicar todas las estrategias puras del jugador *k* (que es el jugador que queremos que gane). De ahí las generaciones sucesivas serán los escenarios que derivan de los escenarios de la primera generación y que son consecuencia de las estrategias puras de los demás jugadores, sin importar si se juega en turnos o simultáneamente, al final el análisis es el mismo. Como el AG necesita determinar el fitness o utilidad de cada individuo (escenario), tomaremos la función de utilidad π^k : S $\rightarrow \Re$ que evaluará cada escenario de acuerdo con los aspectos relevantes al juego (estos aspectos son estrictamente relativos al juego, y deben ser determinados por quien emplee la técnica) y que determinen la utilidad para el jugador *k*.

Parecería que la forma de analizar a los individuos y las generaciones no es efectiva ya que la cantidad de individuos aumentará en cada generación de la misma forma que aumentan el numero de nodos por nivel en el desarrollo del árbol de juego (forma extensiva), pero como cada generación mantiene el mismo número de individuos siempre (sobreviven los mas útiles) la solución disminuye su complejidad algorítmicamente hablando.

Se entiende por escenario (individuo) a cualquier representación completa y única del estado del juego en un instante determinado, codificado en una cadena de caracteres, es decir una representación discreta del estado del juego. Al hacer el cruzamiento se toman un par de escenarios s_i, s_j que darán origen a dos s'_i, s'_j cuyo fitness afectará el fitness de los progenitores y que ocuparan sus posiciones en la siguiente generación. Al final debemos seleccionar una estrategia pura (más alto fitness) de acuerdo al escenario que de origen dicha estrategia en la primera generación. Este análisis se haría cada vez que nos toque "jugar".

La función de utilidad usada para calcular el fitness debe tomar en cuenta criterios relevantes al estado del juego, dichos criterios se representan en coeficientes que reflejan el porcentaje de importancia de ese criterio en ese momento del juego. Obviamente estos coeficientes varían con el tiempo lo que determina que la estrategia general del juego varía conforme el juego avanza. Por ejemplo, si él o los contrincantes se han debilitado, conviene tomar estrategias mas agresivas.

Como cada escenario es una representación discreta, lo tomamos como un patrón para que sea reconocido por una RBP, es decir podemos formar un conjunto de entrenamiento *E* donde cada escenario (patrón) *pi* esta asociado con una salida *si* (aprendizaje supervisado) que es una colección de coeficientes que representan la importancia de los criterios tomados en cuenta en la función de utilidad y que reflejen el estado del juego representado en *pi*. Con ello, una vez entrenada la red, esta será capaz de reconocer el estado del juego representado en un patrón de entrada a la red y botar los coeficientes adecuados para calcular la utilidad (fitness) de dicho escenario (individuo).

3.2 Planteamiento Práctico.

En el caso del ajedrez tenemos:

$N = \{ \text{b, n} \}$ blancas y negras.

A : conjunto de escenarios posibles.

$\{ S^k \}$ $k \in$ N: conjunto de estrategias puras.

El conjunto de estrategias puras para cada jugador esta determinado por todas los movimientos que puede realizar cada pieza de su equipo que no haya sido eliminada. En el momento inicial $S^b = S^n$ .Como necesitamos poder representar discretamente cada escenario, enumeraremos los escaques de la siguiente manera, como se puede ver en la Figura 1.

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

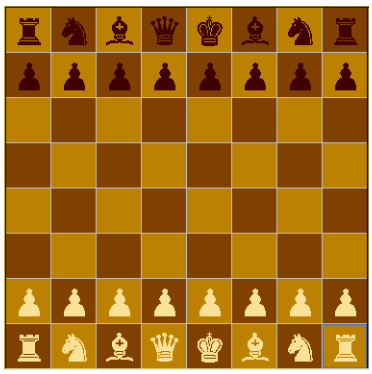

Fig. 1. Numeración de los escaques

Entonces, cada escenario estará representado por una cadena de 64 caracteres donde cada carácter tiene la posición del escaque respectivo. Así, cada alelo o gen podrá tomar los siguientes valores: p (peón) , t (torre) , c (caballo), a(alfil), rey(r), reina(q) o v si esta vacío dicho escaque. Para diferenciar las piezas de un equipo y otro, se tomaran mayúsculas para las blancas y minúsculas para las negras. Entonces en total se tomaran en cuenta de 12 alelos. En la Figura 2 se puede ver la codificación del momento inicial.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	...	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
t	c	a	r	q	a	c	t	p	p	p	p	p	p	p	p	v	v	...	V	V	P	P	P	P	P	P	P	P	T	C	A	R	Q	A	C	T

Figura. 2. Codificación del inicio de la partida

Al aplicarle una estrategia pura, por ejemplo mover el caballo blanco : Cb1 - a3, el escenario resultante se puede ver en la Figura 3.

tcarqactppppppppvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvPPPPPPPPTCARQACT

$$+ \mathrm{Cb1} - \mathrm{a3} =$$

tcarqactppppppppvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvCPPPPPPPPTCARQAvT

Figura 3. Aplicación de estrategia pura

Como ya podemos codificar los escenarios, ahora determinaremos la forma de evaluar los escenarios. En el planteamiento teórico se dijo que los criterios para evaluar el estado del juego eran relativos al juego y deberían ser determinados por quien usará la técnica. Entonces, conociendo el propósito del juego, tomare los siguientes criterios:

- Riesgo del rey contrario.
- Riesgo del rey propio.
- Riesgo de las piezas de nuestro equipo.
- Agresividad del equipo propio (esto se puede determinar contando cuantas de las posiciones del tablero pueden ser atacadas por nuestras piezas).
- Agresividad del equipo contrario.
- Movilidad de nuestras piezas. (se puede tener una idea de la movilidad contando cuantas jugadas puede realizar cada una de nuestras piezas).

Con estos criterios se puede considerar suficientemente representativa la función de utilidad, pero deben tener un coeficiente de importancia, en general, un criterio puede ser tomado con nula (N), baja (B), media (M), alta (A) o muy alta (E) importancia a la hora de evaluar el escenario, entonces tenemos 5 grados de importancia con los coeficientes de importancia siguientes:

- E = 1.00 (100%)
- A = 0.75 (75%)
- M = 0.50 (50%)
- B = 0.25 (25%)
- N = 0.01 (1%)

El *fitness* de los escenarios determina su reproducción, y luego hacemos el *crossover* de la siguiente manera : tomamos dos escenarios s_i, s_j y obtenemos la intersección I de los conjuntos de escenarios posibles que deriven de cada uno de ellos; luego evaluamos todos los escenarios de I tomado los 2 que tienen menor *fitness* (los peores para nuestro jugador k) s_i', s_j', es decir aquellos que probablemente busque el adversario. Estos 2 escenarios ocuparan la posición de sus progenitores en la siguiente generación.

Los patrones de entrada que reconocerá la RBP contienen la posición de todas las piezas del juego, pero considerando al peón como una pieza de segundo orden, consideraremos el numero de peones activos en el juego en vez de la posición de cada uno.

Tomaremos entonces 9 entradas correspondientes a las negras, 8 primeras para los alfiles, torres, caballos, reina y rey, y la novena para el numero de peones negros activos. Otras 9 entradas correspondientes a las blancas codificadas de la misma manera que las negras. Al final, cada patrón de entrada tendrá 18 valores lo que determina el número de neuronas en la capa de entrada de la red, como se puede ver en la Figura 4.

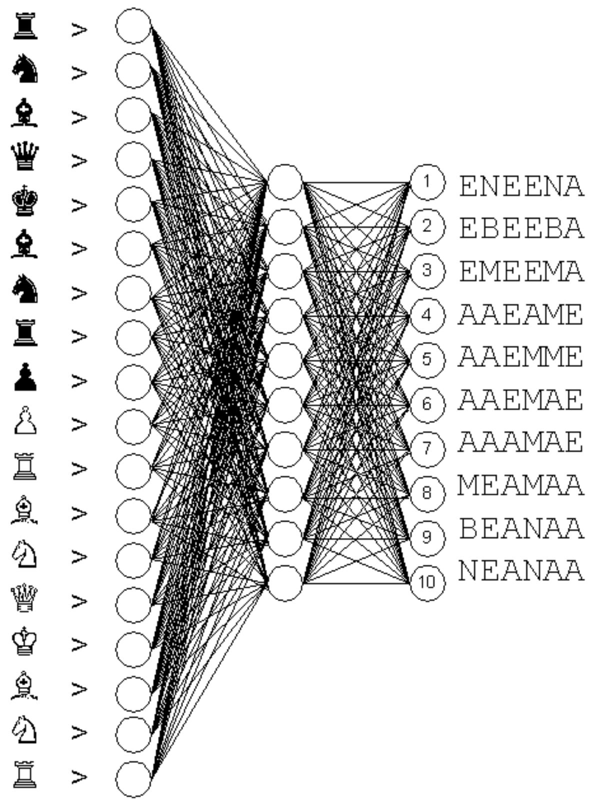

Figura 4. Arquitectura de la Red

Habiendo considerado 6 criterios de evaluación, cada uno con 5 posibles valores, entonces los aspectos relevantes del estado del juego se reflejaran en un arreglo de 6 valores que representan la importancia de cada criterio. Este arreglo es precisamente la salida buscada por la RBP. El problema ahora es que así, las posibles combinaciones de coeficientes en la salida son $5^6 = 15625$ lo que significa que cada patrón puede estar asociado a cualquiera de esas combinaciones, en consecuencia el entrenamiento difícilmente puede resultar efectivo.

Considerando que los objetivos de un jugador de ajedrez son básicamente buscar dar el jaque mate y evitar que el oponente haga lo mismo, entonces cada jugada es una estrategia pura orientada a la consecución de uno u otro objetivo, y si se pudiera, de ambos objetivos a la vez. De hecho, las mejores jugadas son las que cumplen este ultimo requisito. En general cada jugada se puede clasificar de acuerdo al grado en que contribuye a dar el jaque mate (Agresividad) o a evitar que el oponente haga lo mismo (Defensa).

Así, aunque las combinaciones de coeficientes son muchas, varias jugadas tendrán el mismo grado de agresividad y defensa entre sí, lo que nos permite tomar solo unas cuantas combinaciones de coeficientes, que representen las clasificaciones mas comunes. De esta forma, tomamos solo 10 combinaciones de salida, que van desde lo mas agresivo hasta lo mas defensivo.

Con esto, la RBP tendrá en la capa de salida 10 neuronas, activándose cada una por cada una de las 10 combinaciones tomadas, de acuerdo al estado del juego representado en el patrón presentado a la red. Figura 5.

Las letras en cada combinación representan el grado de importancia de cada criterio en el siguiente orden (de izquierda a derecha) : riesgo rey blancas, riesgo rey negras, riesgo de las piezas, agresividad negras, agresividad blancas, movilidad. El sistema juega con las negras, ya que el oponente-usuario debe iniciar la partida y las blancas siempre empiezan. Entonces hay que considerar que los coeficientes de las combinaciones mostrados en la Figura 5 reflejan la importancia de los criterios mencionados con respecto a lo que le conviene a las negras, o sea, al sistema.

Riesgo rey blancas
Riesgo rey negras
Riesgo piezas
Agresividad negras
Agresividad blancas
Movilidad

ENEENA Agresividad+
EBEEBA
EMEEMA
AAEAME
AAEMME
AAEMAE
AAAMAE
MEAMAA
BEANAA
NEANAA Defensa+

E = 1.00
A = 0.75
M = 0.50
B = 0.25
N = 0.01

Figura 5. Combinaciones de coeficientes en las salidas de la red.

La red cuenta entonces con 18 entradas, 10 salidas y una capa oculta de 10 neuronas. Para el entrenamiento se utilizó un coeficiente de aprendizaje α = 0.2. y los pesos fueron inicialmente generados de manera aleatoria. Los patrones que se utilizaron para entrenar la red se obtuvieron de situaciones particulares de juego en donde unos criterios eran eminentemente más importantes que otros, y en situaciones de equilibrio como las primeras movidas de la partida donde cada criterio tiene porcentajes de importancia similares. El funcionamiento del sistema desarrollado esta resumido en la Figura 6.

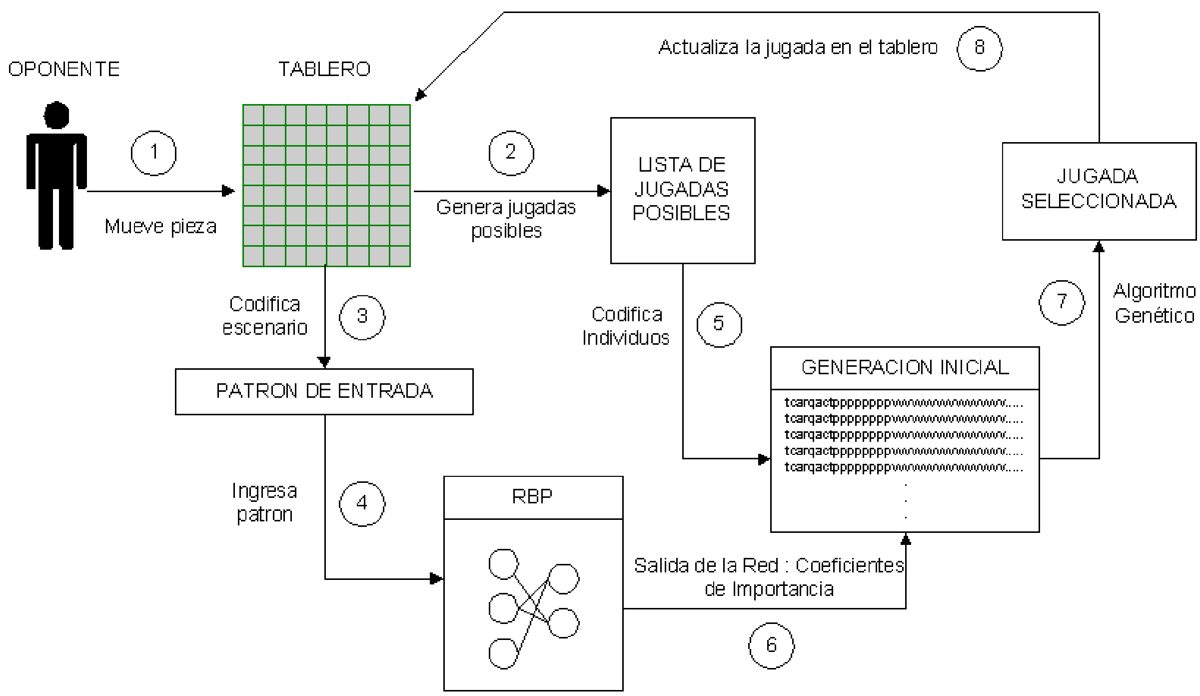

Figura 6. Funcionamiento del sistema.

4. Comparación con las técnicas tradicionales.

Las técnicas tradicionales comúnmente empleadas para la resolución de juegos de este tipo son básicamente tres, si consideramos el problema como uno en el que se busca una solución óptima en un gran dominio: enumerativos (o sea extensivos como el desarrollo del árbol de juego), aleatorios y basados en cálculo [6]. Los enumerativos, es decir extensivos, tienen una complejidad algorítmicamente hablando equivalente a una búsqueda en el árbol de juego, que tienen una complejidad del orden de $O(\sqrt{n})$ (aún después de haber efectuado una "poda" $\alpha - \beta$), donde n es el tamaño del árbol de juego completo [12].

En juegos complejos, tomando como ejemplo al ajedrez, el número de jugadas posibles promedio a lo largo de la partida es 25 – 30 jugadas por turno, es decir que en un momento cualquiera de la partida podemos considerar un árbol de juego equivalente con 30 nodos que salen de la raíz y luego cada uno tiene 30 nodos mas y así sucesivamente. Cómo el análisis se haría sobre los nodos finales (tamaño del árbol), entonces para un desarrollo de árbol hasta el nivel 5 por ejemplo, lo que nos daría una predicción aceptable, el valor de n sería el numero de nodos en el nivel 5, o sea $n = 30^5 = 24300000$ y un orden de $O(\sqrt{n}) = 4929.50$.

En contraste, el uso de un AG con 30 individuos en la generación inicial y que se mantiene constante a lo largo de las demás generaciones, tendría una complejidad que deriva del análisis de todos los individuos en la población. Para 5 generaciones (nivel 5) la población es $m = 30 * 5 = 150$ o sea que el orden finalmente sería $O(m) = 150$. Queda claro entonces que m es mucho menor que n en situaciones similares de juego, lo que significa que el método propuesto tiene una ventaja importante en lo que a tiempo de ejecución se refiere. Como es obvio, la diferencia entre ambas técnicas aumenta mientras mayor sea el número promedio de jugadas en cada turno.

Por otro lado, las técnicas aleatorias seleccionan aleatoriamente estrategias con la esperanza de encontrar una buena. No esperemos obtener buenos resultados. Finalmente los métodos basados en cálculo, requieren en todo caso, que la función que evalúe las alternativas sea una función derivable. En el caso de la función que utilicemos en el análisis del juego sea así (algo medianamente probable) tampoco se puede esperar obtener siempre resultados óptimos (máximos locales, no globales).

Este método puede ser mejorado empleando alguna otra red neuronal de aprendizaje supervisado que demuestre ser útil en el reconocimiento de patrones en el juego que se quiera resolver, y utilizando patrones en el entrenamiento que representen situaciones clásicas de dicho juego, donde los criterios a ser tomados en cuenta con mayor importancia sean ya conocidos.

Partidas realizadas por el sistema.

Partida 1:

e2 - e4, e7 - e5, Af1 - e2, Qd8 - f6, a2 - a4, Af8 - c5, Cb1 - c3, Qf6: f2.

Partida 2:

e2 - e4, e7 - e5, Af1 - e2, Qd8 - f6, Cg1 - h3, Af8 - c5, Enroque, a7 - a5, a2 - a4, Cb8 - c6, c2 - c3, Qf6 - h6, b2 - b4, a5: b4, c3: b4, Cc6: b4, Ac1 - a3, d7 - d5, Ae2 - b5, Ac8 - d7, Qd1 - g4, Enroque, Qg4 - f3, Ad7: b5, e4: e5, Ab5: f1, Rg1: f1, Cg8 - e7, Qf3 - g4, Rc8 - b8, Cb1 - c3, Cb4 - d3, Ta1 - b1, Ac5: a3, Cc3 - b5, Aa3 - c5, Rf1 - e2, Ce7: d5, Re2: d3, Rb8 - a8, a4 - a5, c7 - c6, Cb5 - a3, Ac5: a3, Tb1 - b3, Aa3 - c5, Qg4 - e4, Cd5 - c7, Rd3 - e2, Qh6 - e6, Ch3 - g5, Qe6: b3, Cg5: f7, Qb3: f7, Qe4 - a4, Qf7 - f5, a5 - a6, Qf5 - b1, a6: b7, Ra8: b7, h2 - h4, Th8 - f8, g2 - g3, Tf8 : f2.

Partida 3:

Cg1 - h3, e7 - e5, Cb1 - a3, Qd8 - h4, e2 - e4, Qh4: e4, Af1 - e2, d7 - d5, Enroque, Af8 - c5, Ae2 - b5, Re8 - d8, Ch3 - g5, Qe4 - f4, Cg5: f7, Qf4: f7, d2 - d4, e5: d4, Ac1 - g5, Cg8 - e7, Qd1 - e1, Ac8 - g4, Ta1 - d1, Ag4 : d1, Qe1: d1, Qf7 - e6, Qd1 - e1, Qe6 - f5, c2 - c4, Qf5: g5, c4: d5, Ce7: d5, Ca3 - c4, Qg5 - f5, Cc4 - e5, h7 - h5, f2 - f3, Th8 - h6, g2 - g4, h5: h4, f3: g4, Qf5 - c2, Tf1 - f8, Ac5: f8, Qe1 - f1, Qc2: h2.

Partida 4:

d2 - d4, d7 - d5, Ac1 - g5, Qd8 - d6, Qd1 - d3, h7 - h5, Cb1 - a3, e7 - e6, Enroque, Qd6 - b4, g2 - g4, h5 : g4, h2 - h3, Af8 - d6, c2 - c3, Qb4 - a4, Td1 - e1, Cb8 - c6, b2 - b3, Qa4: a3, Rc1 - d1, Qa3: a2, e2 - e4, Qa2: f2, Cg1 - e2, d5: e4, Qd3: e4, Cc6 - a5, h3: g4, Ad6 - h2, b3 - b4, Ca5 - c4, d4 - d5, e6 - e6, d5 - d6, Cc4: d6, Qe4 - c2, Ac8: g4, Qc2 - a4, Re8 - f8, Qa4 - d7, Ag4: d7, b4 - b5, Cd6 - e4, c3 - c4, Ce4: g5, Ce2 - g1, Th8 - h6, Rd1 - c1, Qf2: e1, Rc1

- c2, Qe1: f1, Rc2 - d2, Qf1: c4, Cg1 - f3, Cg5: f3, Rd2 - e3, Ad7 - f5, Re3: f3, Ta8 - d8, Th1 - f1, Qc4: f1, Rf3 - e3, Ah2 - f4.

Partida 5:

e2 - e4, e7 - e5, d2 - d4, Qd8 - f6, Ac1 - e3, Af8 - b4, Cb1 - d2, e5: d4, Ae3: d4, Qf6: d4, Qd1 - g4, Qd4: d2.

Partida 6:

d2 - d4, d7 - d5, Ac1 - g5, Qd8 - d6, Cb1 - a3, Ac8 - f5, Qd1 - d2, Cb8 - c6, Enroque, Enroque, Cg1 - f3, h7 - h5, h2 - h3, Td8 - e8, g2 - g4, h5: g4, h3: g4, Th8: h1, b2 - b3, Af5: g4, c2 - c4, Qd6: a3, Rc1 - c2, e7 - e6, Qd2 - c1, Qa3: a2. Rc2 - c3, Ag4 - f5, Td1 - d2, Af8 - b4.

Referencias

[1] Bilbao, J.M. *Introducción a la Teoría de juegos no cooperativos.* – Universidad de Sevilla, Matemática Aplicada II, 1971

[2] Binmore, K. *Teoria de Juegos* - McGraw-Hill , 1996. pp 42-43.

[3] Corcho Sánchez, P.I. y Corcho Sánchez P. *Un Curso De Teoría De Juegos.* Facultad de CC. EE. y Empresariales - Universidad de Extremadura, 1998

[4] Davis M. *Teoría del juego.* – Alianza Universidad, 1971

[5] Davis M. *Teoría del juego.* – Alianza Universidad, 1971, pp 19-20.

[6] Goldberg E.D. *Genetic Algorithms in search, Optimization, and Machine Learning,* Addison – Wesley, 1989

[7] Hilera Gómez, J.R. y Martínez, V.J. *Redes Neuronales Artificiales*. – Addison Wesley Iberoamerica – 1995

[8] Holland J. H. *Genetic Algorithms and adaptation.* Universidad de Michigan, 1981.

[9] Kuhn H.W. *Extensive Games and the problem of information*, Annals of Mathematics Studies 28, p. 193, 1953

[10] Zermelo, E. *Una aplicación de estrategias de juego y teoria de juegos* – Proceedings of the Fifth International Congress of Mathmaticians, (II) 501-504 , 1912

[11] Von Neumann, J. y Morgenstern ,O. *The Theory of Games and Economic Behavior.* – Princeton University Press, 1944

[12] Weiss M. A. *Estructuras de datos y algoritmos.* Addison – Wesley Iberoamerica, 1995. pp 432