Comparación de los algoritmos de búsqueda en problemas de videojuegos de estrategia

Comparison of search algorithms in strategy video game problems

José Castillo[1], Jayson Morán[1], Ashley Lan[1], Nicholas Béliz Osorio[2]*
Lic. en Ingeniería de Sistemas y Computación, Facultad de Sistemas Computacionales, Universidad Tecnológica de Panamá [1], Departamento de computación y simulación de sistemas, Faculta de Ingeniería de Sistemas Computacionales, Universidad Tecnológica de Panamá[2]

Resumen Los juegos de estrategia siempre han sido muy gustados por las personas. Juegos como el ajedrez y las damas, por mencionar algunos muy jugados. Han creado rivalidad entre jugadores y hasta se han convertido en deportes jugados por profesionales. Con el surgir de la inteligencia artificial se abre una nueva ventana para probar si las computadoras pueden superar a los humanos. De estas ideas han surgido algoritmos que buscan solucionar problemas a través de diferentes estrategias. En este artículo se estudiaron los algoritmos de minimax y alfa-beta, su definición, funcionalidad y complejidad. Estos algoritmos se implementaron en la emulación del juego de Lara Croft Go, que fue programado en el lenguaje de Java utilizando el IDE de NetBeans. La aplicación consiste en un juego donde un humano se enfrenta a una IA que fue programada con ambos algoritmos y se define por diferentes factores, quien es el mejor resolviendo el juego. La recolección de los datos es definida por los movimientos que realice cada algoritmo para completar los niveles y la cantidad de estados que genere en la búsqueda de la solución. Basados en los datos recopilados durante la ejecución de los algoritmos, se demuestra que el algoritmo más eficiente en la búsqueda de la solución de los niveles implementados es el algoritmo de búsqueda Alfa-Beta, ya que genera menos estados, por lo tanto, evalúa menos nodos y llega a una solución en un tiempo menor que el algoritmo de búsqueda Minimax.

Palabras clave Algoritmo Alfa-Beta, algoritmo de búsqueda, algoritmo Minimax, inteligencia artificial, juegos de estrategia.

Abstract Strategy games have always been very liked by people. Games such as chess and checkers, to mention a few that are widely played, have created rivalry between players and have even become sports played by professionals. With the rise of artificial intelligence, a new window opens to test whether computers can outperform humans. From these ideas, algorithms have emerged that seek to solve problems through different strategies. In this article, the minimax and alpha-beta algorithms, their definition, functionality and complexity were studied. These algorithms were implemented in the emulation of the Lara Croft Go game, which was programmed in the Java language using the NetBeans IDE. The application consists of a game where a human faces an AI that was programmed with both algorithms and is defined by different factors, who is the best solving the game. Data collection is defined by the movements that each algorithm makes to complete the levels and the number of states it generates in the search for the solution. Based on the data collected during the execution of the algorithms, it is shown that the most efficient algorithm in finding the solution of the implemented levels is the Alpha-Beta search algorithm, since it generates fewer states, therefore, it evaluates less nodes and reaches a solution in less time than the Minimax search algorithm.

Keywords Alpha-Beta algorithm, search algorithm, Minimax algorithm, artificial intelligence, strategy games.

* Corresponding author: jose.castillo33@utp.ac.pa

1. Introducción

La teoría de juegos estudia las intervenciones y decisiones que toman los demás personajes, donde el principal debe reconocer esto para que tenga éxito. También se le conoce como un "concepto de solución" donde eligen sus estrategias para maximizar las posibilidades de ganar.

Un ejemplo de esta teoría es cuando usamos la lógica siempre que interactuamos con otra persona, por ejemplo: cuando estamos en una pizzería y queda el ultimo trozo de esta, debemos elegir en si quedarnos con este último trozo o dejar que la otra persona lo tome.

En esta teoría no debemos preguntarnos qué haremos después, sino, debemos preguntarnos qué haremos dependiendo y teniendo en cuenta lo que pensamos que pueden hacer los demás y ellos actuaran según lo que piensen que pueda ser nuestro siguiente movimiento. Nash revolucionó la toma de decisiones en la teoría de juegos y en la economía, donde se estudia las tomas de decisiones e interacciones en

https://doi.org/10.33412/rev-ric.v6.2.2897

donde se conoce como estructuras formalizadas en incentivos.

En el estudio matemático, la teoría de juegos no se ha utilizado solamente en la economía y decisiones empresariales, sino también en la psicología, política, estrategia e incluso, en biología y para ganar al póker. En la representación de esta teoría se suelen usar matrices y árboles de decisiones para comprender como es que de un punto se llega a otro. Pero también los juegos pueden ser resueltos haciendo uso de las matemáticas, por lo que deben ser bastante complejas para encontrar la profundidad.

Existen dos tipos de algoritmos de búsqueda: El primero es la búsqueda ciega o que sólo usa información acerca de si el estado que se está evaluando es o no un estado valido para poder seguir con la búsqueda. El segundo tipo de algoritmo son las búsquedas heurísticas: Estas son lo contrario del método de búsqueda a ciega, donde usan información adicional para realizar sus búsquedas. Para hacer más eficiente este tipo de algoritmo, este hace uso de predicciones del costo para poder alcanzar el estado final o estado meta, esta función que hace todo esto es la heurística.

El objetivo de este artículo es conocer la eficiencia de los algoritmos de búsqueda a través de la comparación de estos en problemas de videojuegos de estrategia.

2. Teoría de juegos

La teoría de juegos estudia las decisiones que tiene un individuo para que tenga éxito de acuerdo con las decisiones que tomen los otros individuos en determinada situación.

Viendo esta definición de la teoría de juegos, podemos definirla como una serie de algoritmos, donde los sujetos, dependiendo del escenario, toman decisiones teniendo en cuenta las elecciones de los demás para poder acertar y llegar a la meta que lo beneficie.

Dentro de lo conocido, “juego”, se define como cualquier estructura en la que se pueden ofrecer recompensas o incentivos preestablecidos, donde implica a varias personas u otros entes racionales, como las inteligencias artificiales o los animales. De modo descendiente, podríamos librar que los juegos son similares a los conflictos.

Siguiendo esta observación, los juegos aparecen constantemente en la vida cotidiana. Así, la teoría de juegos no es únicamente una manera de adivinar el porte de las personas que participan en un problema, esta es útil también para examinar la justa de precios entre dos tiendas que están en la misma calle, así como para muchas otras situaciones.

La teoría de juegos como despacho numeral no se ha utilizado solamente en los juegos, sino también se ha utilizado en la gestión, estrategia, psicología y también en biología [1].

La teoría del juego presenta los siguientes aportes clave:

- La teoría del juego sirve para comprender situaciones sociales entre personajes en competencia y producir la elección de la decisión óptima de personajes independientes y en competencia en un entorno estratégico.
- Muchos de los escenarios incluyen el dilema del prisionero y el juego del dictador, entre muchos otros.
- Se pueden establecer escenarios del mundo real usando la teoría de juegos, en donde existan situaciones como la competencia de precios, el lanzamiento de productos entre otras para predecir sus resultados.

Ejemplos de la teoría del juego:

- El dilema del prisionero: Dos personas son arrestadas, encarceladas y se les fija la fecha del juicio. El fiscal del caso habla con cada prisionero por separado y les presenta una oferta: Si no confiesa y su socio lo hace, será condenado a 20 años y su socio quedará libre. Si ambos confiesan, serán condenados a 5 años de prisión. Si ninguno confiesa, serán condenados a un año de prisión [2].
- El dilema del voluntario: Existe una empresa en la que el fraude contable es desenfrenado, aunque el personal principal no sabe. Algunos empleados del departamento de contabilidad son conscientes del fraude, pero dudan en decírselo al personal principal porque eso daría lugar a que los empleados implicados en el fraude fueran despedidos y muy probablemente enjuiciados. Y si no se hace nada, daría lugar a la quiebra de la empresa y la pérdida de todos los puestos de trabajo.

2.1 Equilibrio de Nash

La teoría de juegos inicia con el estudio de Antoine Augustin Cournot sobre un duopolio con el cual se logra obtener a una versión reducida del equilibrio del Nash, alcanzando lentamente el nivel adecuado de precios y producción. Posterior a esto se puede decir que el fundador formal de la teoría de juegos fue el matemático John Von Neuman [3].

Algunos economistas han tenido el honor de darles el Premio Nobel de Economía por sus trabajos sobre la teoría de juegos. Entre estos destaca Nash, conocido por una película titulada “Una mente maravillosa” y es por eso, que en el equilibrio de Nash es dónde se basan muchas conclusiones en las que se han tomado sobre teoría de juegos aplicada a la vida real.

El equilibrio de Nash es un estado de un resultado en el cual se puede alcanzar, pero cuando se alcanza, ningún jugador puede aumentar la recompensa cambiando las decisiones que ha tomado. También se puede mencionar como “sin arrepentimiento” es decir, cuando la decisión está tomada, el jugador no se arrepentirá de las decisiones que consideren como consecuencias [4].

2.2 Algoritmos de búsqueda con adversarios

En la teoría de juegos se encuentran los algoritmos Minimax y Alfa-Beta, en donde, podemos ver la teoría de juegos como una serie de algoritmos, donde los sujetos dependiendo del escenario, toman decisiones teniendo en cuenta las elecciones de los demás para poder acertar y llegar

a la meta que lo beneficie. Por lo que, estos dos algoritmos hacen referencia a esta teoría [5].

2.2.1 Minimax

Este algoritmo es el más conocido y utilizado para problemas con exactamente dos competidores, donde hay información perfecta y movimientos alternos, es decir, "después de mí, sigues tú". Este algoritmo identifica a cada jugador como, jugador MAX y jugador MIN. MAX es el jugador que inicia el juego, para entenderlo mejor, supondremos que MAX somos nosotros, y marcaremos como nuestro objetivo encontrar el conjunto de movimientos que proporcionen nuestra victoria, independientemente de lo que haga el jugador MIN [6]. En la figura 1 podemos observar el funcionamiento del algoritmo en un árbol.

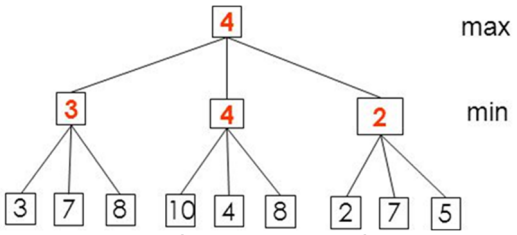

Figura 1. Ejemplo de ejecución del algoritmo de búsqueda Minimax.
Fuente: https://slideplayer.es/slide/1681515/.

2.2.2 Poda Alfa-Beta

Este algoritmo es una técnica mejorable sobre el algoritmo Minimax, donde un punto clave de este algoritmo que debemos tener en cuenta es que, cuanto más profunda sea la exploración, mejor será la toma de decisión sobre la jugada que debemos tomar. Para juegos donde el árbol tiene ramificaciones muy extensas, será de gran ayuda, ya que el cálculo necesario para algunas decisiones será prohibitivo. Por lo que es factible usar poda Alfa-Beta para obviar caminos innecesarios, para así no visitarlos y compararlos, pero para una mejor eficiencia de este algoritmo, se sugiere utilizar heurísticas [6]. En la figura 2 podemos observar el funcionamiento del algoritmo, la línea roja indica la poda del árbol.

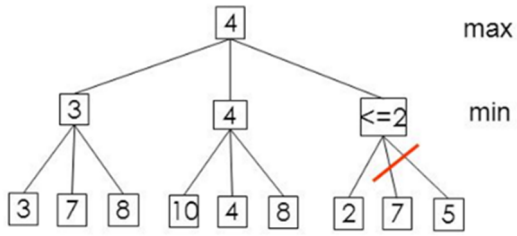

Figura 2. Ejemplo de ejecución del algoritmo de búsqueda Alfa-Beta.
Fuente: https://slideplayer.es/slide/1681515/.

3. Marco metodológico

Tomando en cuenta las informaciones anteriores es importante mencionar que esta investigación es de tipo experimental, debido a que se desarrollaron pruebas experimentales de los algoritmos, en estas pruebas se evaluaron las soluciones presentadas por cada algoritmo y se determinó la mejor solución.

En el caso del juego que se implementó fue crucial realizar una investigación experimental, que consistió en probar el juego, familiarizarse con él, conocer las reglas, técnicas y métodos que se emplean para alcanzar la victoria. De este modo al momento de implementar el juego solo hizo falta tomar en cuenta cada uno de estos conocimientos y llevarlos a ejecución.

La muestra es de hecho no probabilística, se ha elegido este tipo de muestra ya que la población, se definió como el juego que buscamos emular, debemos saber que cada partida completa debe durar máximo 1 minuto por lo que se pensó y se decidió la creación de tres niveles, en cada uno se llevaría a cabo las jugadas del humano y de la inteligencia artificial, para al final demostrar los resultados.

La recolección de los datos se llevó a cabo tomando en cuenta la cantidad de movimientos que realice el usuario o la inteligencia artificial, se evaluó y se determinó mediante estos, quien terminó la partida con un costo de movimientos menor, esta recolección de datos se evalúa al final del juego, mostrando una tabla comparativa de cada nivel jugado en ella, de igual manera se puede ver la serie de movimientos que realizó cada jugador, el propósito no es ver quien gane, ya que ambos logran cumplir el objetivo de finalizar el juego, pero aquel que logró encontrar la meta utilizando la ruta más eficiente, este es quien tiene la victoria.

La segunda recolección de datos que se llevó a cabo es la cantidad de estados que genera cada algoritmo, esto define el tiempo en que los algoritmos logran encontrar los estados finales y las rutas adecuadas a seguir.

4. Desarrollo de la aplicación

Desarrollar la aplicación propuso un reto difícil, ya que no solo es desarrollar la aplicación que lograra resolver los niveles extraídos del juego, sino que también debía poder ser jugado por un humano, todo cumpliendo, además, con las reglas del juego que se seleccionó para realizar las pruebas, en este caso el juego escogido fue Lara Croft Go, el cual cuenta con reglas diversas y en cada nivel se eliminan y agregan reglas nuevas.

Desarrollar la aplicación era fundamental para llevar a cabo el análisis de los algoritmos y compara los resultados, pero antes de desarrollar la aplicación se debió realizar un análisis a fondo del problema, el cual explicaremos en la siguiente sección.

4.1 Análisis del problema

Para llevar a cabo un adecuado análisis del problema se debe fraccionar, es decir, descomponer en pasos más sencillos para que se pueda analizar de mejor manera. Se fraccionó el problema en las siguientes series de puntos:

- Estado inicial: los estados iniciales corresponden a cada uno de los niveles que se implementaron en la

aplicación, cada nivel cosiste en un puzle distinto extraído del juego seleccionado.

- Operadores: los tres niveles cuentan con operadores en común y algunos operadores distintos, los cuales se explicarán a continuación:
  - Operadores en común: los tres niveles cuentan con cuatro operadores común, estos consisten en los movimientos que puede realizar el personaje, subir, bajar, ir a la derecha e ir a la izquierda.
  - Operadores del nivel 1: este nivel cuenta con un operador propio el cual consiste en una serpiente, la cual mira hacia una dirección específica y si el personaje se mueve al cuadrante del mapa que está frente a ella, esta atacará al personaje y se pierde la partida.
  - Operador del nivel 2: este nivel cuenta con dos operadores, el primero es una grieta, la cual se debilita con cada pasada sobre ella, es decir, solo se puede pasar una vez sobre ella y a la segunda vez que se trata de pasar sobre ella el personaje cae y pierde la partida. El segundo operador es un lagarto, el cual, al llegar a un punto del mapa a dos cuadrantes frente a él, este persigue al personaje y si llega a estar a un cuadrante en frente al lagarto, este atacará al personaje y se pierde la partida.
  - Operador del nivel 3: los operadores de este nivel consisten en dos arañas, cada una de ellas se mueve en una columna específica del mapa, suben y bajan sin parar, si el personaje se posiciona un cuadrante frente a una de estas arañas, atacarán al personaje y se pierde la partida.
- Costo de la ruta: el costo de la ruta es el valor correspondiente a cada estado, -1 si es estado es de pérdida o 1 si es estado es de ganancia.
- Meta: la meta de todos los niveles implementados corresponde a un estado final en el que el personaje se posicione en el cuadrante meta, atravesando todo el mapa sin perder.
- Ruta al espacio de estado: la ruta del espacio de estado, serán todos los movimientos que haya tenido que realizar el personaje para llegar desde el estado inicial hasta el estado nuevo.
- Espacio de estado: el espacio de estado está formado por cada uno de los estados que se hayan generado a partir del estado inicial a través del movimiento del personaje a través del mapa.
- Solución: la solución a los tres niveles que se han implementado corresponde a un estado final, en el cual el personaje se posicione en el cuadrante meta del mapa.

4.2 Implementación del juego

La aplicación que se desarrolló incluye tres niveles extraídos del juego Lara Croft Go, cada nivel consiste en un puzle que puede ser resuelto tanto por un humano como por los dos algoritmos de búsqueda. Al inicio de la aplicación se presenta una pantalla donde se selecciona el algoritmo que se desea utilizar. Los niveles del juego de los cuales se extrajeron los mapas de la aplicación son los siguientes:

- Nivel 1: el mapa del nivel 1 de la aplicación (ver figura 3) fue extraído del nivel 1 del primer libro del juego (ver figura 4).

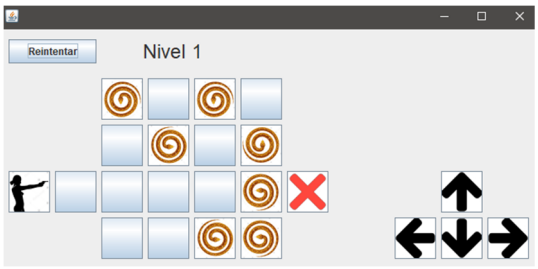

Figura 3. Mapa del nivel 1 de la aplicación.

Figura 4. Nivel 1 del primer libro del juego Lara Croft Go.
Fuente: Juego Lara Croft Go.

- Nivel 2: el mapa del nivel 2 de la aplicación (ver figura 5) fue extraído del nivel 8 del primer libro del juego (ver figura 6).

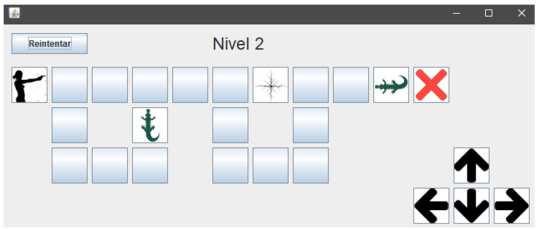

Figura 5. Mapa del nivel 2 de la aplicación.

Figura 6. Nivel 8 del primer libro de juego Lara Croft Go.
Fuente: Juego Lara Croft Go.

- Nivel 3: el nivel tres de la aplicación (ver figura 7) fue

extraído del nivel 1 del libro 2 del juego (ver figura 8).

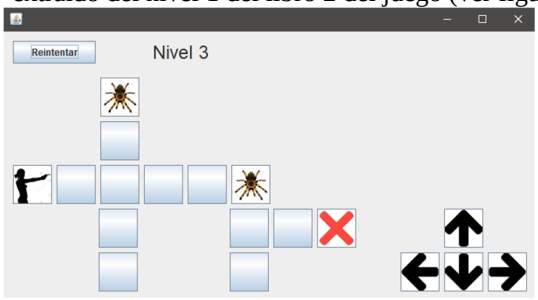

Figura 7. Mapa del nivel 3 de la aplicación.

Figura 8. Nivel 1 del segundo libro del juego Lara Croft Go.
Fuente: Juego Lara Croft Go.

5. Análisis de los resultados

Los datos que se analizarán en esta sección son recolectados durante la ejecución de la aplicación y se muestran al final de la ejecución en una ventana (ver figura 9). Esta ventana consta de tres columnas de datos, la primera corresponde a la cantidad de pasos que realizó el humano para llegar al estado final, la segunda corresponde a la cantidad de pasos que realizó el algoritmo seleccionado para llegar al estado final y la tercera columna corresponde a la cantidad de estados que generó el algoritmo durante el proceso de búsqueda. Para llevar a cabo el análisis de los resultados solo se utilizarán los datos de los algoritmos, por lo tanto, los datos obtenidos de los humanos se ignorarán.

Figura 9. Ventana en donde se muestran los resultados al final de la ejecución de la aplicación.

La tabla 1 que se muestra a continuación contiene los datos recolectados durante la ejecución del algoritmo de búsqueda Minimax. Estos datos serán con los que se lleve a cabo la comparación de la eficiencia de los algoritmos.

Tabla 1. Datos recolectados durante la ejecución del algoritmo Minimax

Nivel	Cantidad pasos realizados por el algoritmo	Cantidad de estados generados por el algoritmo
1	20	1400
2	38	2553
3	10	721

La tabla 2 contiene los datos recolectados durante la ejecución del algoritmo de búsqueda Alfa-Beta. Estos datos junto con los datos recolectados en la tabla anterior serán utilizados en la comparación de la eficiencia de los algoritmos.

Tabla 2. Datos recolectados durante la ejecución del algoritmo Alfa-Beta

Nivel	Cantidad pasos realizados por el algoritmo	Cantidad de estados generados por el algoritmo
1	20	791
2	38	2155
3	10	607

5.1 Análisis de los datos

En esta sección se compara la cantidad de pasos realizados para llegar a la meta y la cantidad de estados generados por los algoritmos para llegar a la meta.

En la siguiente figura se comparan la cantidad de pasos realizados por los algoritmos para resolver cada nivel (ver figura 10). Como se puede observar, ambos algoritmos realizaron la misma cantidad de pasos para llegar desde el estado inicial hasta el estado final. Esto sugiere que ambos algoritmos encontraron la misma ruta para llegar al estado final.

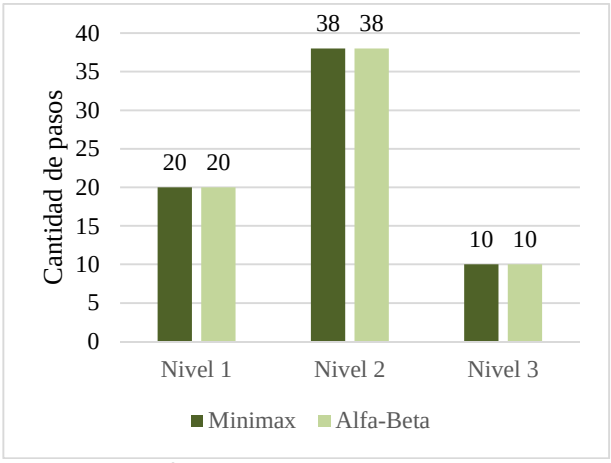

Figura 10. Comparación de la cantidad de pasos realizados por los algoritmos para resolver cada nivel.

En la siguiente figura podemos observar la comparación de la cantidad de estados generados por los algoritmos durante la búsqueda de la solución de cada nivel (ver figura 11), se puede ver que el algoritmo de búsqueda Alfa-Beta generó una menor cantidad de estados para llegar a la solución del problema.

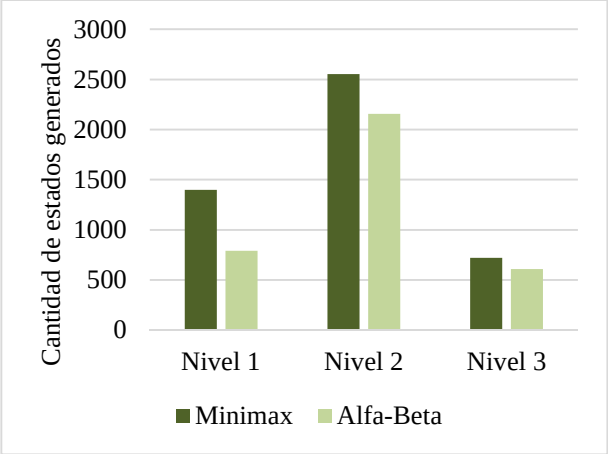

Figura 11. Comparación de la cantidad de estados generados por los algoritmos para resolver cada nivel.

En la siguiente tabla se comparan el promedio de pasos realizados por cada algoritmo para llegar a la solución (ver tabla 3). Como se puede observar, el promedio de pasos de cada algoritmo es el mismo, esto nos sugiere que ambos algoritmos siguieron la misma ruta al estado final, sin importar la cantidad de estados que hayan generado en la búsqueda de la ruta.

Tabla 3. Comparación del promedio de pasos realizados por cada algoritmo

Algoritmo	Promedio de pasos realizados
Minimax	22.7
Alfa-Beta	22.7

En la tabla 4 se puede ver el promedio de estados generados por cada algoritmo en la búsqueda del estado final. Se observa que el promedio de estados generados por el algoritmo de búsqueda Alfa-Beta es mucho menor. En otras palabras, el algoritmo de búsqueda Alfa-Beta generó en promedio 373.7 menos estados que el algoritmo de búsqueda Minimax, lo que nos sugiere que el algoritmo de búsqueda Alfa-Beta es más eficiente que el algoritmo de búsqueda Minimax.

Tabla 4. Comparación del promedio de estados generados por cada algoritmo

Algoritmo	Promedio de estados generados
Minimax	1558
Alfa-Beta	1184.3

5.2 Análisis de confiabilidad

Para analizar la confiabilidad de los resultados mostrados en la sección anterior, se utilizará el coeficiente Alfa de Cronbach, con el cual se obtendrá un valor que dirá si los resultados mostrados son confiables o no.

Para el cálculo del coeficiente Alfa de Cronbach se utilizarán los datos de la tabla 5, los cuales son los datos obtenidos de la ejecución de los algoritmos y los demás datos necesarios para el cálculo del coeficiente.

Tabla 5. Datos utilizados en el cálculo del coeficiente Alfa de Cronbach

Algoritmo	Nivel			Suma
	1	2	3	
Minimax	1400	2553	721	4674
Alfa-beta	791	2155	607	3553
Varianza	185440.5	79202.0	6498.0	628320.5
Suma de la varianza	271140.50			

La ecuación 1 es la utilizada para calcular el coeficiente Alfa de Cronbach.

$$\alpha = \frac{K}{K-1}\left[1 - \frac{\sum S_i^2}{S_T^2}\right] \quad (1)$$

Donde:

- K: Número de ítems
- S_i^2: Sumatoria de varianzas de los Ítems
- S_t^2: Varianza de la suma de los Ítems
- α: Coeficiente de Alfa de Cronbach

Para nuestro cálculo utilizaremos los siguientes valores:

- K: 3
- Si2: 271140.50
- St2: 628320.50

Al introducir los valores mostrados con anterioridad a la fórmula se obtiene que el coeficiente Alfa de Cronbach es igual a 0.85, por lo tanto, se puede decir que la confiabilidad de los resultados mostrados en la sección anterior es buena, en otras palabras, los resultados son confiables.

6. Conclusiones

Como sabemos, el algoritmo Minimax es un método de decisión para minimizar la pérdida en juegos con adversario y así poder potenciar las posibilidades de ganar con información perfecta. Minimax es un algoritmo recursivo y se puede decir que elige el mejor movimiento para ti, suponiendo que tu contrincante escogerá el peor para ti. Este algoritmo aunque es efectivo, ya que marca el camino que se debe seguir, puede no resultar beneficioso cuando se está comparando con el

algoritmo de Alfa-Beta, ya que, en la optimización de el algoritmo de Minimax, es el uso de la técnica de Poda Alfa-Beta, donde se basa en evitar el cálculo de ramas cuya evaluación final no va a poder superar los valores previamente obtenidos, aunque el algoritmo de Alfa-Beta siendo una mejora directa del Minimax.

En esta comparación, con el algoritmo Minimax, el algoritmo Alfa-Beta poda los estados que considera innecesarios evaluar, ahorrando tiempo y espacio de memoria sin recorrer aquellos estados, y de esta manera obtiene el resultado en un corto tiempo. Es por esto, por su eficiencia y corto tiempo de ejecución en comparación con el Minimax, que lo señalamos como el algoritmo que soluciona de mejor manera y logra cumplir el objetivo. Sabiendo esto, se puede concluir que implementando la técnica Poda Alfa-Beta en el algoritmo de búsqueda Minimax, este funcionará de manera más eficiente y rápida que cuando no se aplica.

AGRADECIMIENTOS

Se agradece a la Universidad Tecnológica de Panamá por brindar las herramientas necesarias para llevar a cabo esta investigación y por incentivar a los estudiantes escribir artículos que pueden ayudar a otros estudiantes a conocer más sobre distintos temas. También se le agradece al profesor Nicholas Béliz por motivarnos a escribir artículos que sirven como evidencia de los conocimientos que adquirimos durante nuestra carrera universitaria y que nos pueden ayudar en la vida profesional a sobre salir de un grupo.

REFERENCIAS

[1] I. Gutiérrez, «Muy Financiero,» 11 Septiembre 2001. [En línea]. Available: http://www.muyfinanciero.com/conceptos/teoria-de-juegos/. [Último acceso: 2 Julio 2020].

[2] C. Stokel-Walker, «BBC,» 24 Mayo 2015. [En línea]. Available: https://www.bbc.com/mundo/noticias/2015/02/150220_teoria_de_juegos_que_es_finde_dv. [Último acceso: 2 Julio 2020].

[3] J. Navarro, «El Blog Salmon,» 11 Mayo 2011. [En línea]. Available: https://www.elblogsalmon.com/conceptos-de-economia/que-es-la-teoria-de-juegos. [Último acceso: 2 Julio 2020].

[4] A. Hayes, «Investopedia,» 15 Junio 2019. [En línea]. Available: https://www.investopedia.com/terms/g/gametheory.asp. [Último acceso: 2 Julio 2020].

[5] F. S. Caparrini, «Fernando Sancho Caparrini,» 24 Octubre 2019. [En línea]. Available: http://www.cs.us.es/~fsancho/?e=107. [Último acceso: 2 Julio 2020].

[6] P. A. Fruto, «Ccomputer Science Department,» 12 Junio 2008. [En línea]. Available: https://www.cs.upc.edu/~bejar/ia/material/trabajos/Algoritmos_Juegos.pdf. [Último acceso: 2 Julio 2020].

[7] Google, «Sites Google,» 12 Septiembre 2010. [En línea]. Available: https://sites.google.com/site/portafolionetbeans/que-es-netbeans. [Último acceso: 2 Julio 2020].

[8] Junta de Andalucia, «Junta de Andalucia,» 2 Diciembre 2016. [En línea]. Available: http://agrega.juntadeandalucia.es/repositorio/02122016/a5/es-an_2016120212_9131705/34_colas.html. [Último acceso: 2 Julio 2020].

[9] I. d. P. Centen, «Xerindia,» 13 Febrero 2019. [En línea]. Available: https://www.xeridia.com/blog/el-papel-de-la-inteligencia-artificial-en-la-actualidad. [Último acceso: 2 Julio 2020].

[10] C. Ciuraneta, «MeriStation,» 4 Julio 2017. [En línea]. Available: https://as.com/meristation/2015/08/28/analisis/1440712800_148233.html. [Último acceso: 2 Julio 2020].

[11] Jarkendia, «Vida Extra,» 23 Diciembre 2016. [En línea]. Available: https://www.vidaextra.com/analisis/analisis-de-lara-croft-go-espejito-espejito-quien-es-la-mas-lista. [Último acceso: 2 Julio 2020].