INFORME DE LA ENTREGA 3

Manuel García Jiménez
Universidad EAFIT
Colombia
megarcij@eafit.edu.co

Isaías Labrador Sánchez
Universidad EAFIT
Colombia
ilabradors@eafit.edu.co

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

RESUMEN

Debido al cambio climático el numero de abejas en el mundo está decayendo, lo cual es problema porque ellas son importantes en el proceso de polinización de las flores. Para combatir el problema se ha propuesto sacar al aire libre pequeños drones que lleven polen de flor en flor.

Estos drones necesitan saber cuándo están a punto de chocar entre si para esquivarse entre sí. Problemas relacionados a este los hay en varios ámbitos; la evitación de choques de carros, la formación de superficies uniformemente densas y la animación de videojuegos son todos ejemplos de casos en los cuales es importante crear algoritmos para evitar que hallan objetos en un mismo lugar.

Aquí como solución se ha creado un Quadtree donde se almacenan todas las abejas y se comparan las posiciones de abejas que pertenecen al mismo "nodo" o están cerca desde un punto de vista geométrico.

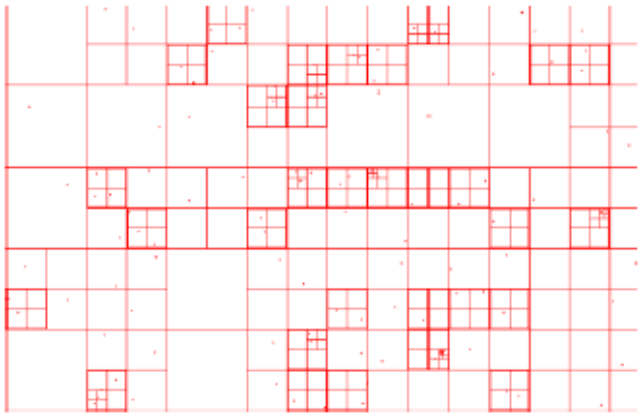
Como resultado al comparar las posiciones en un instante de las abejas, se puedo conocer cuales están lo suficientemente cerca para avisar que están en vía de colisión.

En conclusión, hay muchas maneras distintas de hacer esto, pero las mejores y más optimas son aquellas que evitan comparar las posiciones de abejas que están demasiado lejos como para estar en vía de colisión. Poniendo las abejas en un quadtree se logra esto.

ENTREGAS 1 Y 2

Lo hecho en las entregas 1 y 2 está también puesto en esta carpeta en GitHub llamada "proyecto".

COLISIONES DE ABEJAS CON CUADTREE



Gráfica 1: Grafica de un campo dividido en varios Quadtrees. En cada hoja del árbol hay una sola abeja, la cual posee una posición denotada por una latitud y una longitud.

La implementación se puede encontrar en:

https://github.com/manuGarciaJ/ST0245-033/tree/master/proyecto/codigo

ANALISIS DE COMPLEJIDAD

Complejidades:	
Cuadrado	O(1)
Quadtree	O(1)
Quadtree.show()	O(log(n))
Quadtree.insert()	O(log(n))
Quadtree.createBees()	O(n)
Circulo	O(1)
Abeja	O(1)
Quadtree.query()	O(log(n))
Drawing.enviar()	O(n)
leer()	O(n)
Drawing	O(1)

CRITERIOS DE DISEÑO

La estructura de datos quadtree fue escogida de entre varias otras por los siguientes motivos:

- Permite ver los objetos puestos dentro de ella como puntos con coordenadas bidimensionales, lo cual lo hace más apropiado para este tipo de problemas que un árbol binario.
- Al posicionar los objetos teniendo en cuenta su posición, objetos cercanos geométricamente quedan cerca dentro del árbol, lo cual hace que al buscar objetos cercanos se puedan ignorar los que están lejos, lo cual hace que esta acción sea O(log n). Esto cual es mucho mejor que comparar las posiciones de cada abeja con todas las demás, lo cual es O(n^2).

TIEMPO DE EJECUCION

	10 abejas	100	1000	10000
tiempo (ms)	131	147	1014	102380

Tabla 2: Tiempos de ejecución de las operaciones de la estructura de datos con diferentes conjuntos de datos.

MEMORIA

memoria	10abejas	100	1000	10000
bytes	3183800	3278950	1921005	6602000

Tabla 3: Consumo de memoria de la estructura de datos con diferentes conjuntos de datos.

CONCLUSIONES

Es cierto que los drones pueden ser de gran ayuda para la polinización de las flores en un lugar. Para que sea posible esto, estos drones tienen que ser capaces de detectar posibles colisiones entre sí.

A través de esta implementación de un quadtree donde en los nodos hay abejas, hemos podido disminuir la complejidad de hacer esto de O(n^2) a O(nlogn), lo cual es una mejora considerable.

Esta última solución es bastante mejor que la de la segunda entrega, pues esta permite la lectura de archivos de texto, lo cual permite probar el programa con muchas abejas.

En el futuro se pueden crear algoritmos para que después de que una abeja reconozca estar en vía de colisión cambie de trayectoria y mejorar este programa para que el que 2 abejas se muevan la una hacia la otra sea un criterio que se tenga en cuenta pasa decidir si están en vía de colisión.

REFERENCIAS

El código utilizado para esta solución del problema está basado en el código y seudocódigo expuesto en las siguientes fuentes:

Shiffman(2018), Quadtree (Version 1.0), [source code], https://github.com/CodingTrain/website/tree/master/CodingChallenges/CC_98_1_QuadTree

Wikipedia (2018), Pseudo code (Version 1.0) [Pseudo code], https://en.wikipedia.org/wiki/Quadtree#Pseudo_code