CS7009 Network Applications

Using Information from Github to suggest next language for user to learn

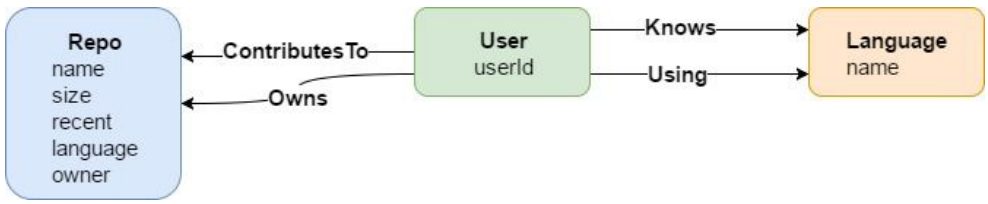
Helen Wallace 11424442 May 2017

Using Github API and Neo4J

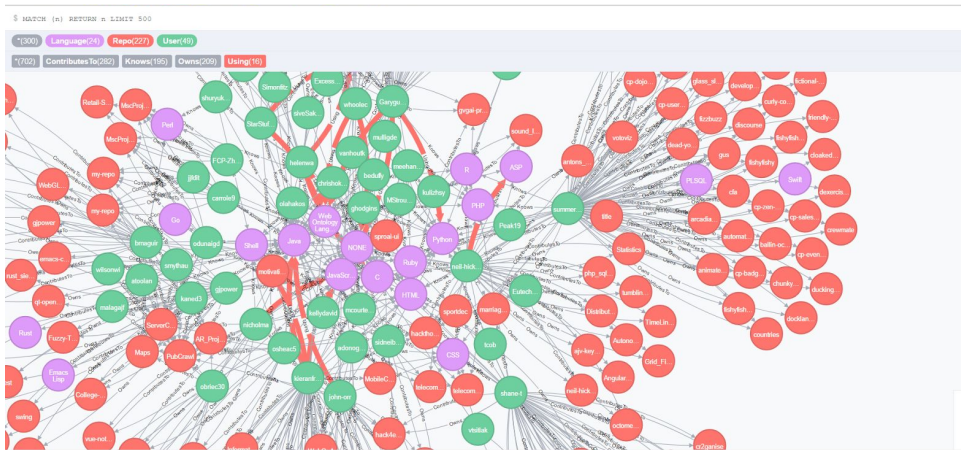
The GitHub API was used to gather a large amount of information. First a login is required via the Github OAuth to obtain a token, this was used to bypass the rate limiter. Following on from this the crawler can be started with a specified start user.

First the we retrieve all the users repositories and their info. Then for each repository all contributors are retrieved. Then the repositories for each contributor are fetched and the cycle goes on for a specified number of hops.

All the information is saved in a Neo4j dataBase. Alongside the information about Users and Repositories information about Language is also added. For each Repository the relevant language node is added to the database. Then each time a user is seen to contribute to a repository a link stating that they Know the language of the repository is added. I the repository is recent (last commit within 6 months) a link stating that the language is in Use is also added. Below is a diagram describing the nodes and possible links.



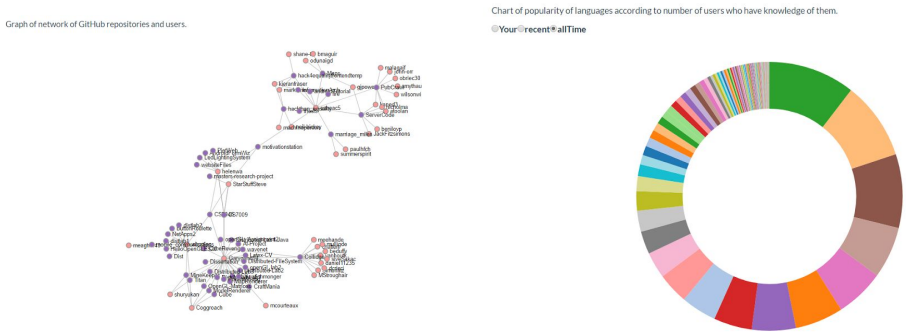
Graph Structure;



Screenshot of graph segment

Yesod and D3.js

A frontend was created using the yesod framework this made getting the OAuth token from GitHub relatively simple. A profile was set up for the logged in user. First a force directed graph of Users and Repositories is displayed. Below this an adjustable pie chart is shown. First it displays the user's languages then switches to show the languages in use by all users recently and over all time. This information is gathered by counting the “Knows” and “Using” links to each Language node



Graph Displayed; Language Chart

Lastly a List of Languages to learn next is displayed. This is gathered by using the path length between the User and the Language and it’s current popularity. Only Paths via the ‘Using’ relation are considered. This ensures

Languages used by users who share languages with the user are considered. The following cipher is used to fetch languages of user XXXX at a distance 3.

```
MATCH (n:User {userId: "helenwa"})
MATCH (l:Language)
MATCH p=(n)-[:Using*3]-(l)
RETURN Distinct( l.name) as name, length(p)
ORDER BY length(p)
```

This has yet to be expanded to include longer path lengths or filter out languages already known by the user.