TÍTULO (DESCRIPCIÓN CORTA DEL PROYECTO. ENTRE 8 Y 12 PALABRAS)

Juliana Henao Arroyave
Universidad EAFIT
Colombia
Jhenaoa4@eafit.edu.co

Gerónimo Zuluaga Londoño
Universidad EAFIT
Colombia
gzuluagal@eafit.edu.co

Mauricio Toro
Universidad Eafit
Colombia
mtorobe@eafit.edu.co

RESUMEN
En el presente se hará un estudio para la aplicación de una estructura de datos llamada pilas, con la cual se tratara de solucionar el problema de colisiones entre abejas roboticas.

1. INTRODUCCIÓN

Desde hace unos años ciertos factores que afectan el ambiente, como el clima, el uso de pesticidas, la destrucción de panales por tala de árboles, entre otros; ha causado que la población de abejas disminuya, lo cual tiene un impacto ambiental que claramente no solo las afecta a ellas, sino que su ida traería eventualmente la desaparición de muchas especies de plantas polinizadas por estos insectos que a su vez traen la desaparición de otras especies. Por esto es tan urgente hallar una solución alternativa a estos eventos que parecen irreversibles. Una manera es creando "abejas" robóticas que tomen el polen y lo lleven a otras plantas para completar la polinización. La creación de estas abejas trae consigo ciertas circunstancias que se deben controlar. En este caso trataremos el hecho de que las abejas podrían chocar unas con otras, la idea es usar estructuras de datos y diferentes algoritmos para hallar la solución para evitar que pasen este tipo de cosas.

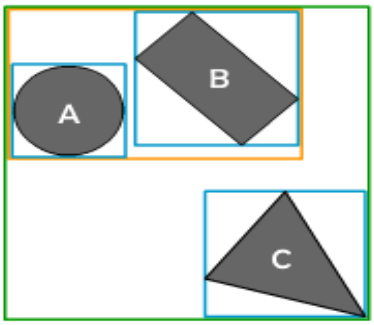
2. PROBLEMA

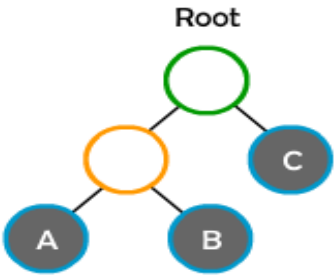
La creación de abejas robóticas (drones), trae consigo un obstáculo a la hora de la implementación de esta idea. Primero hay que considerar el numero de abejas que se tienen que utilizar para cubrir cierta área, con el objetivo de usar menos abejar y lograr cubrir un mayor espacio, sin embargo este no es el mayor problema al que nos enfrentamos al momento de crear estos drones. Teniendo en cuenta lo necesario que es este proceso natural de polinización, es muy claro que va a ser necesario la creación de un numero muy grande de estas abejas robóticas, la consecuencia real de esto es al momento de evitar la colisión entre ellas. Hay que entender que estas abejas son "autómatas" y que por razones de cantidad no van a estar siendo piloteadas por algún humano, lo que genera la necesidad de estarlas monitoreando, algo asi como una torre de control en la aviación, con la diferencia de estar vigilando números abismalmente mucho mas grandes de dispositivos, por lo tanto algo que debe ser realizado por un programa, ya que es casi imposible humanamente estar revisando la ruta o la proximidad de estos dispositivos en posible colisión en números que pasarían fácilmente los millones. A esto se debe la implementación de un algoritmo que evalue la posibilidad o cercanía entre las abejas roboticas y desviarlas automáticamente.

3. TRABAJOS RELACIONADOS

3.1 Dynamic AABB tree

La estructura de datos del árbol AABB es un sistema anticolisión que hace uso de figuras como polígonos, y es aplicado recursivamente para detectar si un rayo o línea esta próximo o tiene una intersección con estas figuras creadas, de esta manera, se puede predecir el contacto entre dos objetos o incluso es usado hasta para videojuegos, como por ejemplo el impacto de una bala o proyectil a los personajes o estructuras de un juego. Este algoritmo tiene muchas aplicaciones, hasta en diseño, y puede ser aplicado en la predicción de colisiones en las abejas robóticas, debido a que tiene aplicaciones hasta en objetos sólidos y de hecho hasta en la simulación de partículas.





3.2 El equipo de Stanford desarrolla software para predecir y prevenir colisiones con drones

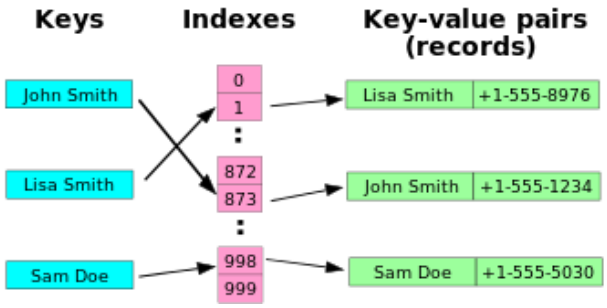
A finales de 2013, Jeff Bezos, tuvo la idea de usar drones para entregar paquetes de Amazon o llevar a cabo misiones de búsqueda y rescate. Unos años más tarde se pudo

realizar este proyecto, pero con esto venían otros factores a tomar en cuenta como un nuevo tráfico aéreo de baja altura. Y así antes de poner a funcionar este sistema de drones de debe definir ciertos parámetros para evitar colisiones o congestión entre estos drones. Ante esto el Laboratorio de Sistemas Inteligentes de Stanford (SISL) busca crear un sistema para gestionar este trafico aéreo no tripulado. Se utilizaron varias soluciones, pero la densidad esperada de los vuelos de drones creó un nivel completamente nuevo de complejidad para el equipo de SISL. "A medida que el número de aviones crece, el problema de evitación se vuelve exponencialmente más complicado, un desafío que los matemáticos llaman la maldición de la dimensionalidad", dijo el estudiante graduado de ingeniería mecánica Hao Yi Ong, que detalla un algoritmo de prevención de conflictos. "Así que tenemos que encontrar mejores formas que la simple búsqueda de fuerza bruta e iterar a través de todas las soluciones posibles".

La solución que propuso Ong para este problema fue separar los conflictos de aviones múltiples en problemas emparejados. Y rápidamente seleccionar la mejor acción para cada par de drones desde una tabla que predice la trayectoria de vuelo de cada dron. El servidor luego coordina cada una de estas soluciones por pares y emite una orden conjunta de prevención de colisiones para todos los drones afectados.

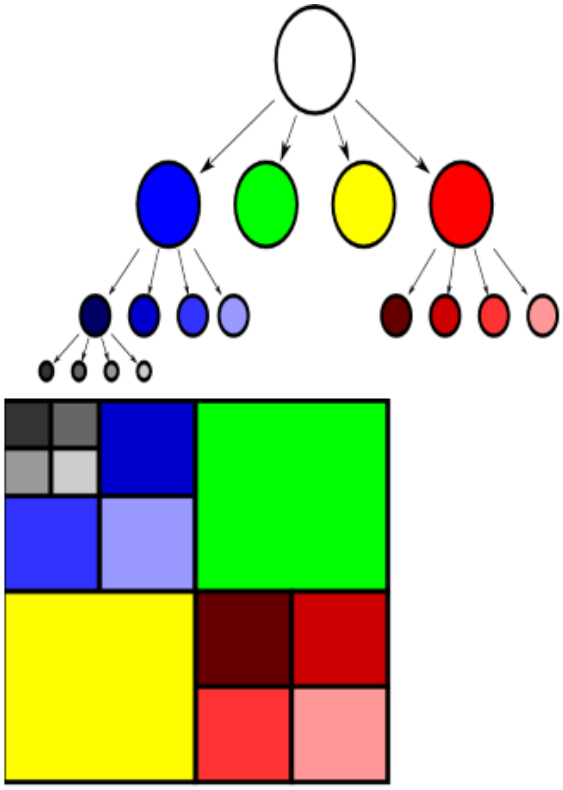
3.3 Hashing

Hashing implica aplicar un algoritmo hash a un elemento de datos, conocido como la clave hash, para crear un valor hash. Los algoritmos hash toman una amplia gama de valores y los asignan a un conjunto de valores más pequeño. Los valores hash se pueden usar para acelerar la recuperación de datos y se pueden usar para verificar la validez de los datos. Una coalición es cuando dos o más teclas hash resultan en el mismo valor hash, Esto causa un problema ya que ya no podemos encontrar rápidamente si los datos están en nuestra tabla hash o no, ya que otra pieza de datos puede tener el mismo valor. La solución a este problema se puede dar de dos maneras, una es colocando las claves colisionantes en el siguiente valor hash libre (encadenamiento separado); u otra es colocando las claves colisionantes en la misma ubicación, utilizando una lista vinculada para vincular todos los valores que coinciden con ese valor hash.



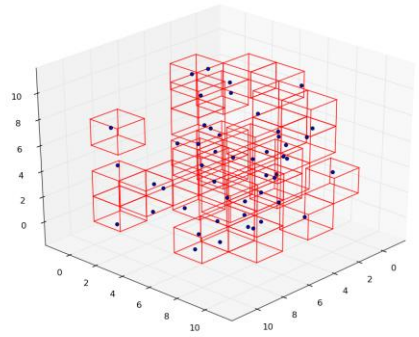
3.4 Quadtrees y Octree

Los Quadtrees (en 2D) y Octree (en 3D) son un tipo de estructura de datos en el que cada nodo en el plano tiene a su vez cuatro nodos hijos. Esto se puede usar para separar nuestro escenario en cuatro áreas iguales recursivamente. Esta estructura de datos permite optimizar el hecho de que en cada interacción tendríamos que comprobar si cada objeto está colisionando con el resto de los objetos en juego



4. Spatial Hashing

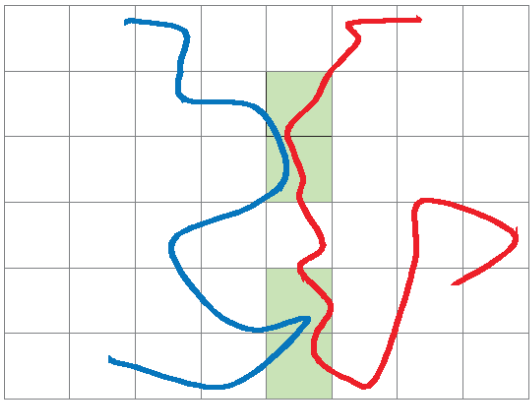
Esta estructura de datos consiste en dividir uniformemente el espacio, en este caso 3D, en contenedores para guardar cualquier dato que tenga una posición en el plano 3D para analizarlos luego. Con esta estructura, se le puede llamar a cada abeja como una partícula que tiene su propia cubeta o subdivisión en el espacio. Una característica importante del hashing son las colisiones, que se dan cuando dos partículas están en el mismo cubo, con esto, si se hacen subdivisiones cubicas de 100x100x100 en las tablas se puede detectar cuando hayan abejas en el mismo cubículo, esto quiere decir una colisión.



Gráfica 1: Visualización del spatial hashing en 3D. (Imagen sacada de: http://www.sgh1.net/posts/spatial-hashing-1.md)

4.1 Operaciones de la estructura de datos

Para el funcionamiento de estas tablas hash se pueden implementar varias operaciones como agregar un casillero, buscar uno en específico, borrar uno y hallar colisiones.



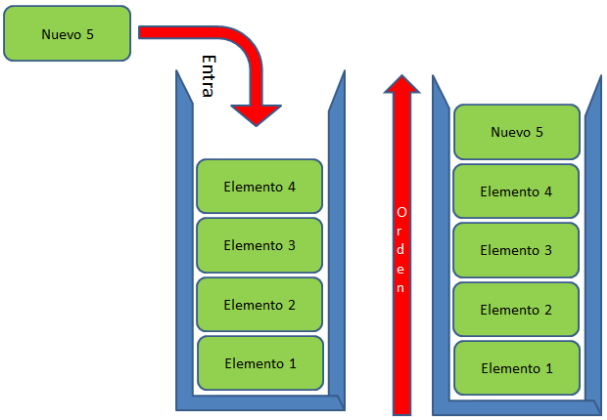
Gráfica 2: Imagen de una colisión en una tabla hash en 2D (Imagen sacada de: https://www.semanticscholar.org/paper/Parallel-Spatial-Hashing-for-Collision-Detection-of-Fan-Wang/4d218db1d4ee8410378efbd07cfff07226b95ce1)

4.2 Criterios de diseño de la estructura de datos

A momento de elegir una estructura de datos apta para el problema a resolver, luego de varias investigaciones acerca de varios tipos de estructuras de datos. Se llego a la conclusión de que el hashing es la más adecuada, ya que hace mucho enfoque en la colisión de particulas, que es precisamente lo principal en el problema de las abejas robóticas. Por lo tanto nos pareció que es muy útil para problemas de este tipo, además de que, según nuestra visión, con esta estructura se puede visualizar mejor el problema en tres dimensiones.

5. ALGORITMOS PARA LA LOCALIZACIÓN DE ABEJAS ROBÓTICAS Y EVITAR COLISIÓN

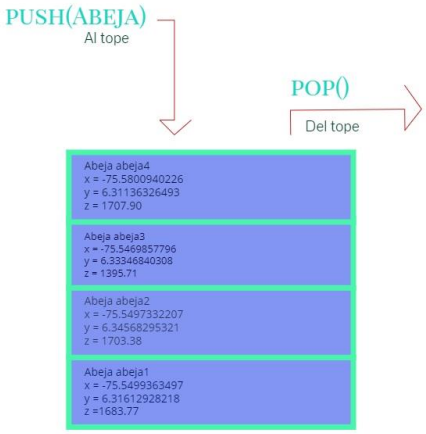
Se utilizaron pilas como estructura de datos para el desarrollo del problema.



Gráfica 3: Gráfica de una pila en la cual se muestra el orden de entrada y de salida.

5.1 Operaciones de la estructura de datos

En las pilas se utilizan dos operaciones. Para ingresar un elemento en el tope de la pila se usa el método push() y para sacar el elemento del tope de la pila se utiliza el pop()



Gráfica 4: Imagen de los métodos push y pop.

5.2 Criterios de diseño de la estructura de datos
Se utilizaron pilas como estructura de datos debido a que nos fue mas eficiente a la hora de la detección de las colisiones que otras estructuras, como por ejemplo una matriz de 3 dimensiones. Además los métodos que su uso es mas necesario en este tipo de problemas, con las pilas tienen una complejidad muy baja. Como por ejemplo push y pop que ambos tienen complejidad algorítmica de O(1) en cualquier caso.

5.3 Análisis de la Complejidad
Con la estructura empleada para este problema se tiene que la complejidad en el peor de los casos en O(n^2).

5.4 Tiempos de Ejecución

# de abejas	t(ms)
10	0
100	1
1000	14
10000	944
100000	91628

5.6 Análisis de los resultados

Número de abejas	Tiempo Con Stack	Tiempo con LinkedList
10	0 ms	1 ms
100	1 ms	4 ms
1.000	14 ms	28 ms
10.000	944 ms	904 ms
100.000	91.628 ms	

Tabla 8: Tabla de comparación entre pilas y linked list

6. CONCLUSIONES

En el desarrollo del trabajo se logro concluir que las pilas son una estructura de datos útil para problemas semejantes a este, que requieren de hacer comparaciones, debido a que sus métodos principales permiten tener una complejidad muy baja, de solo O(1).

Luego de comparar los resultados obtenidos entre varias estructuras de datos, se pudo observar que el tiempo de ejecución con pilas era menor que con las comparadas, como por ejemplo la LinkedList.

6.1 Trabajos futuros

Investigar otras soluciones posibles que quizás nos lleven a una complejidad mas baja.

AGRADECIMIENTOS
Nosotros agradecemos por la ayuda brindada con

Daniel Mesa, monitor, universidad EAFIT. Por los comentarios que nos hizo para mejorar el código.

REFERENCIAS
1. Chipman, Ian. Stanford team develops software to predict and prevent drone collisions, Stanford Engineering Magazine. December 11, 2015. From: Stanford University: https://engineering.stanford.edu/magazine/article/stanfordteam-develops-software-predict-and-prevent-dronecollisions.

2. https://en.wikibooks.org/wiki/Alevel_Computing/AQA/Paper_1/Fundamentals_of_data_structures/Hash_tables_and_hashing#Collisions

3. https://upload.wikimedia.org/wikipedia/commons/7/74/Quadtree%2C_graphical_and_tree_representation.png

4. https://es.wikipedia.org/wiki/%C3%81rbol_AABB

5. http://allenchou.net/wp-content/uploads/2014/02/thinAABB-tree.png

6. http://www.mastergraficos.com/wp/wpcontent/uploads/2015/06/deteccioncolisiones.pdf