TECHNOLOGY AND ALGORITHMS: A SOLUTION TO BEE´S EXTINTION

Isabella Arango Restrepo
Universidad EAFIT
Colombia
iarangor1@eafit.edu.co

Juan David Rengifo Castro
Universidad EAFIT
Colombia
jdrengifoc@eafit.edu.co

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

ABSTRACT

The extinction of the human race is normally seen as sci-fi; however, this could be closer than expected. According to the IUCN (International Union for Conservation of Nature) “nearly one in ten wild bee species face extinction in Europe”. This is something to pay close attention, due to these little pollinators are vital for earth’s life. In fact, the bees are fundamental for plant’s reproduction. In consequence, its extinction implies an authentic catastrophe, increasing the famine, the contamination, and the water reduction. This problem can be solved with the application of efficient data structures in order to control the distribution of robot bees, as hash map that allows to determine bees in risk of collision.

1. INTRODUCTION

It is not a secret that the human damage to earth is a big deal. The overpopulation and our self-destructive behavior have deteriorated the life on earth in almost all the aspects, and bees are not the exception. These little pollinators are an essential part of our life despite we are conscious or not. Sadly, the bee is in danger of extinction. According to BIP (Bee Informed Partnership) between 1988 and 2015 the 42.1% of the colonies died [1].

Keywords

Collision problems, algorithmic solution, extinction of bees.

ACM CLASSIFICATION Keywords

Computing methodologies → Artificial intelligence → Search methodologies → Discrete space search
Computing methodologies → Artificial intelligence → Planning and scheduling → Planning for deterministic actions
Computing methodologies → Modeling and simulation → Simulation types and techniques → Data assimilation
Computing methodologies → Modeling and simulation → Simulation types and techniques → Agent / discrete models

2. PROBLEM

The bee extinction is a major problem, due to this little animal is essential for plants reproduction. The extinction or decrease of its population avoid pollination, something that would increase the famine, the contamination and the water reduction. For this reason, this must be solved as soon as possible.

3. RELATED WORK

Collision detection is a very common problem that is applied in the programming of videogames, searches in multidimensional spaces, analysis of images and even in real life problems. To give an effective solution to this, different data structures are used to reduce the detection time of possible collisions.

3.1 Spatial hash.

A spatial table is a data structure that offers a faster way to solve a collision detection problem. It is a 2D o 3D table that consists of dividing the space into several cells and locating the objects that are in it according to their spatial coordinates. Then, these are placed in a hash table in such a way as to achieve an effective location of those found in the same frame.

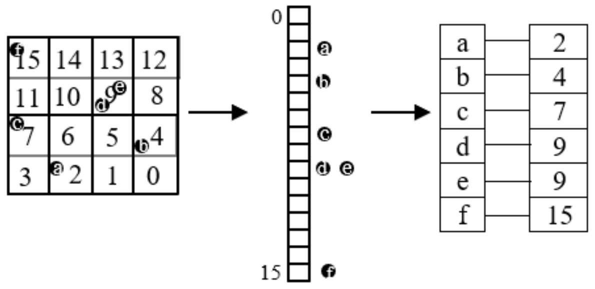

Figure 1: Spatial hash representation.

3.2 AABB tree.

This data structure is very useful and effective at searching objects collisions in an undetermined three-dimensional space. It consists of boxes that are aligned and represent the coordinates of each element [2]. Then, it is very easy to determine if they present an intersection, because it is enough to look if the following is met or not:

maxx1>minx2 & minx1<maxx2 & maxy1>miny1 & miny1<maxy2,

where maxx1 refers the x-coordinate of the upper right corner of object1, maxy1 to the y-coordinate of the same corner, minx1 and miny1 for the lower left corner and so on for the object2.

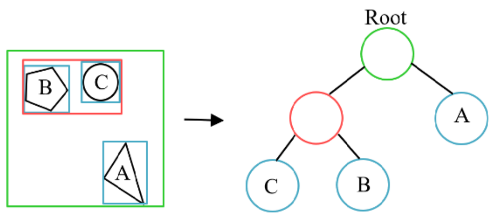

Figure 2: AABB tree representation.

3.3 Quadtree.

The quadtree is a type of spatial decomposition that consists of dividing the space recursively into four cells and when one reaches its maximum capacity, it is divided again into four cubes. They are too useful when detecting collisions, because it allows to evaluate only those objects that may collide, instead of reviewing the entire list of objects that are in space. For this reason, it takes less time to determine those elements that are very close.

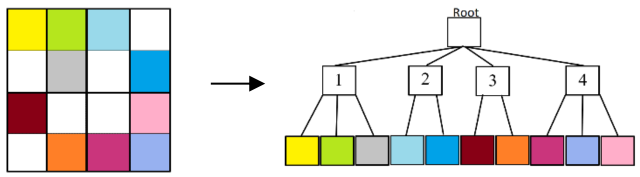

Figure 3: Quadtree representation.

3.4 R-Tree.

This data structure is very used in problems with multidimensional spaces, such as map locations. In addition, it is one of the most efficient methods to find objects that are within a space delimited by a rectangle, which allows searching in a specified area of the tree, instead of searching in each sector that composes the space. An advantage is that it allows to work and analyze large volumes of data without spending a lot of time [4].

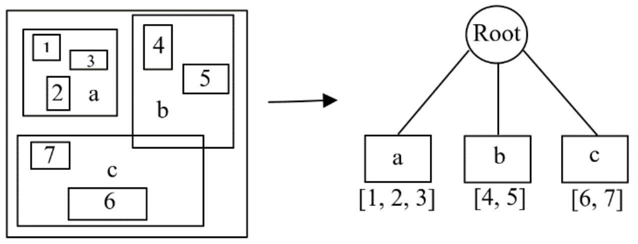

Figure 4: R-Tree representation.

4. ArrayList

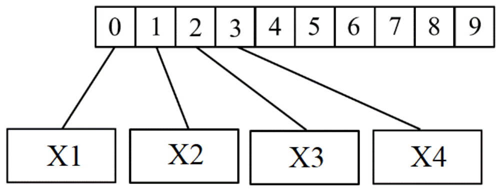

Figure 5: ArrayList of bee´s coordinates. A coordinate is double number that represent the latitude, longitude or height.

The data structure used to solve the problem is an ArrayList that contains the coordinates of the bees. In this case, other three arraylists are used to stock each component of the coordinates.

4.1 Operations of ArrayLists

The principal operation for this problem is the one that determine which bees are with less than 100m between them.

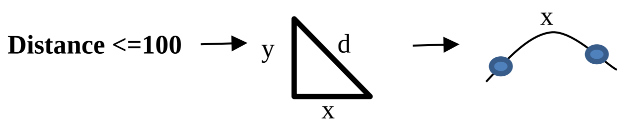

Figure 6: Close Bees operation with arrayLists.

4.2 Design criteria of ArrayLists

Choosing an appropriate data structure that helps to implement a functional and efficient code is very important at the time of thinking in a solution to solve an algorithmic problem. In this case, the ArrayList structure seems to be useful to solve collision problems due to its properties and the ease of working with it. For example, it allows to study different groups of data without knowing exactly their size or if it is going to increase or decrease. Is important to mention, that our method with the highest complexity needs the operation get(), and clearly arraylist have the a very low complexity in this operation against other data structures, such as a Linked list, stack, queue, or a binary three. Also, this data structure has complexities smaller or equals to O(n/2), which is efficient compared to the complexities of other structures. But it is necessary to highlight that this is not the most adequate structure to solve this type of problems, because it becomes a program if it is to work with very large data.

4.3 Complexity analysis

Method	Complexity
closeBees	O(n^2)

Table 1: Table to report complexity analysis.

4.4 Execution time

Operations	10 bees	100 bees	1000 bees	10000 bees
closeBees	3.33ms	23.5ms	394.67ms	23073.67ms

Table 2: Execution time of the method CloseBees for each data set. Time is in milliseconds.

4.5 Memory used

	10 bees	100 bees	1000 bees	10000 bees
Memory used	2MB	3MB	4MB	7MB

Table 3: Memory used for each data set.

4.6 Result analysis

Finally, ArrayLists are a very efficient data structure to solve collision problems because they present very low time complexity compared to other structures and this can be seen in the execution time that it takes to analyze data groups of up to 10000 elements.

5. Hash Table

The hash table is an efficient data structure for collision problems since it has the lowest complexity for insertion, search and deletion (see figure 1).

5.1 Operation of Hash Table

Considering the max and minimum coordinates of the map, divide the space into several parallelepipeds, each of them with the main diagonal of 50m, height of 40m, long and wide of 21.2m, every bee will be added into one of the parallelepipeds according to its spatial coordinates. Different bees will be in the same space, so they will collide with them and with the bees of the parallelepipeds that are around them.

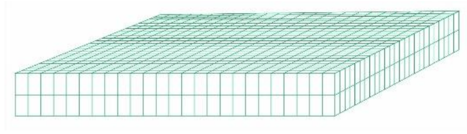

Figure 7: Map divided in several parallelepipeds of 50m main diagonal

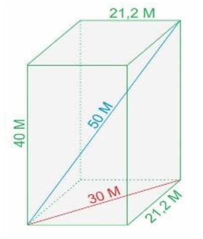

Figure 8: Measures of each parallelepipeds.

In addition, bees that share a parallelepiped are added to a linked list.

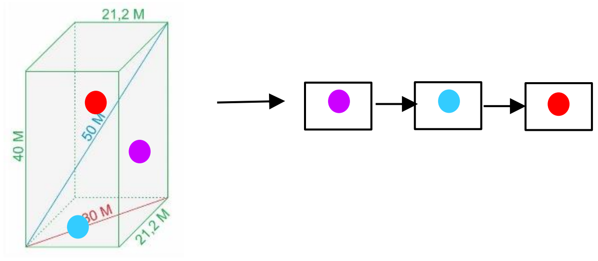

Figure 9: Addition of bees that are in the same parallelepiped to a linkedlist.

5.2 Desing criteria of Hash Table

Creating an efficient algorithm that works fast for any number of objects is the dream of every programmer, but this can only be achieved if the appropriate data structures are used. In this case, Hash table seems to a good one to solve the studied problem because its structure allows to do any operation in constant time regardless of the number of elements. To continue with a low time complexity, is important to choose another efficient data structure to organize the elements in the same parallelepiped, for this the linkedList are recommended because it has a constant complexity for insertion and deletion.

5.3 Complexity analysis

To classify the bees, its necessary to do a loop that go over every bee. Inside it, every operation will be constant because of the insertion to a linkedList. For that reason, this method will have a complexity of O(n). The second process is to compare the proximity, for this it is necessary to go over every parallelepiped, which is a constant number regardless of the number of bees. Inside that loop the operations will also be constant. Therefore, this method also has a time complexity of O(n).

6. CONCLUSIONS

Technology can solve current problems that put human life at risk. However, since technology is a relatively new science, every time a strategy must be looked for that allows to solve problems in a more efficient way.

Hash table are a very useful data structure to solve collision problems. However, it is much more efficient if it is treated with large data values, because this structure grows exponentially. For this reason, although the first algorithm was not so efficient for very large samples, it can be used if small samples are evaluated, otherwise it is recommended to use the second one to reduce the execution time if that is what is desired.

6.1 Future work

This project is very useful to realize the role that technology plays in the actual world and how useful it is if it is used correctly. It would be a very big step if in the future this work is carried out in order to prolong human existence on planet earth, because the work done by bees is of great importance for humanity.

REFERENCES

1. LA VANGUARDIA. Available from https://www.lavanguardia.com/natural/20161005/41771284333/abeja-peligro-humanos.html (2016); accessed 24 August 2018.
2. Runestone. Available from http://interactivepython.org/runestone/static/pythoned/SortS earch/TransformacionDeClaves.html (2012); accessed 24 August 2018.
3. Azure from the trenches. Available from: https://www.azurefromthetrenches.com/introductory-guide- to-aabb-tree-collision-detection/ (2017); accessed 24 August 2018.
4. Wkipedia. Available form https://en.wikipedia.org/wiki/R- tree (2018); accessed 24 August 2018.
5. Evantotuts+. Available from https://gamedevelopment.tutsplus.com/tutorials/quick-tip- use-quadtrees-to-detect-likely-collisions-in-2d-space-- gamedev-374 (2018); accessed 24 August 2018.
6. Wikipedia. Available from https://en.wikipedia.org/wiki/Quadtree#Some_common_uses_of_quadtrees (2018); accessed 24 August 2018.
7. The IUCN Red List of Threatened Species. Available from http://www.iucnredlist.org/news/nearly-one-in-ten-wild-bee-species-face-extinction-in-europe-while-the-status-of-more-than-half-remains-unknown-iucn-report (2018); accessed 24 August 2018.
8. Big-O Cheat Sheet. Available from http://bigocheatsheet.com/ (2018); accessed 20 August 2018.
9. Information Technology Gems. Available from http://infotechgems.blogspot.com/2011/11/java-collections-performance-time.html (2018); accessed 20 August 2018.
10. Enrique Munguía. Available from http://www.enrique7mc.com/ (2018); accessed 22 August 2018