Prime Time

Final Report

ECE 480
November 30, 2016

Ibrahim Abdi, Collin Rokke, Tarin Sultan, Mike Maryn, Vehans Ayvazi

Abstract

The motivation behind our Prime Time project was simply to develop a software from the ground up. Our team decided on the nature of this project, for the purposes of being completable in the time frame provided and contributing to the solution of a long-standing problem. Prime Time is prime number determination program which, if further developed, could lead to detection of truly new and unique numbers for utilisation in security protocols and the likes. The purpose of this program is to make a determination using two methods, fast brain or slow brain as described in Dr. Yin's paper. Meng-Lai Yin, Ph.D is the professor for ECE 480 Software Engineering at Cal Poly Pomona for the Fall 2016 quarter. As a project demonstrating to us as much of the entire process of software engineering as possible, we (Ibrahim Abdi, Collin Rokke, Tarin Sultan, Mike Maryn and Vehans Ayvazi) have designed/developed Prime Time.

Table of Contents

1. Introduction

The purpose of this project is to utilize concepts from the article, *Real-Time Maintenance Prioritization with Learning Capability* by Dr. Meng-Lai Yin and Andrew J. Chan, to solve a problem. In particular, the utilization of two distinct thought processes, the fast brain method and the slow brain method, to solve problems, as well as recall their solutions. With this project the specific problem to solved, was the identification of prime numbers.

2. Project Overview

This software implements a way to identify prime numbers using a graphical user interface to interact with the admin/client and receive their input. The number entered by the user enters the slow brain thought process, which utilizes the prime determination algorithm, and sends back determination times as well as an indication of whether the number entered is prime or not. The second time that same number is entered, the number will skip the algorithm and go straight to the database where the number was previously stored. This part of the software utilizes the fast brain thought process, which allows the software to send back even faster determination times.

3. Design and Implementation

Our software was designed and implemented by using a simple, agile and straightforward programming language, various libraries and a universal database.

3.1 Design

Our design for this software includes the language type, the library types, and the database type that was utilized in order to release a cohesive and operative software.

3.1.1 Programming Language

One of biggest decision we made as a group was what language to use. Coming into the project two out of five of our group members had previous experience in using Python. One of the key reasons of us choosing Python, was the fact that we wanted to gain experience in a language we were not accustomed to. Python, being an extremely easy language to learn, was the perfect fit for us because we needed to pick it up without such a steep learning curve.

3.1.2 Libraries

Mysql.connector:

The mysql.connector is a driver that is utilized to connect to a MySql database server. This driver includes a math library that contains many specific functions necessary for our project. These functions include a square root function, and a function to convert floating point numbers to an integer. Time allows for the use of timer to see how long functions take to execute index.

Tkinter:

Tkinter is a library that we utilized for our software that allows easy GUI development. Since we had used Python as our programming language for this software, we used Tkinter, a Python de-facto standard GUI package, to develop our user interface.

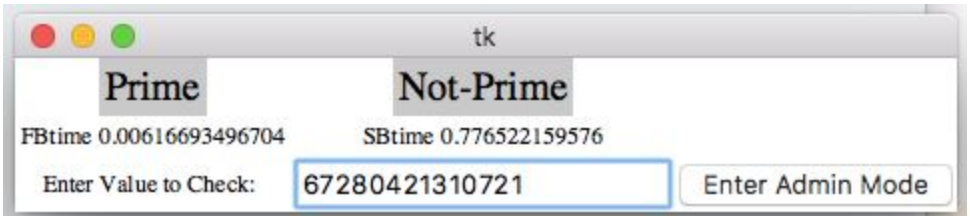
3.1.3 Database

MySQL:

The database chosen for our project. MySQL is an open source relational database, meaning that data is stored and presented in tables, rows, and columns. SQL is the language used for programming and managing data in the MySQL database.

3.2 Implementation

When the code runs a GUI created in Tkinter displays for the user, as shown below



The user then enters a number, the code calculates whether or not the number is prime, then highlights the appropriate term. If not Prime, the term Not-Prime highlights red. If Prime, the term Prime, highlights green, with both the fast brain time and slow brain time being displayed. The fast brain time is the execution time for the program to query the database and determine if the number is prime, the slow brain time is the execution time of the algorithm and determining if a number is prime.

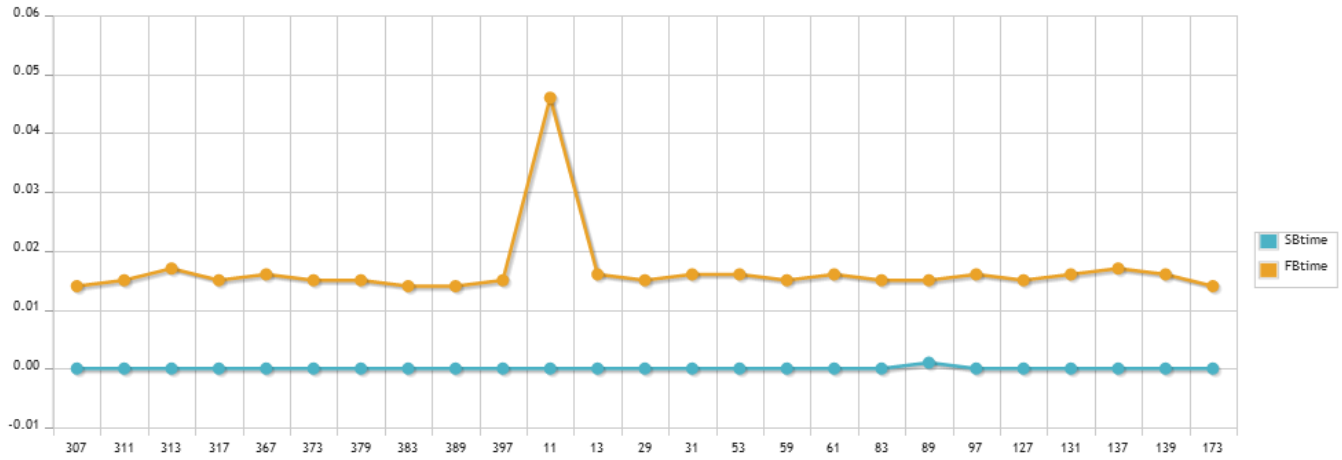
4. Discussion

In this section of our report we would like to discuss the few problems that we encountered during the making of this software, the many successes we had as well as the future potential we have for this software.

Our implementation of printing the database elements using the console resulted in difficulty when trying to interface with the GUI. We attempted to use a textbox with a specified height and width to contain the elements but could not populate the textbox correctly. Having the flush and print functions in a separate child window resulted in problems when referencing our database object as well as buttons and interfaces in the parent window.

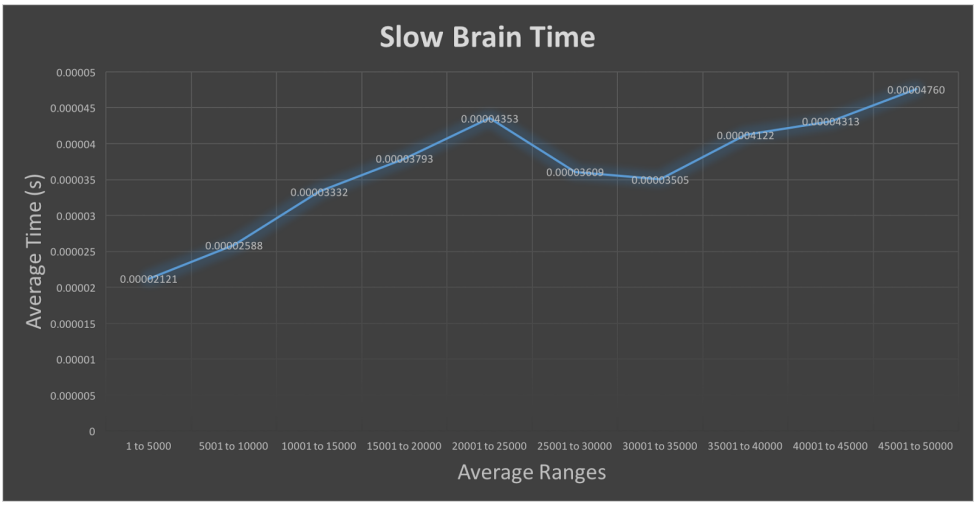
4.1. Results

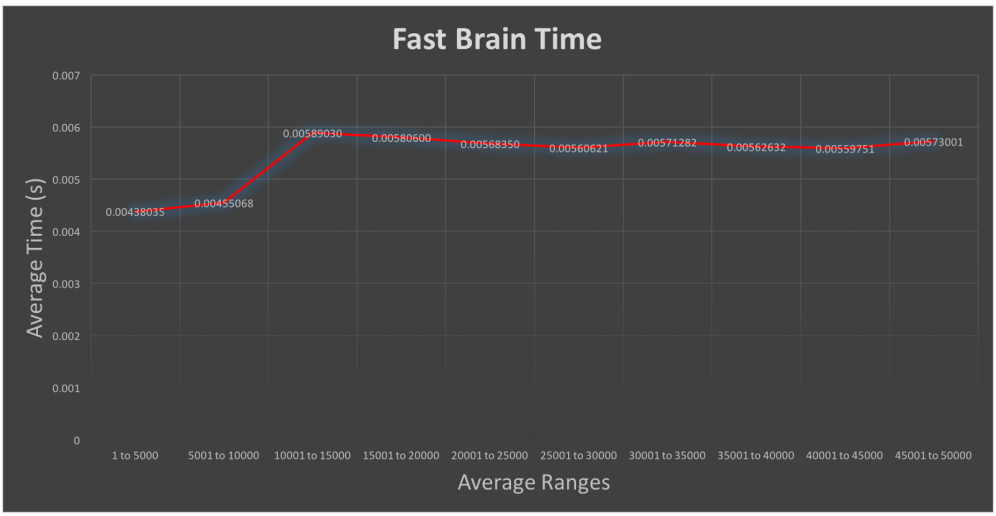
The project was successful in being able to determine if a number is prime or not prime, however this doesn't mean there weren't some issues. The main problem with this final version of the project is that the slow brain is actually faster than the fast brain. Below is a chart, number vs time (X vs Y).



However, once the number does get high enough, roughly in the trillions, the fast brain becomes faster than the slow brain. Disclaimer, this only works when testing the numbers alone in a partially filled database containing only around 10,000 numbers. Realistically, the database would contain trillions of entries so the query time may increase as well, however without access to how MySQL is code we can not be sure.

To further test our program we ran it with two for loops inputting numbers 1 to 50,000. This yielded some interesting results and it show that we overestimated how fast database operations are. It proved to us that our program is only useful for numbers in the range of trillions.





4.2. Future Potential

While we were successful in our implementation of the slow brain/fast brain concept, in the end our project was simple. There is much potential for it in implementing it in more advanced ways. The original idea behind this project was a simulation of a system error (prime number) and a working system (nonprime number). Some systems need a more complex solution, handling multiple errors needing to be fixed in differing orders, as described in Dr. Yin paper. Such a system could benefit from having the solutions stored in a database, rather than calculated every time that particular error occurs.

4.3 Continuation

There were many features we were not able to add to this project which were originally planned. First was the implementation of a game. The game would pull a random number from the data and present it to the user. The user would then have a limited amount of time to guess whether or not the number is prime. The more numbers that the user gets correct in a row, the higher their score.

The other option was to make a our Prime Time project into a Web based/ mobile application. With the rise in popularity of web based and mobile based apps this would have been good practice, however, time constraints and other classes made this unrealistic for us.

Changes to the GUI would also be implemented. Using a more elegant approach as opposed to a text box would give better user interaction with the GUI, and convey information more clearly. However, with that being said, we were able to release an executable software that allows the determination of a prime number to be outputted for any user accessing our software.

5. References

[1] Meng-Lai Yin, Andrew J. Chan, “Real-time Maintenance Prioritization with Learning Capability, ECE Dept, Cal Poly Pomona, CA