Imperial College London
Department of Computing

# Tree of Life Visualisation
# By

Kai Zhong(kz12)

September 2013

Submitted in part fulfilment of the requirements for the degree of
Master of Science in Computing of Imperial College London

# Abstract

The storage, searching and visualisation of big data has become an increasingly important issue in computer science. One of the technique for visualisation of huge hierarchies is the interactive fractal inspired graph(IFIG), which employs fractal geometry to place limitless amounts of information on a single page and show, hide information by the simple actions of zooming and panning. This method is created by Imperial College academic Dr. James Rosindell, who has also employed it for tree of life visualisation where it solved a significant outstanding problem in evolutionary biology: how can large evolutionary trees be visualised effectively for science, education and public outreach.

While the software written by Dr. James Rosindell runs on javascript and html5, my main task is to develop an android application based on his code. Compared with personal computer, the calculation ability of a mobile device is relatively low and the RAM capability is more restricted. The project involves how to use multiple thread and temporary bitmaps to solve performance issue, which greatly shorten the loading time as well as the response time of user interaction. When dealing with larger trees, because of the limitation of RAM capability, dynamical loading and freeing objects is employed in this project, such that the usage of RAM could be maintained under the limitation of an android application. Both the technique of multiple thread and using dynamical loading and freeing objects could be employed in the website software.

# Contents

# List of Tables

# List of Figures

# 1 Introduction

The storage, searching and visualisation of big data has become an increasingly important issue in computer science. One of the technique for visualisation of huge hierarchies is the interactive fractal inspired graph(IFIG), which employs fractal geometry to place limitless amounts of information on a single page and show, hide information by the simple actions of zooming and panning. This method is created by Imperial College academic Dr. James Rosindell, who has also employed it for tree of life visualisation where it solved a significant outstanding problem in evolutionary biology: how can large evolutionary trees be visualised effectively for science, education and public outreach.

While the software written by Dr. James Rosindell runs on javascript and html5, my main task is to develop an android application based on his code. Compared with personal computer, the calculation ability of a mobile device is relatively low and the RAM capability is more restricted. The project involves how to use multiple thread and temporary bitmaps to solve performance issue, which greatly shorten the loading time as well as the response time of user interaction. When dealing with larger trees, because of the limitation of RAM capability, dynamical loading and freeing objects is employed in this project, such that the usage of RAM could be maintained under the limitation of an android application. Both the technique of multiple thread and using dynamical loading and freeing objects could be employed in the website software.

## 1.1 Contributions

The storage, searching and visualisation of big data has become an increasingly important issue in computer science. One of the technique for visu-

alisation of huge hierarchies is the interactive fractal inspired graph(IFIG), which employs fractal geometry to place limitless amounts of information on a single page and show, hide information by the simple actions of zooming and panning. This method is created by Imperial College academic Dr. James Rosindell, who has also employed it for tree of life visualisation where it solved a significant outstanding problem in evolutionary biology: how can large evolutionary trees be visualised effectively for science, education and public outreach.

While the software written by Dr. James Rosindell runs on javascript and html5, my main task is to develop an android application based on his code. Compared with personal computer, the calculation ability of a mobile device is relatively low and the RAM capability is more restricted. The project involves how to use multiple thread and temporary bitmaps to solve performance issue, which greatly shorten the loading time as well as the response time of user interaction. When dealing with larger trees, because of the limitation of RAM capability, dynamical loading and freeing objects is employed in this project, such that the usage of RAM could be maintained under the limitation of an android application. Both the technique of multiple thread and using dynamical loading and freeing objects could be employed in the website software.

## 1.2 Structure of this Report

The storage, searching and visualisation of big data has become an increasingly important issue in computer science. One of the technique for visualisation of huge hierarchies is the interactive fractal inspired graph(IFIG), which employs fractal geometry to place limitless amounts of information on a single page and show, hide information by the simple actions of zooming and panning. This method is created by Imperial College academic Dr. James Rosindell, who has also employed it for tree of life visualisation where it solved a significant outstanding problem in evolutionary biology: how can large evolutionary trees be visualised effectively for science, education and public outreach.

# 2 OneZoom Software Structure

An introduction to how the data is read and how to control the visibility of a drawing element. Setting threshold for displaying element. Potential problem of the software now–¿ when we have larger drawing data set, the loading time and user operation would become seemless.

# 3 Code Refactoring

Redesign part of the code so that now that each drawing elements would take charge of their own drawing manner. Besides, the drawing process

# 4 Performance Issue

Two problems greatly affect the performance of this application. One is over long loading time, the other is slow user interaction response time. Both of the problems are largely influenced by the processing capacity of a mobile phone processor as well as the size of the dataset.

## 4.1 Loading Time

Before the phylogenies tree first gets drawn, the application needs to load data from a string and create objects for each logical unit in it, which represents a species or a genre of some species. Then pre calculation which involves all the nodes on the tree gets called in order to calculate the relative position of each node. Both the cost of loading data and the cost of pre calculation is linear. Hence, the loading time of a phylogenies tree is expected to increase linearly with the size of the data. By using a older version of the application, we can test the loading time of different phylogenies trees on a Sumsang Galaxy S3 device. When loading the mammal tree who has 5XXX species, the loading time is XXX seconds, while loading the bird tree who has XXXX species and tetropads tree who has XXXXX species, the loading time increases to XXX seconds and XXX seconds respectively. The increase of loading time with the increase of species can also be observed in the website software. (Figures of the comparison can be seen below). As the ambition of this project is to build a tree of all existing and existed species, which has over million species, the loading time of that tree could be intolerable.

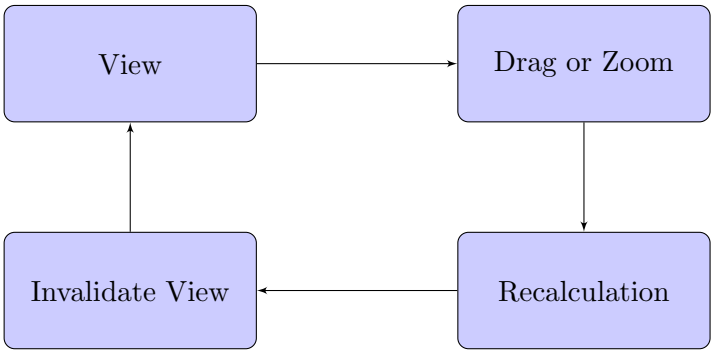
In order to shorten the loading time, several bitmaps of those trees are stored as resources. When a user select one of the tree, the corresponding bitmap would be drawn on canvas. In the meantime, another thread would

be created to load data and do pre calculation. The main UI thread and the calculation thread share a flag by which the view object knows whether the pre calculation is done or not. When the pre calculation is on progress and the user drag or zoom on the tree, the view object will use built in method of canvas class to scale or translate the tree. When pre calculation is finished, the calculation thread would send a message to the main UI thread so that it would invalidate itself.

Essentially, in the previous design, users spend time waiting for the data to be loaded and calculated. In the later design, the tree is drawn using bitmap even before the tree is created. Therefore, from a user perspective, the loading time of any tree could be reduced to unnoticeable.

## 4.2 User Interaction

Another factor that could affect user experience is the slow response time for user interaction. When user drag or zoom on the view, the tree gets recalculate to update the position of each node. Only after the recalculation is done, the view gets updated. Hence, when a user operates on a view, he or she must wait for a whole duration of the recalculation. Things get worse when the user drags the view. It would get a set of mouse move messages, hence that the recalculation would be called for equal times at a very short time. For instance, a single finger move from the middle of the screen to the middle right of the screen could generate XX to XX mouse move messages. The recalculation method will be called XX to XX times after that, which could cause great delay in displaying the tree.




However, the processing of recalculation method is far slower than the speed

of generating new message. Consequently, after the user drags the view, only two or three recalculation methods has been executed, more mouse move messages will be processed and more recalculation methods will be called but only the last one makes sense.

Describe how to use multiple thread and use message passing.

Solving both problems makes use of another thread. One use another thread to load image and when the loading is over, it will send message telling the main thread that it can update itself now. Another use of the thread is to do calculation. When user do some operation, it will update its arguments and sends message to the thread. Then that thread would start calculation. When calculation is ready, it would modify some variable so that the main thread knows that the calculation is ready. It also contains problems like how to synchronize the lastest user operation in the main UI thread and when there are continuous message sending to the aided thread, we should use mechanism like message queueing.

# 5 Memory Issue

# 6 Functionality

# Bibliography

[1] It is more convinient and faster to use `bibtex` instead of writing your bibliography manually.

[2] You can even use a tool like `jabref` to manage and maintain your database of references.