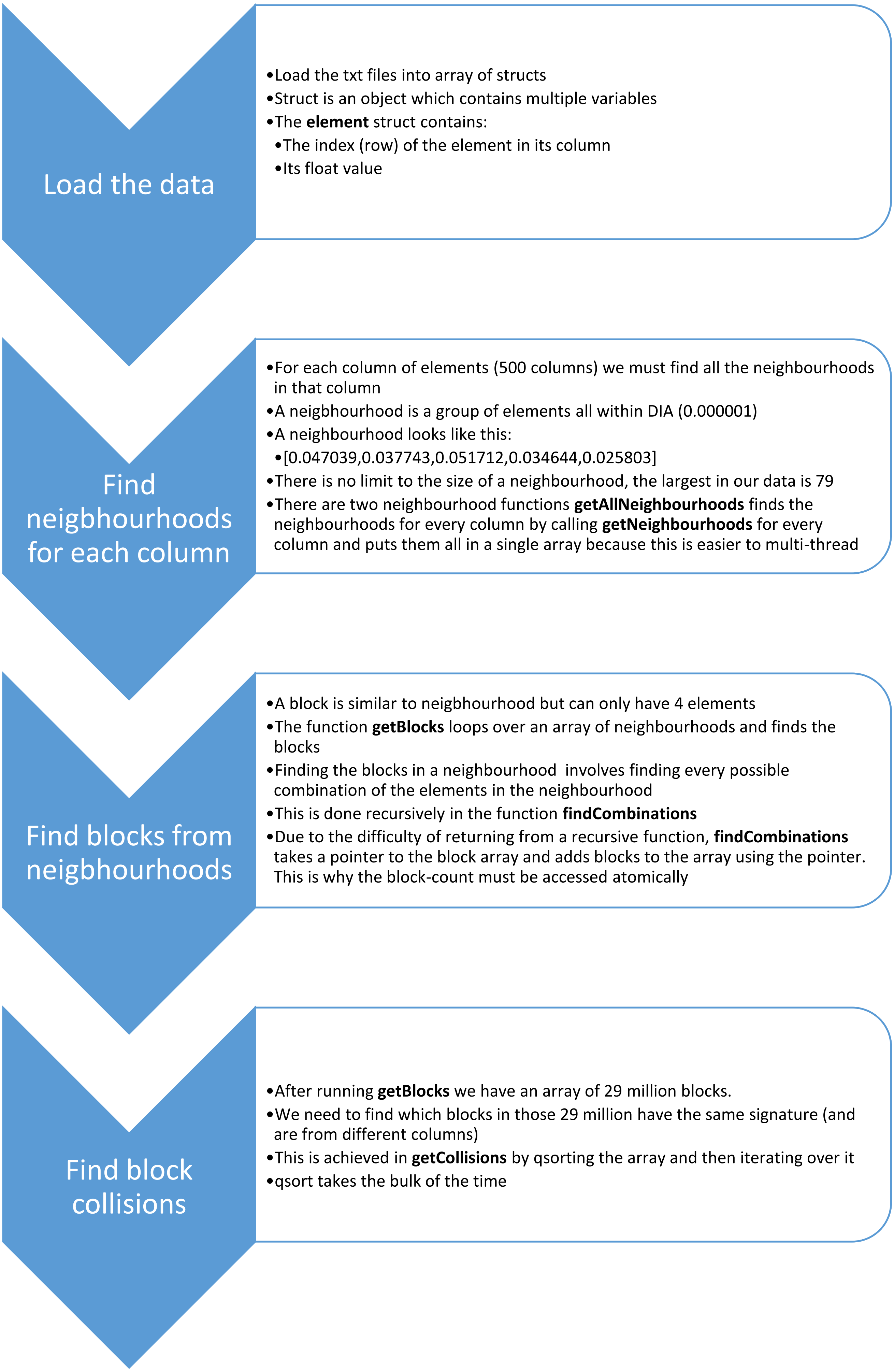

Load the data

- Load the txt files into array of structs
- Struct is an object which contains multiple variables
- The **element** struct contains:
  - The index (row) of the element in its column
  - Its float value

Find neigbhourhoods for each column

- For each column of elements (500 columns) we must find all the neighbourhoods in that column
- A neigbhourhood is a group of elements all within DIA (0.000001)
- A neighbourhood looks like this:
  - [0.047039,0.037743,0.051712,0.034644,0.025803]
- There is no limit to the size of a neighbourhood, the largest in our data is 79
- There are two neighbourhood functions **getAllNeighbourhoods** finds the neighbourhoods for every column by calling **getNeighbourhoods** for every column and puts them all in a single array because this is easier to multi-thread

Find blocks from neigbhourhoods

- A block is similar to neigbhourhood but can only have 4 elements
- The function **getBlocks** loops over an array of neighbourhoods and finds the blocks
- Finding the blocks in a neighbourhood involves finding every possible combination of the elements in the neighbourhood
- This is done recursively in the function **findCombinations**
- Due to the difficulty of returning from a recursive function, **findCombinations** takes a pointer to the block array and adds blocks to the array using the pointer. This is why the block-count must be accessed atomically

Find block collisions

- After running **getBlocks** we have an array of 29 million blocks.
- We need to find which blocks in those 29 million have the same signature (and are from different columns)
- This is achieved in **getCollisions** by qsorting the array and then iterating over it
- qsort takes the bulk of the time