CS467
Fall 2019
GCrawlers: Christopher Beall, Helen Jiang, Brian Metzger

GCrawlers: Project Plan

Introduction:

The GCrawlers team will be creating a graphical web crawler. The web crawler is responsible for searching through links from a starting web page and displaying the results through a graphical interface. With little to no experience, our team is confident that we will develop the skills needed to complete this project. We plan on having weekly meetings through Google Meet to keep each other up to date and to keep track of our progress. We also plan on using Slack to keep each other informed of small changes and in order to contact each other quickly.

User Perspective:

From the end user's perspective, they will enter information into a form on the website. This will include a URL, a link limit for the crawler, a choice between breadth first search and depth first search, and an optional portion of the form that allows them to enter a keyword that is looked for on each page. After this form is submitted, they will be taken to a page that will display a graph of the result of following the links on the URL that they submitted.

This software has multiple use cases which are slightly overlapping. Since the primary function of our software is to identify and follow the links from a source URL, the main use case is for individuals who want to map a site, or identify how pages and even other sites are connected to a particular page. From a website administrator point of view, this could be useful for identifying the different ways that each page is reachable from a specific page of their site. Alternatively, they could also check to see which links take the user away from their domain or even identify links which are broken. From a security perspective, this software could be a useful preliminary reconnaissance tool for penetration testers. It would allow them to map a site that they are attempting to test.

Initial Software Structure:

Our crawler software can be divided into a few different parts: a UI/website, a crawling program, and a data transfer program. The website will be the user-facing portion of this software that takes the user's input parameters and displays the output of the other programs. There will need to be a defined interface between the website and the crawler, between the crawler and the data transfer program, and between the data transfer program and the website in order for this software to function. In the event of broken or bad links, the crawler will log the bad link and move on to another link. It will also track which links have already been followed in order to

prevent loops. As a courtesy to the websites which are being crawled, the function to retrieve all links from a URL will also check for “noindex” meta tags to determine if the owner of the website is attempting to keep the page private. Each component should be subdivided into smaller sub-components which each handle a different part of the software:

UI/Website:

- Input page to gather parameters
- Output page to display results from program
- History page to display recent history of websites crawled

Crawler Program:

- Function for retrieving all links from a given URL
- Function for selecting a random URL from a collection of URLs
- Function for performing DFS on the URLs
- Function for performing BFS on the URLs
- Function for outputting the results to a log file
- Function for detecting keyword on requested page to halt crawl

Data Transfer Program:

- Function for reading the contents of the log file
- Function for connecting to the website
- Function for transferring the data to and from the website

Initial Software Dependencies:

This project will run on a *nix-based server such as Flip. Software versioning will be controlled using GitHub and each team member will be able to use any development environment they feel most comfortable with to work on their assigned portion. The repository will also contain a simple *nix-based setup script that quickly sets up and installs any necessary dependencies of the project. Anticipated software and dependencies are:

- NodeJs (UI/Website)
- Express (UI/Website)
- Handlebars (UI/Website)
- Sessions (UI/Website)
- JQuery (UI/Website)
- Bootstrap (UI/Website)
- Python (Crawler Program)
- Requests library (Crawler Program)
- BeautifulSoup library (Crawler Program)
- OS library (Crawler Program)
- Sockets (Data Transfer Program)
- Forever (All programs)

Team Member Assignments:

Assignment	Due	Format	Assigned To
Project Plan	10/14 @ 11:59 PM	Written Report	All
Week 4 Progress Report	10/21 @ 11:59 PM	Video (2 - 5 minute)	All
Week 5 Progress Report	10/28 @ 11:59 PM	Video (2 - 5 minute)	All
Mid-Point Report	11/04 @ 11:59 PM	Zip File of Source Code/Instructions	All
Week 7 Progress Report	11/11 @ 11:59 PM	Video (2 - 5 minute)	All
Week 8 Progress Report	11/18 @ 11:59 PM	Video (2 - 5 minute)	All
Week 9 Progress Report	11/25 @ 11:59 PM	Video (2 - 5 minute)	All
Final Report	12/06 @ 11:59 PM	Written Report	All
Project Poster	12/06 @ 11:59 PM	PDF with High-Res Images of Product	All

Team Member Project Budgets:

Brian Metzger:

Time Period & Tasks	**Estimate of Hours**
Week 3 - Create main Crawler program file with main structure - Begin functionality to get all links from URL - Create resources to test URL collection function - Write tests for URL collection function	17
Week 4 - Begin function to detect supplied keyword in HTML of URL - Write tests for HTML keyword search function	12

Week 5 - Begin function to output results of crawl to log file (work with Christopher to ensure interoperability with data transfer program) - Write tests of output function	18
Week 6 - Begin DFS function to iterate through the list of links (No randomization implemented yet) - Write tests for DFS function	15
Week 7 - Begin BFS function to iterate through the list of links - Write tests for BFS function	15
Week 8 - Begin function to select random link from the list of links - Write tests for random function	12
Week 9 - Write tests for full integration of all crawler functions - Debug any outstanding issues	13
Week 10 - Create final submission and demonstration	8
Total Time	110

Christopher Beall:

Time Period & Tasks	**Estimate of Hours**
Week 3 - Begin creating structure of data structure program - Begin Work on creating data structure to hold information	17
Week 4 - Finish creating format to transfer data to the website	12
Week 5 - work with Brian to ensure interoperability with data transfer program and crawler output	15
Week 6 - Finish data transfer from crawler to website functionality	15
Week 7 - Debug and test data transfer from crawler to website	12

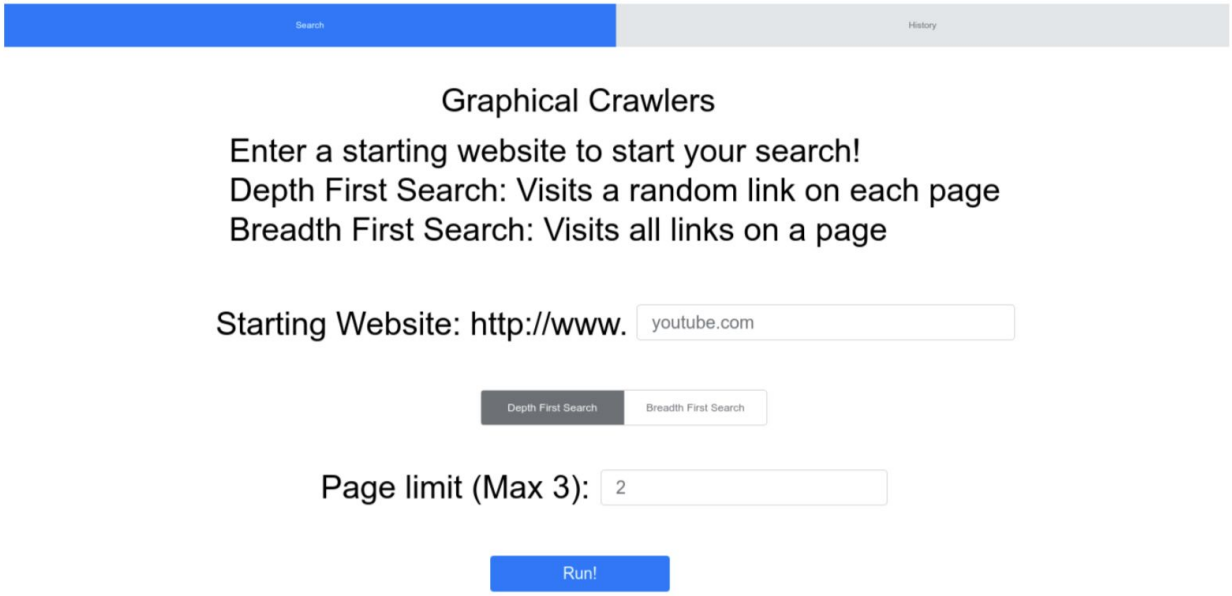
functionality	
Week 8 - Finish data transfer process from website to crawler	13
Week 9 - Debugging and testing of data transfer from website to crawler - Thorough debugging and testing of all data transfer functions	18
Week 10 - Create final submission and demonstration	8
Total Time	110

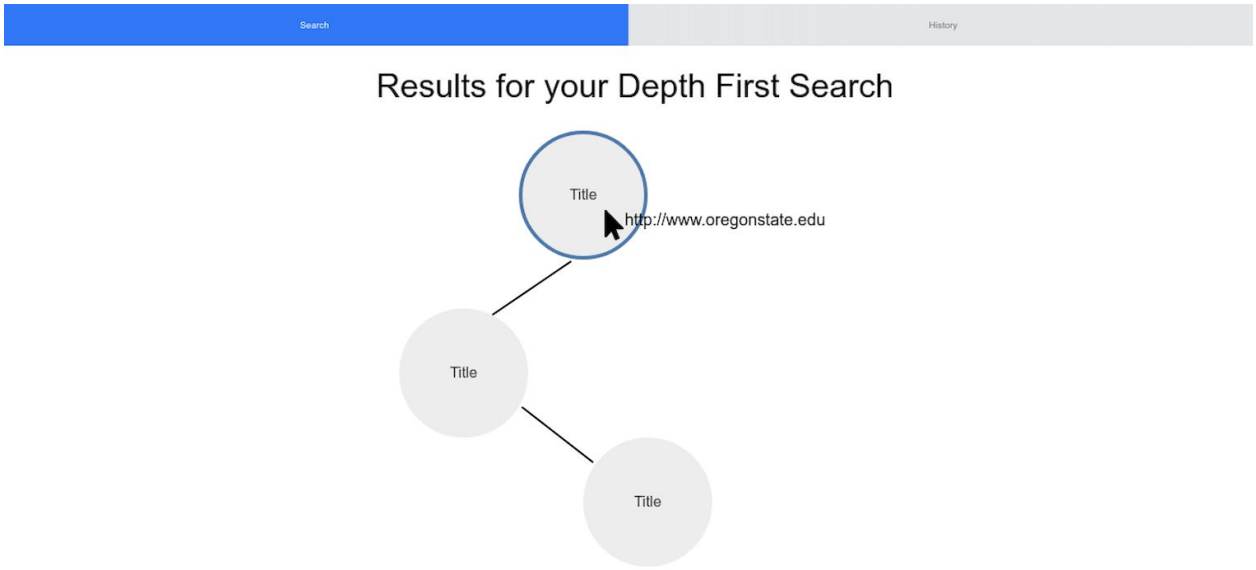
Helen Jiang:

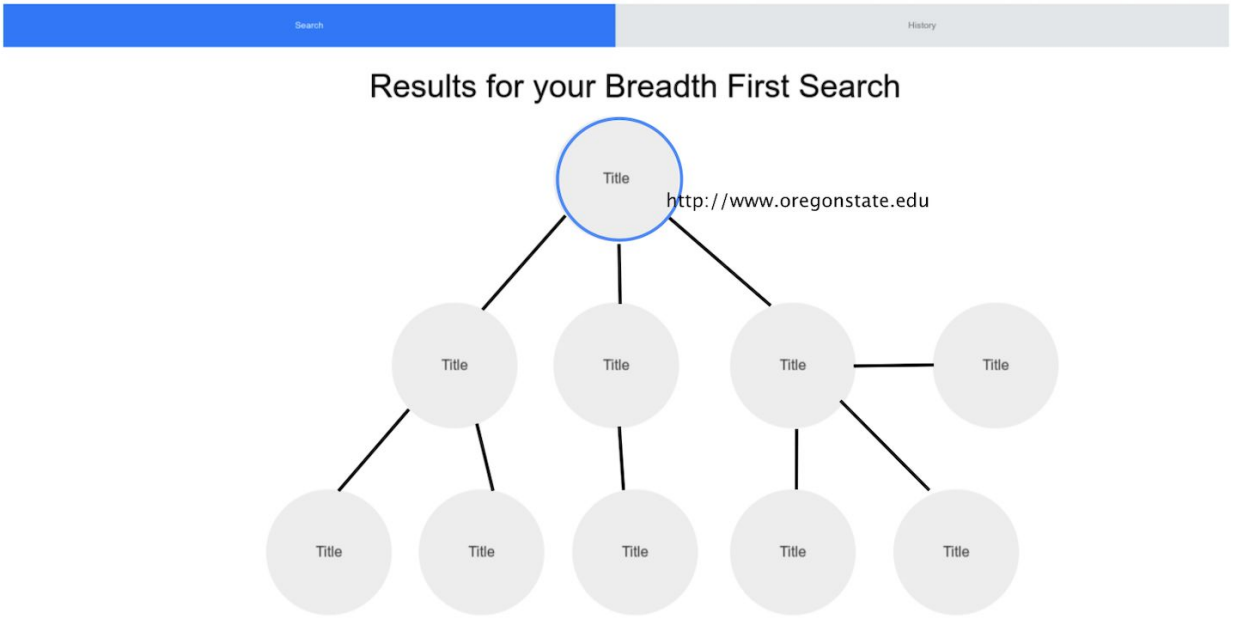
Time Period & Tasks	**Estimate of Hours**
Week 3 - Begin implementing website framework by setting up with NodeJs and express - Implement web interface with HTML/handlebars	12
Week 4 - Add functionality to HTML with NodeJs/express - Start styling with CSS/bootstrap for form	16
Week 5 - Finish styling of forms and initial UI - Test and debug forms - Work with Christopher for relaying of initial queries	16
Week 6 - Work with Christopher for receiving data transfer results	12
Week 7 - Start styling of results with CSS/bootstrap	15
Week 8 - Integrate Sessions for past searches/history - Test and debug Sessions integrations	18
Week 9 - Write tests for full integration of all crawler functions, data transfers with web page - Final debugs	13

Week 10 - Create final submission and demonstration	8
Total Time	110

Prototype:

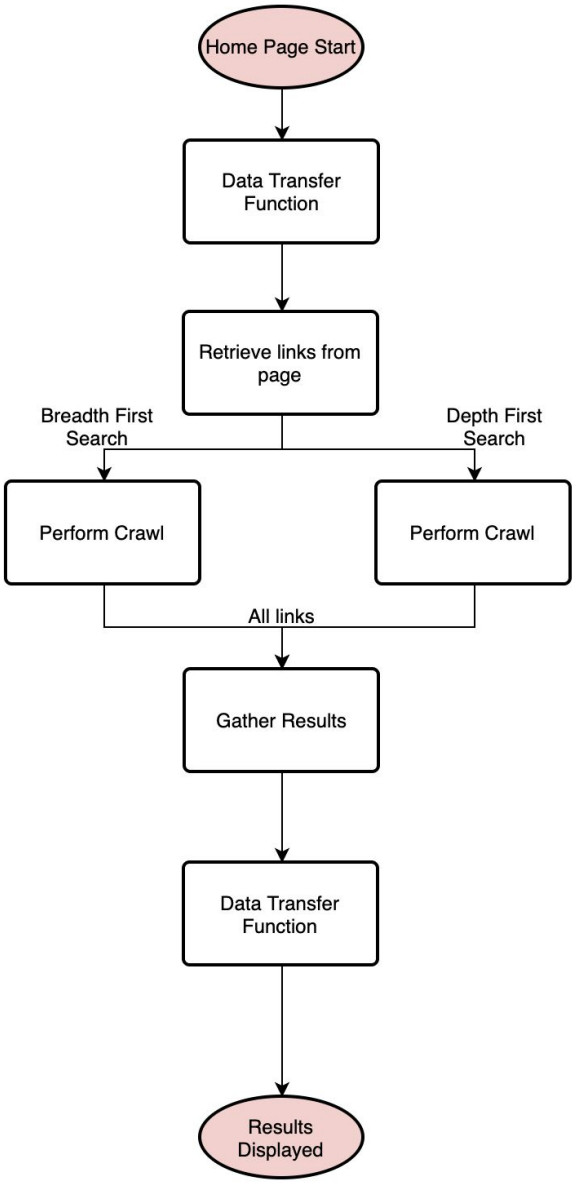



Search
History
Results for your Depth First Search
Title
http://www.oregonstate.edu
Title
Title


Search
History
Results for your Breadth First Search
Title
http://www.oregonstate.edu
Title
Title
Title
Title
Title
Title
Title
Title
Title

Date	Starting Pages	Type	Results	Search Again
1/1/2019	http://www.youtube.com	Breadth	View	Search
1/1/2019	http://www.oregonstate.edu	Breadth	View	Search
1/1/2019	http://www.oregonstate.edu	Depth	View	Search
1/1/2019	http://www.oregonstate.edu	Breadth	View	Search
1/1/2019	http://www.oregonstate.edu	Breadth	View	Search

Flow Chart:



Future Extensions:

- Allow for larger searches by increasing the maximum pages allowed. Larger searches will require more time and memory. If time is allotted, all three group members can contribute to this extension, with most of the work falling under the crawler functions.
- Allow for updating graphics in (almost) real-time. With every few searches, the crawler can send results to the data transfer functions and be displayed on the website. D3.js can be used for real-time updates of the graphics. If time is allotted, all three group members can contribute to this extension.

- A progress bar indicating the progress of the crawler with a cancel button to stop at that time. The crawler will be responsible for sending the progress statistics to the data transfer functions. The progress bar will then be updated on the webpage. All three group members can contribute to this extension as well, with most of the work falling under the UI functions.

Conclusion:
Our team is confident that we can create a graphic web crawler that will be useful for web search engines and other analytical projects. We will primarily use slack and google hangouts for communication, google docs for documentation and git for version control. Project hours will be around 330 hours and we aim to finish by the proposed schedule.