# TITLE (A SHORT DESCRIPTION OF THE PROJECT, BEWEEN 8 AND 12 WORDS

Agustin Restrepo Cadavid
Universidad EAFIT
Colombia
arestrepoc@gmail.com

Sebastian Gonzalez Arango
University EAFIT
Colombia
sgonzalez1@eafit.edu.co

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

**NOTE: To have more information about the sections in this report, please read "*Guía para la realización del Proyecto Final de Estructura de Datos 1.*" For the final version of this report: 1. Detele all text in red. 2. Adjust spaces among words and paragraphs. 3. Change the color of all the texts to black. You should also keep in mind the meaning of the colors:**

**Black text** = To complete for the 1st deliverable

Blue text = To complete for the 2nd deliverable

Violet text = To complete for the 3rd deliverable

**ABSTRACT**
This problem is important because as the population of bees decrease over time, the flowers are going to have difficulty with pollinisation. If it is impossible to stop the bee population from going extinct, one of the possible solutions, is to develop a robot that can replace the job of the bees. If technology is advanced enough to create drones that are small and capable, then collisions between the bees must be addressed. If two bees are in the same area they might collide, so it is necessary and important to create an algorithm that detects when two bees are 100 meters or closer from each other. Problems like collision in terms of objects that do not rotate, and collision in large data sets have to be solved.

To write the abstract, you should answer the following questions in a paragraph: What is the problem? Why is the problem important? Which are the related problems? What is the solution you proposed?, what results did you achieve? , What are the conclusions of this work? Abstract should have at most 200 words.

**Keywords**
AABB tree
Quatree
Spatial Hashing
Dynamic AABB tree
Node

**ACM CLASSIFICATION Keywords**
Only keywords in the ACM classification which can be found at http://bit.ly/2oVE5 2i
You cannot create your own keywords here.

Example: Theory of computation → Design and analysis of algorithms → Graph algorithms analysis → Shortest paths

## 1. INTRODUCTION

The decrease of bees in the environment is a growing problem for agriculture. This is because bees are needed to pollinate many of the plant species. If there are no bees, the plants do not make seeds and can not reproduce. Bees are dying because of pesticides that are put in the plants they pollinate, climate change, and other human made problems. A solution to this problem is to repopulate the bees, but at the rate they are dying, it is important to look for new solutions in case humans can not stop the bees from going extinct. Another solution is to create robots to substitute bees and carry out the process of pollination. The robots have to be small enough and smart enough to carry this out, but a database is needed to keep them organized.

## 2. PROBLEM

When working with these robots it is clear that a main problem, is to keep them organized. They must not be in the same area pollinating the same plants. An algorithm must be set to solve the problem of bees colliding. It must be efficient and work with a large quantity of bees.

## 3. RELATED WORK

### 3.1 AABB Collision for objects in 2D.

When a program has two objects that can be bounded by a box, and can not collide, it is important to create an algorithm that determines if they are colliding. AABB collision, or Axis-Aligned Bounding Box, is an algorithm that prevents two objects, or more specifically, two bounding boxes around an object, from colliding. The algorithm works by checking the edges of boxes, and determining if they are before the opposite edge and after the same edge of another box. For example, if two boxes are on a 2d plane, and the left edge of a box is before the right edge of another and after the left edge, and the same happens in the top edge and bottom edge, the boxes are colliding. It is important to know that the boxes can not rotate. The algorithm will be given the coordinate of the top right corner and bottom left corner of the boxes and the width and height. With this, it determines if any of the sides are overlapping and returns a true or false.

### 3.2 Large amount of Objects

When collision detection is needed for a large amount of objects, the time taken by the computer can increase dramatically as the amount of objects can be huge. This becomes a problem when a lot of objects have to be analysed for collision. Quadtree fixes this, by creating small subsections of the 2d plane, in which more than 1 object are located, then it checks these objects for collision with each other and only with each other, instead of checking for collision with all the other objects.

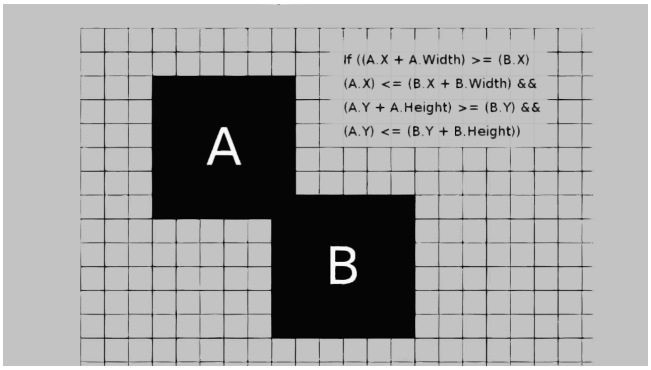
### 3.3 Time taken to calculate collisions

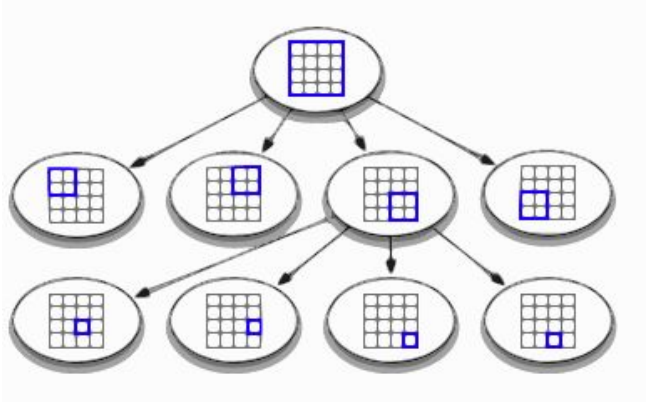
Testing out a high amount of objects during a collision test can be time demanding and it would most likely generate more problems. Spatial hashing fixes all of these problems as it can optimize the objects used in the collision test and the time used. A spatial hash is a dimensional extension of a hash table and what this does is it can store data that's used inside a function to store its value in some sort of buckets, and you use the references of these to find the different information used.

### 3.4 Memory to calculate and coordinate objects

During the tests done, a lot of memory should be used as this problem should be seen as a possible world change. Whats meant by this; to calculate all of the possible outcomes throughout the world, a lot of memory and computing power would be needed. A way to fix this problem with an easy solution would be to optimize all of the algorithms in order to make the whole process work in harmony. Other than the plausible solutions mentioned before, there really aren't more approaches to make this work out as a lot of study has been done for this research by other people. but dynamic AABB trees could also help out by inserting information, removing it and updating it to get important data about the collisions.

Graphs:



## 4. Title of the first data structure designed

Design a data structure to solve the problem and make a figure explaining it. Do not use figures from the Internet.

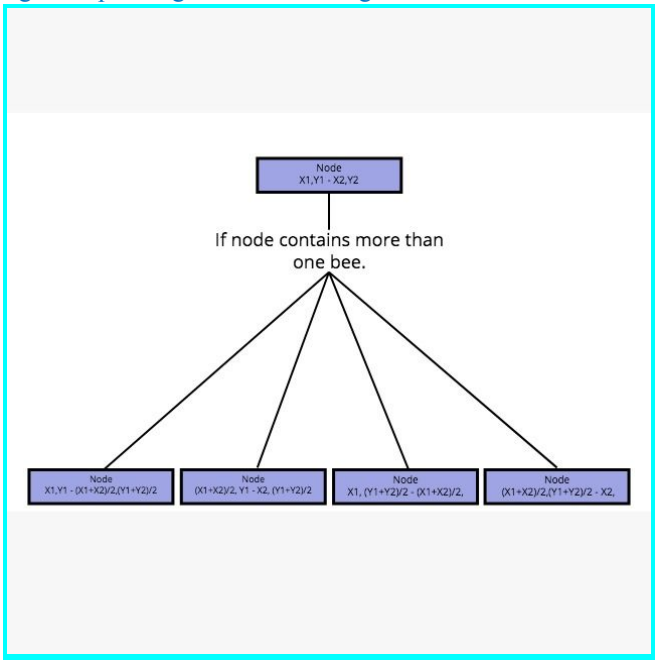



**Figure 1:** Nodes split into four separate blocks when encountered with more than 1 bee in 1 block

### 4.1 Operations of the data structure

A Quadtree Data structure is appropriate to solve this problem because it will drastically reduce the amount of operations the computer must run.

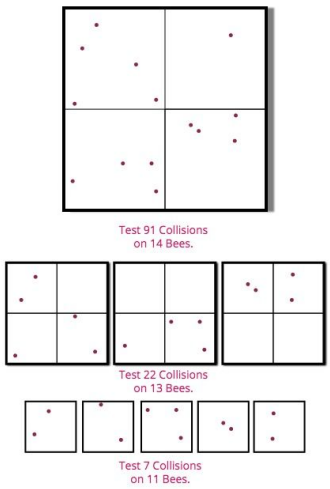



**Figure 2:** The methods used in this data structure are add, search and delete

### 4.2 Design criteria of the data structure

This data structure is very useful for this problem because it is a problem that has to compare many different data points. The amount of time it takes to analyze all points grow exponentially as points are added because each new point has to be compared to all the other points. The Quad-tree algorithm not only marks some points are un necessary to compare, but allows the computer to only analyze likely collisions. The main reason why one would use this algorithm, is to optimize time.

### 4.3 Complexity analysis

Derive the complexity of each operation of the data structure for the worst case and best case, As an example, this is a way to report the complexity analysis:

| Method | Complexity |
|---|---|
| Add | O(n+log(m)) |
| Search | O(n+log(m)) |
| Delete | O(n+log(m)) |

**Table 1:** Number of data points = n

With of Quad-Tree = m

Even thought the complexity does not seem to different from one of a data structure without a quad tree, it is a lot better. This is because the complexity is calculated with the worst case scenario in mind, in which the complexity would be very similar, but in most cases the complexity when using the Quad-Tree is a lot more efficient.

### 4.4 Execution time

Measure (I) execution time and (II) memory used by the operations of the data structure, for the data set found in the .ZIP file.

Measure the execution time and memory used 100 for each data set and for each operation of the data structure. Report the average values.

| | *Conjunto de Datos 1* | *Conjunto de Datos 2* | *...Conjunto de Datos n* |
|---|---|---|---|
| *Creación* | 10 sg | 20 sg | 5 sg |
| *Operación 1* | 12 sg | 10 sg | 35 sg |
| *Operación 2* | 15 sg | 21 sg | 35 sg |
| *Operación n* | 12 sg | 24 sg | 35 sg |

**Table 2:** Execution time of the operations of the data structure for each data set.

### 4.5 Memory used

Report the memory used for each data set

| | *Conjunto de Datos 1* | *Conjunto de Datos 2* | *...Conjunto de Datos n* |
|---|---|---|---|
| Consumo de memoria | 10 MB | 20 MB | 5 MB |

**Table 3:** Memory used for each operation of the data structure and for each data set data sets.

### 4.6 Result analysis

Explain the results obtained. As an example, compare different implementation of the data structure and report the comparison in a table or graph.

**Tabla de valores durante la ejecución**

| Estructuras de autocompletado | LinkedList | Arrays | HashMap |
|---|---|---|---|
| Espacio en el Heap | 60MB | 175MB | 384MB |
| Tiempo creación | 1.16 - 1.34 s | 0.82 - 1.1 s | 2.23 - 2.6 s |
| Tiempo búsqueda ("a") | 0.31 - 0.39 s | 0.37 - 0.7 s | 0.22 - 0.28 s |
| Tiempo búsqueda ("zyzzyvas") | 0.088 ms | 0.038 ms | 0.06 ms |
| Búsqueda ("aerobacteriologically") | 0.077 ms | 0.041 ms | 0.058 ms |
| Tiempo búsqueda todas las palabras | 6.1 - 8.02 s | 4.07 - 5.19 s | 4.79 - 5.8 s |

**Table 4:** Analysis of the results

## 5. Title of the last data structure designed

Design a data structure to solve the problem and make a figure explaining it. Do not use figures from the Internet.


**Figure 1:** Linked List of persons. Una person is a class that contains a name, id number and photo.

### 5.1 Operations of the data structure

Design the oepration of the data structure to solve the problem efficiently. Include one figure to explain each operation.

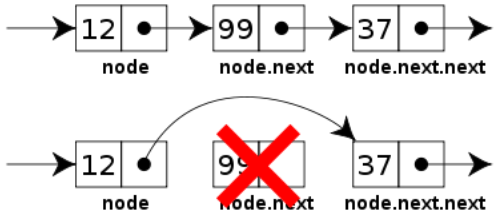



**Figure 2:** Delete operation of a Linked List.

### 5.2 Design criteria of the data structure

Explain objetive criteria that you considered to design the data structure. Examples of objective criteria are efficiency in time and space. Non-objective criteria will lower your grade. Examples of non-objetive criteria are: “I was sick”, “it was the first data structure that I found on the Internet”, “I did it on the last day before deadline”, etc. Remember: This is 40% of the project grade.

### 5.3 Complexity analysis

Derive the complexity of each operation of the data structure for the worst case and best case, As an example, this is a way to report the complexity analysis:

| Mètodo | Complejidad |
|---|---|
| Bùsqueda Fonètica | O(1) |
| Imprimir bùsqueda fonètica | O(m) |
| Insertar palabra busqueda fonètica | O(1) |
| Búsqueda autocompletado | O(s + t) |
| Insertar palabra en TrieHash | O(s) |
| Añadir bùsqueda | O(s) |

**Table 5:** Table to report complexity analysis

### 5.4 Execution time

Measure (I) execution time and (II) memory used by the operations of the data structure, for the data set found in the .ZIP file.

Measure the execution time and memory used 100 for each data set and for each operation of the data structure. Report the average values.

| | *Conjunto de Datos 1* | *Conjunto de Datos 2* | *...Conjunto de Datos n* |
|---|---|---|---|
| *Creación* | 10 sg | 20 sg | 5 sg |
| *Operación 1* | 12 sg | 10 sg | 35 sg |
| *Operación 2* | 15 sg | 21 sg | 35 sg |
| *Operación n* | 12 sg | 24 sg | 35 sg |

**Table 6:** Execution time of the operations of the data structure for each data set.

### 5.5 Memory used

Report the memory used for each data set

| | *Conjunto de Datos 1* | *Conjunto de Datos 2* | *...Conjunto de Datos n* |
|---|---|---|---|
| Consumo de memoria | 10 MB | 20 MB | 5 MB |

**Table 7:** Memory used for each operation of the data structure and for each data set data sets.

### 5.6 Result analysis

Explain the results obtained. As an example, compare different implementation of the data structure and report the comparison in a table or graph.

**Tabla de valores durante la ejecución**

| Estructuras de autocompletado | LinkedList | Arrays | HashMap |
|---|---|---|---|
| Espacio en el Heap | 60MB | 175MB | 384MB |
| Tiempo creación | 1.16 - 1.34 s | 0.82 - 1.1 s | 2.23 - 2.6 s |
| Tiempo búsqueda ("a") | 0.31 - 0.39 s | 0.37 - 0.7 s | 0.22 - 0.28 s |
| Tiempo búsqueda ("zyzzyvas") | 0.088 ms | 0.038 ms | 0.06 ms |
| Búsqueda ("aerobacteriologically") | 0.077 ms | 0.041 ms | 0.058 ms |
| Tiempo búsqueda todas las palabras | 6.1 - 8.02 s | 4.07 - 5.19 s | 4.79 - 5.8 s |

**Table 8:** Analysis of the results

## 6. CONCLUSIONS

To write the conclusions, proceed in the following way. 1. Write a paragraph with a summary, the most important issued of the report. 2. In another paragraph, explain the most important results that you obtained with the last data structure you designed. 3. Compare your first solution with the last solution. 4. At last, explain future work, a future continuation of this project. You can also mention in the conclusions, technical problems that you had during the development of the data structure and its implementation and you solved them.

### 6.1 Future work

Answer, what would you like to improve in the future? How would you like to improve your data structure and its implementation?


## ACKNOWLEDGEMENTS

Identify the kind of acknowledgment you want to write: for a person or for an institution. Consider the following guidelines: 1. Name of teacher is not mentioned because he is an author. 2. You should not mention websites of authors of articles that you have not contacted. 3. You should mention students, teachers from other courses that helped you.

internet ni autores de artículo leídos con quienes no se han contactado. 3. Los nombres que sí van son quienes ayudaron, compañeros del curso o docentes de otros cursos.

As an example: This research was supported/partially supported by [Name of Foundation, Grant maker, Donor].

We thank for assistance with [particular technique, methodology] to [Name Surname, position, institution name] for comments that greatly improved the manuscript.


## REFERENCES


1. AABB. (2018, April 14). Retrieved April 16, 2018, from https://en.wikipedia.org/wiki/AABB
2. Quad Tree. (2018, February 07). Retrieved April 16, 2018, from https://www.geeksforgeeks.org/quad-tree/
3. Spatial hashing implementation for fast 2D collisions. (2009, June 13). Retrieved April 16, 2018, from https://conkerjo.wordpress.com/2009/06/13/spatial-hashing-implementation-for-fast-2d-collisions/
4. Dynamic AABB Tree. (2013, October 21). Retrieved April 16, 2018, from http://www.randygaul.net/2013/08/06/dynamic-aabb-tree/
5. Myopic Rhino. (2009, October 01). Spatial Hashing. Retrieved April 16, 2018, from https://www.gamedev.net/articles/programming/general-and-gameplay-programming/spatial-hashing-r2697/
6. J. (n.d.). Introductory Guide to AABB Tree Collision Detection. Retrieved April 16, 2018, from http://www.azurefromthetrenches.com/introductory-guide-to-aabb-tree-