



Universidad Nacional
ARTURO JAURETCHE

Trabajo Práctico Final

Complejidad Temporal, Estructura de datos y Algoritmos
Primer cuatrimestre de 2020
Comisión 05 - Prof. Leonardo Amet

Alumno:	Almada, Federico
Número de legajo:	27787
Email:	federico.almada1998@gmail.com
DNI:	41.136.004

Índice

1. Introducción	2
1.1. Objetivos generales	2
2. Descripción del juego	3
3. Algoritmo MiniMax	3
4. Diagrama de clases UML	3
5. Problemas encontrados	5
5.1. Soluciones de problemas	5
6. Interfaz del juego	5
7. Consultas	6
7.1. Consulta A	6
7.2. Consulta B	6
7.3. Consulta C	6
8. Mejoras	7
9. Conclusiones	7
10. Bibliografía	8

1. Introducción

El presente informe es para el Trabajo Práctico Final de la materia Complejidad Temporal, Estructura de datos y Algoritmos que consiste en desarrollar un juego de cartas entre dos contrincantes (el usuario y la computadora). El objetivo principal del mismo es programar la Inteligencia Artificial (IA) con la cual la computadora va a elegir sus cartas, esto se hará mediante un árbol MiniMax.

1.1. Objetivos generales

- Implementar el armado del árbol de decisiones.
- Desarrollar la estrategia con la que la IA tomará decisiones favorables.
- Desarrollar la metodología con la que la IA irá descartando sus cartas.
- Presentar el desarrollo del trabajo así como también los problemas hallados y sus soluciones.

Por otro lado, la clase Consulta es la que tiene todos los métodos necesarios con los cuales el usuario en cada momento del juego podrá solicitar una funcionalidad para ver el desarrollo del árbol en cualquier jugada del árbol de soluciones.

5. Problemas encontrados

Durante el desarrollo del juego surgieron distintos problemas, por un lado, el mayor de los problemas fue la hora de la creación del árbol de posibilidades, en el método Inicializar de la clase ComputerPlayer. Por otro lado, en el mismo método, también ocurrieron algunos problemas en relación a la asignación de la función heurística.

5.1. Soluciones de problemas

Para solucionar el armado del árbol, lo que se hizo fue crear un algoritmo recursivo que permita armar el árbol lo más eficiente posible, de tal forma que se pare con el armado del árbol cuando se llegue a un caso base. El caso es cuando se llega a un nodo o carta, donde el límite del juego sea menos a 0. Por otro lado, en relación a la asignación de la función heurística, a medida que el árbol de decisiones vuelve de la recursión, se realizó un booleano que, dependiendo el valor heurístico de los nodos hijos de cada nodo raíz, se maximiza o minimiza su valor de función heurística. Cuando sea el turno de la IA se maximiza el resultado, y cuando sea el turno de la computadora se minimiza su resultado.

6. Interfaz del juego

```
LIMITE: el límite actual es de: 31
Turno Humano
Cartas disponibles:
(6) (5) (9) (8) (4) (2)
El humano está evaluando su mejor opción...

Ingrese una carta ó presione ENTER para abrir las consultas: 4
El humano ha descartado la carta: 4
-----
LIMITE: el límite actual es de: 27
Turno IA
Cartas disponibles:
(1, -1) (3, +1) (7, +1) (10, -1) (11, +1) (12, +1)
La Inteligencia Artificial está evaluando su mejor jugada..
La Inteligencia Artificial ha descartado la carta:7
-----
```

Como se pudo observar, primero se asigna un límite al azar, entre un rango determinado por el juego. El humano va a tener en cada jugada la posibilidad de hacer una consulta. Cuando el humano descarta su carta, se invierte el turno y la IA utiliza la función heurística para decidir cual será su mejor opción. Como se puede ver en la siguiente captura, el color verde indica que es una buena jugada para la computadora, por el contrario, el color rojo indicia que es una buena jugada para el usuario. El usuario tiene la posibilidad de reiniciar el juego cuando lo desee.

```
LIMITE: el límite actual es de: 14
Turno IA
Cartas disponibles:
(1, +1) (3, +1) (5, -1) (7, +1) (8, -1) (11, +1)
La Inteligencia Artificial está evaluando su mejor jugada..
-----
La Inteligencia Artificial ha descartado la carta:7
-----
```

7. Consultas

El jugador cada vez que juega tiene disponible 3 consultas.

- Imprimir posibles resultados a partir de un punto, es decir, los caminos que se forman a partir de un nodo raíz hasta una hoja.
- Escribir una secuencia de cartas y simular las jugadas para ver que pasaría a partir de ese punto.
- Mostrar todas las cartas a partir de un cierto nivel de profundidad.

También, el usuario dispone de 3 utilizades:

- Salir del juego.
- Seguir jugando.
- Reiniciar una partida.

```
-----  
                                CONSULTAS  
-----  
Usted ha ingresado al módulo de consultas. ¿Qué desea hacer?  
  
A. Mostrar resultados a partir de este punto.  
B. Simular una secuencia de posibles jugadas.  
C. Imprimir cartas a partir de una profundidad.  
S. Seguir con el juego.  
R. Reiniciar el juego  
Q. Cerrar el juego.  
-----  
Ingrese una opción:
```

7.1. Consulta A

Si el usuario presiona la tecla A, se muestra consulta que imprime todos los caminos posibles a partir de un cierto punto.

7.2. Consulta B

Si el usuario presiona la tecla C, podrá simular una secuencia de cartas y avanzar para comprobar que pasará en cierto punto. Vale aclarar que al ser una partida simulada, no afectará a la jugada original.

7.3. Consulta C

Si el usuario presiona la tecla C, se hace un recorrido por niveles y se imprime en pantalla un nivel completo con las cartas, si aparece en color celeste, es del humano, sino de la IA.

```
Ingrese una opción: c  
Ingrese un nivel:  
2  
(2, -1)(3, -1)(6, -1)(2, -1)(3, -1)(6, -1)(2, 1)(3, 1)(6, -1)  
Presione una tecla para continuar.
```

8. Mejoras

Para mejorar la experiencia para el usuario, se le ha asignado un Timer desde un rango entre 1,5 seg y 5 seg al descarte de carta de la Inteligencia Artificial para aparentar que está pensando en que carta lanzar. Una mejora posible para el juego podría ser que el turno de los jugadores se invierta cuando el usuario lo desee, o sino, que sea elegido aleatoriamente por el juego. También sería ideal implementar un sistema de dificultades para que la IA le de más oportunidades de ganar al usuario. Por ejemplo, que exista una dificultad 'Normal' en donde la computadora tenga solo un 60 por ciento de posibilidades de ganar.

9. Conclusiones

En conclusión, los distintos problemas que fueron surgiendo en la implementación se pudieron solucionar, luego fue mas sencillo seguir con el desarrollo del juego.

Quedó demostrado que con el algoritmo MiniMax, la IA gana en la mayoría de los casos, sin embargo, hay ciertas ocasiones en donde si el humano juega bien puede llegar a ganar.

Para mejorar la eficiencia del árbol de decisiones se decidió hacer un caso donde que permita frenar el armado del árbol MiniMax.

Por último, el trabajo fue de mucha utilidad para solidificar los conceptos aprendidos durante el transcurso de la cursada.

10. Bibliografía

[1] Devcode.la - "*El algoritmo Minimax y su aplicación en un juego*"
<https://devcode.la/tutoriales/algoritmo-minimax>