Resolución de laberintos usando algoritmos genéticos

Elkin Alejandro Cruz Camacho, Osmond Lisando Pomare Hernández

RESUMEN

El presente artículo explora el mundo de los laberintos con múltiples soluciones usando algoritmos genéticos.

***Palabras Clave*—Agentes inteligentes, algoritmos evolutivos, algoritmos genéticos, maze solving algorithms.**

I. INTRODUCCIÓN

Los laberintos han sido parte de la cultura humana, teniendo como el ejemplo más antiguo conocido el laberinto de Creta, las villas europeas del siglo XVII y XIX las cuales solían tener jardines hechos de setas, y en la actualidad en parques de atracciones.

Un laberinto es una estructura que consiste de varios pasajes ramificados e interconectados de manera complicada, el cual tiene un punto de entrada y un punto de salida, y el propósito es encontrar un camino que conecte ambos puntos.

Resolver problemas de laberintos resulta útil en problemas similares como TSP y direccionamiento de paquetes en una red, dado que los laberintos y estos son subproblemas de Teoría de grafos y Navegabilidad.

Los laberintos se pueden clasificar en múltiples categorías una de ellas es la de "routing" la cual se refiere a como están diseñadas las rutas en el laberinto. Perfect, es conexo, es decir de cualquier punto se puede llegar a otro, no posee caminos ciclos. Braid, es un laberinto sin callejones sin salida. Unicursal, es un laberinto con un único camino y sin ramificaciones. Partial Braid, un laberinto con loops y callejones sin salida[1].

Un algoritmo genético es una búsqueda heurística que imita el proceso de la evolución. Hace uso de una función objetivo y toma conceptos de la evolución como mutación, reproducción sexual, y conceptos biológicos como cromosoma, genotipo y fenotipo[2].

Las estrategias para la resolución de laberintos dependen del conocimiento que se tenga de estos. Algoritmos como Dead-End filling que consiste en rellenar todos los caminos sin salida, y shortest path, que consiste en encontrar el camino más corto desde el punto de entrada hasta el punto de salida requieren conocer de antemano el laberinto, mientras que algoritmos como Wall Follower, que consiste en mantenerse pegado a la pared hasta encofntrar la salida y solo funciona en laberintos que no sean simplemente conexos, Random Mouse, el cual consiste en moverse aleatoriamente en el laberinto hasta encontrar la salida y el algoritmo de Trémaux inventado por Charles Pierre Trémaux, que consiste en ir marcando celdas ya visitadas, devolverse cuando se encuentre un camino sin salida hasta la división anterior y escoger otra ruta, no tienen conocimiento previo del laberinto.

II. PLANTEAMIENTO DEL PROBLEMA

Dado un laberinto de 300X400 celdas *Partial Braid* (con varios caminos desde el punto de entrada al punto de salida) y pesos en cada celda. El proyecto busca encontrar soluciones eficientes usando algoritmos genéticos con soluciones no óptimas.

Se esperan obtener las condiciones, las constantes, en las que debe operar el algoritmo genético para obtener los mejores resultados, en el menor tiempo.

III. HIPÓTESIS

Usando algoritmos genéticos es posible encontrar varias rutas eficientes que resuelvan laberintos cuya característica principal es que no poseen una única solución desde el punto de entrada al punto de salida.

IV. METODOLOGÍA

Los laberintos son generados mediante la siguiente metodología: primero se usa el algoritmo *backtracer (Deep-First Search (DFS))* modificado, de manera que en promedio el 5% de las ejecuciones del algoritmo se escogiese como siguiente seleccionado una casilla ya usada. El laberinto

Elkin Alejandro Cruz Camacho: elacruzca@unal.edu.co, estudiante de Pregrado en Ingeniería - Ingeniería Sistemas y Computación, Universidad Nacional de Colombia.

Osmond Lisandro Pomare Hernández: olpomarehe@unal.edu.co, estudiante de Pregrado en Ingeniería - Ingeniería Sistemas y Computación, Universidad Nacional de Colombia.

obtenido por este procedimiento es de tipo tipo *Perfect (Ver Fig 1)*, luego se seleccionan de manera aleatoria un 10% de las paredes y se eliminan del laberinto. Se sabe que el laberinto resultante es de tipo *Partial Braid (Ver Fig 2)* y se espera que tenga también múltiples soluciones sin bucles. A cada celda se le asigna un peso aleatoriamente.

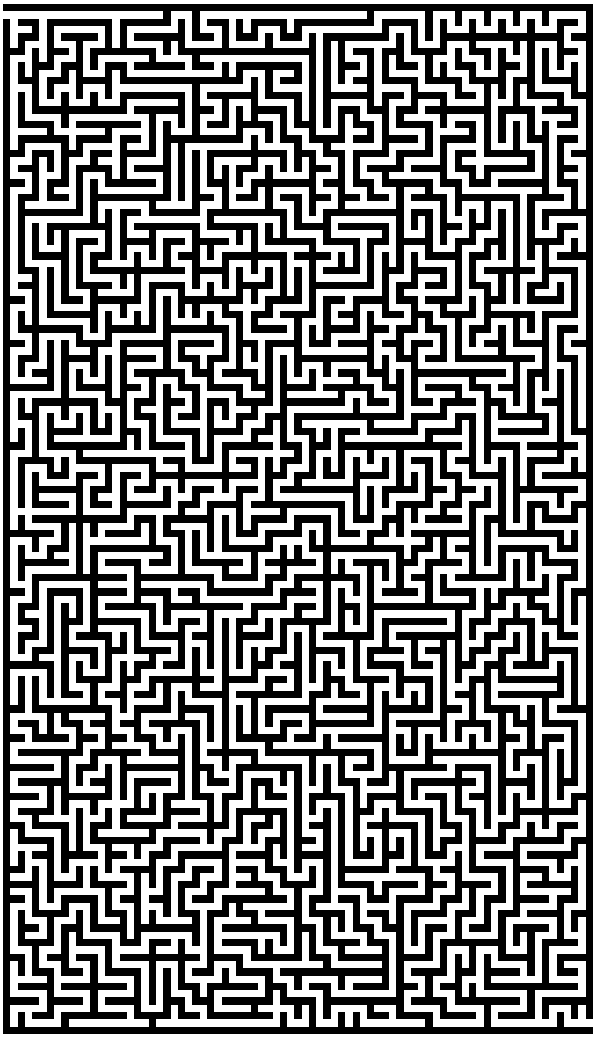

Fig. 1. Laberinto generado aleatoriamente mediante de DFS (El 5% de las veces se seleccionaba un elemento que no estuviera en la cabeza de la *stack*). De tamaño 70x40 (en los ejemplos aquí presentados se verán únicamente laberintos de 70x40 dadas que un laberinto de 300x400 es difícilmente visible a esta escala).

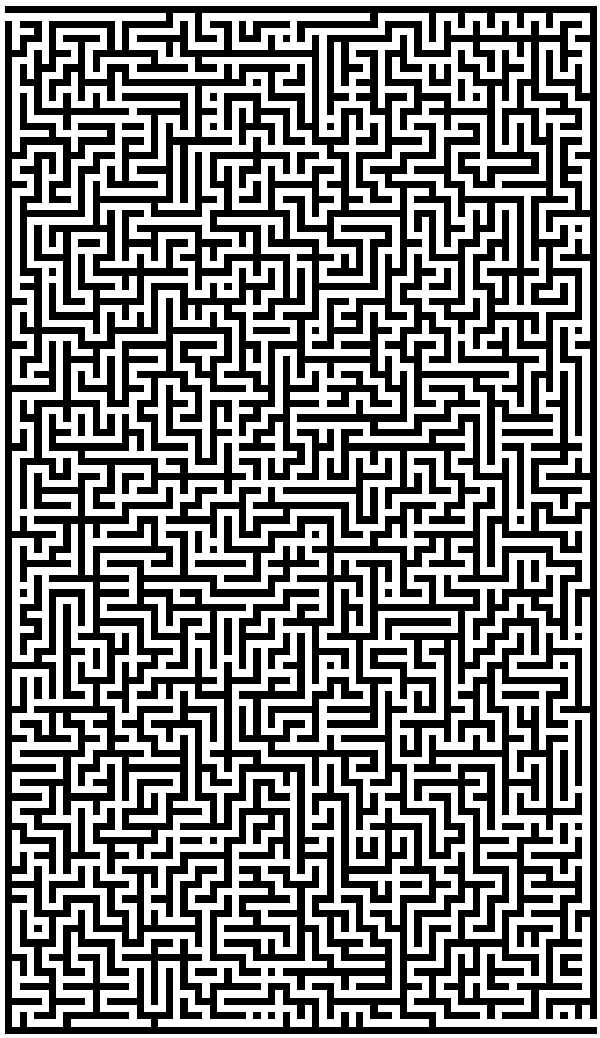

Fig. 2. Laberinto de Fig. 1 eliminando el 40% de las paredes (para los laberintos de 300x400 el porcentaje de paredes eliminadas será de un 10%)

Especificaciones del algoritmo genético:

- Cromosoma: Cada individuo será representado por una cadena de longitud variable, la cual indica el camino tomado por cada individuo *(Ver Fig 3)*.
- Función objetivo: suma total de celdas recorridas del inicio del laberinto al final. Por tanto se considera como mejor individuo a aquel que tiene un resultado de la función objetivo menor.
- Número de individuos: de 30 como población inicial, con una población total de 100 indviduos.
- Recombinación: cuando los caminos de dos individuos se cruzan es posible obtener en un tercer camino que comparte partes de caminos de sus padres. Un individuo es dominante y de él el hijo obtendrá la base (el camino principal), y se cambiará una sección de su camino por la de su otro padre *(Ver Fig 4 y 5)*.

- Mutación: se eligen dos puntos aleatorios en la ruta de un individuo y se busca otro posible camino entre esos dos puntos.

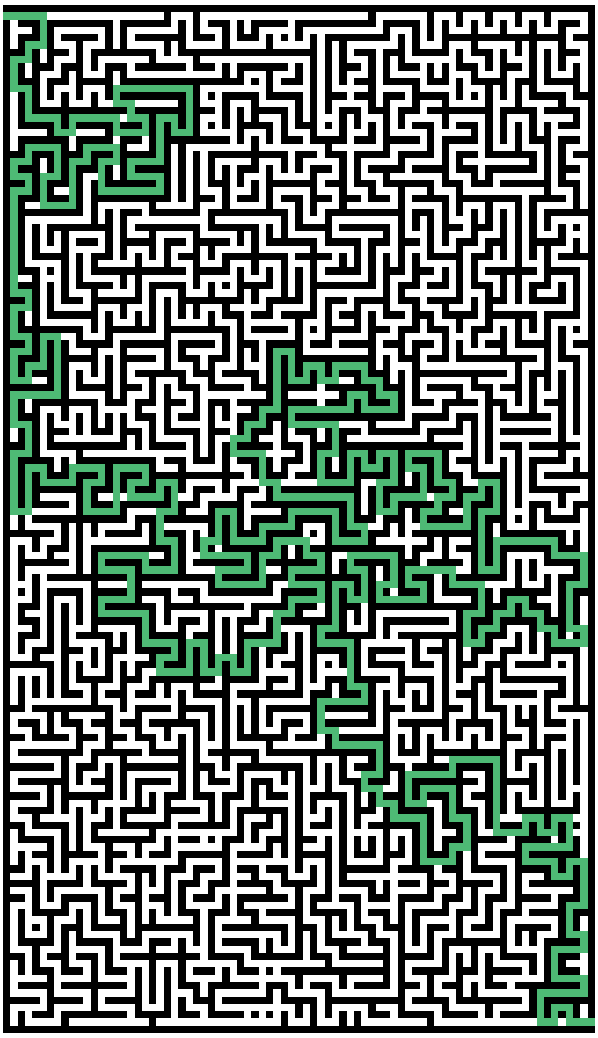

Fig. 3. Individuo, una solución del laberinto.

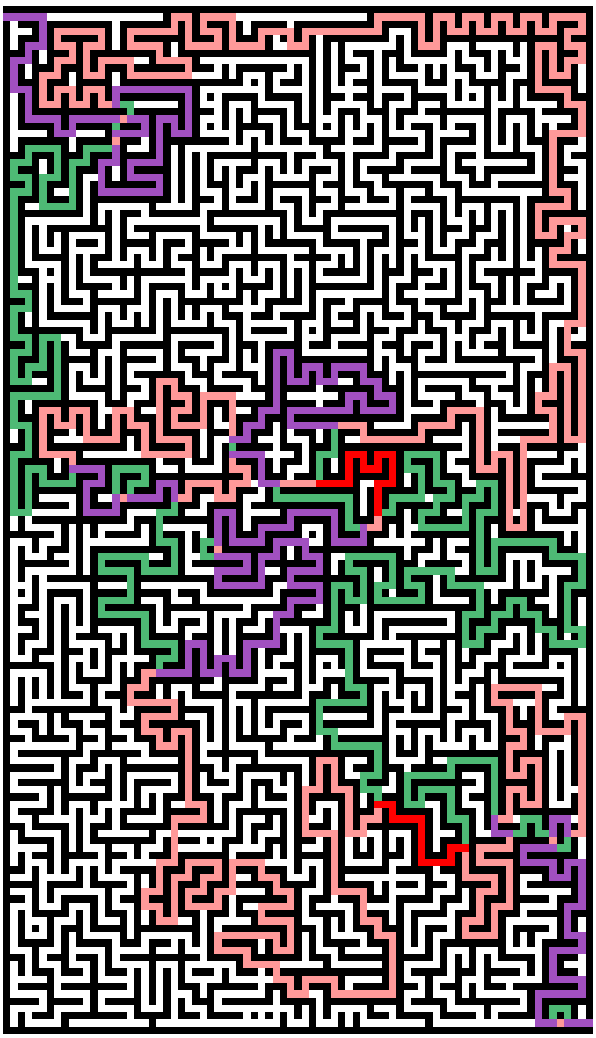

Fig. 4. Dos individuos y las zonas de los caminos donde se cruzan. Caminos: en rosa y verde; puntos comunes: en morado y rojo (en rojo dos cruces elegidos aleatoriamente para realizar la recombinación combinación). Individuo predominante: Rosa.

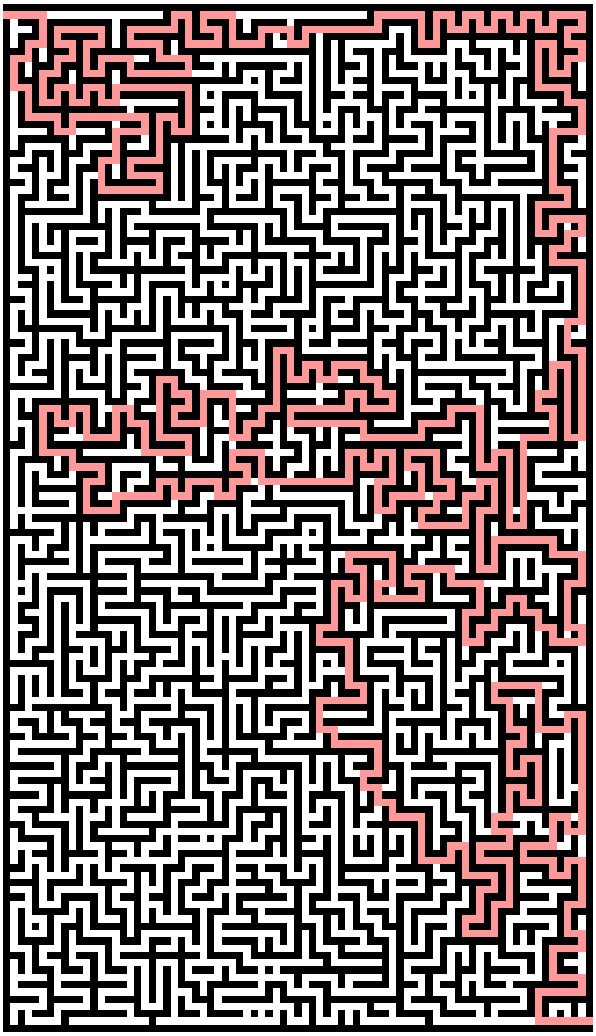

Fig. 5. Resultado de la recombinación de los dos caminos *(ver Fig 4)*, individuo que comparte caminos con sus padres.

V. RESULTADOS ESPERADOS

Se espera obtener varios caminos solución que resuelvan los laberintos eficientemente. Con posibilidades de encontrar un algoritmo que pueda ser plasmado con facilidad en papel, a diferencia de -por ejemplo- un agente desarrollado en una red neuronal.

VI. IMPACTO DEL PROYECTO

El proyecto tendrá repercusión en áreas que involucren teoría de grafos como se ha mencionado anteriormente, dado que se espera que las soluciones obtenidas tengan similitud con algoritmos de búsqueda en grafos.

REFERENCIAS

[1] W. D. Pullen. (2013, 11, 11). *Think Labyrinth: Maze Algorithms* [En Internet, Online]. Disponible en: http://www.astrolog.org/labyrnth/algrithm.htm.

[2] S. J. Russell, P. Norvig. *Artificial Intelligence: A Modern Approach*. Upper Saddle River, NJ: Prentice Hall Pa., 2009, pp. 126-129.

[3] T. Keeley, S. Audtrech. *Explorations in Maze Traversal Algorithms*. Proceedings of CSE 331, Data Structures Fall 2000, 2000, p. 25. Disponible en: http://www.cse.nd.edu/~cseprog/proj00/proceedings.pdf.

[4] M. Foltin. "Automated Maze Generation and Human Interaction", Tesis de Grado, Facultad de Informática, Universidad de Masaryk. 2008. Disponible en: http://www.martinfoltin.sk/mazes/thesis.pdf.

[5] Astolfo, David, Mario Ferrari, and Giulio Ferrari. Building Robots with LEGO Mindstorms NXT. Syngress, 2007, p. 371-390. Disponible en: http://www.sciencedirect.com/science/article/pii/B9781928994671500677.

[6] Choubey, Nitin S. "A-Mazer with Genetic Algorithm." *International Journal of Computer Applications* 58.17, 2012, p. 48-54.