Proyecto:
Board Game AI

Sistemas Inteligentes (curso 21/22)

Aram Pérez Dios (alu0101244488@ull.edu.es)

enlace al repositorio del proyecto en github:
https://github.com/alu0101244488/Board-Game-AI

Introducción

Propósito, objetivos y justificación del proyecto

El proyecto implementado en este caso se trata de un agente simple que actúa como contrincante en algún tipo de juego de mesa, en este caso el ajedrez (Board Game AI).

El propósito de este proyecto es el de la creación de un agente desde cero y su implementación en un juego. Para este proyecto el ajedrez es un elección perfecta ya que el juego contiene un conjunto de normas predefinido, esto es, una base sobre la que partir y además proporciona suficiente dificultad y opciones como para que sea necesario la generación de algún tipo de agente capaz de interactuar o en este caso responder a las jugadas del jugador.

Como se ha explicado anteriormente, el ajedrez se sustenta bajo un grupo de reglas bien conocido. Esto es exportable a apartados como el campo de juego, ya que no se trata de un campo continuo, sino de uno discreto, de un tablero con cuadrículas. Por esto la creación de las reglas y del agente se limita mucho mejor que en otros tipos de juegos. La interacción del agente se limita así a la elección de un movimiento válido en el tablero y se le da al agente un único propósito.

Descripción del Proyecto

El proyecto consiste en la implementación de un agente en un entorno controlado en el cual su única tarea es la toma de decisiones que en este caso se traduce en la elección de una ficha de ajedrez para mover y una posición donde moverla. Para esta implementación se crea desde cero al agente.

Se trata por lo tanto de implementar alguna estrategia por la cual el agente con únicamente la información a su alcance pueda tomar una decisión. El ajedrez al tratarse de un juego de información perfecta (juego en el que todos los agentes conocen toda la información del juego) facilita enormemente las posibilidades. Pero esto a su vez es un problema ya que el ajedrez cuenta con una gran cantidad de posibles movimientos y estados posibles por lo que la elección de un movimiento u otro no es trivial.

Proyectos similares

Existe gran variedad de fuentes sobre lo propuesto como proyecto. El inconveniente es que gran parte de estos se basan en los mismos aspectos por lo que esta gran variedad de fuentes se queda limitada rápidamente. Además existen pocos, pero que son grandemente conocidos proyectos, pero que se alejan del alcance de este proyecto ya que abordan cuestiones más complejas. Así trabajos similares encontrados encontrados serían los siguientes:

- **DeepBlue**: Programa de inteligencia artificial de IBM que fue capaz de ganar a Kasparov
- **AlphaZero**: Inteligencia Artificial de DeepMind que es capaz. Aunque muy diferente a lo que se trata de implementar en este proyecto, es bastante conocida.
- **Recursos web sobre la implementación de IA en el ajedrez**: Existe una gran cantidad de recursos (Blogs, posts, tutoriales, etc) en la web sobre la implementación de distintas estrategias aunque no exista una verdadera variedad sobre diferentes aproximaciones a este problema

Recursos a usar

Para implementar el proyecto se ha decidido emplear ciertas herramientas ya que se cree que estas son las adecuadas para el proyecto. Estas son las siguientes:

- **Unity**: Entorno para la implementación del proyecto ya que cuenta con herramientas para facilitar todo lo relacionado con la visualización y los elementos de UI con los que el usuario interactúa.
- **Github**: herramienta de control de versiones la cual se usará no solo para mantener un control del proyecto sino para almacenar toda la documentación que se genere.
- **ProjectLibre**: Software para la gestión de proyectos que ha permitido entre otras cosas generar el cronograma del proyecto.

Tecnologías de IA para implementar el proyecto

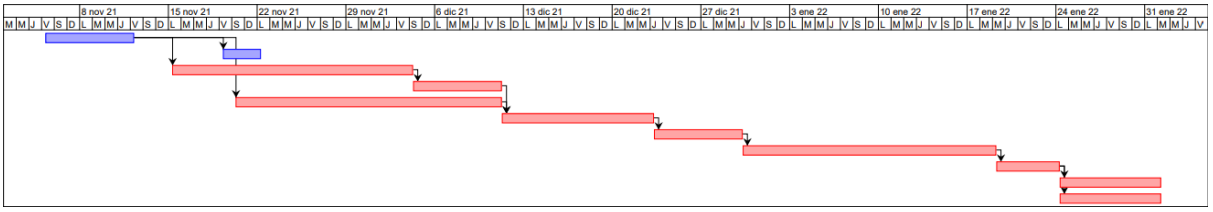
Como ya se ha explicado, el propósito es la implementación de un agente desde cero por lo que la acción a tomar es la implementación de algoritmos que representarán las distintas estrategias del agente. En este caso, este algoritmo es el minmax el cual se amolda perfectamente a las características del proyecto.

Desarrollo de la aplicación

Lista de tareas y cronograma

Lista de tareas y diagrama de Gantt del proyecto:

	Nombre	Duracion	Inicio	Terminado	Predecesores
1	Creación de la ACP	7 days	5/11/21 8:30	12/11/21 8:30	
2	Elaboración del cronograma	3 days	19/11/21 8:30	22/11/21 8:30	1
3	Implementación del juego en el entorno unity	19 days	15/11/21 8:30	4/12/21 8:30	1
4	Implementación de elementos básicos de la UI	7 days	4/12/21 8:30	11/12/21 8:30	3
5	Investigación para la creación del agente	21 days	20/11/21 8:30	11/12/21 8:30	1
6	Implementación básica del agente en el entorno	12 days	11/12/21 8:30	23/12/21 8:30	5;4
7	Seguir investigando nuevas estrategias para implementar	7 days	23/12/21 8:30	30/12/21 8:30	6
8	Implementación de comportamientos más complejos	20 days	30/12/21 8:30	19/01/22 8:30	7
9	Añadir más elementos a la interfaz para alterar al agente	5 days	19/01/22 8:30	24/01/22 8:30	8
10	Generación de informe final	8 days	24/01/22 8:30	1/02/22 8:30	9
11	Elaboración de la presentación final	8 days	24/01/22 8:30	1/02/22 8:30	9



El cronograma se divide en dos partes. Una en la que se busca implementar la interfaz y todos los aspectos relacionados con la aplicación en sí. Y una segunda parte en lo que se tratará es de crear al agente y las estrategias que tomará (algunas de estas tareas, como lo es la investigación para la implementación del agente se podrá realizar de forma paralela a el resto). La fecha que se añade a continuación indica la fecha estimada de finalización de la actividad:

- Implementación del juego del ajedrez (04/12/2021)
- Creación de la interfaz de usuario (11/12/2021)
- Investigación de para implementar al agente (11/12/2021)
- Introducir estrategias para el agente (23/12/2021)
- Investigación de estrategias más complejas (30/12/2021)
- Introducir estrategias más complejas para el agente (19/12/2021)
- Creación del informe final del entregable (01/02/2021)
- Creación de las diapositivas para la presentación del proyecto (01/02/2022)

Objetivos medibles del proyecto y criterio de éxito asociados

El éxito del proyecto se mide en gran medida por el comportamiento del agente por lo que se toma como parámetro para medir el éxito los siguientes factores:

- El agente ha de cumplir con los requisitos de alto nivel. Esto es: la implementación de alguna estrategia que emplea el agente para realizar su elección de movimiento.
- El agente es capaz de actuar de forma adecuada en respuesta al entorno actual y actuará acorde a la estrategia utilizada.
- La modificación de los parámetros en la estrategia del agente resulta en un cambio efectivo en el comportamiento del agente.
- El agente es capaz de ganar en una partida al usuario.

Problemas encontrados

Durante la realización del proyecto se han encontrado varios problemas que aunque de baja gravedad, afectan al desarrollo del proyecto. Aquí se encuentran todos los registrados todos los problemas que se han encontrado durante la realización del proyecto:

- Elegir la plataforma para la implementación del proyecto: al principio del proyecto se dudó entre dos posibles opciones, el empleo de librerías o la utilización de la plataforma de unity. Sin haber afectado gravemente al progreso del trabajo, se considera un problema.

- Implementar el juego en Unity: Encontrar una forma de implementar el juego y que al mismo tiempo pueda estar preparado para proveer toda la información necesaria al agente es uno de los problemas actuales. Al mismo tiempo la implementación del juego lleva más tiempo del que se le había asignado en el cronograma por lo que esto resulta en retrasos en el resto del proyecto.

- Elección de la primera estrategia del agente: existiendo variedad de estrategias bien documentadas en la web se da la situación de que algunas de estas pueden ser más complicadas que otras y dado que lo que se pretende es buscar una forma sencilla de comprobar el buen funcionamiento del agente, la elección de la estrategia se complica.

- Implementación del agente en el entorno: una vez ya implementado el juego en sí, el siguiente paso es implementar al agente. En primera instancia la implementación del agente para que reciba toda la información necesaria puede ser más difícil de implementar de lo que parecía al comienzo del proyecto. Al mismo tiempo, como se dijo antes, dependiendo de la primera estrategia que se implemente puede ser más o menos complicada su implementación.

- Implementación del juego en Unity: uno de los problemas de Unity es la ejecución de los distintos scripts que se usan ya que es el propio motor es el que decide el orden en el que se ejecutan. Esto resulta en que un script puede estar accediendo a un valor de otro script que todavía no se ha inicializado. Unity para esto provee varias funciones en su API como lo es Awake(), que asegura su ejecución antes de que comience el juego. Sin embargo este no ha resultado efectivo por lo que se ha tenido que buscar soluciones alternativas a este problema ya solucionado.

Descripción del funcionamiento

Funcionamiento del agente

El agente se comporta como un agente de un sistema pizarra ya que aunque sea el único agente, este tiene que obtener toda la información del tablero y esta es su única fuente de recopilación de datos. Además el agente no cuenta con ningún tipo de aprendizaje. Este genera por fuerza todos los estados posibles y no es capaz de generalizar o realizar ninguna inferencia de los movimientos realizados tanto por el propio agente como por el jugador. Su funcionamiento es el siguiente:

Primero genera todos los posibles movimientos (profundidad 1) ya que está obligado a mover alguna ficha a alguna posición válida (si eligiéramos una profundidad de 0) el agente generaría aún así todos los movimientos posibles y escogería uno aleatoriamente. Podemos elegir aquellos movimientos con mayor valor objetivo y el agente puede escoger o el último movimiento con mayor valor de la función, el primero, o de entre todos los mejores, elegir uno aleatorio.

En cuanto lo que se pretende es intentar deducir jugadas posteriores (profundidades del árbol de 2 o más), se generar todas las jugadas posibles intercalando las del jugador y las del agente. Una vez se llega a la profundidad deseada, se calcula el valor de la posición del tablero alcanzada (para ello a cada pieza se le asigna un valor) y se propaga el valor a los nodos padre dependiendo de si se trata de una jugada del jugador o una del agente. Y al final se escoge aquella jugada la cual corresponde con la mayor profundidad.

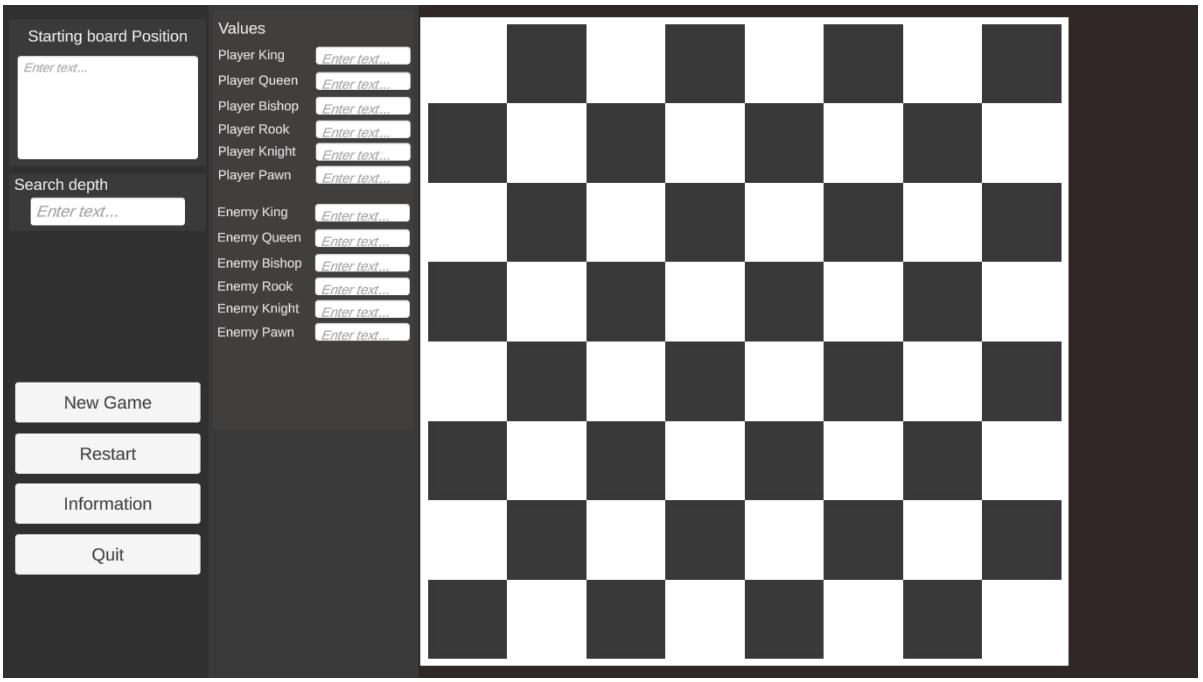
Funcionamiento del programa

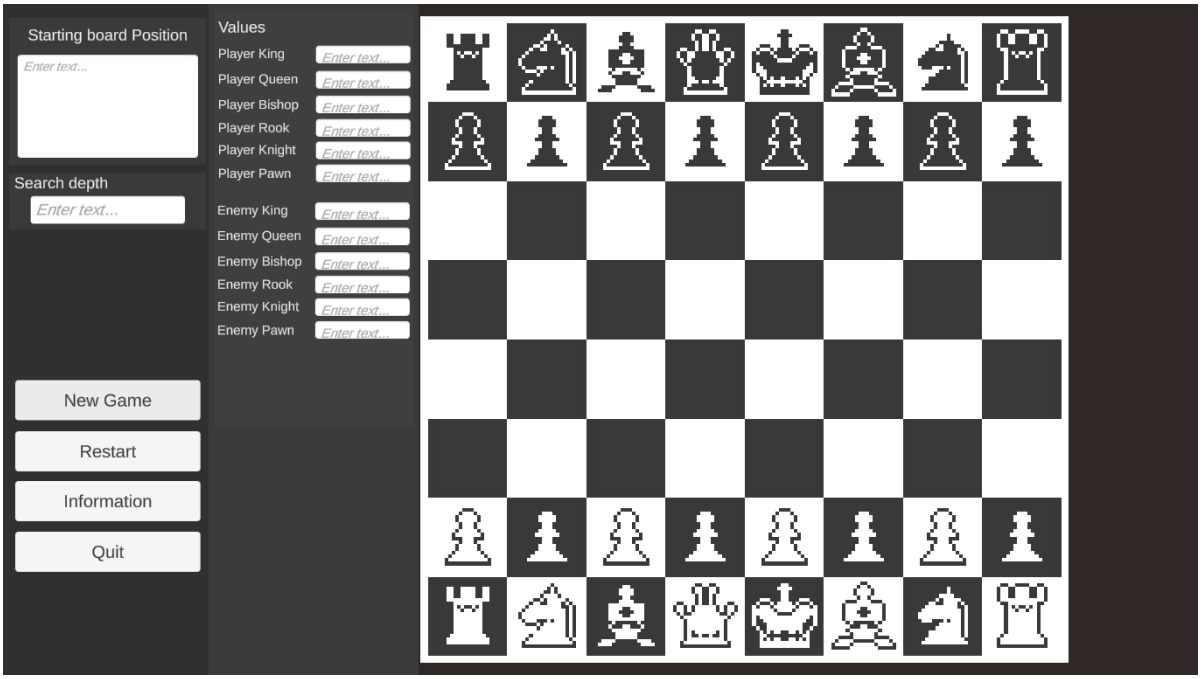
El programa permite interactuar de una forma sencilla. Para ello se proporcionan diferentes campos y botones para modificar el estado inicial del tablero o poder modificar el comportamiento del agente durante el juego. Se nos permite modificar varios parámetros del agente. Estos son:

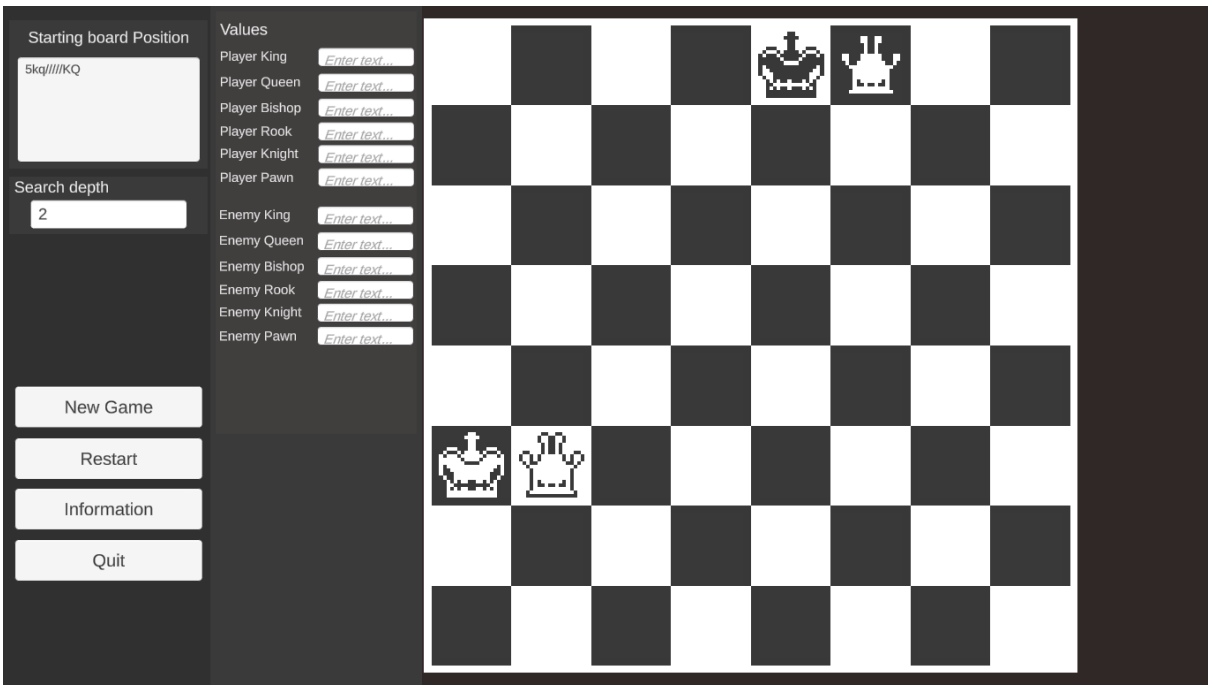
- **Profundidad de búsqueda**
- **Valor de las piezas**

Otros parámetros que se pueden modificar es la posición inicial del tablero, esto es, las piezas que se encuentran en el tablero al inicio de la partida.

Imágenes sobre el programa en funcionamiento:







Enlace al repositorio del proyecto en github: https://github.com/alu0101244488/Board-Game-AI