A new algorithm to improve the gathering of information on mobile devices

Scoil Mhuire Gan Smál – Blarney

Project number : 2632

By Robert Gabriel

SUMMARY / ABSTRACT

INTRODUCTION

INTRODUCTION

INTRODUCTION

INTRODUCTION

INTRODUCTION

SUMMARY / ABSTRACT

This must be a brief summary of the entire project from start to finish. I have taken out some (in blue) but more must be added.

The title of my project is “, A new algorithm to improve the gathering of information on mobile devices" and it is about the improvement of the gathering of information from databases using a new search algorithm that I have developed. Even though the title says it is for mobile devices it is not limited to mobiles. The idea is an algorithm that improves the search and information gathering from servers and the cloud in the blink of an eye on a mobile device. This means getting faster results from a mobile device using Global Positioning System (GPS), Wi-Fi?

I will go into more details about the algorithm later. The idea for the project came from me being at the BT young scientist business and innovation workshop where we were in a hotel and the Wi-Fi was really terrible. I wasn’t able to send emails or do anything, as the Internet access was consistently weak. It was also the same date as the release of the iPad 2 and I wasn’t able to watch it as the stream kept cutting out. After this I knew I wanted to do something about speed and reliable connection when searching the internet.

At home I quickly realised that my project from last year already had some of the code and ideas that I needed for my new idea. I was very lucky then that I happened to meet a local businessman, Mr Kieran Hyland. He was very interested in what I was doing and offered me an internship with his company for the summer.

It was a few weeks later, when my now mentor Mr Kieran Hyland came to look over my 2011 young scientist project (Automated monitoring of computer SQL servers by way of a merit system) and told me that he thought I already had a suitable algorithm inside my software. The algorithm I developed for last year’s project was based around users actions and choices on the computer, so what was stopping me from building a new updated version and using it to pull information from a server?

With the motivation to get faster and more reliable internet speed and the foundation of my aligorthim already built there was nothing stoping me from trying to develop an improved algorithm to improve the gathering of information on a mobile deivce.

Lots more ……

INTRODUCTION

I will start with a very short bullet point summary first:

- 1265 hours of work.
- That's 12 months of work.
- 7 versions of the algorthim.
- Tested in Cork Institute of Technology. Is this true?
- Patented ideas using the algorthim with Hymed and help from Mr Michael O Connor.
- Tested by former heads of Samsung developments.
- Seen by the head of IDA Ireland

I started my Young Scientist project back in February after coming back from the BT young scientist business and innovation workshop. As I metioned the Wi-Fi was quite week and was extremely unreliable and very slow. This was my first real taste of weak broadband speeds; I'm lucky to say I have quick speeds of 6 Megabyte in Blarney. So my goal was to make a programme to manage download of information for weak internet speeds. I really felt that I had fire in my belly again after I came back from Dublin.

For me, winning prizes doesn't mean a lot unless what you do is innovative and helps change peoples lives, even in a the small way. For example the software I made last year changed my live forever. It made my family life easier and showed me and other people that nothing can stop someone with dyslexia from reaching top marks in a field where being able to read is such an important skill. So from about mid-March I really got stuck into my new project.

The steps that went into creating an algorthim that can pull information for a server as quiclky as possblie came mostly from research but also some luck. So in the following pages I will start with how I started to where I finished and finally to were it got realy tested. I placed the algorithm in a software package to show the real power of it because as this qoute puts it

“An algorithm must be seen to be believed.”
Donald Knuth

INTRODUCTION

Back to March 2011, were I really started this project I didn’t do much research to begin with. What I did was try to make simple progammes from the start to get a grasp on the whole idea of pulling information and data from an external server. From my experiences from last years project about sql data bases I alrealy knew the basics of handling sql data from within a progamme. Now the main aim was to get it working on a live server or a cloud database. First I looked over last years project code (which I now think I did a horrible job of writing).

```
<add name="Tallus3.Properties.Settings.TallusDatabaseDataSet3ConnectionString"
connectionString="Data Source=U:\Tallus.sdf" providerName="Microsoft.SqlServerCe.Client.3.5"
/>
</connectionStrings>
```

The above snippit of code is from last year project. I used it to connect to an sql database file (.sdf) on a network called U. So I experimented using different services that already exist for databases in the cloud, services like dropbox. That didn’t work out great because it wouldn’t let me have more then one process running at any time. So what I did is revist my code from last year and edited it with some improvments.

```
<connectionStrings>
<add
connectionString="Data Source=199.34.228.100\Files\BTYOUNG\Tallus.sdf"
providerName="Microsoft.SqlServerCe.Client.3.5" />
</connectionStrings>
```

As you may have guessed the server is running on Windows Server 2008, its easier for me to code in C# and C++. The main change in the above code is that I placed the sdf database file on my web server, with address 199.34.228.100 (or projectbird.com) and the programme is able to communicate through open and closed pockets in the server and update the database file. This makes it easy for me to pull information down for developing apps in the future.

INTRODUCTION

Now I had part of the software that would pull information from a server when making a request. I started looking up other types and popular company's progammes that were already out there. Dropbox, as I mentioned is the most popular progamme for easily updating and storing files in the cloud but I didn't need to replace the files just get the information from the database file. So I thought about it and developed a website that would do what I needed to do (at this stage). So I looked into php and other website building languages and got a grasp on how they work.

What I found, is that php, and websites in general, use similar calls to recivce data from a database file to that of the C# language. The code below is the php code for receiving and displaying information from a database when a request is made calling from different tables within the database.

```
<?php  // Connects to your Database
 mysql_connect("your.hostaddress.com", "username", "password") or die(mysql_error());
 mysql_select_db("Database_Name") or die(mysql_error());
 $data = mysql_query("SELECT * FROM friends")
 or die(mysql_error());
 Print "<table border cellpadding=3>";
 while($info = mysql_fetch_array( $data ))
 {
 Print "<tr>";
 Print "<th>Name:</th> <td>".$info['name'] . "</td> ";
 Print "<th>Pet:</th> <td>".$info['pet'] . " </td></tr>";
 }
 Print "</table>";
 ?>
```

As I said the above code is for requesting and dispalying information from a database in the PHP language. So I studied the code and made a version of it in the C# language , which can be seen below. It wasn't easy, there was trouble displaying the information for quite a while because it wasn't connecting to the database (which I later learnt to be a fault in the connection line).

Below is the code I made to pull information from a database possible.

INTRODUCTION

The textbox1.text is an input and when the input is searched against the table in the data and found, it loads and pulls the information from the rows (ie. Names, address etc).

```
using System.Threading;
using System.Runtime.InteropServices;
dynamic login =this.studentsinformationTableAdapter.ScalarQuery(textBox1.Text, textBox2.Text);

if (login == null)
{
string searchFor = “textBox1.Text”;
int results = 0;
DataRow[] returnedRows;

returnedRows=tallusDatabaseDataSet.Tables["Studentsinformation"].Select("FirstName='" +
searchFor + "'");

results = returnedRows.Length;
if (results > 0)
{
DataRow dr1;
dr1 = returnedRows[0];
 (dr1["FirstName"].ToString());
 (dr1["Age"].ToString());
 (dr1["Points"].ToString());
}
else
{

}
}
```

Now before you move onto the next part of my booklet. I have built and placed a server to hold and allow data to be transferred to and from a database file. I have also built a bit of code in C# to allow an application to get and download the information and display it on screen.

Why is this important??

INTRODUCTION

I would probably leave out the next two pages, or most of it anyway.

To start off have been very lucky to meet Mr Kieran Hyland of hymed systems.To say that Kieran didn't change my life would be a lie. So how did Kieran and I collaborate? Back in September 2010 Kieran and I became acquainted by chance when he was collecting his daughter from a play date with my younger sister. Kieran casually asked if I had any interest in the upcoming BT Young Scientist Competition. I told Kieran all about my aspirations to take part in the competition. The months quickly ran by. Fast forward to February 2011. I was after doing well in the competition, winning a category award and two special awards. Kieran called to our house, to warmly congratulate me on my success. I naturally gave him an indepth demonstration of the project, Automated monitoring of computer SQL servers by way of a merit system.

Kieran provided me with a fresh perspective on my project. Using his entrepreneurial skills, Kieran gave me both positive and negative feedback, mostly negative. Concept wise, it wasn't great. I clearly had a lot to learn. From a development perspective however, he thought it was great. It became apparent that software coding was my forté. Kieran set me a personal challenge. He wanted to see what I was capable of.

Fast forward another week. Kieran asks if I could develop a working concept of moving files around a server automatically. I took to my room, with the Young Scientist project still in the back of my mind. Leveraging what I had learnt from my project, I began to develop Kieran's concept. It was develped within two days.

Kieran and I met up in a local café and I showed him what I had come up with. I showed him the project he asked for. He asked me to compare last years project with the one I had just developed for him. I couldn't see many similarities. However it emerged that I had developed a very useful algorithm. This algorithm was present in both projects. Kieran saw something in my original project that I hadn't. He was happy to see that I had used the same algorithm subconsciously. Kieran offered me the opportunity to fine tune my coding skills by contributing to future projects as an intern with his company. After this lunch, I was eager to further develop this algorithm that was hidden within my project.

INTRODUCTION

About two days after showing Kieran what I could do, I started to look again at last years Young Scientist project and this algorthim. When I examined it again I realised that, using this algorithm, it was possible to adapt to different speeds and settings to get information and display it as quickly as possible.The way I designed it , it could be fitted into many different types of software such as internet broswers for large scale search operations.
I began to see that this highly adaptable algorithm had serious potential.

So I went back to Kieran with the alogrthim telling him I had separated it out from the rest of the code (it can be seen on page 33). I said it was great that I had made it by mistkae but asked if he could help me. I needed to make sure it was as stable as possible and to learn what its strongest points were. Keiran said he was more then happy to help but asked if I could make a progamme that makes use of the algorthim so it could be tested. I said sure thing.

The progamme in question is to pull information from a database of people who signs up on a website (using a smart phone or computer) with their name, address and contact details plus their photo.

I realised myself that this could be used to track people and products. If someone bought beer or other alcohol and there was a code on it and their Garda ID card was used it would be possible to track who bought what product. I now know this kind of system already exists and is called track and trace and is used for medical devices and medicines.

In the next pages I will talk about the code that is making this work and run well, then we get back to Kieran as it's the next few weeks I spent building the code that really made this project before it got tested.

INTRODUCTION

Let's get straight into it. The code below is written in java script and is used in the web broswer, web app aspect of the set up. When a user vists "http://store.projectbird.com" they input a number in the textbox and click search. The code springs into action. What it does is it adds the address of the file location plus the number they inputed and ".jpg" and goes to that address. It shows the picture and details of that person in an instant with the alorgthim making it work as fast as it does.The code for mobile apps and the desktop apps is different though and I will talk about them next in the report.

```
<script language="javascript">
<!--//
/*This Script allows people to enter by using a form that asks for a
UserID and Password*/
function pasuser(form) {
if (form.pass.value== form.pass.value )
{
location="http://store.projectbird.com/files/theme/" + form.pass.value +".jpg"
} else {
alert("No results were found :(")
}
}
//--></script>
<center>
</tr>
<font size="3"></font></b></i></h1></td>
<td height="0" bordercolor="#FFFF00" style="border: 0px solid #FF0000"><form
name="login"><inputname="id" type="text">
</td></tr>
<tr><td bordercolor="#FFFFFF" height="30"  style="border: 0px solid #FF0000"><h1><i><b>
        <font size="3"></font></b></i></h1></td>
        <td bordercolor="#FFFF00" style="border: 0px solid #FF0000"><input name="pass"
type="password"></td></tr>
<tr>
```

<td bordercolor="#FFFF00" style="border: 50px solid #FF0000" height=""> <input type="button" value="Search" onClick="pasuser(this.form)"> </td>

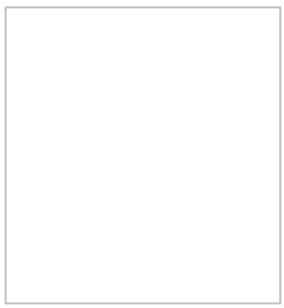
INTRODUCTION

Now that I have explained the java script that is running on the website (I know I havent explaned the alogrithim yet) I will explain the phone app, desktop app (windows 7 and windows 8 metro version) and the web apps for any moble device. What's different about the set up for the website and the apps themselves is the fact that the alorgthim is built into the apps and not into the server. It will be easy to understand as you see the code.

You do need to explain the algorithm at this stage; in particular what makes it different/better/ faster than previous systems. A lot of the explanation in the Experimental methods should probably be moved to an appendix describing the general principles of database searches and how they work.

INTRODUCTION

So for the mobile app the algorithm in use is highlighted in green. The reason for building a mobile app is because mobile applications and their market place get to a bigger range of people. With the use of the algorithm I have developed it speeds up the gathering of information from a server extremely quickly no matter what your signal is. As you can see below the code is quite small because a lot of the background working is in the algorithm itself.



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using Microsoft.Phone.Controls;
namespace PhoneApp8
{
   public partial class MainPage : PhoneApplicationPage
   {
      public MainPage()
      {
         InitializeComponent();
      }
         private void button1_Click (object sender, RoutedEventArgs e)
         {
            if (textBox1.Text == textBox1.Text)
            {
Tardis.start;
textBlock1.Text =“http://store.projectbird.com/files/theme” + textBox1.Text “.jpg” ;
textBlock2.Text =” http://tiny.cc/Tardis” + textBox1.Text
textBlock3.Text = ” http://tiny.cc/Tardis3” + textBox1.Text;
```

```
textBlock4.Text = ” http://tiny.cc/Tardis4” + textBox1.Text;
textBlock5.Text = ” http://tiny.cc/Tardis77” + textBox1.Text;
image1.Source =” http://tiny.cc/Tardis78” + textBox1.Text “.jpg” ;
Tardis.stop;

            }
            else
{

            }
```

INTRODUCTION

Remember Kieran hyland that I metented eailer well have I showed him what I did with being able to improve speed of gathering information. He had other ideas of being able to

Sir this is where I was going to let you fill in where about the people we emailed and how I changed from my first idea to kierans ivids concept.

EXPERIMENTAL METHODS

In computer science, a search algorithm is an algorithm for finding an item with specified properties among a collection of items. The items may be stored individually as records in a database; or may be elements of a search space defined by a mathematical formula or procedure, such as the roots of an equation with integer variables; or a combination of the two, such as the Hamiltonian circuits of agraph.

In computer science, a sorting algorithm is an algorithm that puts elements of a list in a certain order. The most-used orders are numerical order and lexicographical order. Efficient sorting is important for optimizing the use of other algorithms (such as search and merge algorithms) that require sorted lists to work correctly; it is also often useful for canonicalizing data and for producing human-readable output. More formally, the output must satisfy two conditions:

The output is in nondecreasing order (each element is no smaller than the previous element according to the desired total order);

The output is a permutation, or reordering, of the input.

Since the dawn of computing, the sorting problem has attracted a great deal of research, perhaps due to the complexity of solving it efficiently despite its simple, familiar statement. For example, bubble sort was analyzed as early as 1956. Although many consider it a solved problem, useful new sorting algorithms are still being invented (for example, library sort was first published in 2004).

EXPERIMENTAL METHODS

Overview

This is the easiest to implement and the most frequently used search algorithm in practice. Unfortunately the sequential search is also the most ineffective searching algorithm. However, it is so commonly used that it is appropriate to consider several ways to optimize it. In general the sequential search, also called linear search, is the method of consecutively check every value in a list until we find the desired one.

Basic Implementation

The most natural approach is to loop through the list until we find the desired value. Here's an implementation on PHP using FOR loop, something that can be easily written into any other computer language.

Complexity

As I said at the beginning of this post this is one of the most ineffective searching algorithms. Of course the best case is when the searched value is at the very beginning of the list. Thus on the first comparison we can find it. On the other hand the worst case is when the element is located at the very end of the list. Assuming that we don't know where the element is and the possibility to be anywhere in the list is absolutely equal, then the complexity of this algorithm is O(n).

Different cases

We must remember, however, that the algorithm's complexity can vary depending on whether the element occurs once.

Is it so ineffective?

Sequential search can be very slow compared to binary search on an ordered list. But actually this is not quite true. **Sequential search can be faster than binary search** for small arrays, but it is assumed that for n < 8 the sequential search is faster.

Application

The linear search is really very simple to implement and most web developers go to the forward implementation, which is the most ineffective one. On the other hand this algorithm is quite useful when we search in an unordered list. Yes, searching in an ordered list is something that can dramatically change the search algorithm. Actually searching and sorting algorithms are often used together.

A typical case is pulling something from a database, usually in form of a list and then search for some value in it. Unfortunately in most of the cases the database orders the returned result set and yet most of the developers perform a consecutive search over the list. Yet again when the list is ordered it is better to use binary search instead of sequential search.

PageRank is a probability distribution used to represent the likelihood that a person randomly clicking on links will arrive at any particular page. PageRank can be calculated for collections of documents of any size. It is assumed in several research papers that the distribution is evenly divided among all documents in the collection at the beginning of the computational process. The PageRank computations require several passes, called "iterations", through the collection to adjust approximate PageRank values to more closely reflect the theoretical true value.

A probability is expressed as a numeric value between 0 and 1. A 0.5 probability is commonly expressed as a "50% chance" of something happening. Hence, a PageRank of 0.5 means there is a 50% chance that a person clicking on a random link will be directed to the document with the 0.5 PageRank.

Simplified algorithm

Assume a small universe of four web pages: **A**, **B**, **C** and **D**. The initial approximation of PageRank would be evenly divided between these four documents. Hence, each document would begin with an estimated PageRank of 0.25.
In the original form of PageRank initial values were simply 1. This meant that the sum of all pages was the total number of pages on the web at that time. Later versions of PageRank (see the formulas below) would assume a probability distribution between 0 and 1. Here a simple probability distribution will be used—hence the initial value of 0.25.

If pages **B**, **C**, and **D** each only link to **A**, they would each confer 0.25 PageRank to **A**. All PageRank **PR( )** in this simplistic system would thus gather to **A** because all links would be pointing to **A**.

$$PR(A) = PR(B) + PR(C) + PR(D)$$

This is 0.75.

Suppose that page **B** has a link to page **C** as well as to page **A**, while page **D** has links to all three pages. The *value of the link-votes is divided among all the outbound links on a page*. Thus, page **B** gives a vote worth 0.125 to page **A** and a vote worth 0.125 to page **C**. Only one third of **D**'s PageRank is counted for A's PageRank (approximately 0.083).

$$PR(A) = \frac{PR(B)}{2} + \frac{PR(C)}{1} + \frac{PR(D)}{3}$$

In other words, the PageRank conferred by an outbound link is equal to the document's own PageRank score divided by the normalized number of outbound links **L( )** (it is assumed that links to specific URLs only count once per document).

$$PR(A) = \frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)}$$

In the general case, the PageRank value for any page **u** can be expressed as:

$$PR(u) = \sum_{v \in B_u} \cdot \frac{PR(v)}{L(v)}$$

i.e. the PageRank value for a page **u** is dependent on the PageRank values for each page **v** out of the set $\mathbf{B_u}$ (this set contains all pages linking to page **u**), divided by the number $L(v)$ of links from page **v**.

As in a different story to google in doing research Microsoft hasen't released any information on the alogorthim that is running bing but luckily for me google engineers tried a trick to see if bing was copying their results. From my browsing of the web for artices about Bing's algorithm which is also used by yahoo, it is very similar to Google's but with minor changes. As the following starts

Bing has been criticised by competitor Google, for utilising user input via Internet Explorer, the Bing Toolbar, or Suggested Sites to add results to Bing. After discovering in October 2010 that Bing appeared to be imitating Google's auto-correct results for a misspelling, despite not actually fixing the spelling of the term, Google set up a honeypot, configuring the Google search engine to return specific unrelated results for 100 nonsensical queries such as *hiybbprqag*. Over the next couple of weeks, Google engineers entered the search term into Google, while using Microsoft Internet Explorer, with the Bing Toolbar installed and the optional Suggested Sites enabled. In 9 out of the 100 queries, Bing later started returning the same results as Google, despite the only apparent connection between the result and search term being that Google's results connected the two.

Microsoft's response to this issue, coming from a company's spokesperson, was clear: "We do not copy Google's results." Bing's Vice President, Harry Shum, later reiterated that the search result data Google claimed that Bing copied had in fact came from Bing's very own users. Shum further wrote that "we use over 1,000 different signals and features in our ranking algorithm. A

small piece of that is clickstream data we get from some of our customers, who opt-in to sharing anonymous data as they navigate the web in order to help us improve the experience for all users." Microsoft commented that clickstream data from customers who had opted in was collected, but said that it was just a small piece of over 1000 signals used in their ranking algorithm, and that their intention was to learn from their collective customers. They stated that Bing was not intended to be a duplicate of any existing search engines.

This is very good and interesting. I must talk to you about the speed calculation

Humming Bird is an alogrithim I deveopled to help speed up the search and gathering of information from servers when a person is randomly searching the web on their smart-phone or computer. Humming Bird can be calculated for collections of documents and webpages of any size and return them to the user in the fastest time possble. It is assumed in several research papers that the speeds of broad band in areas of Ireland and many other countries is quite slow in comparsion to some countries. The Humming Bird computations require several passes, called "karmas", through the collection to adjust approximate speeds values/times to more closely adapt to always give the quickest results when pulling information.

A probability is expressed as a numeric value between $1X10^1$ and $20x10^1$ probability is expressed as a "5% chance the result is done and no help is need in the table (which can be seen on pages 21-23). Hence, a humming bird of $1x10^1$ means there is a 5% chance of the search needing to go on. Unlike a humming bird of $20x10^1$, which uses a lot more code to help speed up the gathering of information from a server. So the nearer to 1 $X10^1$ the less code is needed. The higher to twelve the more code and tricks needed. The higher the number the less high resolution of photo is downloaded .

Simplified algorithm

Assume a small website of four web pages: **1**, **2**, **3** and **4**. The initial approximation of humming bird would search and find these in four different databases. Each webpage would begin with an humming bird rating of $1X10^1$.

So when a user is searching/imputing a number or term to search that is **S**. the location is also caught by using their current ip-address and that is **L.** Also used is the time to finish the search so, if it took .20 of a second to finish a search before returning it to the user, that time is saved and placed into the algorthim and divided by the same time but with the 10^1 and the time it took is **T,** which is determined by using a simple ping call.

$$R = \frac{S + SL - L + T}{T \; x \; 10^1}$$

So for example, using the above information a user wants to search for page 3. They input the number into the search box, as soon as they do that that the number 3 (search term) is placed in the algorthim as well as the ip-address or location of the dievce. The search is already taking place before the user can click search/enter. Within that time of when the user clicks the button, the search is done and a time of how long it took has been recorded and placed into T. Below you can see a step by step guide of what happens.

$$Results = \frac{Search\ term + (Search\ Location - Location + Time}{Time \; x \; 10^1}$$

Humming bird alorgthim

$$R = \frac{S + (SL - L) + T}{T \; x \; 10^1}$$

$$Results = \frac{Search\ term + (search\ location + Location) + Time}{Time \; x \; 10^1}$$

$$Results = \frac{Batman + (0.94 - 0.52) + .13}{.13\ x\ 10^1}$$

$$Results = \frac{i(info) + 0.42 + .13}{.13\ x\ 10^1}$$

$$Results = \frac{0.42 = -.13}{.13\ x\ 10^1}$$

$$Results = 0.42 = -.01$$

$$Results = -0.1\ \%\ 0.42$$

$$Results = .2380\ seconds$$

$$Results = i(info) in\ .238\ seconds$$

When doing the testing for this algorithm I started to go with a waterfall model as it limited the number of errors appearing. A **waterfall model** is a sequential design process, often used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing, Production/ Implementation and Maintenance.

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of

software implementation. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs(errors or other defects).

Software testing can be stated as the process of validating and verifying that a software program/ application/product:

1. meets the requirements that guided its design and development;
2. works as expected; and
3. can be implemented with the same characteristics.

Software testing, depending on the testing method employed, can be implemented at any time in the development process. However, most of the test effort occurs after the requirements have been defined and the coding process has been completed. As such, the methodology of the test is governed by the software development methodology adopted.

Most of the pages above are not your own so should probably be in an appendix as I said earlier

There was private testing

Facebook testing

21 cms testing

School testing

21cms where very good to test out my algorithm for me and stress test it.

21CMS is the brainchild of Franco De Bonis and Mark Gaffney. With many years of experience in web marketing and development, the two formed a company to service the web needs of SMEs across Ireland and the UK.

However, they see a future world where smartphones and other mobile devices will dominate web browsing. A world where businesses will connect with their customers wherever they may be and at any time, not just when the customer is sitting in front of a PC at home or work. The problem is that everywhere they looked they could see complex web management systems and completely disconnected and equally complex web marketing systems, many of which are priced beyond the reach of typical SMEs.

They therefore resolved to deliver a solution to answer this need and so development of 21CMS began in the autumn of 2008. In the spring of 2011 (more than two years later) development of the first phase was completed and 21CMS was launched to the world.

I sent my software on to Mark to be tested and these are the tests they did

Grey Box Testing involves having knowledge of internal data structures and algorithms for purposes of designing the test cases, but testing at the user, or black-box level. The tester is not required to have a full access to the software's source code. Manipulating input data and formatting output do not qualify as grey box, because the input and output are clearly outside of the "black-box" that we are calling the system under test. This distinction is particularly important when conducting integration testing between two modules of code written by two different developers, where only the interfaces are exposed for test. However, modifying a data repository does qualify as grey box, as the user would not normally be able to change the data outside of the system under test. Grey box testing may also include reverse engineering to determine, for instance, boundary values or error messages.

Performance testing is in general executed to determine how a system or sub-system performs in terms of responsiveness and stability under a particular workload. It can also serve to investigate, measure, validate or verify other quality attributes of the system, such as scalability, reliability and resource usage.

Load testing is primarily concerned with testing that can continue to operate under a specific load, whether that be large quantities of data or a large number of users. This is generally referred to as software scalability. The related load testing activity of when performed as a non-functional activity is often referred to as *endurance testing*. *Volume testing* is a way to test functionality.

Stress testing is a way to test reliability under unexpected or rare workloads. *Stability testing* (often referred to as load or endurance testing) checks to see if the software can continuously function well in or above an acceptable period.

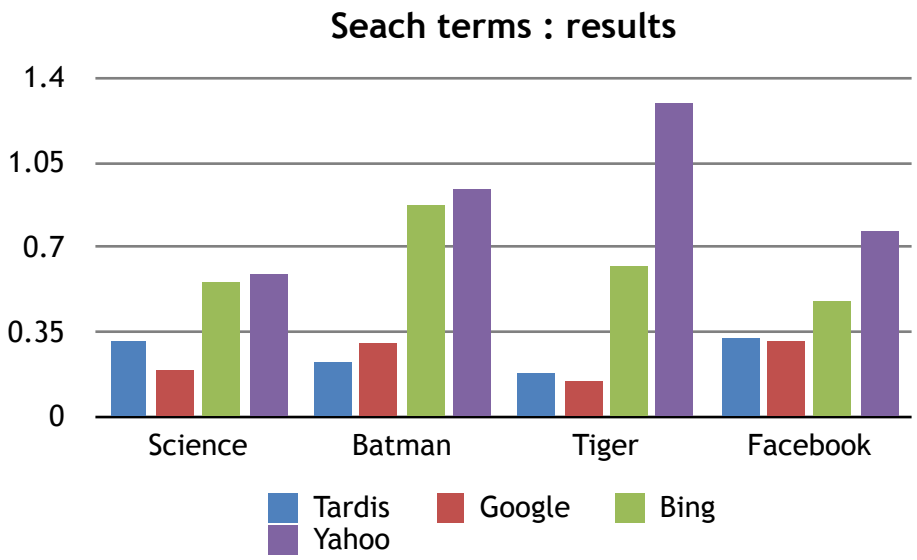
There results can be seen on page ()

Rob you'll have to explain these results to me and to the judges in some detail

My alorgthim and site

Recommendation	Grade
Performance grade:	95/100
Minimize redirects	99
Avoid bad requests	100
Remove query strings from static resources	97
Leverage browser caching	100

Recommendation	Grade
Specify a Vary: Accept-Encoding header	100
Serve static content from a cookieless domain	97
Minimize request size	100
Specify a cache validator	100



Recommendation	Grade
Performance grade:	92/100
Serve static content from a cookieless domain	57
Combine external JavaScript	58
Specify a Vary: Accept-Encoding header	85
Leverage browser caching	100

Minimize DNS lookups	**100**	
Minimize redirects	**100**	
Combine external CSS	**100**	
Parallelize downloads across hostnames	**100**	
Remove query strings from static resources	**100**	
Avoid bad requests	**100**	
Specify a cache validator	**100**	
Minimize request size		**100**

google

Bing

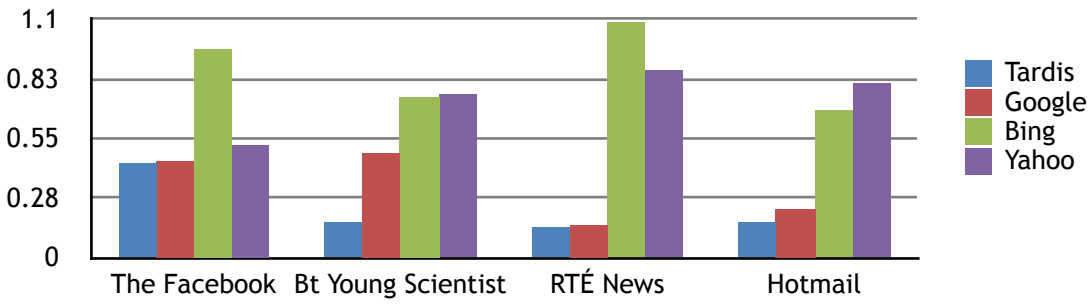
Recommendation	**Grade**
Performance grade:	84/100
Minimize redirects	**0**
Remove query strings from static resources	**50**
Serve static content from a cookieless domain	**57**

Recommendation	Grade	
Leverage browser caching	**96**	
Minimize request size	**100**	
Combine external JavaScript	**100**	
Combine external CSS	**100**	
Parallelize downloads across hostnames	**100**	
Avoid bad requests	**100**	
Minimize DNS lookups	**100**	
Specify a cache validator	**100**	
Specify a Vary: Accept-Encoding header		100

yahoo

Recommendation	Grade
Performance grade:	91/100

Recommendation	Grade
Minimize DNS lookups	75
Minimize request size	100
Combine external JavaScript	65
Leverage browser caching	100
Avoid bad requests	100
Minimize redirects	100
Combine external CSS	100
Parallelize downloads across hostnames	100
Remove query strings from static resources	100
Serve static content from a cookieless domain	100
Specify a cache validator	100
Specify a Vary: Accept-Encoding header	100


Search Terms : long words
1.1
0.83
0.55
0.28
0
The Facebook
Bt Young Scientist
RTÉ News
Hotmail
Tardis
Google
Bing
Yahoo

Need you to help fill in the CONCLUSIONS AND RECOMMENDATIONS

Asim not sure whats best to write