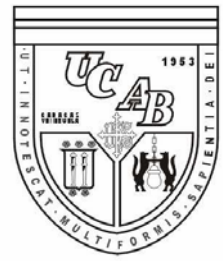


UNIVERSIDAD CATÓLICA ANDRÉS BELLO

FACULTAD DE INGENIERÍA

ESCUELA DE INGENIERÍA INFORMÁTICA

Desarrollo de un jugador artificial de GO basado en redes neurales evolutivas

TRABAJO ESPECIAL DE GRADO

presentado ante la

UNIVERSIDAD CATÓLICA ANDRÉS BELLO

como parte de los requisitos para optar al título de

INGENIERO EN INFORMÁTICA

REALIZADO POR: Luján Toro, Alejandro.

PROFESOR GUIA: Pereira, Wilmer.

FECHA: Octubre de 2004.

Dedicatoria

A mis padres, por impulsar constantemente mi desarrollo personal y profesional. Les debo todo lo que he logrado, lo que estoy logrando y lo que lograré.

A Dani, porque junto a ti y gracias a ti estoy encontrando el camino a la sabiduría infinita. Hoy es sólo el comienzo de una larga vida juntos de retos y victorias.

Agradecimientos

A Alex Lubberts, cuya investigación fue parte importante de la inspiración para este trabajo, y quien ofreció un valioso aporte en las diferentes fases del proyecto.

A Alexis Hernández por ofrecer sus valiosísimos conocimientos en el ámbito de GO y apoyar de forma entusiasta el proyecto desde su inicio.

A Carola, por haberme apoyado en los momentos mas complicados de este trabajo, y de mi carrera. Por siempre estar ahí, gracias infinitas.

A mis compañeros de estudio y de trabajo, especialmente a Carlos, Dennis, Guillermo y Hernán, quienes han sabido apoyarme y compartir mis esfuerzos cuando lo he necesitado.

Índice de contenidos

Dedicatoria i

Agradecimientos ii

Índice de contenidos iii

Índice de tablas vii

Índice de figuras viii

Resumen ix

Capitulo I 1

I.1 Introducción 1

I.2 Planteamiento del problema 2

I.3 Objetivos 3

Objetivo general 3

Objetivos específicos 3

I.4 Limitaciones y alcances 4

Capitulo II: Marco teórico 5

II.1 El juego de GO 5

Reglas de juego 6

Suicidio 6

Regla de KO 7

Fases del GO 7

II.2 Redes Neurales 9

II.3 ¿Por qué redes neurales para GO? 11
II.4 Algoritmos Evolutivos 12
II.5 SANE 14
Evolución Simbiótica 14
SANE: Poblaciones paralelas 15
Coevolución competitiva 16
Fitness Sharing 18
Shared Sampling 19
Hall of Fame 20
Algoritmo evolutivo definitivo 22
Capitulo III: Metodología 23
Capitulo IV: Desarrollo 24
IV.1 Investigación 24
IV.2 Decisiones de diseño 24
Conexiones 24
Estructura a tres segmentos 26
IV.3 Diseño 27
IV.4 Implementación 29
Paquete CoCoSane: 29
Paquete GoCoCoSane 29
Paquete NeoGo 30
Paquete GO 30

Detalles de implementación 30
Pesos de las conexiones 30
Selección de jugada: mejor jugada válida 30
Regla de KO 31
Sistema de Puntuación 31
Configuración dinámica 32
Protocolo de comunicación GTP 33
IV.5 Corridas 33
IV.6 Pruebas 35
Capitulo V: Resultados 36
V.1 Tablero 5x5 36
CoCo_100 37
CoCo_200 38
Neo_343434 39
Neo_206020 40
Neo_666666 41
Neo_4012040 42
V.2 Tablero 9x9 43
V.3 Estrategias desarrolladas 43
Ojos 43
Separación de grupos 43
Captura 44

Capitulo VI: Conclusiones y Recomendaciones 45
VI.1 Conclusiones 45
Tablero 9x9 45
Tablero 5x5 45
Cantidad de neuronas 45
Estructura a tres segmentos 45
Framework SANE 46
VI.2 Recomendaciones 46
Paralelización 46
Estructura a tres segmentos 46
Particularización 47
Bibliografía 49
Apéndice 1: Glosario de términos de GO 51
Apéndice 2: Juegos 54
Juego #1 55
Juego #2 56
Juego #3 57

Índice de tablas

Tabla 4.1: Configuraciones de corridas evolutivas..... 34

Índice de figuras

Figura 2.1: Suicidio. 6

Figura 2.2: KO 7

Figura 2.3: Red neural “feed forward” 9

Figura 2.4: Ojos 11

Figura 4.1: Evolución de redes con numero fijo de conexiones. ... 25

Figura 4.2: Evolución de redes completamente conexas. 25

Figura 4.3 Resultados de corrida evolutiva utilizando puntuación de GnuGo. 32

Figura 4.4 Resultados de corrida evolutiva utilizando cálculo interno de puntuación. 32

Figura A1.1: Escalera 51

Figura A1.2: Ojos. 53

Resumen

En este trabajo especial de grado se exploraron una serie de técnicas del área de las redes neurales evolutivas, utilizando como experimento la implementación de un jugador artificial de GO. Las técnicas exploradas fueron las siguientes:

- Algoritmo Evolutivo SANE
- Coevolución
- Shared Sampling
- Fitness Sharing
- Hall of Fame

Durante el trabajo se siguieron las etapas de recolección de requisitos, investigación, diseño, implementación del prototipo y evaluación. Se utilizaron dos diseños de redes: la tradicional feed forward de una capa oculta, y una estructura a tres segmentos diseñada específicamente para el juego de GO. Para la fase de evaluación se utilizo a GNUGO como jugador externo.

Los resultados muestran un aumento en el nivel de juego en el tablero 5x5 a lo largo de la evolución, sin obtener mejoras significativas al aumentar el número neuronas o al utilizar la estructura a tres segmentos.

Finalmente, este trabajo especial de grado propone una implementación del mecanismo Coevolutivo Competitivo SANE bajo un diseño orientado a objetos fácilmente extensible para otros dominios.

Capitulo I

I.1 Introducción

Una de las técnicas mas importantes del área de la Inteligencia Artificial es el uso de las redes neurales artificiales. Su buen desempeño en el análisis de patrones y su rapidez de procesamiento las convierte en estructuras sumamente interesantes para una variedad de tareas, y en particular en el estudio de juegos de tablero como el GO. Al combinar estas dos capacidades con técnicas de aprendizaje artificial, se vislumbra la posibilidad de encontrar un mecanismo que provea el motor de un jugador de GO de cierto nivel. Mas aún, la validez de las técnicas estudiadas podría extrapolarse a otros dominios, lo que multiplicaría el valor de un posible resultado.

El propósito de este trabajo especial de grado es evaluar un conjunto de técnicas relacionadas con las redes neurales artificiales evolutivas, entre las cuales tenemos:

- Algoritmo Evolutivo SANE
- Coevolución
- Shared Sampling
- Fitness Sharing
- Hall of Fame

Adicionalmente, se experimentará particularizando la topología de las redes neurales para evaluar el beneficio que pueda traer esto en términos de velocidad de aprendizaje.

Las redes neurales han sido de gran utilidad en áreas tan disímiles como visión artificial, reconocimiento de voz, diagnósticos médicos, traducción de textos, data mining, etc. Por otro lado, los algoritmos evolutivos también forman parte importante del área de la Inteligencia Artificial, y plantean metas del aprendizaje artificial que poblarían las páginas de libros de ciencia ficción. Participar en la investigación, y colaborar en el desarrollo de estos temas puede significar un aporte muy significativo para la comunidad investigativa, aporte que podría eventualmente propagarse a la industria.

Hasta la fecha se han realizado una serie de trabajos en el campo de las redes neurales evolutivas, entre los que se pueden citar a [ROSI97]. En particular, aplicados a GO, vale la pena resaltar las siguientes investigaciones: [LUMI01], [RIMO98], [RUNO03]. Aunque el nivel de los jugadores de GO obtenidos en estos trabajos sigue sin alcanzar al de un profesional, la mejora en este nivel da esperanzas acerca del desempeño de las redes neurales evolutivas como técnica utilizada en el desarrollo del motor del juego.

I.2 Planteamiento del problema

El juego de GO, originario de Asia, es uno de los juegos de mesa mas antiguos de los que se tiene referencia. Sus reglas son muy sencillas, pero las estrategias para ganar pueden ser muy elaboradas, lo que lo hace muy interesante para el campo de los jugadores artificiales: programas que jueguen contra oponentes humanos o inclusive contra otros programas. Algunas de sus características, como el tamaño del tablero, lo diferencian significativamente de otros juegos de mesa como el Ajedrez. En el

caso de este último, se han desarrollado programas de muy buen desempeño, logrando inclusive a vencer al campeón mundial, como es el caso de Deep Blue de IBM [DEEP]. Sin embargo, algunas de las técnicas utilizadas para programar jugadores de Ajedrez serían completamente inadecuadas para un jugador artificial de GO, lo que presenta el reto de buscar técnicas mas eficientes.

I.3 Objetivos

Objetivo general

Desarrollar un programa que sea capaz de jugar GO, utilizando las técnicas de Redes Neurales Artificiales y Computación Evolutiva.

Objetivos específicos

1. Diseñar un prototipo de redes neurales artificiales que, dada una configuración del tablero de GO, determine la próxima jugada a realizar;
2. Determinar un mecanismo evolutivo que permita obtener redes neurales artificiales de mejor desempeño que el prototipo;
3. Implementar el mecanismo mencionado anteriormente con el fin de obtener redes neurales artificiales capaces de efectuar una jugada dada una configuración del tablero;
4. Enfrentar al programa resultado con otros programas jugadores de GO, como GNU GO y Jimmy-5.0, para evaluar su desempeño y compararlo con el de otros trabajos semejantes.

5. Analizar los resultados de las pruebas mencionadas en el inciso anterior.

I.4 Limitaciones y alcances

Por lo general, la arquitectura de un juego de computador contiene al menos dos elementos principales: el motor y la interfaz. El motor se encarga de mantener el universo virtual necesario para representar el ambiente del juego, mientras que la interfaz presenta los elementos de dicho universo al usuario. Este Trabajo Especial de Grado se encargará de desarrollar el motor del juego, y se utilizará un framework de dominio público para la interfaz.

Por otro lado, dada la incertidumbre que se cierne en el manejo de las redes neurales artificiales y la escasez de trabajos en ésta área, los objetivos no incluyen el logro de un jugador artificial de mayor nivel que los existentes, sino explorar que tan adecuadas son las técnicas de Inteligencia Artificial mencionadas en el área de los jugadores artificiales. Evidentemente lograr un jugador que supere a los existentes sería un gran beneficio, pero no es el indicador determinante de éxito en éste trabajo particular.

Capitulo II: Marco teórico

II.1 El juego de GO

El juego de GO es un juego de mesa de reglas muy sencillas, que se desarrolla en un tablero cuadriculado, típicamente de 9x9 o 19x19 intersecciones. Los jugadores, que juegan uno con el color blanco y el otro con el color negro, toman turnos alternos en los que pueden colocar piezas en las intersecciones libres del tablero.

Se define un ***grupo***[1] como dos o mas piedras adyacentes. Un grupo tiene tantas ***libertades*** como intersecciones adyacentes libres. Cuando un jugador ocupa la ultima libertad de un grupo enemigo, ese grupo es ***capturado*** y eliminado del tablero.

En su turno, un jugador puede decidir pasar en lugar de colocar una piedra, y el juego termina cuando los dos jugadores pasan de forma consecutiva.

El ***territorio*** de un jugador se define como las intersecciones rodeadas únicamente de piedras de su color. Al final del juego se realiza un conteo de territorio de cada jugador, al que se substrae las piedras capturadas por el enemigo. El objetivo del juego es obtener la mayor cantidad de territorio posible.

En principio, las reglas hacen parecer que GO es un juego sencillo, pero quien se aventure a jugar unos pocos juegos descubrirá pronto que desarrollar estrategias eficientes es una tarea muy

[1] Los términos específicos que se introduzcan por vez primera estarán resaltados en negritas y cursivas, para indicar que hay una entrada correspondiente en el glosario.

compleja. Para ilustrar cuan profundo es el juego, en países como Japón, China y Corea hay ligas profesionales de jugadores de GO, y escuelas de jóvenes que dedican su vida al GO como una carrera.

Reglas de juego

Durante su turno, un jugador puede realizar una jugada válida o pasar. Una jugada válida cumple con las siguientes condiciones:

- La intersección en la que se desea jugar esta vacía
- No es una jugada suicida
- No viola la regla de KO

Suicidio

Se define una jugada suicida como aquella que resulte en un grupo del propio color sin libertades. Un ejemplo elemental es una piedra colocada en una intersección rodeada de piedras del oponente. Esta movida eliminaría automáticamente la piedra recién jugada. Como excepción, una jugada no es suicida si resulta en la captura de un grupo del oponente. La figura 2.1 ilustra ambas situaciones:

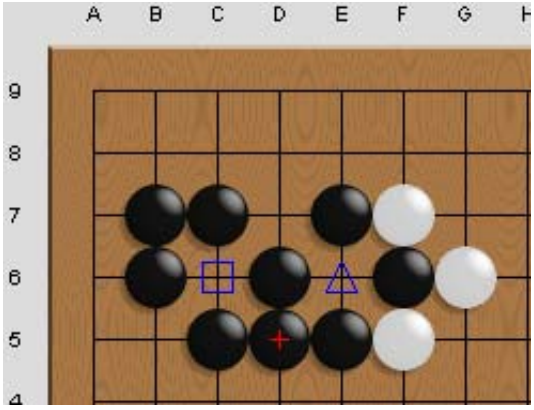


Figura 2.1: Suicidio.
Blanco no puede cometer suicidio en C6 porque esa intersección no tiene libertades. Sin embargo, la jugada E6 esta permitida porque implica la captura de la piedra negra F6, por lo que no es considerada una jugada suicida.
(Ilustración elaborada con [JAGO])

Regla de KO

La regla de KO establece que no se debe repetir la configuración del tablero a lo largo del juego. Esta condición no es demasiado común, pero su cumplimiento evita la posibilidad de repetir un ciclo de jugadas indefinidamente (condición especialmente importante cuando están jugando programas en lugar de personas).

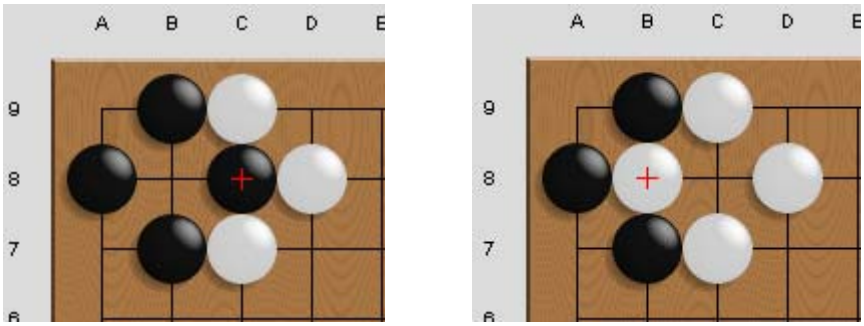


Figura 2.2: KO. *Negro C8 está amenazado (izquierda), y blanco decide capturarlo jugando en B8 (derecha). Negro podría pensar en jugar en C8 de nuevo para capturar a blanco B8, pero eso devolvería al tablero al estado de la figura A, lo que violaría la regla de KO. Por lo tanto, negro está obligado a elegir otra jugada.* (Ilustración elaborada con [JAGO])

Fases del GO

El juego de GO se puede dividir en tres fases que se desarrollan en el siguiente orden:

- **Fuseki**: es la primera fase del juego, y consiste principalmente en esbozar en el tablero zonas que los jugadores reclaman como propias. El juego tiende a ser mas ***elástico*** y el contacto entre grupos es mínimo.

- **Medio Juego**: es la fase usualmente mas larga del juego, en la que los jugadores establecen batallas de vida o muerte por las zonas del tablero que reclamaron en la fase de fuseki. El juego es mucho menos elástico y se caracteriza por un mayor contacto entre grupos, ataques y defensas mas frontales.
- **Yose**: es la fase terminal del juego, en la que se cierran las últimas zonas y se termina de demarcar de forma definitiva las fronteras entre los territorios de cada jugador.

La elasticidad de una jugada se define como la proximidad que tenga esa jugada a los grupos del mismo jugador. Una jugada muy elástica está muy alejada del resto de los grupos del mismo jugador, y una muy poco elástica está muy cercana a otros grupos del mismo color. Evidentemente, una jugada muy elástica es muy arriesgada pero avanza rápidamente en la conquista por el tablero, mientras que una jugada muy poco elástica es mas segura pero avanza mas lentamente.

Las tres fases del juego son sumamente importantes: esbozar muy cobardemente territorios en el tablero reclama poco territorio, pero hacerlo de forma muy osada puede resultar en zonas débiles. De forma semejante, cerrar los últimos huecos en los territorios de forma incorrecta puede definir un juego muy parejo. En la fase de medio juego es donde normalmente se define una mayor cantidad de puntos, y las habilidades de batalla son cruciales para el éxito.

Es importante destacar la diferencia en el estilo de juego en cada fase en términos de elasticidad y agresividad. Esta diferencia se tomará en cuenta en el diseño de las redes neurales como parte del experimento.

II.2 Redes Neurales

Una red neural es una estructura con fundamentos matemáticos que permite procesar datos, apoyada en la construcción de unidades de procesamiento llamadas neuronas, que se relacionan con sus neuronas vecinas por medio de conexiones que simulan las sinapsis en los cerebros de los animales. La figura 2.3 ilustra el funcionamiento de una red neural:

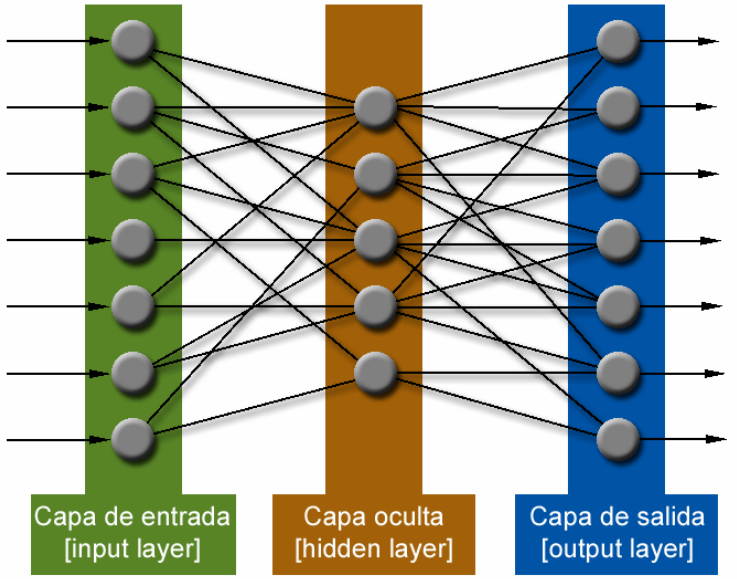


Figura 2.3: Red neural "feed forward".
(Fuente: elaboración propia)

Las redes neurales contienen una capa de entrada, por donde se reciben los datos que se van a procesar. De forma semejante, presentan una capa de salida, que contendrá los datos de salida ya procesados.

Adicionalmente, las redes neurales contienen una o mas capas ocultas, que son las que realmente realizan el procesamiento de los datos, típicamente evaluando una sumatoria ponderada de los datos de entrada en una función matemática. Ejemplos de funciones utilizadas son: sigma, sigmoide y escalón.

Las redes neurales presentan una serie de propiedades que pueden ser explotadas de forma ventajosa para ciertos tipos de problemas:

- Son **adaptativas**, lo que les permite "aprender", es decir, obtener resultados gradualmente mejores basadas en los datos con los que se les alimenta. Usualmente esto se logra con una fase de entrenamiento y con una serie de algoritmos de aprendizaje de complejidad variable.
- Son **paralelizables**: si el sistema sobre el cual laboran lo permite, la paralelización de la carga de procesamiento es fácilmente alcanzable, lo que evidentemente puede contribuir con la velocidad de procesamiento.
- No requieren de una definición **algorítmica** de la solución. Esta es una de sus mayores ventajas, ya que teóricamente permite enfrentar problemas no solucionables por medio de métodos algorítmicos, e inclusive encontrar soluciones que puedan estar mas allá de la capacidad de quien crea la red.

En el caso particular de este trabajo, una red se puede alimentar con la información del tablero en un turno determinado, esperando de su salida la recomendación acerca de cual jugada realizar.

La materia de Redes Neurales Artificiales es sumamente extensa, y abarca desde la definición matemática hasta aplicaciones reales de las redes neurales, pasando por una variedad de modelos y propuestas de implementación. Los siguientes textos ofrecen desarrollos de este tema en detalle: [BISH][HAYKIN][BRMO].

II.3 ¿Por qué redes neurales para GO?

El desarrollo de un jugador artificial de GO obliga a explorar la forma en la que el humano aprende a jugar. Las heurísticas que desarrolla el ser humano al jugar son sumamente complejas, y muchos las han estudiado e implementado, con mayor o menor éxito (Ver [GNUGO], [SMAGO]). Una de las observaciones realizadas es que el humano analiza el tablero con un particular enfoque en los patrones que allí se forman. En esta línea, es muy común encontrar términos como "una buena (o mala) formación", "una figura muy sólida", "una figura defensiva (u ofensiva)", etc. Dadas las excelentes capacidades que han demostrado las redes neurales para análisis de patrones, es obvia la tendencia a investigar la posibilidad de utilizarlas para el análisis de tableros en GO.

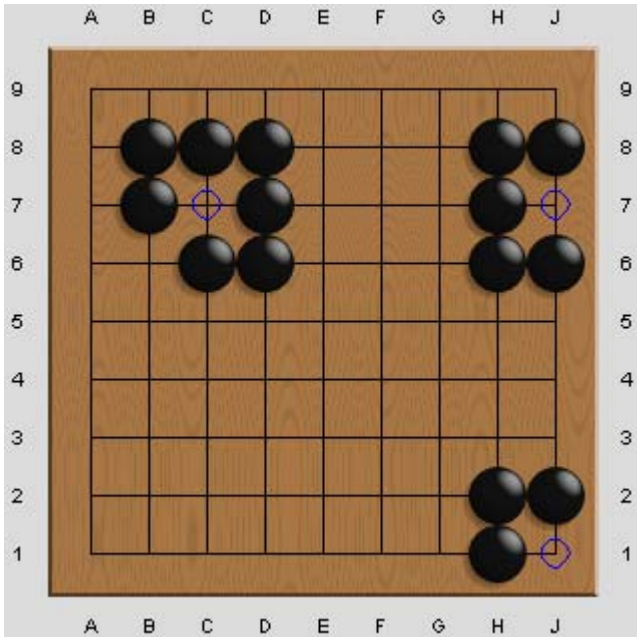


Figura 2.4: Ojos. *Un patrón importantísimo es el "ojo" (marcados con círculos). Éstos son vitales para la supervivencia de grupos pues es imposible que el oponente capture un grupo con dos ojos. La capacidad de formar y detectar ojos es, por lo tanto, crucial para un jugador. Otros patrones se pueden observar en los juegos, algunos de importancia ofensiva y otros con intención defensiva.* (Ilustración elaborada con [JAGO])

II.4 Algoritmos Evolutivos

Los algoritmos evolutivos son técnicas de búsqueda de soluciones basadas en la teoría Darwinista de la evolución natural. Estos algoritmos parten de un conjunto inicial de soluciones, e iteran realizando una serie de pasos, entre los que se encuentran evaluación, cruce y mutación. Cada iteración es denominada una generación, y la idea principal es que cada generación recombine las mejores soluciones para obtener un nuevo conjunto de soluciones potencialmente mejores, que sustituyen a las menos aptas de la generación actual, de forma de elevar el nivel global de desempeño de la población. El objetivo final es que, luego de un número finito de generaciones, el algoritmo obtenga un conjunto de soluciones de mejor desempeño que las originales, en búsqueda finalmente de la solución óptima.

En breve, los pasos de un algoritmo evolutivo tienen los siguientes objetivos[2]:

- **Evaluación**: determina el desempeño (fitness) de un individuo, bien por la evaluación de una función matemática o la aplicación de un experimento (en el caso de éste trabajo, el desarrollo de un juego de GO). En cualquiera de los casos, el resultado de la evaluación debe ser un valor que se pueda comparar con el de los demás individuos, manteniendo una relación de orden (mayor, igual o menor) con estos. Típicamente la evaluación resulta en un número real.

[2] El área de los algoritmos evolutivos es sumamente extensa. Se recomienda consultar [XXX]

- **Cruce**: una vez que se han seleccionado los mejores individuos de la población, se procede a cruzarlos para obtener nuevos individuos. Este cruce puede tener varias formas, pero se apoya siempre en el concepto del gen del individuo como elemento que almacena su información, y en la recombinación de ésta información genética para obtener nuevos individuos.
- **Mutación**: este paso permite modificar de forma aleatoria algunos individuos resultantes del cruce con el fin de explorar soluciones diferentes a las planteadas en la población. La mutación tiene como objetivo evitar la convergencia hacia valores subóptimos mediante la exploración de nuevas soluciones.

En este trabajo, cada red neural será un individuo de la población (por lo tanto se hablará de redes neurales evolutivas), y se utilizará un algoritmo evolutivo para perseguir la optimalidad en el desempeño de las redes. Adicionalmente, las neuronas serán parte de una población paralela, idea que se desarrollará en la próxima sección.

La selección de la función de evaluación puede ser en muchos casos mucho menos que trivial. Volviendo al experimento de éste trabajo, ¿qué función de evaluación puede determinar el nivel de juego de un individuo? ¿cómo determinar que una jugada es mejor que otra? Dado que las consecuencias de una jugada pueden evidenciarse luego de muchos turnos, se plantea un problema difícil al efectuar tal evaluación. Un enfoque viable es no evaluar el

desempeño de una red basados una sola jugada sino en un juego completo. De esta forma, se puede enfrentar la red a un contrincante, y siguiendo las reglas del juego se determina quién gana, obteniendo una medida clara del desempeño de la red.

II.5 SANE

SANE (Symbiotic Adaptive Neuro-Evolution[3]) es, en palabras de sus autores:

> "…un método novedoso de aprendizaje... que evoluciona una población de neuronas a través de algoritmos genéticos para formar una red neural capaz de ejecutar una tarea." [MOMI94]

La segunda versión de SANE maneja las redes como una población paralela a la de neuronas. La primera publicación que define SANE es [MOMI94], pero [MOMI98] ofrece mas detalles y algunas mejoras. En este trabajo se utilizará SANE como algoritmo evolutivo, complementándolo con las técnicas de **coevolución competitiva, fitness sharing, shared sampling y hall of fame,** técnicas que se describirán en breve.

Evolución Simbiótica

En los algoritmos evolutivos tradicionales, un individuo de la población es evaluado por medio de una función F, de forma que el individuo representa una solución a esa función. Este enfoque coloca al individuo en una posición egoísta, donde su desempeño depende en poco o ningún grado del desempeño de sus

[3] Neuro-Evolución Adaptativa Simbiótica

compañeros de población. Sin embargo, una serie de problemas plantean la posibilidad (y en algunos casos, la necesidad) de definir individuos cuya evaluación dependa de otros, es decir, donde el individuo represente una solución parcial a la función F. Tal es el caso de las redes neurales, donde se pueden plantear neuronas como individuos de una población, con la consecuencia de que la evaluación de una neurona depende del desempeño de la red completa. Este es el concepto que rige la evolución simbiótica: individuos que necesitan del desempeño de sus compañeros para su propia supervivencia [MOMI98]. El objetivo de los individuos es formar parte de una solución, y combinarse de forma efectiva con otras soluciones parciales.

SANE: Poblaciones paralelas

La propuesta inicial de SANE forma redes tomando individuos de la población de neuronas de forma aleatoria. Sin embargo, la forma en que se combinen las neuronas es evidentemente decisiva en el desempeño de estas, y una segunda versión de SANE maneja dos poblaciones de forma paralela: una población de neuronas (PN) y una de redes (PR). Cada individuo de la población de neuronas define un conjunto de conexiones ponderadas con otras neuronas dentro de una red. Por su lado, la población de redes comprende individuos que se construyen combinando individuos de la población de neuronas, y el desempeño de la red será mayor mientras mejor se combinen las neuronas. Ambas poblaciones se evolucionan de forma paralela, con el fin de obtener no sólo neuronas mas capaces, sino combinaciones mas eficientes de esas neuronas.

Defínase PR como la población de redes, y PN como la población de neuronas. El algoritmo de SANE, como se define en [MOMI98], está conformado por los siguientes pasos:

1. Limpiar fitness de cada neurona y red de las poblaciones.
2. Seleccionar las N neuronas de PN a las que haga referencia una red de PR.
3. Crear una red neural a partir de las neuronas seleccionadas.
4. Evaluar la red en la tarea de rigor.
5. Asignar al fitness de la red el valor resultado de la evaluación.
6. Repetir los pasos 2 al 4 para cada individuo en la población PR.
7. Asignar al fitness de cada neurona el fitness de las 5 mejores redes en las cuales participó.
8. Realizar las operaciones de cruce y mutación en ambas poblaciones

A continuación se mencionarán las técnicas que se utilizarán en este trabajo, y finalmente se verá en qué fases del algoritmo se aplicará cada una.

Coevolución competitiva

Al definir el mecanismo de evaluación, surge el siguiente problema: ¿quién (o qué) es el contrincante? A primera vista surgen dos posibilidades: enfrentar a la red contra un humano o contra otro jugador artificial. La primera tiene un inconveniente obvio: la evolución de una población requiere miles o cientos de miles de evaluaciones[4]; la cantidad de horas hombre que se necesitarían para ejecutar esa cantidad de juegos descarta completamente esta

[4] Esta cifra evidentemente varia mucho según el problema y el tamaño de las poblaciones, pero pueden fácilmente alcanzar números de esta magnitud.

posibilidad. Es necesario evaluar entonces las ventajas y desventajas de la segunda opción.

Muchos trabajos se han llevado a cabo efectuando evaluaciones contra otros sistemas capaces de realizar la tarea en cuestión. En principio esta estrategia puede presentar las facilidades de rapidez y costo, y en ocasiones ha sido de gran utilidad. Sin embargo, se han resaltado algunas deficiencias de este enfoque. En particular, [RIMO98] mencionan que:

> "Un problema peculiar… es que la estrategia desarrollada usualmente explota debilidades encontradas en el oponente particular, en lugar de representar buenas habilidades de GO generales."

El hecho de que el jugador se adapte al estilo de juego de su oponente es un riesgo muy grande. En lugar de obtener un buen jugador de GO general, se obtiene un excelente contrincante para un oponente en particular.

Por otro lado, además de adaptarse el estilo de juego, se ha observado la tendencia a que el proceso de aprendizaje se estanque una vez que se alcanza el nivel de juego del oponente. Lubberts lo menciona claramente en su trabajo:

> "Las redes evolucionadas contra GNUGO están limitadas por el nivel de juego del jugador existente, mientras que las redes coevolucionadas continúan mejorando".

Esta limitante es sumamente importante, ya que si el objetivo final es obtener un jugador artificial de nivel elevado, lo ideal sería que pudiese aprender indefinidamente, hasta lograr un nivel de juego

óptimo. Por otro lado, ¿cual es el punto de enseñar a una red a jugar si no va a superar el nivel de sus contrincantes?

La **Coevolución** está definida como la evolución paralela de dos poblaciones. Este concepto toma fuerza cuando se conjuga con la posibilidad de evaluar a los individuos de forma competitiva, como es el caso de la mayoría de los juegos. La Coevolución Competitiva plantea la evolución de dos poblaciones que sirven cada una como banco de ejemplos para la otra. De esta forma, en cada generación se siguen los siguientes pasos con las poblaciones P1 y P2 en las fases de evaluación:

- Al momento de evaluar a P1, ésta toma el papel de host y P2 el de parásito. Cada individuo de P1 se evalúa con un subconjunto de P2.
- En la fase de evaluación de P2, ésta toma el papel de host, mientras que P1 toma el papel de parasito. Cada individuo de P2 se evalúa con un subconjunto de P1.

Dadas las condiciones adecuadas, las dos poblaciones elevarán su nivel a cada generación, lo que le ofrece a cada población un conjunto de ejemplos aproximadamente de su nivel. Este esquema ofrece eliminar la limitante que impone la evaluación contra un elemento externo en lo que se refiere al nivel de éste; se vislumbra la posibilidad de que las dos poblaciones continúen elevando su nivel indefinidamente.

Fitness Sharing

Tradicionalmente el desempeño de un individuo se calcula como la suma de las evaluaciones que se le realizan. Sin embargo, este

método tiene una particularidad: supóngase que hay un parásito P sumamente fuerte, que logró derrotar a todos los hosts menos al individuo H. Supóngase también que H sólo logró derrotar a P y perdió contra el resto de los parásitos. Si H tiene un número de victorias menor que el promedio de su población, es muy probable que obtenga un valor de fitness bajo, disminuyendo su probabilidad de sobrevivir. Es fácil observar que la información genética de H es valiosa, ya que logró derrotar a un parásito dominante, y puede ser conveniente preservar a H. Fitness Sharing busca premiar a los individuos atípicos como H, planteando una función de fitness diferente, definida de la siguiente forma [ROSI95]:

$$F_i = \sum_{1}^{O} \frac{1}{N_j}$$

donde F_i es el fitness del individuo i, *O* el conjunto de oponentes derrotados por i y N_j es la cantidad de derrotas que ha sufrido el oponente j. En el ejemplo mencionado anteriormente, H recibirá un valor alto de fitness por haber derrotado a P. La idea es que la habilidad de derrotar a P se propague por la población en las generaciones siguientes [LUMI01].

Shared Sampling

Dado un esquema de coevolución competitiva, evaluar a una población contra todos los individuos de la población parásito puede representar una carga de procesamiento fuerte. Es posible minimizar esta carga seleccionando una muestra de la población parásita. La selección de esta muestra podría obtenerse de forma aleatoria, pero dado que existe mayor información de los

individuos, puede realizarse una selección mas informada. En particular, el fitness y el rastreo de las victorias de los individuos pueden ser utilizados para este propósito. Una selección cuidadosa de la muestra puede facilitar que la muestra sea diversa y representativa. Shared sampling propone un método para seleccionar la muestra similar al usado con shared fitness para calcular los fitness. El algoritmo es el siguiente:

```
1.  Para cada oponente O de la generación anterior:
2.  O.beatBy = 0
3.  Para i=0 hasta N (N=tamaño deseado de muestra):
4.          Para cada parasito P que no este en la muestra:
5.                  P.sample_fitness=0
6.                  Para cada oponente O derrotado por P:
7.                          P.sample_fitness+=1/(1+O.beatBy)
8.          P2 = oponente con máximo fitness
9.          Agregar P2 a la muestra
10.         Para cada oponente O de la generación anterior:
11.                 Si P2 derroto a O entonces O.beatBy++
```

(Traducción del algoritmo presentado en [ROSI95])

De esta forma se persigue reducir el tamaño de la muestra, sin perder la calidad de la población parasito.

Hall of Fame

Un posible problema de los algoritmos evolutivos es la falta de memoria a largo plazo: las habilidades obtenidas en una generación pueden diluirse a largo plazo si los contrincantes no explotan esas

habilidades, ya que los individuos de las generaciones actuales no necesitan de esa habilidad para sobrevivir. Un mecanismo que busca combatir este problema es el uso de **Hall of Fame**, que consiste en almacenar, a lo largo de las generaciones, un salón de la fama con individuos de generaciones anteriores que han sido altamente exitosos. De este salón de la fama se extraerán individuos que fungirán como contrincantes en la fase de evaluación. Si las habilidades características de las generaciones anteriores están representadas en estos individuos, y estos forman parte de los casos de evaluación, los individuos de la generación actual necesitarán derrotarlos para obtener un buen valor de fitness. De esta forma, se busca perpetuar las habilidades que se obtuvieron en las generaciones anteriores.

Algoritmo evolutivo definitivo

Luego de definir las técnicas a utilizar, se puede ver en qué lugar encajan dentro del algoritmo evolutivo, para así obtener los pasos definitivos. Defínase PR como la población de redes, y PN como la población de neuronas, el algoritmo sería el siguiente:

1. Limpiar fitness de cada neurona y red de las poblaciones.
2. Seleccionar una red R de la población PR.
3. Seleccionar N oponentes de la población parásito, utilizando **shared sampling**
4. Seleccionar M oponentes del **hall of fame**
5. Evaluar la red contra los (N+M) oponentes seleccionados en los pasos 3 y 4.
6. Asignar al fitness de la red R el valor resultado de la evaluación, utilizando **fitness sharing**
7. Repetir los pasos 2 al 6 para cada individuo en la población PR.
8. Agregar el mejor individuo de la población PR al **hall of fame,** si no existe ya.
9. Asignar al fitness de cada neurona el valor de la evaluación de las 5 mejores redes en las cuales participó.
10. Realizar las operaciones de **cruce** y **mutación** en ambas poblaciones (PN y PR).

(Algoritmo extraído del código fuente de Lubberts, A. con modificaciones menores propias)

Capitulo III: Metodología

La metodología a seguir para el desarrollo del Trabajo Especial de Grado fue una adaptación del modelo de construcción de prototipos [PRESS], que contempla los siguientes pasos:

- **Recolección de requisitos**
- **Investigación y Análisis**
- **Diseño**
- **Construcción de prototipo**
- **Adaptación**
- **Evaluación del prototipo**

Como se puede observar, se agregó una fase de Investigación y Análisis, debido a que el nivel de conocimiento en el área no es suficientemente completo como para realizar un diseño acertado únicamente con el insumo de los requisitos. La fase de adaptación surge naturalmente como la separación del desarrollo del motor del juego y de la interfaz. Para esta última se utilizara un programa de dominio público: JaGo.

Los resultados de este trabajo se publicarán para uso público, de forma que otros trabajos puedan utilizarlos, posiblemente para realizar nuevas iteraciones en las que se construyan otros prototipos.

Capitulo IV: Desarrollo

IV.1 Investigación

La primera fase del trabajo consistió en investigar los temas relacionados y buscar trabajos anteriores similares. Por los resultados positivos que se han reportado utilizando SANE se decidió tomar éste como algoritmo evolutivo. En relación a éste algoritmo se encontraron investigaciones diversas, entre las cuales se puede destacar a [RIMO98] [LUMI01]. El primero utiliza SANE con un enfoque tradicional, no coevolutivo, y evoluciona las redes contra WALLY, un jugador artificial de GO ampliamente conocido. El segundo trabajo mencionado emplea SANE con el enfoque de coevolución competitiva y utilizando las técnicas de Shared Sampling, Fitness Sharing y Hall of Fame. Es importante destacar que en su trabajo, Lubberts demuestra la efectividad de estas tres técnicas.

En el trabajo de Alex Lubberts se utiliza la versión 2 de SANE implementada en java, y éste facilitó el código resultado de su trabajo, el cual sirvió como punto de partida para el desarrollo.

IV.2 Decisiones de diseño

Conexiones

En algunos de los trabajos realizados anteriormente, las redes neurales no eran completamente conexas, sino que por lo contrario, las neuronas tenían un numero fijo de conexiones. Sin embargo, se decidió explorar la posibilidad de utilizar redes completamente

conexas, para lo cual se llevó a cabo un experimento que podría sugerir cual estrategia daría mejores resultados. Este experimento consistió en dos corridas evolutivas con los mismos parámetros, con la única diferencia de que en la primera las redes serían completamente conexas y en la segunda corrida las neuronas tendrían un numero fijo de conexiones. Los resultados se resumen en los siguientes gráficos:

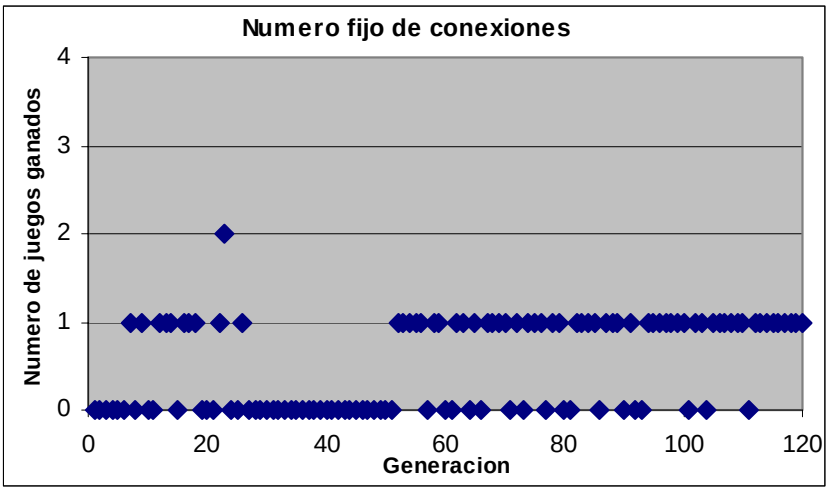


Figura 4.1: Evolución de redes con numero fijo de conexiones.
Los puntos representan la cantidad de juegos ganados por las redes neurales en cada generación. Nótese el estancamiento en el aprendizaje de las redes neurales.

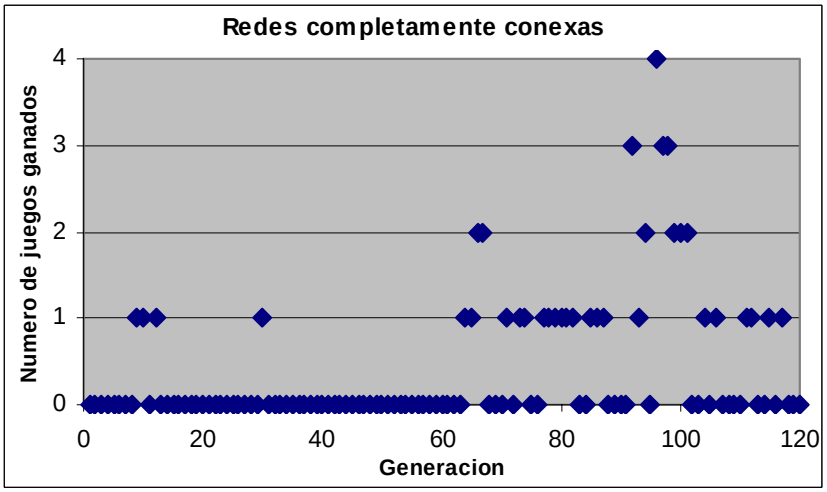


Figura 4.2: Evolución de redes completamente conexas.
Las redes muestran un nivel mas elevado de aprendizaje que en la corrida anterior.

Como podemos observar, las redes completamente conexas ofrecen una mayor cantidad de victorias. Por esta razón se decidió utilizar esto como principio de diseño.

Adicionalmente, se puede definir la no conexión entre dos neuronas como un caso especial de conexión, donde el peso de la conexión sea cero. En este caso, queda abierta la posibilidad de que el proceso evolutivo "desconecte" neuronas llevando el valor de la conexión a cero.

Estructura a tres segmentos

Siguiendo la sugerencia de Alexis Hernández, asesor experto en GO, se decidió experimentar en segunda instancia con una arquitectura de red particular, con las siguientes características: la capa escondida estaría separada en tres segmentos, cada uno encargado de una fase diferente del juego: Fuseki, Medio Juego y Yose. Esta idea responde a la naturaleza del estilo de juego en estas tres fases:

- **Fuseki**: se refiere a la primera fase del juego, donde las jugadas tienden a ser mas elásticas y el territorio es demarcado mas abiertamente.
- **Medio juego**: es la fase principal del juego, donde se realiza la mayor parte de los movimientos, a la vez que el ataque y la defensa son las estrategias primordiales. Se pueden presentar muchas situaciones de vida o muerte que definen el resultado del encuentro.
- **Yose**: es la denominación que recibe la fase final del juego, donde la elasticidad de las jugadas es mínima, y la

preocupación principal es cerrar las últimas áreas indefinidas del tablero.

Cada segmento de neuronas escondidas estará activo en la fase que le corresponde. Al inicio del juego, el segmento encargado del Fuseki estará activo, y una neurona especial (que no pertenece a ninguno de los tres segmentos mencionados) se encarga de decidir cuando ha finalizado el Fuseki para dar inicio a la próxima fase de medio juego. De forma similar, una neurona especial se encargara de decidir cuando ha finalizado el medio juego, dando paso a la etapa final de Yose.

Por los momentos las neuronas "de paso" están implementadas de forma que sólo tomen en cuenta la densidad del tablero, con valores predeterminados en los pesos para este fin. Cuando el tablero alcanza cierta densidad, se pasa de un segmento al próximo. Un análisis heurístico del tablero podría dar mejores resultados, o quizás la inclusión de las neuronas de paso en el proceso evolutivo.

Posteriormente se analizarán los resultados de esta estructura, y se compararán con los obtenidos al utilizar el diseño original.

IV.3 Diseño

Tomando como referencias los trabajos revisados en la fase de investigación, se decidió el siguiente diseño inicial:

1. **Tablero:** se trabajará con dos tamaños de tablero: 5x5 y 9x9 (este último es el tamaño que típicamente utilizan los principiantes para aprender) con el fin de identificar

diferencias en velocidad de aprendizaje y tiempos de evolución.

2. **Organización**: una red Feed Forward[5] con una capa escondida de neuronas. Se realizarán experimentos con dos tipos de organizaciones: 1) una capa escondida de forma tradicional y 2) una capa escondida dividida en la estructura de tres segmentos expuesta en la sección anterior “Decisiones de diseño”.

3. **Entradas**: Las redes tendrán 2N entradas, donde N es el número de intersecciones del tablero (50 para un tablero 5x5, 162 para 9x9). Una de las entradas indicará si hay una piedra de color blanco en una intersección determinada y la otra si hay una piedra de color negro. Dadas las características del juego, no pueden estar encendidas las dos entradas simultáneamente.

4. **Salidas**: Las redes tendrán N salidas (N es el número de intersecciones), una para cada intersección del tablero. Cada salida será un numero real en el rango (0..1) que indicará cuan conveniente es jugar en esa intersección. Mientras mayor sea el valor, mas conveniente es realizar la próxima jugada en esa intersección.

5. **Tamaño**: siguiendo los resultados mostrados en [RIMO98] se estableció como tamaño de prueba una capa escondida de 500 neuronas para el tablero 9x9. Para el tablero 5x5 se tomaran redes de 100 neuronas, valor tomado de

[5] Una red Feed Forward es aquella donde todas las neuronas de una capa están conectadas únicamente con neuronas de las capas superiores.

[LUMI01]. Se experimentará con otros tamaños para observar el resultado de esta variación.

6. **Conexiones**: Cada neurona de la capa escondida recibirá conexiones de todas las neuronas de la capa de entrada y se conectará con todas las neuronas de la capa de salida.

IV.4 Implementación

Se partió de la implementación existente de SANE en Java, realizando una serie de modificaciones para mejorar el diseño orientado a objetos y aprovechar capacidades ofrecidas por Java (serialización, herencia, constructores, etc.). Estos son los paquetes mas importantes que resultaron del rediseño de las clases:

Paquete CoCoSane:

Contiene lo referente al algoritmo coevolutivo SANE competitivo, así como las implementaciones de Network (Red Neural) y Neuron (Neurona). La clase Evolution es la principal del sistema y coordina los detalles de la evolución, como el número de generaciones, llamadas a los métodos de evaluación, etc. No contiene detalles del dominio, de forma que la evaluación de las redes neurales deben ser implementadas por alguna subclase de Network.

Paquete GoCoCoSane

Contiene principalmente la Clase Network que extiende a CoCoSane.Network, e implementa el método de evaluación de la red, específicamente para redes que jueguen GO. Hace uso de las clases del paquete GO para efectuar los juegos entre redes.

Paquete NeoGo

Implementa el diseño de tres segmentos explicado anteriormente, para lo que crea subclases de Network y NetworkPopulation.

Paquete GO

Aquí residen una serie de clases necesarias para desarrollar juegos de GO, tales como Board (tablero), Game (juego), Player (jugador), Tournament (torneo) y otros. También permite la creación de jugadores basados en Redes Neurales o en programas externos. Esta última implementación es útil para enfrentar a redes neurales contra programas como GnuGo, Wally o cualquier jugador de GO que implemente el protocolo GTP.

Para una información mas detallada de las clases y su funcionamiento, refiérase a la documentación JavaDoc publicada junto con el código fuente en [GIIAR].

Detalles de implementación

A continuación se presentan los detalles mas importantes de la implementación:

Pesos de las conexiones

Las neuronas son creadas al inicio de la evolución con una serie de conexiones a las capas de entrada y salida. Cada conexión tiene un peso con un valor aleatorio en el rango [-0.5, 0.5].

Selección de jugada: mejor jugada válida

La red neural, luego de ser alimentada con el tablero, se activa para obtener una serie de salidas (para un tablero de 9x9 se obtendrán 81 salidas, una para cada intersección). Para efectuar la

jugada, se selecciona la salida de mayor valor que represente una jugada válida, tal y como se definió en el marco teórico.

Adicionalmente, se implementa un mecanismo para que el jugador pueda pasar: si ninguna jugada válida supera un valor mínimo[6], el jugador considera que no hay jugadas valiosas que hacer, y pasa el turno a su oponente.

Regla de KO

Para asegurar el cumplimiento de la regla de KO, durante cada juego se almacena un historial de todos los estados del tablero. Si alguna jugada intentara devolver el tablero a alguno de estos estados, se considera como una jugada inválida.

Sistema de Puntuación

El código fuente facilitado por Lubberts incluía un método para estimar la puntuación de los juegos. Sin embargo, se realizaron dos pruebas preliminares para evaluar este sistema. La primera prueba evaluó 1000 tableros (elegidos aleatoriamente) con ambos métodos, revelando que en mas de un 50% de los casos la puntuación del algoritmo interno muestra una puntuación diametralmente opuesta a Gnugo[7]. La segunda prueba buscaba observar el efecto de esta diferencia en el proceso evolutivo. Con este fin se efectuaron dos corridas con parámetros idénticos, con la única diferencia de que en la primera se realizaron las puntuaciones con GnuGo y la segunda con el algoritmo interno. Los resultados mostrados en la graficas 4.3 y 4.4 muestran un comportamiento

[6] Se utilizo 0.5 como valor mínimo, imitando el utilizado en [LUMI01]

[7] Resultado que declara ganador al jugador opuesto al declarado por gnugo.

errante y un retraso en el aprendizaje al utilizar el algoritmo interno comparado con la otra corrida. Por esta razón, se decidió utilizar a gnugo como árbitro de los juegos, para decidir la puntuación de éstos.

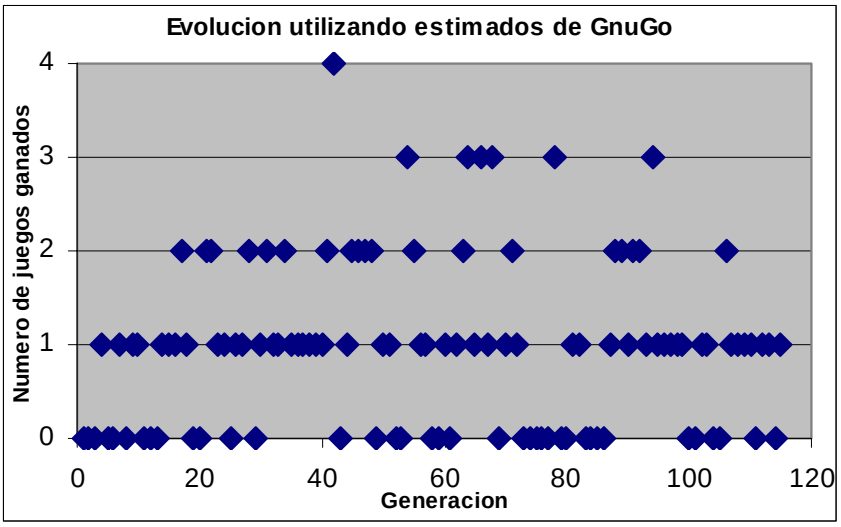


Figura 4.3 Resultados de corrida evolutiva utilizando puntuación de GnuGo. *Se puede observar un mejoramiento progresivo en el nivel de juego.*

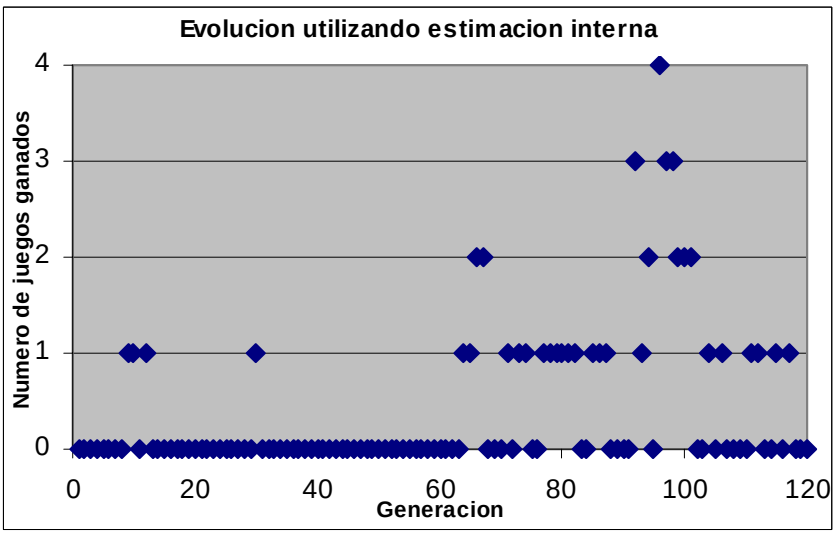


Figura 4.4 Resultados de corrida evolutiva utilizando cálculo interno de puntuación. *Nótese el retraso evidente en el aprendizaje de las redes.*

Configuración dinámica

Para facilitar las corridas evolutivas, se implementó un mecanismo de configuración dinámica, que lee de un archivo de texto los parámetros que se deseen modificar. Algunos de estos parámetros

son: número de generaciones, cantidad de neuronas y redes en las poblaciones, número de neuronas en las capas escondidas, etc.

Protocolo de comunicación GTP

Para establecer la comunicación entre las redes y el jugador externo (GNUGO) se utilizo el protocolo GTP [GTP], diseñado especialmente para este fin.

IV.5 Corridas

Una vez implementado el mecanismo evolutivo, se procedió a efectuar las corridas, con diferentes configuraciones para evaluar el efecto de éstas en el aprendizaje de las redes. Se implementó un mecanismo de checkpoints, basado en los mecanismos de serialización de Java, para evitar posibles pérdidas de datos en caso de alguna falla o interrupción del programa.

Las configuraciones utilizadas se muestran en la página siguiente.

Tabla 4.1: Configuraciones de corridas evolutivas.
(Fuente: elaboración propia)

	CoCoSane					
Nombre Corrida	**coco_100**	**coco_200**	**coco250**	**coco500**	**coco500A**	**coco2000**
Muestra	25	25	20	20	20	20
Muestra HOF	3	3	3	3	3	3
Población de neuronas	2000	2000	2000	2000	4000	20000
Población de redes	200	200	200	200	400	400
Neuronas Escondidas	100	200	250	500	500	2000
Tamaño tablero	5	5	9	9	9	9

	NeoGo					
Nombre Corrida	**343434**	**206020**	**666666**	**4012040**	**166166166**	**100300100**
Muestra	25	25	25	25	25	25
Muestra HOF	3	3	3	3	3	3
Población de neuronas	990	1000	1980	2000	4980	5000
startgame	330	200	660	400	1660	1000
midgame	330	600	660	1200	1660	3000
endgame	330	200	660	400	1660	1000
Población de redes	200	200	200	200	400	400
Neuronas Escondidas	99	100	198	200	498	500
startgame	33	20	66	40	166	100
midgame	33	60	66	120	166	300
endgame	33	20	66	40	166	100
Tamaño tablero	5	5	5	5	9	9

Leyenda
Muestra: tamaño de la muestra tomada de la población parásito para evaluar a la población host.
Muestra HOF: tamaño de la muestra tomada del salón de la fama (hall of fame) para evaluar a la población host.
Startgame: número de neuronas en el segmento de inicio de juego.
Midgame: número de neuronas en el segmento de medio juego.
Endgame: número de neuronas en el segmento de fin de juego.
CoCoSane: estructura tradicional de red neural: una capa escondida homogénea.
NeoGo: redes neurales con estructura a tres segmentos.

IV.6 Pruebas

El comportamiento esperado de las redes neurales que evolucionan es que tengan un mejor nivel de juego a medida que avanzan las generaciones. La única manera de comprobar esto es enfrentarlas con un jugador externo. En este caso se eligió GNUGO por varias razones:

- Es un proyecto de código abierto
- Es ampliamente conocido en el área de Computer GO
- Permite graduar el nivel de dificultad (de 1 a 10)
- Implementa un protocolo conocido de comunicación: GTP.

Con el fin de realizar las pruebas necesarias, las corridas evolutivas almacenaron las 4 mejores redes de cada generación. Esto permite monitorear el nivel de las redes a lo largo de la evolución y el efecto de las diferentes configuraciones en la velocidad de aprendizaje. Una vez terminadas las corridas, se efectuaron torneos en los cuales las redes obtenidas jugaron contra GNUGO. Los resultados de estos torneos para las diferentes configuraciones se muestran en el próximo capítulo.

Capitulo V: Resultados

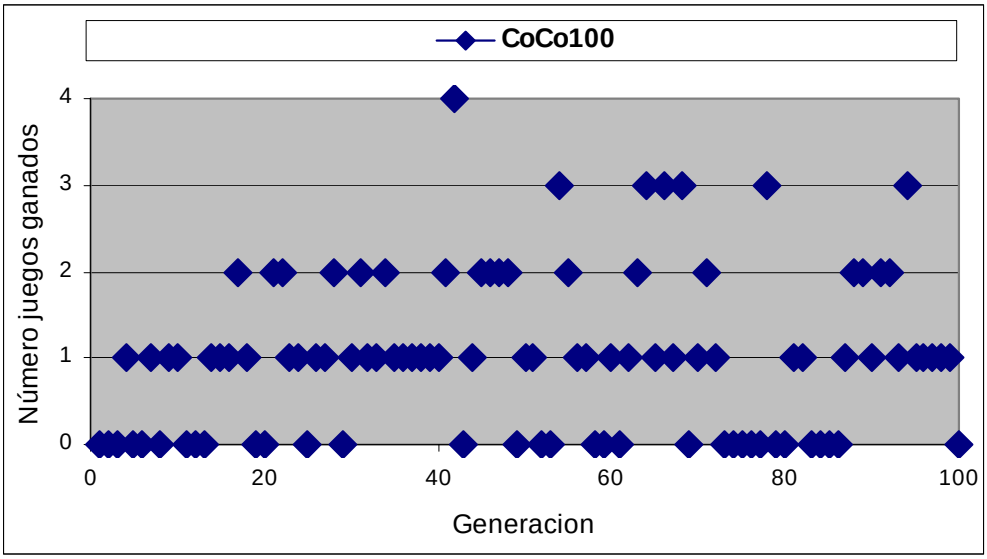
V.1 Tablero 5x5

A continuación se presentan los resultados de los torneos realizados en cada generación para las distintas configuraciones de los tableros 5x5. En las gráficas, el eje X representa cada una de las generaciones, mientras que el eje Y indica la cantidad de juegos ganados por parte de las redes neurales al jugar contra GnuGo. Para cada generación se almacenaron las 4 mejores redes neurales, las cuales jugaron una vez con el color blanco y una vez con el color negro, de forma que en cada generación se realizaron 8 juegos.

Es importante notar que el tablero 5x5 es un tablero muy pequeño, y de hecho es un problema resuelto: si el jugador que tiene el primer movimiento (negro) juega la partida correctamente, es imposible que pierda. Esto justifica un hecho notable en todas las corridas de este tamaño: las redes nunca ganan mas de 4 juegos por generación. Mas aún, los resultados arrojaron que en el 99% de los juegos ganados por las redes éstas utilizaban el color negro. Estos hechos parecen indicar que para las redes es prácticamente imposible ganar utilizando blanco. Dadas las características de este tablero (problema resuelto), esto no representa falta de habilidad, sino una particularidad del juego.

CoCo_100

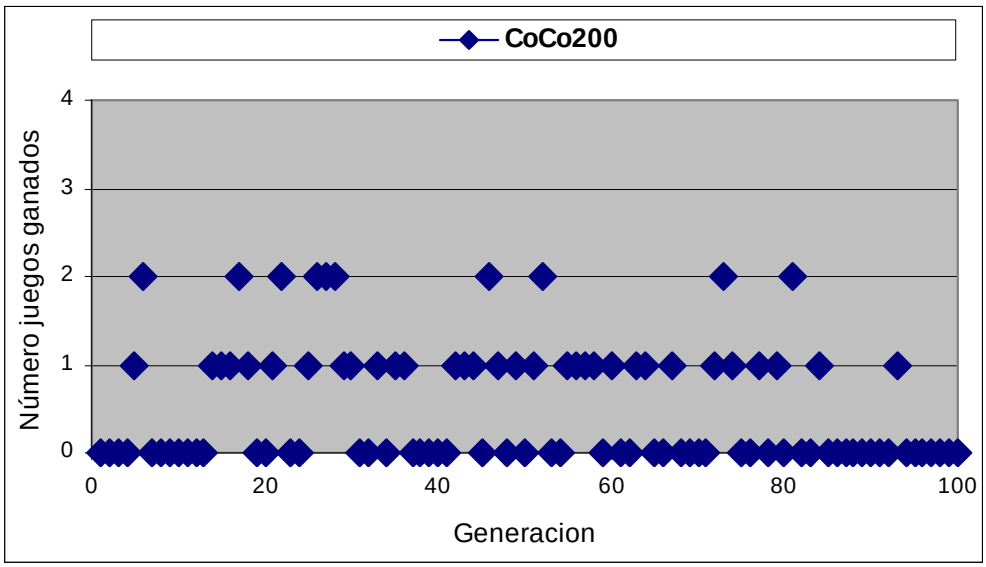
Esta corrida utiliza una población de redes de 200 individuos, cada una con 100 neuronas en la capa escondida. Se puede observar un definitivo mejoramiento en el nivel de juego. Ya en la cuarta generación se comienzan a presentar victorias sobre GNUGO y para la generación 17 se comienzan a ganar hasta 2 juegos por generación. El pico en la generación 42 indica que el juego de las redes es prácticamente perfecto en ese momento[8], y varios puntos en 3 victorias en las generaciones sucesivas indican que el nivel de juego se mantiene elevado.



[8] Tomando en cuenta que es prácticamente imposible ganar un juego como blanco, sólo se pueden ganar un máximo de 4 juegos (como negro).

CoCo_200

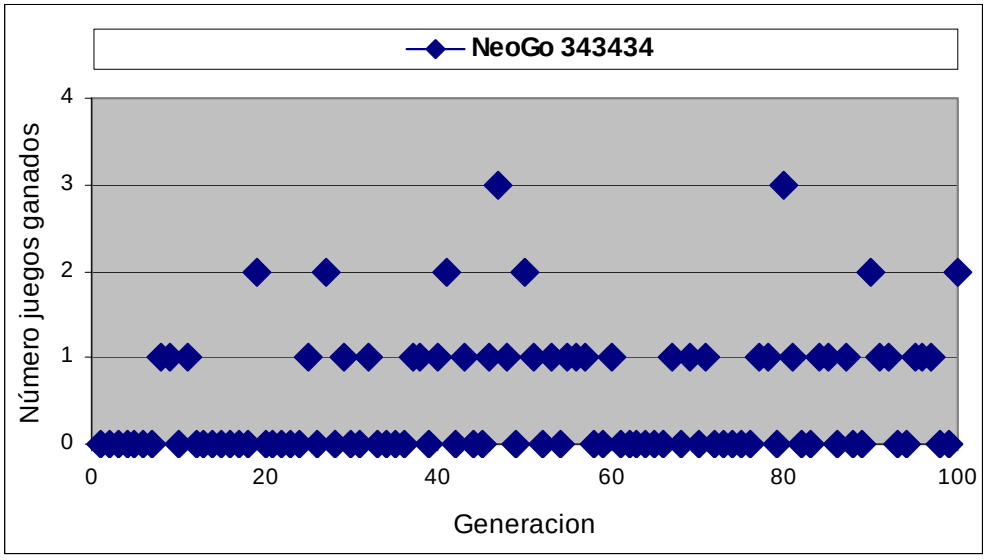
Esta corrida es muy semejante a la anterior, con la única diferencia que las redes tienen 200 neuronas en lugar de 100. Aunque hay un comportamiento relativamente semejante al de CoCo_100, los datos no muestran evidencias de una mejora en el nivel de juego ni en la velocidad de aprendizaje comparados con aquel.



Neo_343434

Esta es la primera corrida que ensaya con la estructura a tres segmentos. Con intención de comparar con CoCo_100, se utilizan un numero semejante de neuronas, distribuidas de forma uniforme en los tres segmentos: 34 neuronas en cada uno para un total de 102 neuronas en la capa escondida.

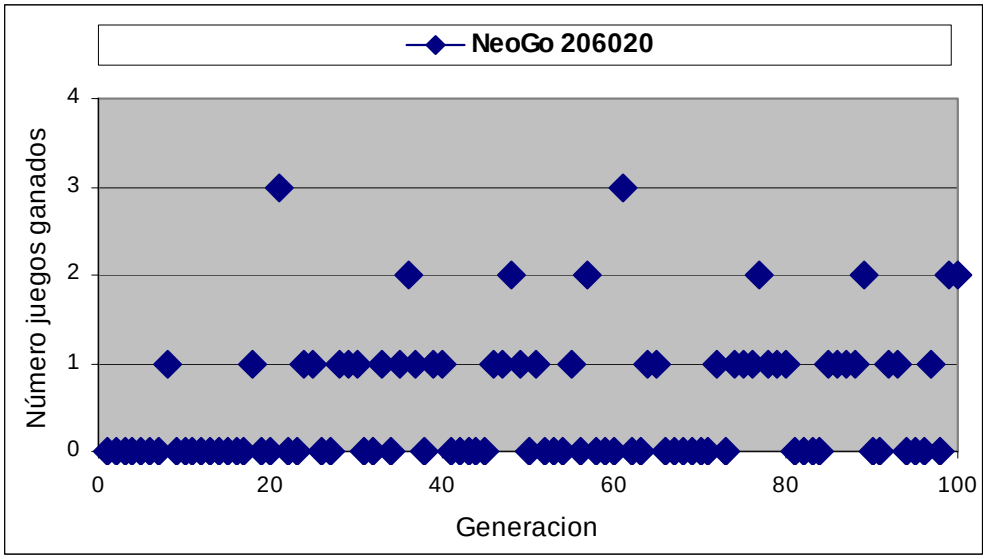
Esta corrida también muestra un mejoramiento en el nivel de juego a medida que avanza la evolución, pero no evidencia mejoras significativas con respecto a CoCo_100.



Neo_206020

Un segundo experimento con la estructura a tres segmentos, pero esta vez distribuyendo de forma desigual las neuronas. Esta vez los segmentos de inicio y fin de juego tienen 20 neuronas cada uno y el de medio juego tiene 60 neuronas.

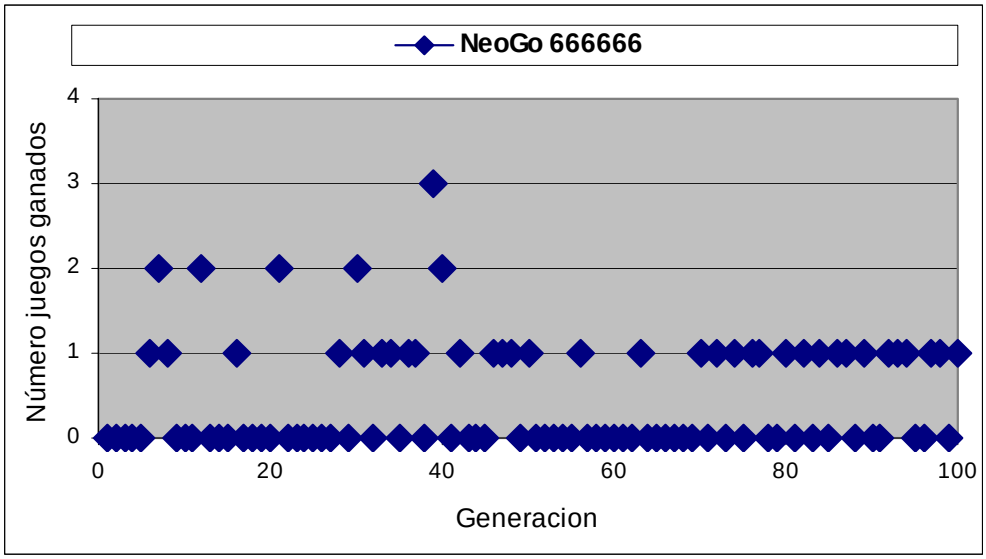
Una vez mas, se observa un buen nivel de aprendizaje, pero no se evidencian mejoras con respecto a CoCo_100.



Neo_666666

El tercer ensayo con la estructura a tres segmentos se asemeja a CoCo_200, distribuyendo las 200 neuronas en partes iguales: 66 neuronas en cada segmento para un total de 198.

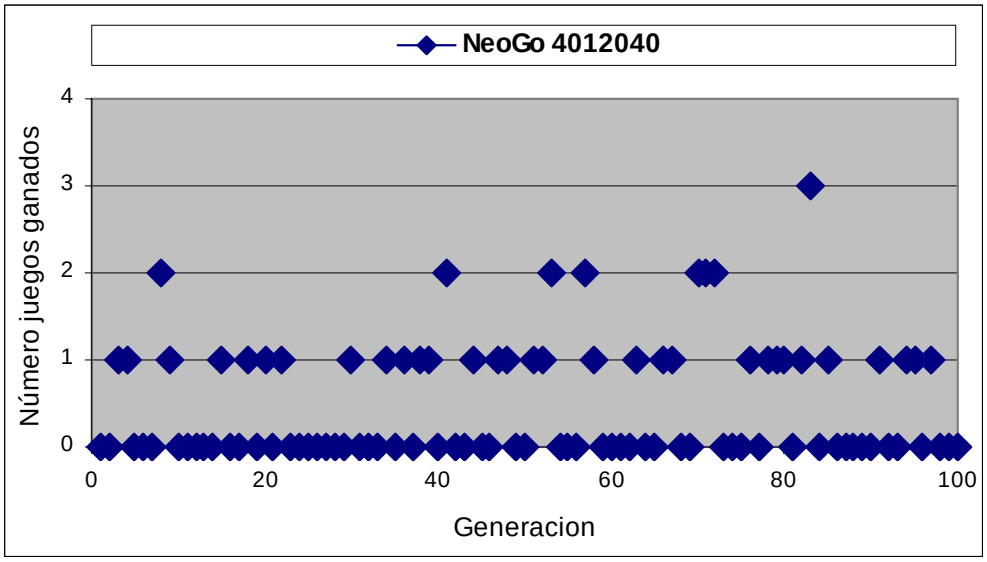
La grafica obtenida es semejante a la de CoCo_200, y aunque aquí se observe un pico en la generación 39 con 3 victorias, CoCo_200 parece mantener el nivel de 2 victorias mas constantemente que Neo_666666.



Neo_4012040

Una vez mas se experimenta con la estructura de tres segmentos, esta vez distribuyendo 200 neuronas de forma irregular en los tres segmentos: 40 neuronas para la primera y última fases, y 120 neuronas para la fase de medio juego.

De nuevo es necesario poner esta corrida en perspectiva con CoCo_200, y observamos que el nivel de juego es bastante semejante, así como la velocidad de aprendizaje.



V.2 Tablero 9x9

Para el tablero 9x9 se realizaron 6 corridas evolutivas (descritas en la tabla 4.1) y no se observaron evidencias de aprendizaje. Los juegos realizados entre las redes neurales y GnuGo resultaron todos en victorias para éste último. Por razones de tiempo no se exploraron mas generaciones.

V.3 Estrategias desarrolladas

A lo largo de los juegos se pudieron identificar algunas estrategias importantes desarrolladas por las redes neurales, como son la formación de ojos, separación de grupos del oponente y captura. El Anexo 2 ilustra estas estrategias con algunos de los juegos realizados por las redes.

Ojos

El comportamiento que mas se observa es el de formación de ojos, figura imprescindible para sobrevivir. Aun en casos donde las redes efectúan mas jugadas de lo necesario al final del juego, éstas mantienen suficientes ojos para garantizar su supervivencia.

Separación de grupos

Una estrategia importante en el juego de GO es separar grupos del oponente, pues al estar separados tienden a ser mucho mas débiles. Las redes muestran este comportamiento en un número de juegos, donde ocupan posiciones que separan dos o mas grupos del oponente.

Captura

La captura es una estrategia esencial en las batallas de vida o muerte. Las redes aparentemente demuestran cierta habilidad para identificar cuando la captura de un grupo del oponente representa su oportunidad de sobrevivir y efectúan la captura a tiempo.

Capitulo VI: Conclusiones y Recomendaciones

VI.1 Conclusiones

Tablero 9x9

Dadas las condiciones del experimento, el aprendizaje para un tablero 9x9 no es factible, pues luego de 100 generaciones aun no se observó una tendencia a mejorar la calidad de juego. Por limitaciones de tiempo no se pudo ensayar con un mayor número de generaciones.

Tablero 5x5

El experimento en tableros de 5x5 muestra que las técnicas utilizadas en conjunción con el algoritmo coevolutivo SANE competitivo (Hall of Fame, Shared Sampling y Fitness Sharing) son efectivas al aplicarse como método de aprendizaje para las redes neurales en el ámbito de GO para este tamaño de tablero.

Cantidad de neuronas

Bajo las premisas y condiciones del experimento, el aumento en la cantidad de neuronas para un tablero de 5x5 no parece dar un aporte significativo a la velocidad de aprendizaje ni al nivel de juego alcanzado.

Estructura a tres segmentos

La estructura a tres segmentos, que representa un primer intento de particularización de las redes, no ofreció una mejora en el

proceso de aprendizaje. Sin embargo, mantiene la tendencia de avance de la estructura tradicional, lo que es un indicio positivo.

Framework SANE

Se obtuvo exitosamente un framework que contiene una serie de clases, escritas en java, que facilitan la implementación de Coevolución Competitiva SANE para otros dominios. Las clases fueron implementadas con especial atención en un buen diseño Orientado a Objetos y en facilitar el uso para otros dominios. Todo el código fuente, junto con su documentación está publicado en [GIIAR].

VI.2 Recomendaciones

Paralelización

Uno de los elementos decisivos en el alcance de este trabajo es el tiempo que necesitan las corridas en tableros medianos y grandes. Una corrida evolutiva para un tablero 9x9 puede tomar hasta 10 días[9]. Evidentemente, realizar experimentos con estos lapsos de tiempo es un problema. Una de las posibles soluciones es paralelizar la aplicación para aprovechar las capacidades de múltiples procesadores con el fin de disminuir drásticamente estos tiempos, lo que permitiría correr un mayor número de generaciones y experimentar con distintas configuraciones.

Estructura a tres segmentos

Una de las limitaciones importantes de la implementación de la estructura a tres segmentos de este trabajo es la estrategia de

[9] En un procesador Pentium IV de 2.8Ghz con 512Mb de memoria RAM.

recompensa. El fitness del individuo sigue siendo calculado de forma global, es decir, basado en el resultado del juego y no en el resultado de cada fase. Esto puede tener una implicación negativa para el algoritmo evolutivo, pues puede estarse premiando incorrectamente a los individuos: si una red neural jugó brillantemente durante el inicio del juego pero falló en el medio juego tendrá un fitness bajo. Se podría explorar la posibilidad de utilizar un fitness para cada segmento, y que en el caso mencionado el fitness del segmento de inicio de juego fuera alto, independientemente del resultado final del juego. Sin embargo, esto presenta serias dificultades, como por ejemplo la identificación de la calidad de juego en cada fase, parámetro que es difícilmente cuantificable.

Por otro lado, otro punto importante es el paso de una fase a otra del juego. La implementación actual pasa del fuseki al juego medio y del juego medio al yose basada en la densidad del tablero. Una posible oportunidad de mejora es explorar estrategias mas sofisticadas para identificar el final de una fase y el comienzo de la próxima.

Particularización

Aun cuando la estructura a tres segmentos no mostró en este caso una mejora en el aprendizaje, es muy posible que los esfuerzos para obtener mejores jugadores de GO basados en redes neurales deban concentrarse en particularizar las redes e insertar un mayor nivel de conocimiento acerca del dominio. Esto puede incluir reconocimiento de patrones predefinidos, bases de datos de secuencias establecidas y uso de estructuras de datos particularizadas. Esto puede resultar en estrategias desarrolladas

con mayor facilidad y en menor tiempo, finalmente logrando un aprendizaje mas veloz y de mejor nivel.

Bibliografía

[BISH] Bishop, C. (1995). Neural Networks for Pattern Recognition. Editorial Oxford University Press.

[BRMO] Martín del Brío, B. y Sanz Molina, A. (2001) Redes neuronales y sistemas difusos (Segunda edición). Alfa Omega.

[DEEP] Deep blue, jugador artificial de ajedrez. http://www.research.ibm.com/deepblue/

[EXPGO] Explorer, Jugador artificial de GO: http://www.cs.ualberta.ca/~mmueller/cgo/explorer.html

[IGF] Intelligent GO Foundation: http://www.intelligentgo.org/

[GIIAR] Grupo de Investigación en Inteligencia Artificial, UCAB: http://www.ucab.edu.ve/ingenieria/informatica/giiar/tesis.htm

[GNUGO] GNU GO: http://www.gnugo.org

[GOLD] Goldberg, D. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. Editorial Addison-Wesley Professional.

[GTP] Gunnar F. (2002). Specification of the Go Text Protocol, version 2, draft 2. http://www.lysator.liu.se/~gunnar/gtp.

[HAYKIN] Haykin , S. (1998). Neural Networks: A Comprehensive Foundation (Segunda edicion). Prentice Hall.

[JAGO] JaGo, un cliente gráfico gratuito para el juego de GO. http://www.rene-grothmann.de/jago/

[LUMI01] Lubberts, A. y Miikkulainen, R. (2001). Co-Evolving a Go-Playing Neural Network. Coevolution: turning adaptive algorithms

upon themselves. Birds-of-a-feather workshop, Genetic and Evolutionary Computation Conference.

[MOMI94] Moriarty D. y Miikkulainen R. (1994) Efficient Reinforcement Learning Through Symbiotic Evolution.

[MOMI98] Moriarty D. y Miikkulainen R. (1998) Forming Neural Networks Through Efficient and Adaptive Coevolution.

[NEUNRG] Neural Networks Research Group, University of Texas: http://nn.cs.utexas.edu/

[PRESS] Pressman, R. (1998). El Producto. En Ingeniería del Software, un enfoque práctico (pp. 24-25) (Cuarta Edición). Mc. Graw Hill.

[RIMO98] Richards, N., Moriarty, D. y Miikkulainen, R. (1998). Evolving Neural Networks to Play Go.

[ROSI95] Rosin, C. y Belew, R. (1995). Methods for Competitive Coevolution: Finding opponents worth beating. Cognitive Computer Science Research Group, University of California, San Diego.

[ROSI97] Rosin, C. (1997). Coevolutionary Search Among adversaries. Trabajo de grado, Doctorado en Ciencias de la Computación, Universidad de California, San Diego.

[RUNO03] Russel, S. y Norving, P. (2003). Artificial Intelligence, a modern approach (Second Edition). Prentice Hall.

[SENSEI] Sensei's Library http://senseis.xmp.net/

[SMAGO] Smart Go Program: http://www.smartgo.com/

[USGO] American Go Association: http://www.usgo.org/index.asp

Apéndice 1: Glosario de términos de GO

- **Captura**: cuando un grupo de piezas pierde su última libertad gracias a una jugada del oponente, las piedras que lo conforman son retiradas del tablero, y se consideran capturadas. Estas piezas serán tomadas en cuenta para el cálculo del resultado final.
- **Elasticidad**: dos piezas guardan una relación de elasticidad entre ellas dependiendo de cuán lejanas estén entre si en el tablero. Se utiliza el término jugada elástica para indicar que una piedra esta relativamente alejada de los otros grupos de su color. Por el contrario, una jugada poco elástica esta muy cercana a algún grupo.
- **Escalera**: formación particular de piedras que asemejan la forma de una escalera. La siguiente figura ilustra el patrón mencionado:

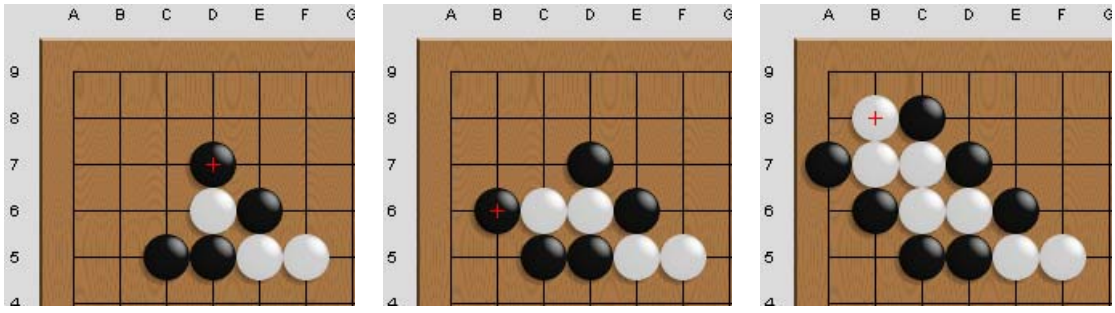


Figura A1.1: Escalera. *Blanco D6 esta amenazado (izq.) y forma una escalera al intentar escapar de negro con C6 (centro), quien lo persigue hasta acorralarlo en la esquina superior izquierda (der.). (Ilustración elaborada con [JAGO])*

- **Fuseki:** fase inicial del juego.
- **Grupo**: una o mas piezas del mismo color posicionadas de forma adyacente forman un grupo. Nótese que la adyacencia es únicamente horizontal y vertical, no diagonal.
- **Libertad**: un grupo tiene tantas libertades como intersecciones adyacentes libres. Si un grupo se queda sin libertades, es considerado capturado y eliminado del tablero.
- **Medio juego:** fase principal del juego, posterior al fuseki y anterior al Yose. En esta fase se desarrollan las batallas de vida o muerte y normalmente es la que define una mayor cantidad de puntos.
- **Ojos**: Una libertad interna de un grupo que no puede ser removida (porque esta rodeada de piedras de un color, de forma que la jugada del oponente en esa posición esta prohibida por ser suicida). Un grupo solo puede garantizar su supervivencia en el tablero si forma (o puede formar) dos ojos.

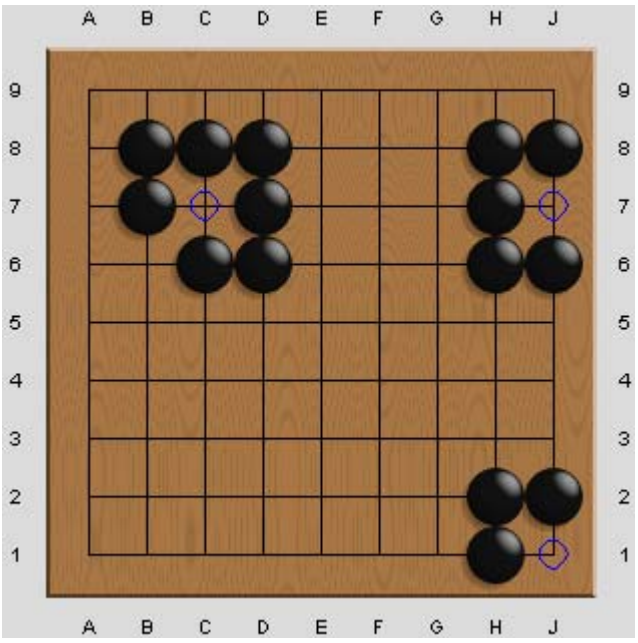


Figura A1.2: Ojos.
Las intersecciones marcadas con círculos azules son ojos de los grupos que las rodean.
(Ilustración elaborada con [JAGO])

- **Piedra**: una pieza de color blanco o negro que utilizarán los jugadores para ocupar las intersecciones.
- **Vida o muerte**: una serie de jugadas en las que dos o mas grupos batallan por capturar al otro grupo o ser capturados.
- **Yose:** fase terminal del juego, donde el destino de los grandes grupos ya se encuentra definido. Esta fase consiste en terminar de demarcar las zonas del tablero.

Apéndice 2: Juegos

Durante la fase de evaluación de las corridas evolutivas se guardaron algunos juegos en los que las redes resultaron victoriosas, para observar su comportamiento. En las siguientes páginas resaltaremos algunas de las estrategias identificadas por medio de la observación de estos juegos.

Los juegos se almacenaron utilizando el formato SGF (Smart GO Format), formato diseñado específicamente para guardar secuencias de juegos de GO. Puede encontrar mas ejemplos de juegos entre las redes neurales y GnuGo en la siguiente página web:

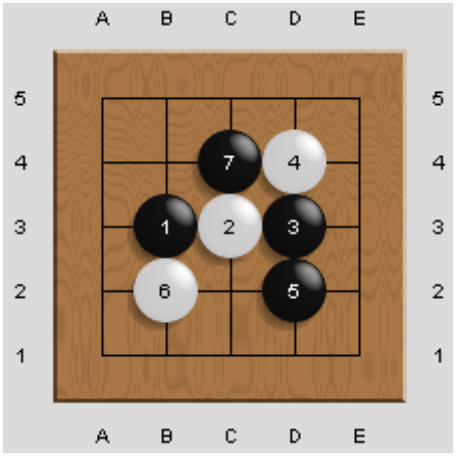
http://www.ucab.edu.ve/ingenieria/informatica/giiar/tesis.htm

Para ver los juegos se necesita un lector de archivos SGF (hay un número de ellos gratuitos disponibles en Internet). La siguiente dirección contiene una lista de lectores de archivos SGF:

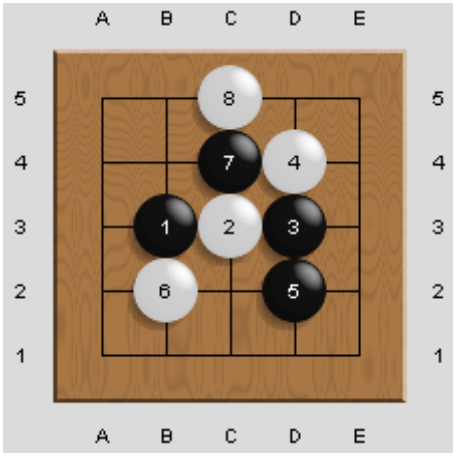
http://gobase.org/software/editors/

Juego #1

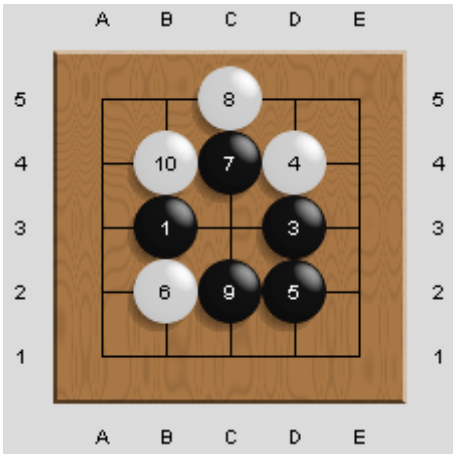
En este juego se identifican varias estrategias importantes: separación de grupos del enemigo, captura de piedras y reconocimiento de amenaza de los grupos propios.



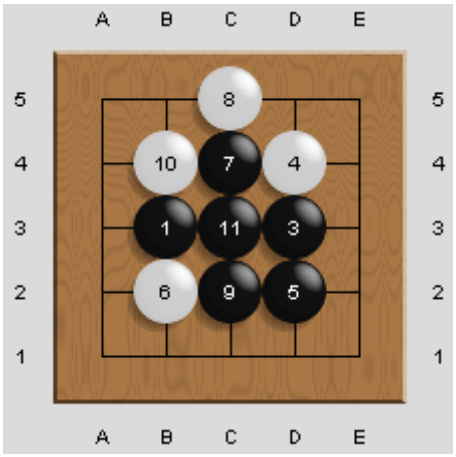
La jugada 7 de Negro cumple una doble función: separa dos grupos de blanco (2 y 4) a la vez que amenaza a 2.



La respuesta de blanco es amenazar en 8.



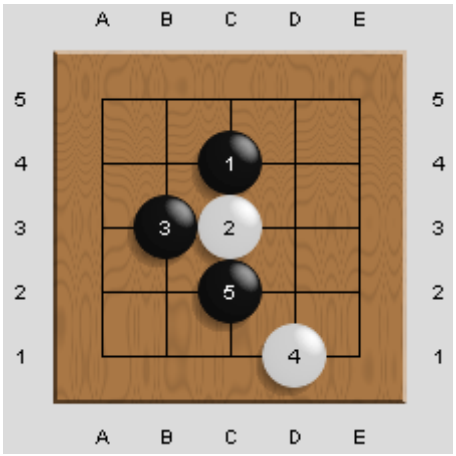
Negro captura a blanco con 9, contra lo cual blanco vuelve a amenazar con 10.



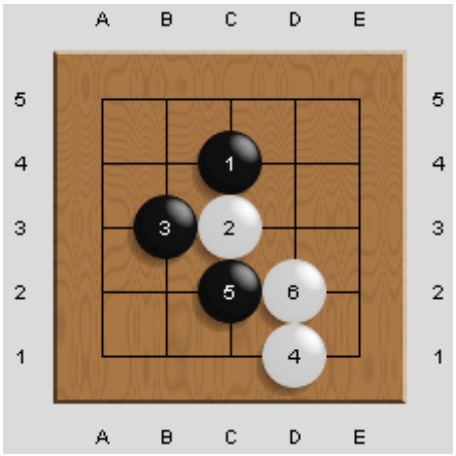
Negro parece reconocer la amenaza y salva su piedra al conectarla al resto del grupo con 11.

Juego #2

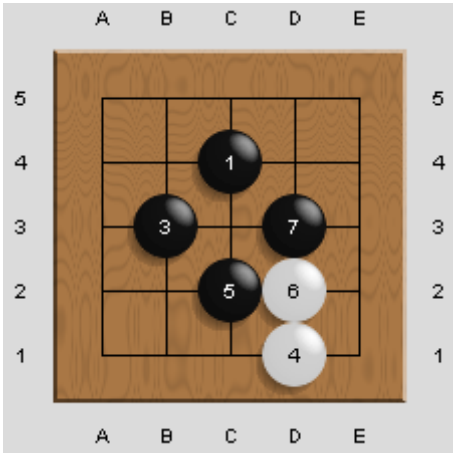
Este juego ilustra una estrategia importantísima: la formación de ojos por parte de negro. Por otro lado, con no menos importancia, se observan de nuevo estrategias de captura y separación de grupos.



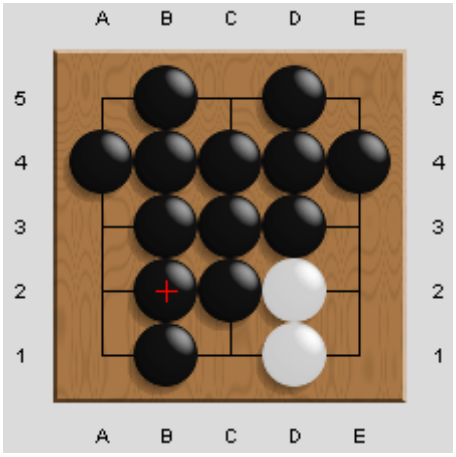
De forma semejante al juego anterior, la jugada 5 de Negro separa dos grupos de blanco (2 y 4) a la vez que amenaza a 2.



Blanco hace un intento por unir sus grupos con 6.



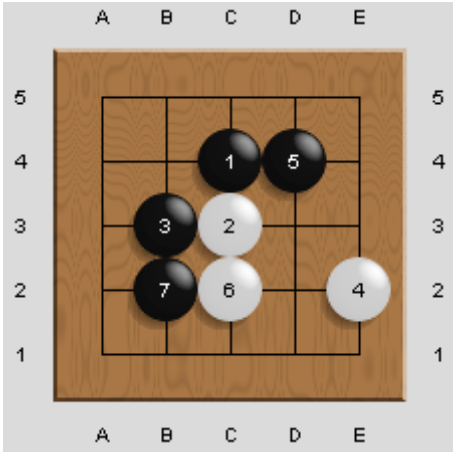
Con 7 negro evita que se unan los grupos de blanco y obtiene una captura valiosa.



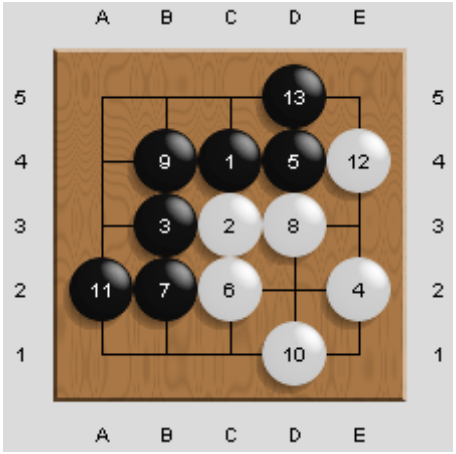
Negro culmina su juego asegurándose la victoria al formar ojos en A5, C5 y E5.

Juego #3

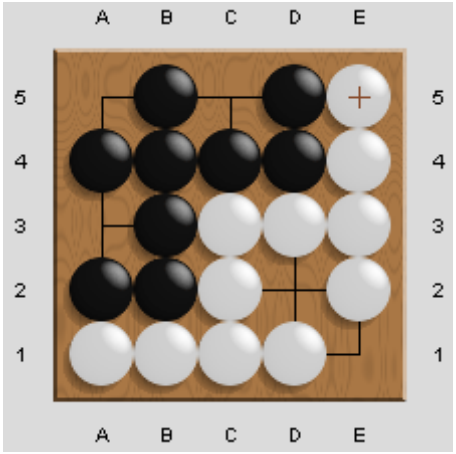
Este juego es particular pues se perfila mas pacífico que los anteriores, y muestra el atractivo de que blanco no se retira tan temprano. Una vez mas, negro logra asegurar suficientes ojos y territorio para sobrevivir.



Ambos jugadores parecen estar asegurando su lado del territorio



Continua el avance del juego, y las posibilidades de batalla disminuyen.



Efectivamente, el juego termina sin ninguna captura. Cada jugador aseguró su territorio, pero negro logró apoderarse de mas intersecciones, obteniendo otra victoria.