



Undergraduate Research Opportunity Program
(UROP) Project Report

Search Engine Optimizations for VisuAlgo

By

Wang Zi

Department of Computer Science

School of Computing

National University of Singapore

2014/2015

Undergraduate Research Opportunity Program
(UROP) Project Report

Search Engine Optimizations for VisuAlgo

By

Wang Zi

Department of Computer Science

School of Computing

National University of Singapore

2014/2015

Project ID: U160010

Advisor: Dr. Steven Halim

Deliverables:

Report: 1 Volume

Program: 2 Packages

Database: 1 Set

Abstract

VisuAlgo was first conceptualized by Dr. Steven Halim as a self-paced studying tool for his students to better understand data structures and algorithms. Dr. Steven has led a team of students from NUS School of Computing to implement a series of intuitive visualization and animation, from simple sorting algorithms to complex graph data structures.¹ I am privileged to become one of the developers to further build and promote VisuAlgo. My UROP can be divided into two phases but with one grand theme: using all my research on modern search engines to make a working webpage takes advantages of all of those. During the first phase, I studied about modern search engines' features and working mechanisms. Then I need to use my research results to promote VisuAlgo's page ranking for English searching queries. Subsequently, we targeted on internationalizing VisuAlgo to promote its page ranking for searching queries in other languages. Hence, it is critical for us to find a way to translate the whole website in a sustainable and user-friendly way. This leads to my second phase of research. Before I started phase two, I also individually worked on a development case for VisuAlgo to master its back-end structure. And finally, thanks to our united effort, the project is completed successfully beyond our expectations.

Subject Descriptors:²

B.2.4 Algorithms

D.1.7 Visual Programming

D.2.5 Testing and Debugging

D.2.7 Distribution, Maintenance and Enhancement

Keywords:

Data Structure and Algorithm, Internationalisation, Modern Search Engines,
Visualization and Animation

Implementation Software:

Mac OS X Yosemite, Google Chrome, XAMPP, MySQL Server

Implementation Language:

HTML, CSS, JavaScript, MySQL, Google Translate API, PHP

¹ About VisuAlgo. 2011. <http://visualgo.net> (Accessed 02/04/2015).

² The 1998 ACM Computing Classification System. 1998. <http://www.acm.org/about/class/1998/> (Accessed 02/04/2015).

Acknowledgement

The completion of my UROP project is made possible with the guidance and support from the following people.

I want to express my genuine gratitude to **Dr. Steven Halim** for being a most committed, helpful and inspiring supervisor. I do sincerely appreciate your many hours of consultation and meetings, inspirations and expertise, and most thankfully, your willingness to offer me constant guidance and support which is like a torchlight during my UROP journey. It is an invaluable experience to have taken your data structure and algorithm classes, which has been a great inspiration of my computer science fundamental studies. I was inspired by your creation, passion, and unparalleled expertise. I am so fortunate to have known you and taken your class since my first year in School of Computing. I have learnt so much from you not only of computing knowledge but also the professional and responsible attitude towards a project. I do sincerely apologies for any inconvenience I have brought to you. Thanks for your considerations and constant support Dr. Steven. They have become my best memories in NUS.

I would also like to extend my gratitude to the CP3101B (Web Programming and Applications) project team mates, namely **Anand Sundaram, Nguyen Hoang Vu, Nguyen Viet Dung, Savin Varshney**. It has been a great experience working with you on the VisuAlgo Internationalization project. All those commits we made, emails we sent and codes we programmed have become a great learning journey for me. Thank you so much.

Finally, to my dearest **parents and grandma**, my appreciation and gratitude transcend words. Your unconditional love and accompany is always helping me to overcome barriers and obstacles. Being as your daughter or grand daughter always does myself proud and fortunate. To my dearest **grandpa**, please rest in peace. You always live in my heart. My sincere gratitude goes to you.

Table of Contents

Title Abstract	i ii
Acknowledgement	iii
1 Introduction	
1.1 VisuAlgo's Front-End and Back-End	2
1.2 Project Description	5
2 Study of Modern Search Engines	
2.1 The use of Modern Search Engines	6
2.2 Modern Search Engines' Working Mechanism	7
2.3 Modern Search Engine Optimization	10
3 Development Case of VisuAlgo: Floyd's Cycle Finding Algorithm	
3.1 Methodology	11
3.2 Program Design	11
3.3 Implementation	12
3.4 Case Summary	14
4 Internationalization Implementation	
4.1 Study of Google Translate API & Web Translator	15
4.2 Program Design	17
5 Conclusions	
5.1 Summary	23
5.2 Limitations	24
5.3 Recommendations for Future Work	24
<u>References</u>	25
Appendix - Program Listing	26

1 Introduction

1.1 VisuAlgo's Front-End and Back-End

My search engine optimizations research is especially tailored to the website VisuAlgo, “which was conceptualized in 2011 by Dr. Steven Halim as a tool to help his students better understand data structures and algorithms, by allowing them to learn the basics on their own and at their own pace.”¹ Moreover, VisuAlgo contains many advanced algorithms that are discussed in Dr. Steven Halim’s book (*Competitive Programming 3*).² Currently, there is an average about 2000 sessions pre day by visitors from all over the world.³ As a creative online studying tool, “VisuAlgo is specifically designed for NUS students who are taking various data structures and algorithm classes (e.g. CS1010, CS1020, CS2010, CS2020, CS3230, and CS3233).” Worth mentioning, VisuAlgo is an ongoing project, which means that it is continuously enhancing and extending its features as well as functions to make it possible for more data structures and algorithms to be visualized.

After an introduction of VisuAlgo, I would like to move on to elaborate its front-end. The home page of VisuAlgo presents all the available animations of the data structures and algorithms, including sorting, bitmask, linked list, stack, queue, hash table, binary heap, binary search tree, AVL tree, graph data structures, union-find disjoint sets, segment tree, binary indexed tree, generic recursion tree/DAG, graph traversal, minimum spanning tree, single-source shortest paths, network flow, graph matching, suffix tree, suffix array, (computational) geometry and Floyd’s cycle finding. Each of the data structures and algorithms I mentioned serves as a searching query I used initially to do page ranking experiments, which will be explained in detail later. Within each animation box, the user can choose either exploration mode to play with the data structure or get to know more under the tutorial mode. Moreover, the current home page has 11 foreign language versions, namely English, Chinese, Russian, Indonesian, Japanese, Korean, Thai, Vietnamese, Bengali, Dutch and German. Besides teaching mode, there is a part that facilitates online testing which is currently only accessible to specific users who are taking the related modules. However, the training section is available to every visitor of the website and it offers plenty of practice.

¹ About VisuAlgo. 2011. <http://visualgo.net> (Accessed 02/04/2015).

² Steven Halim. 2013. *Competitive Programming*. 3rd ed.

³ Google Analytics. 2015. <http://www.comp.nus.edu.sg/~stevenha/cp3101b/visualgo.html#google-analytics> (Accessed 02/04/2015).

Thanks to Dr. Steven's clear guidance and explanation, I managed to implement the visualization of Floyd's Cycle Finding algorithm.¹ During this development experience, I got an insight into VisuAlgo's back-end, which is of vital importance to the final internationalization phase. As I mentioned above, currently only the home page, is available in 11 foreign languages. At the original stage, our method was to use one index.html root file based in English and use JavaScript file to detect user browser's default language settings so that to invoke corresponding language page written in JavaScript. However, this method only enables the content within the <title></title> to be indexed by Google since Google usually does not crawl the content within the JavaScript page. The browser can execute JavaScript and produce content on the fly - the crawler cannot. However, only if the crawler finds your content, it can become searchable.² Hence, a restructure of the code is badly needed. And after our one semester's work, we managed it. The English version is kept as the index.html as the home page and all the other 10 translated pages will be redirected from this index.html and display the html page within which the translations are hard coded in html. In this way, we make it possible for the search results for queries in languages other English can also hit the top. However, the issue is still not completely solved since hard coding makes it extremely tedious for further maintenance and extension. The overhead is too high for every update since we need to have human resource understanding all the 11 languages to update each every time. The Google translation cannot be directly used here since most of the times the translations are too robotic and hard to read. And the translation of the dynamic contents regarding animations, online testing and randomly generated tutorial questions cannot be hard coded. Hence, we need to think of a well-rounded method to solve this internationalization issue, which will be explained in the following paper.

In addition to the home page file directory, I would also like to introduce the back-end of the implementation of the animation and visualization. Thanks to Dr. Steven and previous developers, VisuAlgo has a well-defined and powerful graph library programmed in JavaScript files that define all the useful constants and functions. Within this library, all the basic drawing elements used for complex animations are defined properly, such as the node and the edge. Then for each every of the available data structures and algorithms, there will be a specific XWidget.js relating to the individual X.html (for example, X could be list) which

¹ Tortoise and hare. 2014. http://en.wikipedia.org/wiki/Cycle_detection (Accessed 02/04/2015).

² What the user sees, what the crawler sees. 2014. <https://developers.google.com/webmasters/ajax-crawling/docs/learn-more> (Accessed 02/04/2015).

controls the animations of each operations as well as the display of status and pseudo codes. Besides common CSS files that define the visualization area, there is a specific set of CSS files corresponding to each of the data structure or algorithm.

It is extremely beneficial for my project and future studies with knowledge of the back-end implementation of VisuAlgo. The whole project would focus on extending and optimizing current status of VisuAlgo.

1.2 Project Description

The title of this project is “ Search Engine Optimizations for VisuAlgo”. As given in the description in the UROP project admin, “A search engine is a type of computer software used to search data in the form of text or a database for specified information. Search engines normally consist of spiders which roam the web searching for links and keywords. The project is to investigate modern search engine technologies like Google Page Rank.”¹

As mentioned previously, my work under Dr. Steven can be divided into two main phases. For the first phase, I would do research on the working mechanisms of some main modern search engines, such as Google, Yahoo, Bing and Baidu, and experiment with my research results to raise the page ranking of VisuAlgo given various searching queries. For the second phase, this part is more technical involving the implementation of the internationalization of VisuAlgo with a version that is easy for future maintain and extension. Between these two parts, it is very insightful of Dr. Steven to delegate me a visualization development job on VisuAlgo which makes me have a deeper understanding of the implementation and back-end structure of VisuAlgo.

I would also like to explain this project from another perspective. We started from the original status: we had a 6th to 8th ranking for tailored searching queries like “visualizing data structures and algorithms through animation”(content within the <title></title>tab), lower than 5 pages or none ranking for more generic searching queries like “sorting animation” or “bitmask visualization”, let alone other foreign languages whose results are even worse. Yes, we were intended to hit the top for all the relevant searching queries and in all 11 languages. And the exciting part of this project is that we hit the top! However, we have a more ambitious target status: we want to make the whole website internationalized and also sustainable for future maintenance and extension. And, we eventually made that possible. The following paper will explain how this is done in detail.

¹ Undergraduate Research Opportunity Programme. 2014.
<https://mysoc.nus.edu.sg/~projadm/projlist.php?proj=urop&pj=U16001&type=detl> (Accessed 02/04/2015).

2 Study of Modern Search Engines

2.1 The use of Modern Search Engines

We initially chose those popular search engines, namely Google, Bing, and Baidu as our target research subjects. Then, I started with a learning of their most-updated search functions. Even though, search engines to most people may just serve as a result fetcher of their search queries, the discoveries of modern search engines' new technologies was amazing. I noticed that the idea of simply matching keywords is not appreciated anymore. Instead, modern search engines are endeavoring on understanding why people are searching the queries and guess what they want.¹ The knowledge graph is the most updated technology used to fulfill this ambition.² For example, if we search "Da Vinci", we can observe there is a panel appears on the right of the search result page offering information on artists from the same era or similar style and even more portraits on Culture Renaissance. The panel offering all the potential and related options is called a knowledge graph. This state-of-the-art technology has been used by both leading companies: Google and Baidu, which is declared as a revolutionary improvement. Instead of simply giving out the searching result, this feature tries to lead users to explore their search with insights based on the most popular questions other users ask related to this topic. In other words, these search engines are no longer satisfied with simply answering users' single questions. They want to lead users search so that a single query is turned into a list of them.

Moreover, it is worth mentioning that both Google and Baidu have made the search easier by showing results as you type.³ By predicting the search queries before the user finished typing, the user can stop typing as soon as the user sees what he or she needs in the bar, which definitely makes searching more user-friendly and time-saving. Google calls it "Google Instant".⁴ There are still a lot of other features such as image search, voice search, real-time result synchronization and so on and so forth.⁵ The study of the functions of modern search engines gives me an updated idea on what are the core values of current searching technology and also let me think about how to align my research approach with it.

¹ Inside Search by Google. 2014. <http://www.google.com.sg/intl/en/insidesearch/> (Accessed 03/04/2015).

² The Knowledge Graph. 2014. <http://www.google.com.sg/intl/en/insidesearch/features/search/knowledge.html> (Accessed 03/04/2015).

³ Baidu Search Predict. 2014. <http://jingyan.baidu.com/article/da1091fbd723e1027949d644.html> (Accessed 03/04/2015).

⁴ Meet the new way to search. 2014. <http://www.google.com/insidesearch/features/instant/> (Accessed 03/04/2015).

⁵ Google Search Features. 2014. <http://www.google.com/insidesearch/features/> (Accessed 04/04/2015)

2.2 Modern Search Engines' Working Mechanism

After the learning of searching features, I went one level deeper. I did an in-depth research on the search algorithm and page-ranking algorithm used by different search engines. Since the page-ranking algorithm is the top secret for those search engine companies, I read many books and online blogs to get a view. And, more importantly, I also tested out and re-concluded those unofficial conclusions by doing many experiments on my own website. I will explain this in the following section.

First, I need to understand how does a modern search engine work, in other words, how is search done from algorithm to answers. Take Google as an iconic example,¹ “search starts with the web. It is made up of 60 trillion individual pages. And it is constantly growing.” However, when we are visiting websites, we do not visit pages individual by individual. Instead, we follow the links created by the site owners from page to page. Similarly, this is how crawler navigates the web. The most well known crawler is called “Googlebot” that is used by Google.² Crawlers go from link to link, copy the data on the webpages, generate a HTML snapshot and send back to back-end servers. In addition, crawlers will pay special attention to new sites, changes to existing sites and dead links.³ They work quite frequently so that new sites can be indexed as soon as possible.⁴

After crawling, the millions of gigabytes data is kept track by indexing. Considering the huge amount of data, the crawled pages should be gathered and organized in a central system that creates an index for future looking up. As a result, when we search, Google's algorithms look up your search terms in the index to find the appropriate pages. The search process gets much more complex from here. Advanced algorithms are required to first understand and then search efficiently for what we want, which involves query understanding, autocomplete, spelling checking, synonyms and so on.

Then, after all the related results are filtered at the back-end, the search engine needs to do a page ranking, which is one of the core topic of my project. According to Google, it uses over 200 factors to weigh a page such as freshness, safety, user context, site and page quality and so on. However, Google never release what exactly are these 200 factors and

¹ How Google Search Works. 2014. <http://www.google.com/insidesearch/howsearchworks/thestory/> (Accessed 03/04/2015).

² Googlebot. 2014. <https://support.google.com/webmasters/answer/182072?hl=en> (Accessed 03/04/2015).

³ Ask Google to crawl and index your URL. 2014. <https://support.google.com/webmasters/answer/6065812?hl=en> (Accessed 03/04/2015).

⁴ About Google's Regular Crawling. 2014. <https://support.google.com/webmasters/answer/34439?hl=en> (Accessed 03/04/2015).

how they are weighed. Thus, I did a lot of experiments with my own website and also tried it on VisuAlgo with the help of Dr. Steven as well on both Google and Baidu.

We are confident with our site and page quality and safety. So we focused on other factors. We first refactored our code to be more crawler-friendly by including more potential searching query information within the html files. Later, I found the importance of secondary page links. To be more specific, the more secondary page links we have, the higher our pages would likely be ranked. So we started from 3 languages, namely English, Indonesian and Chinese. We wrote Facebook posts recommending our websites and other localized forum posts. The post on renren has been shared by 511 times so far since the issued date on 1st Oct, and has reached a view time of 2250, which makes this post came out to be the first when we search the related queries on Baidu. In this sense, we hope that by redirecting, we direct more and more volume of clicks to our website and as a result, make it a primary link to our website by long-term accumulation. And we finally made it. Our website ranking arose fast beyond expectations. The experiment turned out a success. Subsequently, this method is applied to other 8 languages as well. Below is a chart comparing the page rank of our site between the original and now. (form: [previous](#)/now, based on Google, timestamp: 07/04/2015, 20:00pm)

	<title> content Search	Data Structure Visualization	Data Structure Animation	Algorithm Visualization	Algorithm Animation
English	6 th / 1 st	p.g 2, 6 th / 3 rd	None / 1 st	None / 1 st	None / 1 st
Chinese	p.g 2, 6 th / 1 st	None / 10 th	None / p.g 2, 7 th	None / p.g.2, 3 rd	None / 1 st
Indonesian	None / 1 st	None / 2 nd	None / 2 nd	None / 5 th	None / 5 th

Trying more generic searching queries on Google in Chinese:

Chinese Terms	Bitmask Visualization	AVL Tree Visualization	Graph Data Structure Visual	Segment Tree Visualization	MST Visualization
Rank	None / 4 th	None / 6 th	None / p.g.2, 4 th	None / 1 st	None / 10 th
Chinese Terms	Bitmask Animation	AVL Tree Animation	Graph Data Structure Anim.	Segment Tree Animation	MST Animation
Rank	None / 1 st	None / 2 nd	None / 1 st	None / 1 st	None / 4 th

During the experiment, I also noticed one confusing major mechanism difference between Baidu and Google. Baidu gives secondary link like blogs, forum posts a much higher rank than the primary link of the original website. This may sound not that logical, but it is the “truth” underlying Baidu we found after hundreds of search experiments. However, Google focuses more on primary links. This difference leads us to adjust our strategies for different search engines. And in my own opinion, this difference may due to the searching habits of different user groups since Baidu mainly focuses on Chinese searching. Chinese users may want to read more posts other than looking for the original website. Or probably, this may also be where their distance from Google lies in.

2.3 Modern Search Engine Optimization

During my research, I also covered the topic of search engine optimization, where blag-hat as well as white-hat techniques are discussed. Blag-hat techniques attempt to improve rankings in ways that are disapproved by the search engines, or involve deception. One blag hat technique uses text that is hidden, either as text colored similar to the background, in an invisible division, or positioned off screen. Another method gives a different page depending on whether the page is being requested by a human visitor or a search engine, a technique known as cloaking. More specifically like, page stuffing that uses a lot of duplicates. Selling and farming links are also popular. A link farm is a collection of web pages that all interlink with one another in order to increase each page's rank. When search engines detect a link selling scheme or link farm, they flag all involved sites.

However, for white-hat techniques, the crawlers read Web pages and index them according to the terms that show up often and in important sections of the page. Title, headers should definitely use the keywords and try to mention it though out the web page but not too many times otherwise will be categorized as keyword stuffing which is a blag-hat technique. Moreover to determine the quality of a web page, most automated search engines use link analysis, which means the search engine looks to see how many other web pages link to the page in question. It will help raise the page ranking by exchanging links with other related websites.

3 Development Case of VisuAlgo: Floyd's Cycle Finding Algorithm

3.1 Methodology

For VisuAlgo, we do not have a step-by-step developer manual. Hence, the requirement for me is to develop and code by learning and deducing from current code and graph library. Frankly speaking, my knowledge in front-end web development using HTML and JavaScript was elementary level. It was a great challenge for me to do the self-learning by reading thousands of links of code from multiple developers with different programming style and write out my own. Many of the time, I was so desperate to give up. However, thanks to Dr. Steven for willing to offer me consultations to lead me out of my frustration. He suggested me choosing a data structure with similar drawing elements like linked list as my starting point. I cracked the code function by function to see what change will my modification lead to and summarize what I want. I finally found out the pattern and complete the programming. I would like to conclude this methodology as read-ask-and-crack.

Another method I used a lot is divide-and-conquer. I used to hear this mentioned much from the textbook. However, this time makes me understand deeply how important this method is especially for a not small program. Thanks to Dr. Steven by helping me to define a clear working scope for each time segment and divide the big challenge into smaller parts which I could solve each at a time. And naturally the issue was solved completely in the end. [The current link of this website is: <http://visualgo.net/cyclefinding.html>]

3.2 Program Design

Thanks to Dr. Steven who has given me plenty of freedom of designing this Floyd's Cycle Finding page. I got much chance to transform what I understood from paper to animated visualizations, which is very exciting. For the operation panel, I designed a create operation that allows both random create and user defined create. After many experiments, I observed that the rho shape of the function $f(x) = (x^2 - 1)\%mod^1$ where mod is optimally being a prime number is the best. Hence, I suggested using this function to generate random rho for demonstration. And for user defined, we would also like to include those one-degree polynomials. Hence, I thought of the way using the form of $f(x) = (ax^2 + bx + c)\%mod$, where a, b, c are coefficient ranging from [-999 to 999] as long as the page can hold the picture. In this way, a or b could be zero to control the degree of the polynomial functions.

¹ Corman, T. et al, 2009, *Introduction to Algorithm*.

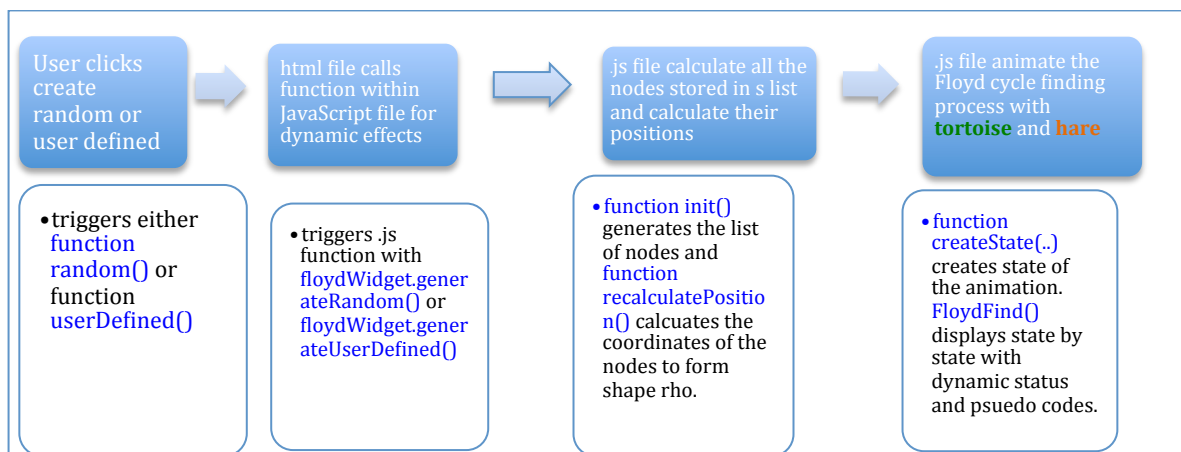
3.3 Implementation¹

My implementation phase was painful but fruitful. I would like to conclude this phase with several difficulties I met and how I solved.

- **How is the visualization triggered and animation drawn?**

I spent days and hours reading and understanding the code from previous developers, trying to find the clue. I hesitated whether I should modify a template or start from scratch. With my elementary level of web-development knowledge, how to understand the flow of the code? I am not smart. I programmed, failed, programmed, failed, failed and failed. However, what I have learnt from the failure makes them worth failing.

The below flow chart illustrates well how the visualization is realized.



- **Two parents for one node?**

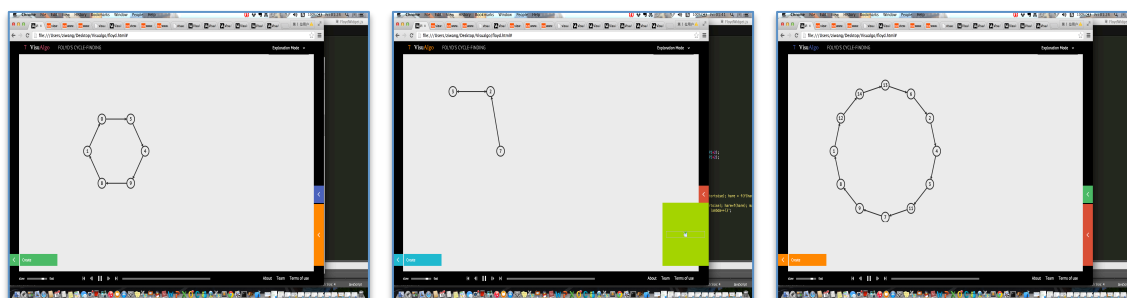
For the starting point of the cycle in rho, there will be two edges pointing to it. Then how to define its parent-child relationship for the edge drawing?

I originally wanted to modify the node object by having two parent attributes, `parent1` and `parent2`. However, I found it not reasonable since I need to modify the whole library because of one node. There must be other solutions. I observed that the edge numbers used are natural numbers corresponding to the increase of nodes. Hence, I solved the problem by using a unique negative edge number (-1) representing the edge from the one node before the code starting to the starting node. The cycle starting node's parent is hence defined as the "ending node" of the cycle.

¹ Please refer to the attached code folder named `VisuAlgo_FloydCycle`.

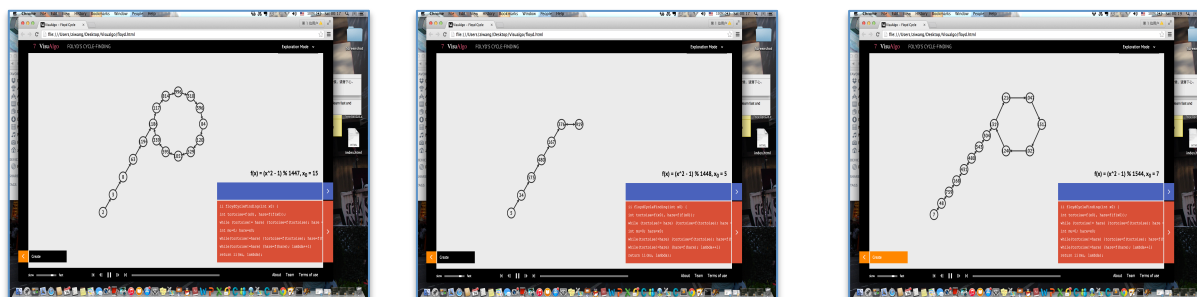
- **How to draw the rho shape?**

As explained previously, after the `init()` function initiate all the nodes generated for the cycle. It is the `recalculatePosition()` function that calculates out the x and y coordinates of each every nodes and positions them on the canvas. Below are snapshots some of very first “rho” I drew:



It can be seen from above, I could not position the tail of the circle in the end only can draw a regular polygon instead of a rho. And the position of the tail in the second graph is also wrong, since the tail of the rho should be on its left.

I solved this difficulty by thinking from the opposite. I start from positioning the circle to draw the regular polygon first. I experimented many times to find a proper center point for the canvas. Then, I calculated out the proper radius catering to different number of nodes in the cycle. Then I deduced the formula of the coordinates of the vertexes of a regular polygon from a given center point. I used the $(center_x - radius, center_y)$ as the coordinate of the last point of the tail and draw the tail with a around 35 degree slop backwards. And below are the rho shapes now:



- **Animating two nodes at the same time?**

Unlike the linked list templates I chose, the animation of Floyd’s Cycle requires showing the animating of two vertexes at the same time, namely the tortoise and hare in the Floyd’s cycle finding algorithm. I spent a lot of time coding out the `FloydFind()` function that

implements the animation of the graph as well as generating dynamic status and pseudo code.

Based on my observation of the previous animation algorithm, I noticed the primary elements that underlies the animation is the each every state created by the createState() function. Hence, instead of the previous:

createState (internalListObject, vertexTravesed, edgeTraversed), I changed it to:
createState (internalListObject, tortoiseTraversed, hareTraversed, edgeTraversed). Each state will simultaneously keep track of two set of vertex respectively, which enables the current animation we see.

3.4 Case Summary

In conclusion, the Floyd's Cycle finding features is online used for demonstrations. It may still need further testing and enhancements. Thanks to Dr. Steven, further samples have also been added corresponding to his book competitive programming 3. I am quite proud with my calculation to draw the rho. I think the shape of rho is properly drawn from an aesthetic standard. For the animation part, I do sincerely appreciate Dr. Steven who has optimized based on my version and made it nearly perfect.

One of my regret on this project is that I did not have enough time to refactor my code so that makes it more abstractive and neat. This is an invaluable experience for me to keep in my mind the importance of readability the program. Moreover, if I could have a chance to improve the program more, I would have chosen test-driven development¹ as my initial programming methodology, which means that I will design the test cases while develop the functions. In this case, I think the time spent on afterwards test phases will be shortened and the program will also be more robust.

¹ Test-Driven Development. 2014. http://en.wikipedia.org/wiki/Test-driven_development (Accessed 05/04/2015).

4 Internationalization Implementation

4.1 Study of Google Translate API & Web Translator

Google Translate has become more and more powerful these years. Its huge language database makes it possible to translate freely among hundreds of languages. However, due to its robotic translation property, some sentences especially long sentences' translations do not make sense. So how can I take advantage of its merits and avoid the demerits?

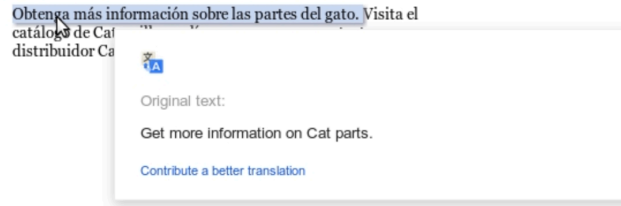
I first experimented with the web services offered by Google Translate called “Google Web Translator”. I was lucky to become one of its last batch of users. It is a pity that this free website service will not be available after 16th March 2015.¹ This service is easy to use. When user submit their website domain, it will automatically generate a piece of unique code for your website. You can insert the piece of code before the <head> tag of your html file. Then there will appear a language selection dropdown menu with more than hundreds of languages to be translated into. The disadvantage of this website is that it can only translate static web pages. When we intended to apply it on the animation pages, there will be no translation or a long lag because of the Google translate server is waiting for a static HTML snapshot to translate. Hence, this cannot fulfill our requirements since all our animations are dynamic.

However, I still gained one inspiration from this web service. The Google Web Translator has a feature that enables contribution. This is a great idea. Since the Google translations are so robotic to translate some sentences, it offers the visitors of the website a right to contribute to the translation of the website as they wish. The contributions submitted by the visitors will be collected and sent to the owner of the website and he could decide whether to use the contribution to replace the current translation or not. This mechanism is so brilliant that it combines both robotic translation and human wisdom in an easy way. [Illustration graph can be found below this paragraph.] I would also apply this mechanism in the internationalization of VisuAlgo which would be explain in detail in the program design part.

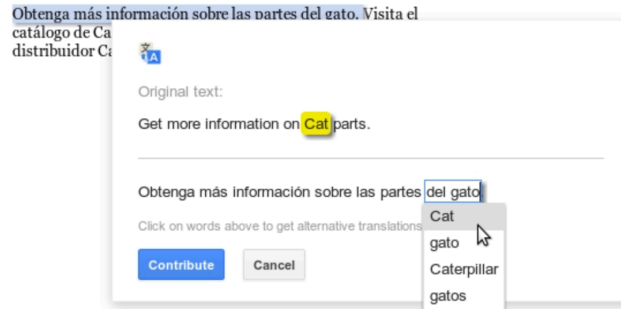
¹ Website Translator. 2015. <https://translate.google.com/manager/website/suggestions> (Accessed 05/04/2015).

Tip: These suggestions are submitted by visitors to your site. If you'd like to use one of the translation suggestions submitted by a user, select it and approve. Here's how:

When a visitor views the translation of your site using the Website Translator plugin, they can suggest a better translation by hovering the mouse cursor over the translation they'd like to correct. The phrase or sentence will be highlighted and a tooltip window displaying the original text will appear.

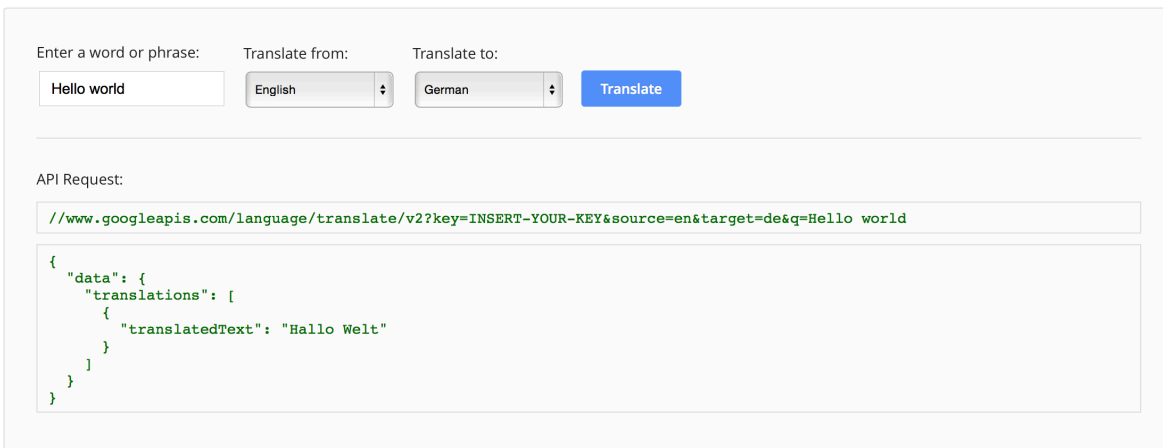


The visitor can then click on Contribute a better translations and submit the corrected translation



The suggestion will appear in Website Translator for you or another authorized Editor to approve. Once approved, the suggestion becomes a correction and will be shown whenever any user translates a page containing the source text into the target language.

Moreover, I also did research on Google Translate API. This API service is pricing based on the usage. However, it offers a free trail so I managed to play with it. “Google Translate API provides a simple programmatic interface for translating an arbitrary string into any supported language.”¹ A graph below shows its action:



¹ How Google Translate API Works. 2014. <https://cloud.google.com/translate/#features> (Accessed 05/04/2015).

Each of the translate request will generate an API request to the Google Language libraries and fetch the results dynamically. It is able to detect a language and translate it using a RESTful API, which stands for Representational State Transfer.¹ The power of this API is that we can invoke the translate API using REST from JavaScript, using the callback query parameter and a callback function. Hence, in this way we will be able to translate the changing status for each animation dynamically. However, there are also some limitations of the use of this API. For example, user cannot get multiple translations at one time of a word. This feature is only available via the web Google translate interface.

As a summary of the study of this two Google translate services. I have gained much inspiration on our internationalization project. And I will endeavor to implement the combination of their merits on our website to achieve the best result.

4.2 Program Design

During the program design phase, we were facing three major issues.

¹ Rest. 2015. https://cloud.google.com/translate/v2/getting_started#REST (Accessed 05/04/2015).

- **What translation mechanism should we take up?**

Based on my previous research on Google Translate API and Google Website Translator, I suggested we taking a similar but optimized translation method. We can use Google Translate API to translate the webpage once and store our translated string in the database. This translation would be robotic and even with errors.

In addition, we also enable our own contribution system for only authorized users to contribute. Their contributions will be stored in the database upon the submission. And we will also have admins for each respective language that will login to the system regularly and check the contributions. The admins can either approve the contribution or not. Once they approve the contribution will replace the original translation. Dr. Steven also emphasizes that we need to have at least 2 admins for each language in case of improper behavior of the admins. In this way, we are no longer limited to 11 languages and our translations will be sustainable with the users' constant contributions.

- **How should we manage our database?**

As mentioned previously, the translation of the website needs to be easy for maintenance and update. Hence, the use of a database is a must.

First, the database needs to store the translations in different languages string by string. As suggested by Dr. Steven, we need a unique index for each of the string in VisuAlgo. The delegation of the indexes to strings should be segmented. In other words, we will have first 200 indexes for the home page strings. For each animation box of the data structure or algorithm, we leave 1000 indexes for each. And within this 1000- index, we further segment it for status, pseudo code, control panel and tutorial strings. The index will not be used up at one time. The remaining space is leaved blank for the future extension of updates. In this way, we make the system friendly for maintenance and extension.

In addition, the database should store all contributions from the users for admins' approval. The contributions will be stored together with a foreign index being the index of the string which is unique. In this scenario, we also need to save the information of admin usernames and their corresponding passwords in our database.

4.3 Implementation and testing¹

¹ Please refer to the attached code folder named VisuAlgo_Internationalization.

Technical details for implementation:

Front-end: JavaScript for changing translated text, calling AJAX for suggesting translations.

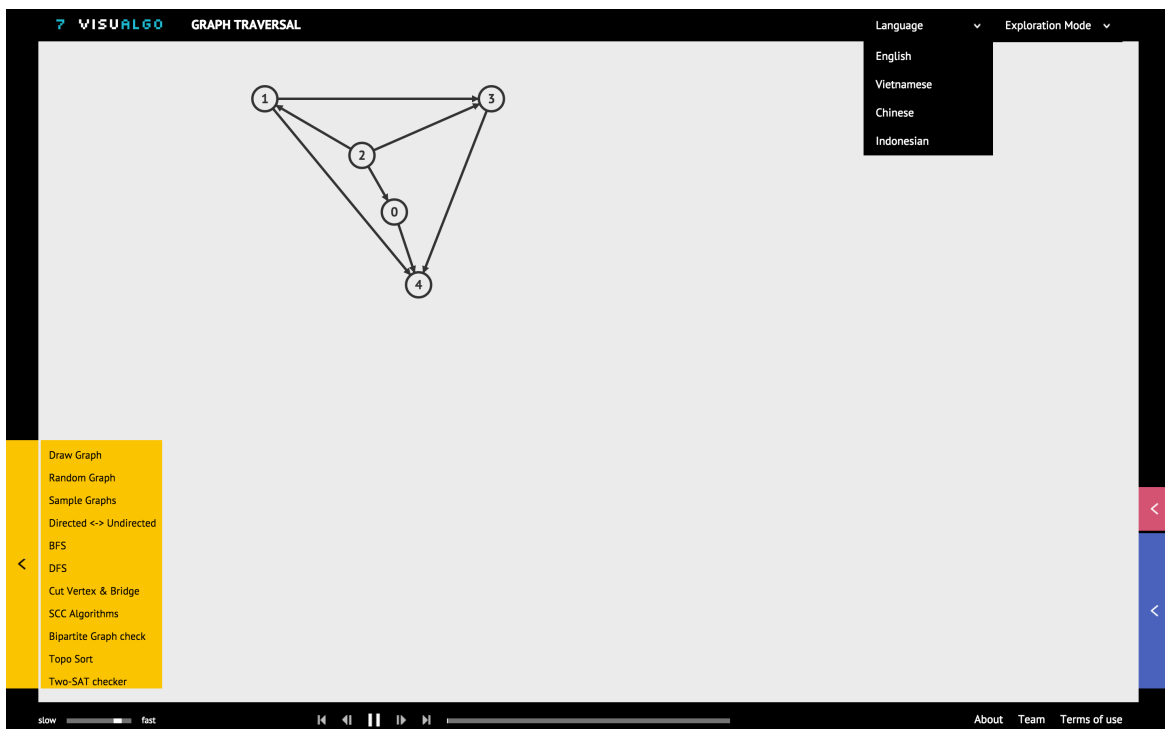
Back-end: MySQL database for hosting database.

PHP for rendering web pages and serving AJAX calls.

I am taking charge of this internationalization with CP3101B team8 as a web application case for their course project. We have implemented our prototype idea into two visualizations, namely dfsbfs.html and CycleFinding.html. Below are some screenshots of our web pages.

Screenshot_1: dfsbfs.html page in English, together with 4 languages to choose.

Screenshot_2: CycleFinding.html page in Indonesian, together with 4 languages to choose.



7 VISUALGO FLOYD SIKLUS-FINDING

Bahasa
 English
 Vietnamese
 Chinese
 Indonesian

Exploration Mode

$f(x) = (3x^2 + 7x + 5) \% 97, x_0 = 62$

Kura-kura Set dan referensi hare
 $t=40$ and $h=41$

```

int t=f(x0), h=f(f(x0)); // t=tortoise, h=hare
Phase 1: while (t!=h) { t=f(t); h=f(f(h)); }
Phase 2: mu=0; h=x0;
while (t!=h) { t=f(t); h=f(h); mu++; }
Phase 3: lambda=1; h=f(t);
while (t!=h) { h=f(h); lambda++; }
return pair(mu, lambda);

```

Membuat Acak Custom, $f(x) = (3x^2 + 7x + 5) \% 97, x_0 = 62$ GO

lambat cepat

Tentang Tim Syarat Penggunaan

Admin page for approving/rejecting translations from contributors:

VISUALGO ADMIN Logout

Hello, admin

Dashboard

- Dashboard
- Languages
- Contributors
- Add Contributor
- Registrations

Language	New Contributions	Status
Indonesian	13	Approved
Chinese	32	Approved
Vietnamese	71	Approved

VISUALGO ADMIN Logout

Hello, admin

Dashboard

Languages

- Vietnamese
- Chinese**
- Indonesian

Contributors

Add Contributor

Registrations

Chinese

ID	English	Wang
100	Graph Traversal	图的遍历 ✓ ✗
104	Draw Graph	绘制图表 ✓ ✗
106	Sample Graphs	示例图表 ✓ ✗
108	BFS	广度优先搜索 ✓ ✗
109	DFS	深度优先搜索 ✓ ✗
121	BFS	广度优先搜索 ✓ ✗
122	DFS	深度优先搜索 ✓ ✗
123	BFS	广度优先搜索 ✓ ✗
126	GO	开始 ✓ ✗
127	About	关于 ✓ ✗
128	Team	团队 ✓ ✗
216	BFS is completed. Orange edges create a BFS tree.	广度优先搜索完成。橙色边创建广度优先搜索树。 ✓ ✗
254	ignore!	请忽略 ✓ ✗
316	is free, push	被释放, 推 ✓ ✗

Contributors' page for contributions to the translation with view of their status:

VISUALGO CONTRIBUTOR Logout

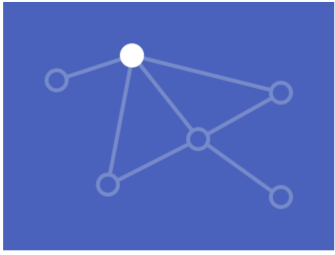
Hello, Wang

Dashboard

Sections


Pages open for translation

Graph Traversal



[Contribute Now!](#)

Cycle Finding



[Contribute Now!](#)

Hello, Wang

Dashboard

Sections

Graph Traversal

Cycle Finding

Cycle Finding

Save your translations

ID	Content	Current Translation	Your Translation	Status
1100	Floyd's Cycle-Finding	弗洛伊德的循环查找	<input type="text"/>	
1103	Language	语言	<input type="text"/>	
1104	Create	创建	<input type="text"/>	
1105	Random	随机	<input type="text"/>	
1107	GO	开始	开始	Approved
1108	About	大约	关于	Rejected
1109	Team	球队	团队	Pending
1110	Terms of use	使用条款	<input type="text"/>	
1111	slow	慢	<input type="text"/>	
1112	fast	快	<input type="text"/>	
1113	Given a function (restricted to $f(x) = (A \cdot x^2 + B \cdot x + C) \% M$ for this visualization) and an initial value x_0 , the cycle-finding problem is defined as the problem of finding the starting index μ (?) of the cycle and the	给定的函数（仅限于 $F(x) = (A \cdot x^2 + B \cdot x + C) \% M$ 此可视化）和一个初始值 x_0 。循环调查问题被定义为查找的起始索引 μ 的问题。 (μ) 的循环和序列迭代函数值的周期长 (λ) ； $\{x_0, x_1 = F(x_0), x_2 =$	<input type="text"/>	

5 Conclusions

5.1 Summary

I do not really want to write this summary part of my UROP since I do not want it to end. I have learnt so much and grown so much from this research experience under Dr. Steven's supervision. One of the best parts of this UROP is the platform of VisuAlgo, where I could put what I have learnt and researched into practice. We can witness the changes and progress made every time after a hard work. My sincere gratitude goes to Dr. Steven for his generous guidance and expertise. I would like to summarize what I have learnt from this project from two aspects.

First, for technical skills, this UROP is a super intensive learning journey. When I first started, my knowledge in web-development was just elementary. After my developing experience with VisuAlgo, now, I am more confident with my web-development technical skills. The importance is not how much I have known but the learning method I have mastered. The knowledge learnt by practice will never fade away. Moreover, I have also learnt a lot from Dr. Steven how to do individual research. I need to search a lot, read a lot and most importantly, experiment a lot. My experiment with Google Translation API using java was a failure. He encouraged me that even though it is a failure, it is still a meaningful experiment as long as I have learnt from it. I will remember the keys on design experiments and prototypes for research theory.

Then, my soft skills are also exercised tremendously throughout this UROP. Unlike other modules, UROP emphasizes on self-learning. The time is tight and the learning bell-curve is steep. It is important to have good use of resource and a fast-learning method. Many of times, I thought I could not complete the project. I was so desperate. However, every time I still managed to solve the issue. There is no problem that cannot be solved as long as I am determined enough and willing to work hard. The attitude is of critical importance of a project. Whenever dealing with a project, I need to be responsible for it. For the second phase, I am also working within a team, my teamwork ability is further exercised.

To conclude, the experience of this UROP is invaluable. Both my technical skills and soft skills are improved greatly. I am still hoping to contribute to VisuAlgo after the completion of this project.

5.2 Limitations

As mentioned above, based on our experiments with Baidu search engine, its page ranking weighs secondary links much higher than the primary website link. Our post comes first in the searching result but not the primary source. This may require further research on Baidu.

One of the limitations for us to internationalize the online testing parts and the tutorial parts using our method will result in a short lag since currently these parts need to talk to the server asking for randomly generated questions. If we could restructure the current scheme and keep a local HTML snapshot each time of all the questions and then display one by one, the lag may be significantly reduced. There exist some restrictions when we were intended to work with the server.

5.3 Recommendations for Future Work

For future work, it would be great if the online-testing and tutorial parts' working mechanism could be refactored with a better combination of PHP and HTML. Taking the advantage of the HTML snapshot to shorten the lag.

As the internalization of VisuAlgo is completed, we may need more work on promoting it to the computer science lovers around the world. And also make them using our tutorial features catering in their languages. Issues and new needs will emerge again as the user group becomes larger. However, we are always ready to fix the problem and satisfy new needs. As stated in the beginning, VisuAlgo is an on-going project. It will never satisfy with what has already achieved. I do sincerely hope it will help more and more computing lovers and even every single student on their data structure and algorithm learning.

VisuAlgo has an unmeasurably bright future. I had a fantastic experience contributing to it. I hope that there will be more developers joining our team to build VisuAlgo. I recommend a system developer manual can be written gradually for the ease of future developers.

References

About VisuAlgo. 2011. <http://visualgo.net> (Accessed 05/04/2015).

Amy N. Langville & Carl D. Meyer. *Google's PageRank and Beyond: The Science of Search Engine Rankings*. 1st ed. Princeton University Press. 2012.

Anthony, Holdener. *Ajax: The Definitive Guide*. 1st ed. 2008

Baidu Webmasters. 2013. <http://zhanzhang.baidu.com> (Accessed 05/04/2015).

Christoper D. Manning, Prabhakar, Raghavan & Hinrich, Schutze. *Introduction to Information Retrieval*. 1st ed. Cambridge University Press. 2008.

David, Amerland. *Google Semantic Search: Search Engine Optimazation (SEO) techniques That Get Your Company More Traffic, Increase Brand Impact and Amplify Your Online Presence*. 1st ed. Que Publishing. 2013.

David, Flanagan. *JavaScript: The Definitive Guide: Activate Your Web Pages*. 6th ed. 2011.

Drik Klicker. 2014. Website Penalty on Google Product Forums. 14 Setepember. <https://productforums.google.com/forum/#!category-topic/webmasters/crawling-indexing--ranking/kowJW0izpfo> (Accessed 05/04/2015).

Eric, Schmidt & Jonathan, Rosenberg. *How Google Works*. 1st ed. Grand Central Publishing. 2014.

Google Official Webmaster Central Blog. 2009. Optimize Your Crawling & Indexing. <http://googlewebmastercentral.blogspot.sg/2009/08/optimize-your-crawling-indexing.html> (Accessed 05/04/2015).

Google Official Webmaster Central Blog. 2010. Unifying Content under Multilingual Templates. <http://googlewebmastercentral.blogspot.sg/2010/09/unifying-content-under-multilingual.html> (Accessed 05/04/2015).

Google Official Webmaster Central Blog. 2011. Beyond PageRank: Graduating to Actionable Metrics. <http://googlewebmastercentral.blogspot.sg/2011/06/beyond-pagerank-graduating-to.html> (Accessed 05/04/2015).

Google Official Webmaster Central Blog. 2011. More Guidance on Building High-Quality Sites. <http://googlewebmastercentral.blogspot.sg/2011/05/more-guidance-on-building-high-quality.html> (Accessed 05/04/2015).

Google Search Features. 2013. <http://www.google.com.sg/intl/en/insidesearch/features/> (Accessed 05/04/2015).

Google Webmasters. 2014. <http://www.google.com.sg/webmasters/> (Accessed 05/04/2015).

How Google Search Works. 2011. <http://web.archive.org/web/20111104131332/http://www.google.com/competition/howgooglesearchworks.html> (Accessed 05/04/2015).

How Search Works, crawling and indexing. 2013. <http://www.google.com.sg/intl/en/insidesearch/howsearchworks/crawling-indexing.html>

(Accessed 05/04/2015).

Inside Search by Google. 2013. <http://www.google.com.sg/intl/en/insidesearch/> (Accessed 2014-10-10).

Jerry, Lee. *Ajax Programming for the Absolute Beginner*. 1st ed. 2008.

Jon, Duckett. *Web Design with HTML, CSS, JavaScript and jQuery Set*. 1st ed. 2014

Larry, Ullman. *PHP and MySQL for Dynamic Web Sites: Visual QuickPro Guide*. 4th ed. 2011

Peter, Kent. *Search Engine Optimization for Dummies*. 5th ed. For Dummies. 2012

Robin, Nixon. *Learning PHP, MySQL & JavaScript: with jQuery, CSS & HTML5*. 4th ed. 2014

Steven, Halim. *Competitive Programming*. 3rd ed. 2013

Steven, Skiena. *The Algorithm Design Manual*. 2nd ed. 2008

Thomas, Cormen et al. *Introduction to Algorithm*. 3rd ed. 2009

Trevor, Strohman & Donald, Metzler. *Search Engines: Information Retrieval in Practice*. 1st ed. Pearson Education. 2009.

Wikipedia. 2014. Search Engine Optimization.
http://en.wikipedia.org/wiki/Search_engine_optimization (Accessed 05/04/2015)

Wu Jun. *The Beauty of Mathematics*. 2nd ed. People's Post Press, 2011.

Appendix - Program Listing

Attached Program Packages List to the email:

Zip Folder: VisuAlgo_FloydCycle.zip

Zip Folder: VisuAlgo_Internationalization.zip