COMP5313 LARGE SCALE NETWORK

ASSIGNMENT 2 REPORT

STACKOVERFLOW ORIENTED LEARNING AND PROGRAMMING

Network Analysis on stackoverflow.com

May 12, 2017 [1]

Lin Han 460461265

[1]Powered by $\LaTeX$

Contents

Introduction

The stackoverflow.com site is becoming more and more important for students and IT professions. This crowd sourcing website with reputation/vote system now has over 7 million users raising and answering 14 million questions. What behind it is a large scale network connecting users, knowledge and almost everything about programming.

This project focuses on the programming language aspect: which languages are popular, what the key knowledge area of one specific language is, and how programming language evolves. In the following part of this report, the method of retrieving data will be given first. After that, analysis based on retrieved data will be presented. At the end of this report, we will reach a conclusion on the results derived from stackoverflow.com's data.

Method

The project is divided into two separate part: 1.spider based on Scripy and 2.data analysis part.

Scripy Spider

The spider of this assignment is a relatively simple one, which contains only a spider program and the item class giving the information that what we want from web page. Because stackoverflow.com has no anti-spider schema, a simple spider is enough to crawl data we want.

The code follows the simplest design of Scrapy framework:

```python
# -*- coding: utf-8 -*-

# Define here the models for your scraped items
# See documentation in:
# http://doc.scrapy.org/en/latest/topics/items.html

import scrapy

class StackoverflowItem(scrapy.Item):

    links = scrapy.Field()
    views = scrapy.Field()
    votes = scrapy.Field()
    answers = scrapy.Field()
    tags = scrapy.Field()
    questions = scrapy.Field()
    time_stamps = scrapy.Field()
```

items.py

```
import scrapy
from stackoverflow.spiders.items import StackoverflowItem

class StackoverflowSpider(scrapy.Spider):

    name = "stackoverflow"

    def start_requests(self):
        urls = ['http://stackoverflow.com/questions?page={page}&sort=newest&pagesize
            =50'.format(page=page)
                for page in range(166000, 167000)]
        for url in urls:
            yield scrapy.Request(url=url, callback=self.parse)


    def parse(self, response):
        for index in range(1, 51):
            sel = response.xpath('//*[@id="questions"]/div[{index}]'.format(index=
                index))
            item = StackoverflowItem()
            item['votes'] = sel.xpath('div[1]/div[2]/div[1]/div[1]/span/strong/text()
                ').extract()
            item['answers'] = sel.xpath('div[1]/div[2]/div[2]/strong/text()').extract
                ()
            item['views'] = "".join(sel.xpath('div[1]/div[3]/@title').extract()).
                split()[0].replace(",", "")
            item['questions'] = sel.xpath('div[2]/h3/a/text()').extract()
            item['links'] = "".join(sel.xpath('div[2]/h3/a/@href').extract()).split("
                /")[2]
            item['tags'] = sel.xpath('div[2]/div[2]/a/text()').extract()
            item['time_stamps'] = sel.xpath('div[2]/div[3]/div/div[1]/span').extract
                ()
            yield item
```

spider.py

Results and Data Analysis

The retrieved data is in JSON format, contains the link, the question, tags, number of views, number of votes, number of answers, and the created time for each single question. For easy analysis purpose, there are two retrieving order: newest and most voted.

Popularity

Since stackoverflow.com is a crowd sourcing website with reputation system, we can easily make a guess that the popularity distribution of questions follows **Power Law**, which leads to an imbalanced distribution. And the real data prove this.

After sorting 1 million the most voted questions in descending order, we can draw the line graph below:

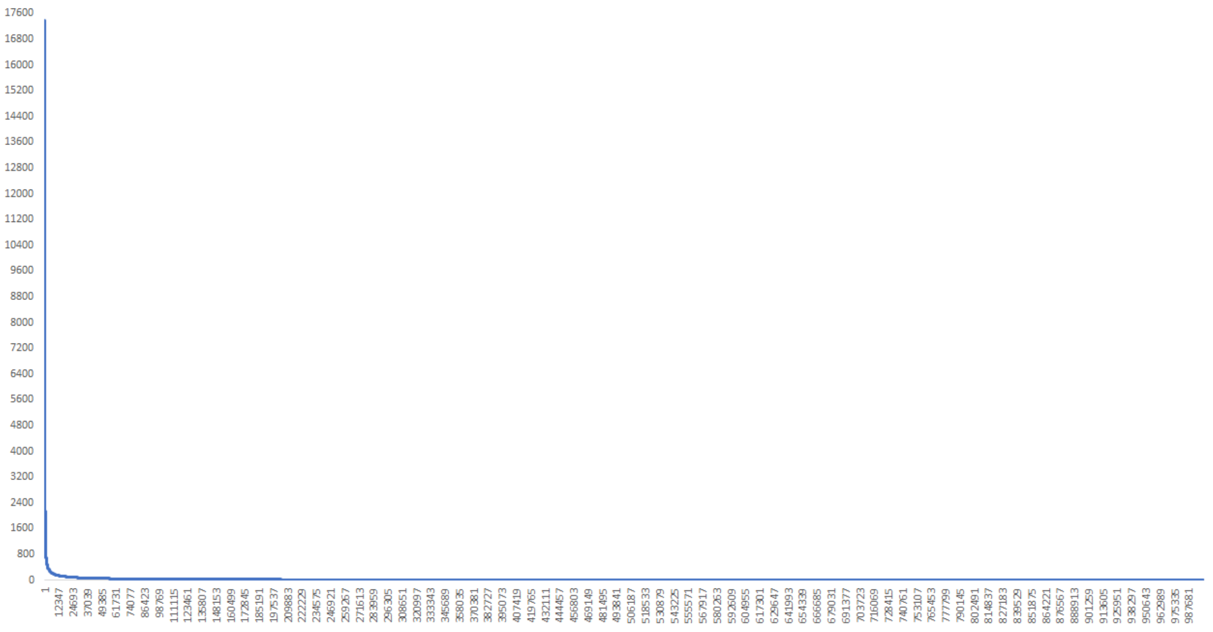

Figure 1. Popularity distribution (votes)

The same distribution also can be seen from distribution based on number of answers.

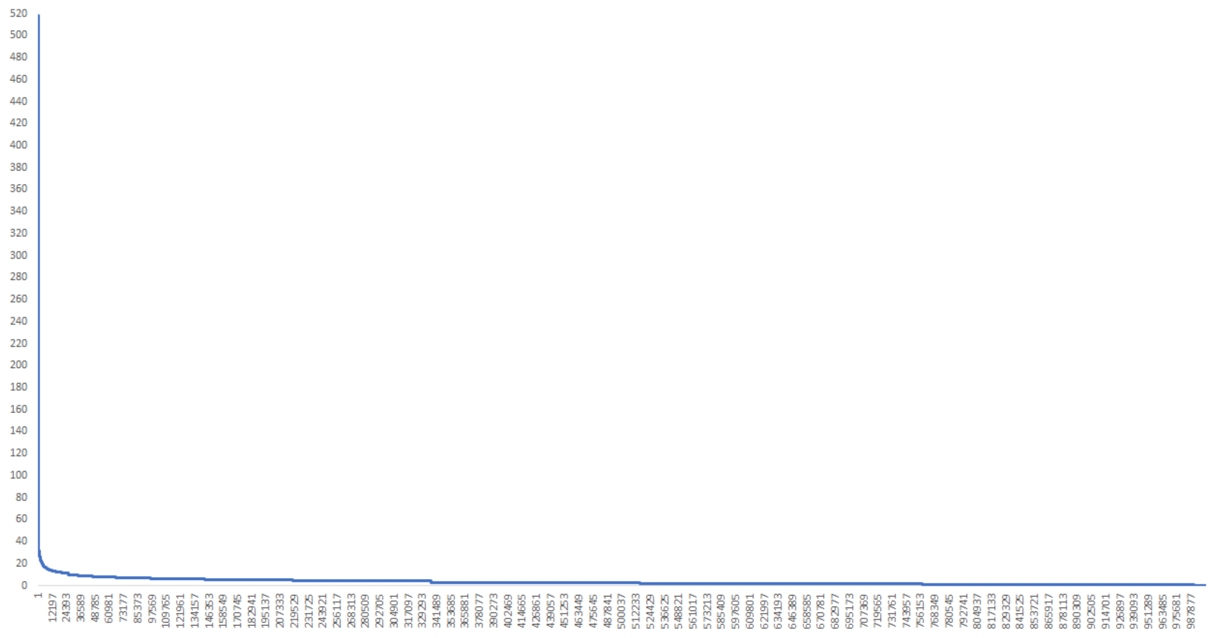

Figure 2. Popularity distribution (answers)

Hence, we can easily conclude that **Power Law** applies in a reputation/votes crowd sourcing system like stackoverflow.com.

Key Knowledge of Language

In addition to the popularity that we can directly see from our data, there are some inner correlation of how technologies are connected with each other. All data used in this part are top 50,000 most voted questions.

In this project, we pick Python as example and analyze how technologies are connected within Python.

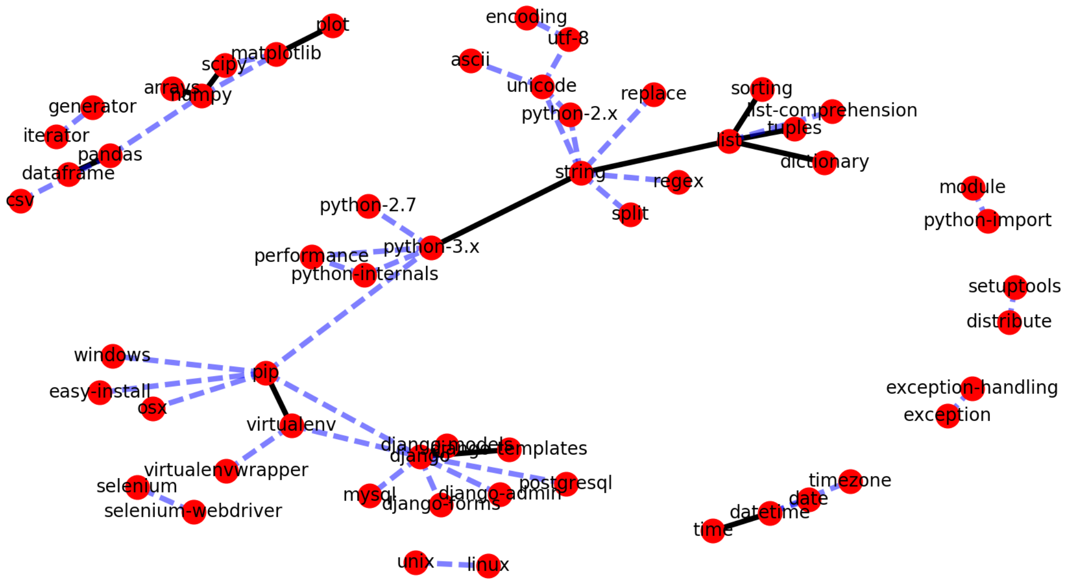

Figure 3. Technologies Graph (Python, weighted)

In Figure 3, vertices are tags, edges are tags that appears within the same question, normal line represents a high correlation(more than 10 times here), and dashed line represents a low correlation. We can see that there are several components in the graph. Each of them targeting a specific application of Python. For example, the left upper corner component shows the techniques one have to master for scientific computing using Python.

In this way, this analysis would give significant help for one to form learning road map before really learning a language or technology.

Language Evolution

Within such a large network like stackoverflow.com, how a language evolves over time can also be seen. Below are three snapshots of number of questions related to major programming languages, taking at a specific time within a year.

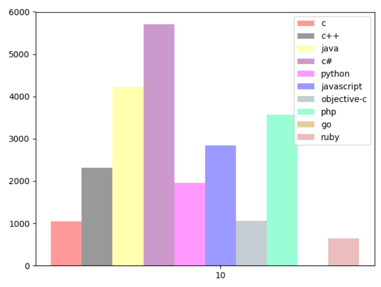

Figure 1: Language Snapshot 2010

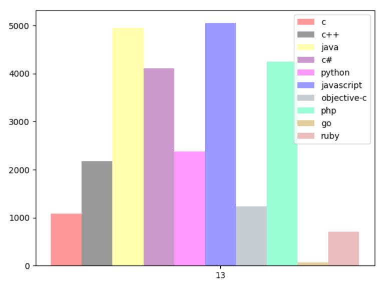

Figure 2: Language Snapshot 2013

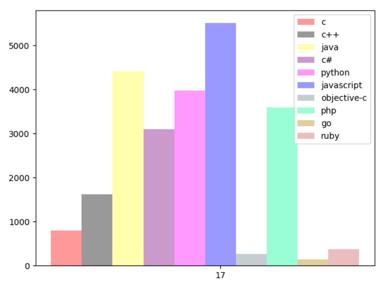

Figure 3: Language Snapshot 2017

As we can see, with the trend of World Wide Web, related languages like Python, JavaScript have a growing popularity. In addition to the trend, new version of programming language can also lead a evolution. One example is the release of Java8, in the 2017 data set, there are five questions both having the tags *functional-programming* and *java* while there is only one question in the other two periods.

Conclusion

From this project, we can see that stackoverflow.com not only gives an overview on popularity of questions on programming techniques, but also shows the highly-concerned knowledge of one language and its evolution.

Using this analysis, we can have a better plan on how we balance efforts during learning or developing. Pick the most popular and potential language and concentrate more on the hardest part of it. This may make it easy to be a coding master.

REFERENCES

1. Easley, D., & Kleinberg, J. (2010). *Networks, crowds, and markets: Reasoning about a highly connected world.* Cambridge University Press.

2. Scrapy, https://scrapy.org/