On convergence rates of game theoretic reinforcement learning algorithms☆ Zhisheng Hu a , Minghui Zhu a,∗ , Ping Chen b , Peng Liu c

El problema planteado en este paper es que en un juego multijugador donde solo se tiene información de uno mismo, y el objetivo es ganar contra uno o más oponentes con una serie finita de acciones que se tiene de forma única, pero solo se conoce la propia. En este caso se debe plantear la forma maximizar la eficiencia de las acciones que se posea y tener la mejor combinación, para poder tener un mejor resultado que los contrincantes. Además dependiendo de las acciones que un individuo posea se le asigna una utilidad, que mide qué tan eficiente es su combinación de habilidades y cada habilidad por separado. Un individuo va evaluando su set de acciones según la interacción con otros individuos ya que desconoce las acciones que posee el contrincante.

El set de acciones para un individuo siempre es comparado según su utilidad con los sets anteriores de este mismo individuo. Al final de cada interacción del aprendizaje el algoritmo cuenta con un ratio de exploración que lo hará cambiar uno o más valores de manera random, y este ratio tiene la posibilidad de ir cambiando con el tiempo, con estos radios se hicieron múltiples pruebas para comprobar los mejores resultados. El hecho de que cada individuo solo conozca su propio set hace un gran ahorro en memoria para la máquina donde se ejecute.

Se aplicó un algoritmo de reforzamiento por aprendizaje. El algoritmo original utilizaba árboles que generaban sus ramas utilizando diferentes cadenas de markov, y esto concluía al generar el árbol y encontrar la rama que tuviera el maximo potencial estocastico. Sin embargo esto fueron tan eficientes al momento de implementarse con juegos suavemente aciclicos, refiriendose a que a las acciones que se realizarán podía haber una cantidad finita de respuestas válidas o correctas. Debido a lo anterior se implementó en su algoritmo de reforzamiento por aprendizaje la busca de un equilibrio de nash, ya que para cada juego suavemente cíclico existe como mínimo un equilibrio de nash.

En conclusión, se propuso un algoritmo de reforzamiento por aprendizaje que maximiza el potencial estocástico. Según las características del algoritmo se clasifica como un algoritmo voraz inteligente con detalles semejantes a simulated annealing por su ratio de exploración y variación del mismo.

Hu, Z., Zhu, M., Chen, P., & Liu, P. (2019). On convergence rates of game theoretic reinforcement learning algorithms. pg. 90–101.

Intelligent Level Generation for Super Mario using Interactive Evolutionary Computation

Ching-Ying Cheng Ya-Hung Chen Tsung-Che Chiang

El problema planteado en este paper es la generación de mapas de Mario Bros. que tengan una dificultad intermedia, que desde la perspectiva de los jugadores no sea ni muy complicado ni demasiado difícil pasar estos niveles. Los autores plantean que niveles muy complicados traen frustración a los jugadores y los niveles demasiado difíciles son aburridos.

Como solución se plantean una solución de Interactive Evolutionary Computation que puede representarse como un algoritmo genético con evaluación humana. Los autores proponen 5 posibles tipos de escenario y combinaciones de 2 de estas 5 opciones para la generación de un mapa que consta de 5 escenarios.

El primer grupo de mapas creado serían mapa generados de manera aleatoria. Seguido a esto, un grupo de testers jugarían los mapas k cantidad de veces y realizan observaciones en un formato entendible para el programa, con características como dificultado, puntuación como mapa, etc. Además de esto el algoritmo identifica valores según el desempeño de cada jugador como la cantidad de salto, el tiempo, monedas recolectadas etc. Con la información de la evaluación y la información recolectada de los mapa, califica según algoritmos cuales mapas han cumplido más las expectativas de los testers. El programa escoge los 2 mapas con mejor puntuación generada y hace dos combinaciones de estos, seguido se realizan mutaciones en 2 de los 5 escenarios de los mapas de manera aleatoria. Seguido a esto se volverían a evaluar los mapas, k veces los mapas generados, y otra cantidad los mapas ya existentes. En caso de que los nuevos mapas generados tengan malas evaluaciones habrá sido una generación defectuosa por ellos se creó data histórica para poder realizar una regresión modal.

Ching-Ying, C., Ya-Hung, C., & Tsung-Che Chiang, C. (2013). *Intelligent Level Generation for Super Mario using Interactive Evolutionary Computation*.

Designing virtual bots for optimizing strategy game groups

Entre los videojuegos actuales los más populares son los juegos multijugador, entre estos los juegos con grupos. Este paper trata de la inteligencia artificial que no controla a un único individuo, sino a un grupo, creando una estrategia para conseguir un objetivo. El problema principal tratado es el dilema de explotacion-explotacion (explotacion como aprovechamiento). En este paper se realizan bots para los juegos Unreal Tournament y Counter Strike, juegos muy populares de batalla de equipos, centrándose en el juego de capturar la bandera.

Debido a que el sistema consta de varios entes se usa un paradigma de sistema multiagente, centrándose en diseñar una estrategia con el conjunto de entes, teniendo como meta principal su objetivo sin importar cómo se llegue a cumplir el mismo. El algoritmo asigna roles a cada integrante del equipo de forma que cada uno tiene un comportamiento un poco diferente al resto.

Los bots de manera individual poseen 2 estados, exploración y reconocimiento, esto es la base del algoritmo. No pueden realizar ambas acciones al mismo tiempo y mientras estén en su estado de exploración no podrán comunicarse con otros bots. El estado de exploración hará que los bots se muevan entre los sectores del mapa en busca de la bandera enemiga para lo que utiliza cadenas de markov con las que se marca los movimiento a seguir entre áreas. Por otro lado el estado de reconocimiento los dejara quietos mientras analizan lo que les rodea. El intervalo en el que cambian entre un estado y otro varía en cada bot y el el principal factor cambiante en su comportamiento. Esta el la representación del dile de exploración-explotación.

Lo importante es la probabilidad de cada bot para encontrar la bandera a ser capturada para lo que se usa la posición inicial de donde comenzó el personaje, la cantidad de posiciones en el mapa en forma de malla y si índice de cambio entre estados. Debido a que cada bot tiene un índice de cambio y posicion inicial diferente cada uno tiene probabilidades diferentes de encontrar la bander. Cuando alguno de los bots encuentre el objetivo este mandará una señal a sus compañeros indicandoles hacia donde dirigirse, sin embargo cuando un bot está en estado de exploración este no podrá recibir el mensaje.

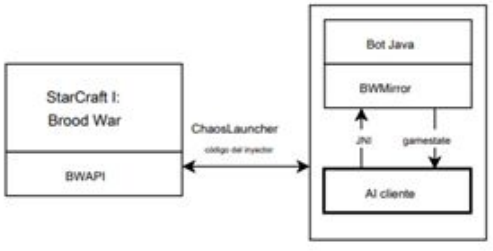
Para que el algoritmo sea eficiente en necesario crearlo usando algoritmos genéticos que hagan que pueda mejorar con la práctica, donde se representarán los datos de cada bot, se entrenan, y se mezclaran lo que tengan el mejor desempeño seguido de una mutación. Con esto se imita la evolución biológica.
Por último aplica la regresión lógica que es un modelo de clasificación probabilística con el que se predicen variables, un problema al momento de tener una falta de factores importantes para encontrar el objetivo.

Bedia, M. G., Castillo, L., Lopez, C., Seron, F., & Isaza, G. (2016). *Designing virtual bots for optimizing strategy-game groups. Neurocomputing, 172, 453–458.*

STAICRAFT: a Starcraft-based educational platform for Artificial Intelligence

La inteligencia artificial es muy común hoy en día. Se ha convertido en una de las características esenciales en el desarrollo de productos que los seres humanos utilizan a diario. Por otro lado, los videojuegos han conseguido el interés por la comunidad investigadora de IA, ya que ofrecen entornos en los cuales se pueden aplicar diferentes técnicas de IA. Especialmente estrategias en tiempo real. Este proyecto se centra en la toma de decisiones que deben de ser tomadas en intervalos cortos de tiempo con el fin de ganar la partida. En estos tipos de juego, algunos conocidos como Warcraft 3, World of Warcraft, Age of Empires y Starcraft. La contribución se llama STAICRAFT, una plataforma de IA educativa que le permite crear desde cero un bot de Starcraft que emplea diferentes técnicas de IA. Se puede trabajar en:

- Ruta-planificación mediante la búsqueda heurística.
- Árboles de comportamiento, árboles de decisión, maquinas de estados finitos, o de planificación automatizado con el fin de toma de decisiones



Analizando este paper nos ayuda a resolver nuestro problema con la toma de decisiones de nuestra IA para la toma de decisiones a través de árboles de comportamiento y máquinas de estado finito.

A pseudo-polynomial algorithm for mean payoff stochastic games with perfect information and few random positions

En este paper Se consideran de dos personas la suma de sus movimientos estocásticos dado por un digrafo G =(V.MI), con premios locales R: E -> Z, y tres tipos de posiciones negro V_B , blanco V_W , y aleatorio V_R, formando una partición de V. En este paper, se muestra que en los BWR-juegos con un número constante de posiciones aleatorias pueden ser resueltos en un tiempo polinomial. Más preciso, en cualquier BWR-juego con $|V_R| = O(1)$, un punto de silla en las estrategias estacionarias puros uniformemente óptimos se pueden encontrar en polinómicas veces en $| V_W| + |V_B|$, la máxima recompensa en locales absolutos, y el denominador común de las probabilidades de transición.

El resultado principal visto se puede considerar como solución parcial de este problema para el caso cuando el número de posiciones aleatorias es fijo. Para un Juego BWR denotamos por $n = | V_W| + |V_B| + |V_R|$ el número de posiciones, por k = máxima $\{|V_R| ,1\}$ el número de posiciones aleatorias 3 y suponga que los beneficios locales son integrales con el valor absoluto máximo R, y todas las probabilidades de transición son racionales con el denominador común D. El resultado principal de este documento es el siguiente:

Teoría 1: Un juego BWR G se puede resolver en $(nDk)^{O(k)}$ R · tiempo polylog R

Para juegos de pago de terminal estocástico con terminales t, se puede demostrar que v (G) ≤ t + 1. Por lo tanto, el Teorema 1 extiende la trazabilidad de parámetros fijos de los juegos estocásticos simples con respecto al número de posiciones aleatorias y la solvencia pseudo-polinomial de los juegos de pago medio determinista Es importante tener en cuenta que el resultado anterior requiere una nueva técnica para resolver los juegos BWR. La aproximación complejidad reclamada.
El enfoque para demostrar el teorema 1 se basa en gran medida en reducir el cálculo de estrategias uniformemente óptimas para un juego BWR general en el caso de juegos BWR ergódicos, es decir, aquellos en los que el valor (óptimo) no depende de la posición inicial, y muestra que Este caso especial se puede resolver en tiempo (nDk)O (k) R · tiempo polylog R. Nuestro algoritmo para el caso ergódico se basa en transformaciones potenciales que cambian la recompensa local sin cambiar la forma normal del juego. Nos gustaría llevar las recompensas locales a tal forma que cada movimiento localmente óptimo también sea globalmente óptimo. Comenzando desde cero potenciales, el algoritmo sigue seleccionando un subconjunto de posiciones y reduciendo sus potenciales hasta las recompensas óptimas localmente (máximo para blanco, mínimo para negro y promedio para Aleatorio) en diferentes posiciones se vuelven lo suficientemente cercanas entre sí, o se obtiene una prueba de no ergodicidad en forma de una cierta partición de las posiciones. En más detalles, el algoritmo procede en fases. En una fase, dividimos el rango de recompensas locales óptimas actuales en cuatro regiones, y seguimos reduciendo el potencial de algunas de las posiciones para que ninguna posición pueda escapar de las dos regiones intermedias. La fase finaliza cuando la región superior o inferior se vacía, o se encuentra

una prueba de no ergodicidad. Tenga en cuenta que se sugirió un algoritmo para juegos BW, también basado en posibles reducciones. Por lo tanto, las posiciones aleatorias con arcos que salen del nivel medio pueden tener que escapar de ese nivel después de la reducción potencial. En nuestro algoritmo, superar esta dificultad relajando el nivel medio a un conjunto de niveles.

Dado un juego BWR G = (G, p, r), introducimos un mapeo x: V → R, cuyos valores x (v) se llamarán potenciales, y defina la función de recompensa transformada rx: E → R como:

rx (v, u) = r (v, u) + x (v) - x (u), donde (v, u) ∈ E.

No es difícil verificar que las dos formas normales del juego Gx obtenido y el juego G original son las mismas y, por lo tanto, los juegos Gx y G son equivalentes. En particular, sus estrategias óptimas (estacionarias puras) coinciden, y sus funciones de valor también coinciden: μGx = μG. Este resultado fue extendido

para la clase más general de juegos BWR en en el juego transformado, se da el valor de equilibrio μG (v) = μGx (v) simplemente por la recompensa local máxima para v ∈ V W, la recompensa local mínima para v ∈ V B y la recompensa local promedio para v ∈ V R. En este caso decimos que el juego transformado está en forma canónica. Para definir esto más formalmente, usemos la siguiente notación a lo largo de esta sección: Funciones dadas f: E → R yg: V → R, definimos las funciones $M^{-}[f]$, $M^{-}[g]$: V → R.

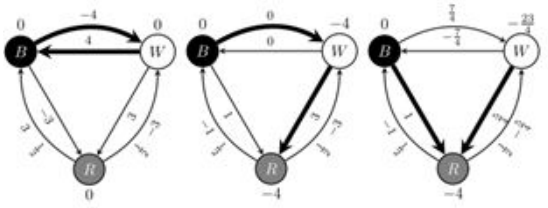
$$\bar{M}[f](v) = \begin{cases} \max_{u|(v,u)\in E} f(v,u), & \text{for } v \in V_W, \\ \min_{u|(v,u)\in E} f(v,u), & \text{for } v \in V_B, \\ \sum_{u|(v,u)\in E} p(v,u)\, f(v,u), & \text{for } v \in V_R. \end{cases}$$

$$\bar{M}[g](v) = \begin{cases} \max_{u|(v,u)\in E} g(u), & \text{for } v \in V_W, \\ \min_{u|(v,u)\in E} g(u), & \text{for } v \in V_B, \\ \sum_{u|(v,u)\in E} p(v,u)\, g(u), & \text{for } v \in V_R. \end{cases}$$

Ergodic BWR-games

Dado un juego BWR G = (G, p, r), calculemos m (v) = M^{-} [r] (v) para todos v ∈ V. El algoritmo procede en fases. En cada fase cambiamos iterativamente los potenciales de tal manera que el rango de m se reduce en un factor de 3/4, o descubrimos una descomposición contra ergódica. La función de recompensa local inicial en la siguiente fase se obtiene transformando las recompensas iniciales por la función potencial obtenida en la fase actual. Una vez que el rango se vuelve lo suficientemente pequeño, dejamos de repetir estas fases, lo que nos permite concluir que el juego es ergódico. Ahora describimos informalmente los pasos en una fase. El paso general de una fase, llamado bombeo, consiste en reducir todos los potenciales de las posiciones con valores m en la mitad superior del rango por la misma constante δ; Decimos en este caso que esas posiciones están impulsadas. En cada paso, elegiremos δ como la constante más grande, de modo que ningún valor m abandone la mitad media. A continuación, damos una descripción más formal de la fase h = 0, 1, ... Denotemos por xh el potencial al final de la fase h - 1, donde establecemos x0 = 0. Dada una función f: V → R , denotemos por f + y f - sus valores máximos y mínimos respectivamente. En todo momento, denotaremos por [m] = [m−, m +] y [r] = [r−, r +] el rango de funciones myr, respectivamente, y dejaremos que M = m + - m− y R = r + - r−. Dada una función potencial x: V → R, denotaremos por mx, Mx, etc., las funciones m, M, etc., con r reemplazada por la recompensa transformada rx en las definiciones anteriores. Dado un subconjunto I ⊆ [m], deje Vx (I) = {v ∈ V | mx (v) ∈ I} ⊆ V. En el siguiente algoritmo, establecer siempre será un intervalo cerrado o semicerrado dentro de [m] (y no dentro de [mx]). Por conveniencia, escribiremos (·) xh como (·) h, donde (·) podría ser m, r, r +, etc. (por ejemplo, m− h = m− xh, m + h = m + xh).

A continuación, se muestra un ejemplo simple que ilustra cómo funciona el algoritmo de bombeo. Las posiciones W y B están controladas por el maximizado y el minimizador, respectivamente. La posición R es una posición aleatoria con probabilidad 1/2 en ambos arcos salientes.



La aplicación de este paper en nuestro trabajo seria con las pocas posiciones que tenemos que analizar, entonces podemos predecir cuando un personaje intente esquivar o moverse a través de estas fórmulas.
G. Vladimir, B. Endre, E. Khaled y M. Kazuhisa, «A pseudo-polynomial algorithm for mean payoff stochastic games with perfect information and few random positions,» 2019.

Finding Comfortable Settings of Snake Game Using Game Refinement Measurement

En este paper la problemática es el progreso del juego de snake el objetivo de este juego es recoger frutos tanto como sea posible. Un jugador pierde cuando la serpiente se estrella contra la pared o se estrella en sí. Para la solución de este problema es hacer un refinamiento la cual es basada en el concepto de progreso del juego y la información del juego el progreso. Tener información completa del progreso del juego, es decir, después de su conclusión, el progreso del juego x (t) será dado como una función lineal del tiempo t con $0< t < t_k$ y $0<x(t) < x(t_k)$ muestra una ecuación.

$$x(t) = \frac{x(t_k)}{t_k} t$$

Sin embargo, la información del juego el progreso dada por la ecuación. Es desconocida durante el periodo en el juego. La presencia de incertidumbre durante el juego, a menudo hasta que el fi momentos nales de un juego, razonablemente renders progreso del juego como exponencial. Por lo tanto, un modelo realista del progreso información del juego se da por la ecuación.

$$x(t) = x(t_k)\left(\frac{t}{t_k}\right)^n$$

aquí norte representa un parámetro constante que se da en base a la perspectiva de un observador del juego considerado. A continuación, la aceleración del progreso información del juego se obtiene mediante la derivación de la ecuación . Resolverlo en $t = t_k$ tenemos la ecuación.

$$x''(t_k) = \frac{x(t_k)}{(t_k)^n} t^{n-2} n(n-1) = \frac{x(t_k)}{(t_k)^2} n(n-1)$$

En este paper se usarár para incrementar la dificultad del bot en cada periodo.

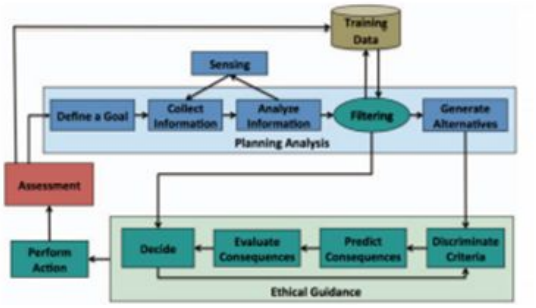
P. Anunpattana, H. Iida y C. Panumate, «Finding Comfortable Settings of Snake Game Using Game Refinement,» Japan, 2016.

A Decision Making Model for Ethical (Ro)bots

En este paper nos habla sobre los comportamientos exhibidos por las máquinas, ya sean deterministas, semiautónomos o totalmente autónomos, a menudo tienen consecuencias para el bienestar de las personas y, por lo tanto, tienen una dimensión ética. Por lo tanto, la relevancia ética de la acción de una máquina que surge de haber tomado una decisión, independientemente del proceso real de toma de decisiones, es aún mayor porque la máquina seleccionó una acción entre los cursos de acción alternativos. Si una máquina toma un curso de acción que tiene consecuencias perjudiciales y es posible que esa máquina haya elegido un curso de acción diferente, especialmente uno que podría no podría tener estas consecuencias. Según la MIT los robots se deben de ver como algoritmos incorporados pero los algoritmos solo son bots no robots, los procesos de toma de decisiones y sus características éticas se aplican para ambos. Además, característica clave en un (ro)bot es la capacidad autónoma de toma de decisiones. Otra característica esencial de un robot es si puede responder adecuadamente a una amplia variedad de situaciones. Para un robot Hay 3 enfoques principales para el gobernar la ética y el comportamiento autónomo:

1. Puede tener innumerables leyes que controlan de sus acciones.
2. Puede calcular las consecuencias de sus acciones y evaluar esas consecuencias acuerdo a principios utilitarias.
3. Puede diseñar con un sistema de aprendizaje con la capacidad de adquirir un comportamiento ético a partir de experiencias(Un sistema de virtudes éticas).

Para permitir que un robot tome decisiones éticas a partir de experiencias habría que requerir un método de maching learnig. Existen 3 tipos de maching learning el aprendizaje supervisado, no supervisado y reforzado. En el supervisado enseñas a la máquina a correlacionar de entrada y salida usando las etiquetas. Un ejemplo para que se pueda realizar el desempeño se tienen que etiquetar con nuevas entradas y salidas. El no supervisado obtiene solamente datos de entrada sin supervisión y aprende que una entrada tiene patrones asociados con una salida. El reforzado aprende de un entorno dinámico y puede archivar su desempeño en una secuencia de acciones. En este paper se presenta el modelo de toma de decisiones en el cual hay 10 etapas etapas en el proceso de toma de decisiones de un dispositivo robótico: se dividen en 2 grupos: el grupo de análisis de planificación y el grupo de Orientación ética. La función principal del análisis de planificación es determinar si una elección de acción requiere un análisis ético individual o situado o si puede atajar una decisión. Si requiere un análisis ético, entonces entran en juego las cuatro etapas en el grupo de orientación ética. Definir la meta, recolectar información, analizar información, filtrar y generar alternativas.



1. Definiendo una meta: Un robot multifuncional podría tener objetivos programables y un robot totalmente autónomo podría definir sus propios objetivos.
2. Recopilación de información: Los sistemas robóticos suelen adquirir información de sensores, redes, telemetría y humanos. La información que necesitan se obtiene del sonido, la luz, el contacto físico y otros métodos de entrada.
3. Análisis de la información: Los sistemas robóticos luego analizan esta información de entrada para crear representaciones del mundo real de los objetos que necesitan razonar, sintetizar e integrar estos datos para hacer predicciones y ejecutar tareas de planificación.
4. Filtrado: el robot analiza situaciones similares ya experimentadas y las vuelve a usar esto solo este proceso ocurre en máquinas que tiene un aprendizaje experimental.
5. Generación de alternativas: Si la etapa de filtrado indica que es necesario pasar por el proceso de tomar una decisión, el robot debe producir una lista de acciones alternativas de las cuales seleccionar
6. Criterios de discriminación: Dado un conjunto de acciones alternativas entre las cuales elegir, el sistema robótico ahora debe evaluar las alternativas generadas de acuerdo con las reglas deontológicas.
7. Predicciones de consecuencias: Si la acción que se debe seleccionar debe evaluarse de acuerdo con los principios consecuencialistas (etapa 8), se deberán calcular las implicaciones de cada alternativa
8. Evaluación de consecuencias: Las consecuencias de cada acción alternativa deben ser evaluadas
9. Tomar una decisión: Después de evaluar las consecuencias de cada acción alternativa, el robot seleccionará el acto que sea óptimo con respecto a la meta u objetivo de la tarea general y óptimo con respecto a la ética.
10. Realización de acciones: Esta etapa es la que lleva al compromiso con la acción.

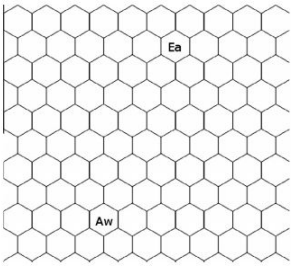
Este modelo de relación se puede aplicar a nuestro trabajo, debido a que un sistema inteligente debe de tener un modelo de toma de decisiones para poder tomar una elección correcta según un régimen ético.

F. Alaieri y A. Vellino, «A Decision Making Model for Ethical (Ro)bots,» Canada , 2017.

Fuzzy Tactics: A scripting game that leverages fuzzy logic as an engaging game mechanic

El presente paper da a conocer la pasada problemática de los centros para el desarrollo de videojuegos. Muchos son los problemas como por ejemplo: las animaciones de los personajes jugables como no jugables, implementar un gameplay adaptativo (juego de calidad), etc. Sin embargo, el problema más crítico según (Pirovano & Luca Lanzi, 2014) es el uso de inteligencia artificial para dar vida y más interacción a un videojuego. Se habla concisamente del juego Black and White, juego en el que se juega a ser dios (Wikipedia, 2019), del cual se extrajeron ideas originales para la creación de un nuevo juego basado en los principios de la lógica difusa.

Se planteó la creación de un video juego llamado Fuzzy Tactics que es una fiel muestra de como es que el algoritmo de lógica difusa trabaja. Posicionado en un tablero hexagonal, se deberá derrotar a los enemigos en un formato de juego por turnos en donde la E representa a los enemigos y la A representa a los aliados:



(Pirovano & Luca Lanzi, 2014)

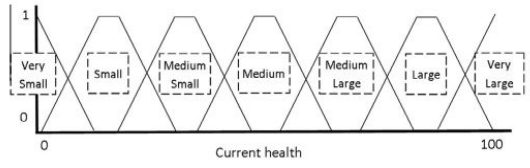
Algunas de las reglas generadas para Fuzzy Tactics según este escenario son:

- IF distance IS very-small THEN Ea holds
- IF distance IS small THEN Ea attacks
- IF distance IS medium-small THEN Ea attacks
- IF distance IS small THEN Ea attacks
- IF distance IS medium THEN Ea attacks
- IF distance IS medium-large THEN Ea attacks
- IF distance IS large THEN Ea attacks
- IF distance IS very-large THEN Ea attacks

Se puede apreciar como el enemigo podría eliminar a nuestra unidad. Por lo que se generan las siguientes reglas en respuesta a esta situación.

- IF distance IS very-small THEN Ea holds
- IF distance IS small THEN Ea attacks
- IF distance IS medium-small THEN Ea attacks
- IF distance IS small THEN Ea attacks
- IF distance IS medium THEN Ea attacks
- IF distance IS medium-large THEN Ea attacks
- IF distance IS large THEN Ea attacks
- IF distance IS very-large THEN Ea attacks
- IF distance is very-small THEN Aw attacks
- IF distance is small THEN Aw attacks
- IF distance IS medium-small THEN Aw moves
- IF distance IS small THEN Aw moves
- IF distance IS medium THEN Aw moves
- IF distance IS medium-large THEN Aw moves
- IF distance IS large THEN Aw moves
- IF distance IS very-large THEN Aw attacks

Básicamente, la inteligencia artificial del juego verifica sus reglas contra el estado actual de la batalla, las cuales tan representadas en las variables difusas a las que se pueden acceder. Todos los personajes pueden acceder a todas las variables difusas establecidas hasta esos momentos. Estas variables están definidas en conjuntos difusos representado en trapecios como se pueden apreciar en la siguiente imagen:



(Pirovano & Luca Lanzi, 2014)

La salida del motor de inferencia difusa determina, por lo tanto, la acción que toma la unidad (huir, atacar, moverse, etc.) y a qué unidad apuntar (enemigo, aliado, sí mismo) en función del estado de batalla actual.

Para concluir, Fuzzy tactics hizo uso de lógica difusa en la lógica de un juego de estrategia. Esto funciona como un ejemplo de como sistemas expertos pueden ser utilizados dentro del diseño de videojuegos.

Pirovano, M., & Lanzi, P. L. (2014). *Fuzzy Tactics: A scripting game that leverages fuzzy logic as an engaging.* Milano, Italia: Elsevier Ltd.

Designing virtual bots for optimizing strategy-game groups

En los últimos años la industria de los videojuegos ha crecido exponencialmente. Las características que hacen más atractivas a un juego es el reto que tiene le jugador para competir contra la inteligencia artificial del mismo sistema experto. Es por ello que muchas empresas no saben cómo optimizar la inteligencia artificial en el desarrollo de sus juegos para brindar una experiencia mucho más equilibrada en el dilema de "exploración vs. explotación" (Bedia, Castillo, Lopez, & Isaza, 2015)

Es por estas razones que el equipo de creador de este paper se propuso la creación de un juego que incluyera la inteligencia artificial retratada en Unreal tournament 2004, primer video juego de disparos en primera persona, en la que lo enemigos pueden ser representados por la misma inteligencia artificial del juego o por otro jugador en una red entre computadoras.

Se hará uso de algoritmo genético y la cadena de Markov para la creación de un juego con la temática de captura la bandera. Definiendo el algoritmo genético en el modelo propuesto:

- Fitness Function: Fitness = min(Ttotal)

- Población: todos lo miembros de los equipos, conformados por 3 jugadores cada equipo.

- Configuraciones experimentales:

- El algoritmo se ejecutara 2900 veces

- El procedimiento de selección es la selección de truncamiento en el 50% de la población

- La mutación se realiza agregando términos gaussianos a los pesos.

- El cruce se realiza promediando los valores de peso de ambos padres con una probabilidad de 10%

De acuerdo al modelo de markovian podemos obtener la probabilidad que tiene cada bot para obtener la bandera.

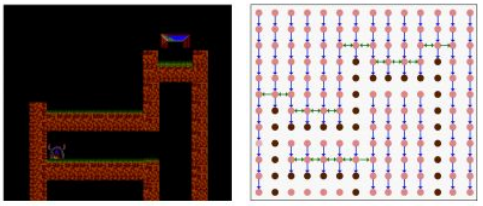
Bedia, M., Castillo, L., Lopez, F. S., & Isaza, G. (2015). *Designing virtual bots for optimizing strategy-game.* Neurocomputing.

AN EMPIRICAL STUDY ON COLLECTIVE INTELLIGENCE ALGORITHMS FOR VIDEO GAMES PROBLEM-SOLVING

La tecnología bioinspirada se hara un estudio a una serie de tecnologías utilizadas en Starcraft, Phisical traveller salesman Problem, entre otros. En este paper se propone remodelar el juego The Lemmings Game en un problema de satisfacción de restricciones.

Para trabajar este proyecto se escogieron dos algoritmos, el Genetic algorithm y el algoritmo Ant Colony Optimization. Estos algoritmos presentan características similares, ya que ambos trabajan con una población sobre la cual se aplicaran los algoritmos así también se deben elegir dos sobre otros tomando en cuenta la Fitnes Function definida.

La técnica definida en el documento se basa en modelar el juego The lemmings Game como un problema de satisfacción de restricciones donde las variables denotadas como X son las diferentes posiciones de los niveles y los posibles valores D son las habilidades que los lemmings pueden ejecutar en cada posición dentro de un grafo. En primer lugar el mapa está compuesto de dos dimensiones de la cual se puede extraer un grafo como el siguiente.



(Antonio, 2015)

La finalidad del uso de genéticos es para llegar a automatizar la tarea de los lemmings de llegar a su destino. El valor resultante de la principal ecuación representada debe ser maximizado. La primera ecuación representa la cantidad de lemmings que hay en el mapa. La segunda toma en cuenta el tiempo total que se le da y el tiempo empleado por el lemming para resolver el nivel. La tercera ecuación se usa para favorecer aquellos caminos que usan menos acciones o menos habilidades.

$$
\begin{aligned}
S(Ind) &= TotalLemmings - BlockersUsed(Ind) - ExplodedLemmings(Ind) \\
T(Ind) &= MaxTime - Time(Ind) * S(Ind) \\
A(Ind) &= TotalActionsAvailable - ActionUsed(Ind) \\
F(Ind) &= \frac{T(Ind) + A(Ind) + S(Ind)}{MaxTime + TotalActionsAvailable + TotalLemmings}
\end{aligned}
$$

(Antonio, 2015)

Antonio, G. P. (2015). *AN EMPIRICAL STUDY ON COLLECTIVE INTELLIGENCE ALGORITHMS FOR VIDEO GAMES PROBLEM-SOLVING.* España: Computing and Informatics.