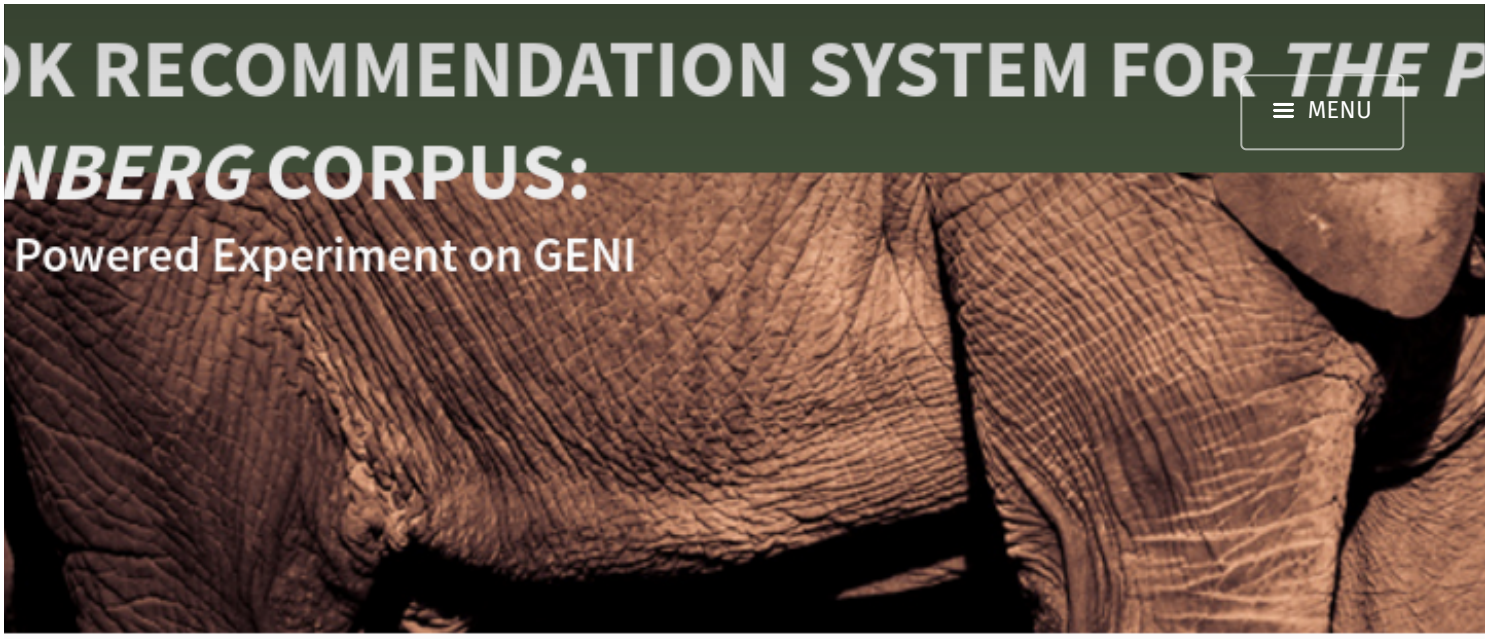


	Relevance Score
anor Putnam and Arlo Bates	0.5127368468
Other Stories, by Anonymous	0.4070709409
ndrew Lang, by Andrew Lang and Others	0.3817256423
A. J. Glinski	0.3742766082
ce and His Travelling Cloak, by Dinah Maria Mulock	0.3553404975
ce, by Miss Mulock (Pseud. of Maria Dinah Craik)	0.3537180176
ith Gautier	0.3405963303
ll from Fairy Tales and Folklore,	0.3386015326
y Jules Claretie	0.3292568203
Rider Haggard	0.3009157590

A Hadoop-powered book recommendation system

Kimberly Devi Milner

29 FEBRUARY 2016 on project, distributed systems, hadoop

This project shows how to use Hadoop to process a large text corpus (in this case, the Project Gutenberg public domain book dataset) and use it

to power a simple book recommendation engine.

This experiment should take about 60-90 minutes to run.

To reproduce this experiment on GENI, you will need an account on the GENI Portal, and you will need to have joined a project. You should have already uploaded your SSH keys to the portal and know how to log in to a node with those keys.

- Skip to Results
- Skip to Run my experiment

Background

Apache's Hadoop is an open-source storage and processing framework that can scale out to thousands of servers, and handle petabytes of data. Hadoop was produced shortly after Google's 2004 File System research paper.

Hadoop's processing works by splitting both data and processing jobs onto clusters of commodity hardware. It can use many machines to process jobs in parallel, and also keep working even if one machines fails (an idea known as fault-tolerance).

Hadoop's distributed storage system is also scalable. This means that users can continue to add capacity to a full file system by adding more workers to the cluster, without disproportionately influencing the performance of the cluster.

Our Recommendation System

Our project uses Hadoop to find instances of relevant keywords in the Project Gutenberg text corpus and recommend books that contain many instances of those keywords.

We approached the problem of creating a recommendation system as follows:

We generated a word frequency list from the entire corpus (a one-time step that makes subsequent queries much faster). We downloaded all of the available Project Gutenberg books over a period of a couple of days, storing them on a GENI VM (an XOXLarge type). Once we had the books (~36,000 books at 17GB), we created a word frequency list (~ 3.6GB), where every line contained a stemmed word, a particular book id, the total number of words in that book, and the number of times the stemmed word appeared.

So, when given a set of words to look for, we only need to process that list, rather than the entire corpus.

A single MapReduce job is then used to process the file.

In the map stage, the lines containing the words of interest are printed. These lines are the input to the reducer job, which applies a simple term frequency algorithm to produce a relevancy score S_b for every book passed to it

$$S_b = \frac{1}{W} \sum_{w=1}^{W} TF_w$$

where W is the number of search terms, and TF_w is the term frequency of word w in a book and is computed as

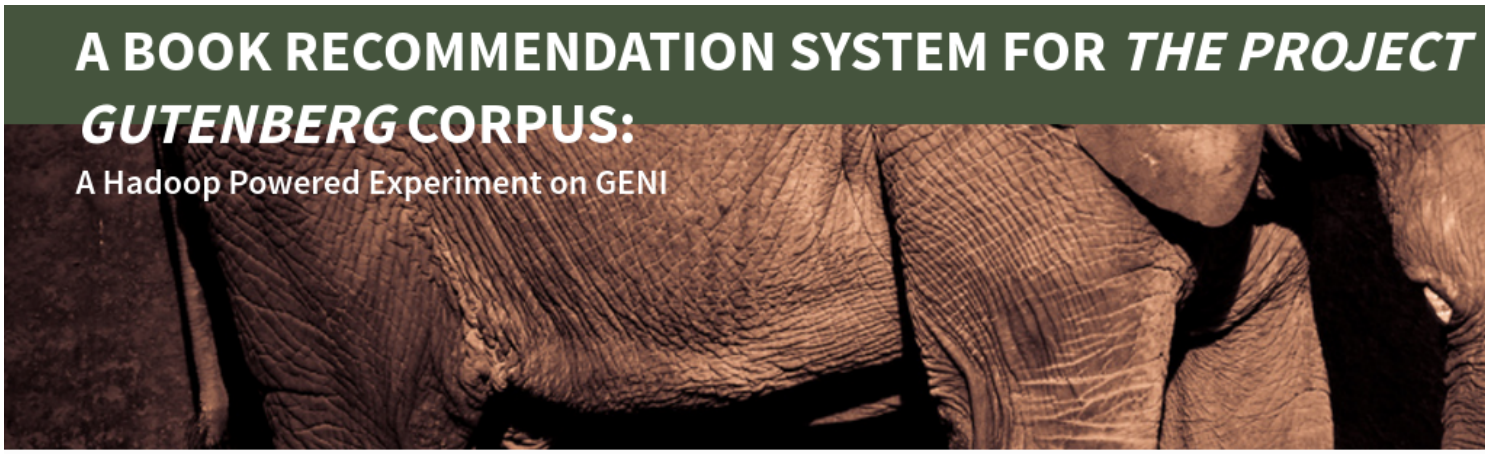
$$TF_w = \frac{c}{t\alpha}$$

where c is the number of times word w appeared in a particular book, t is the total number of words in that book, and α is just an amplifying constant.

The books with the highest scores are then passed to the user.

Results

The following books were recommended for the terms "frog" and "prince":

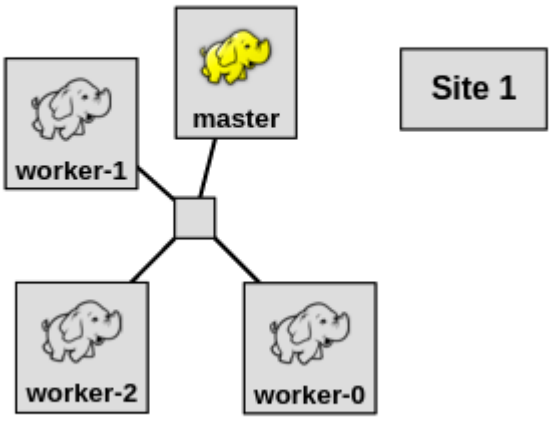


Book	Relevance Score
Prince Vance, by Eleanor Putnam and Arlo Bates	0.5127368468
The Frog Prince and Other Stories, by Anonymous	0.4070709409
The Fairy Books of Andrew Lang, by Andrew Lang and Others	0.3817256423
Polish Fairy Tales, by A. J. Glinski	0.3742766082
The Little Lame Prince and His Travelling Cloak, by Dinah Maria Mulock	0.3553404975
The Little Lame Prince, by Miss Mulock (Pseud. of Maria Dinah Craik)	0.3537180176
The Usurper, by Judith Gautier	0.3405963303
Stories to Read or Tell from Fairy Tales and Folklore,	0.3386015326
Prince Zilah, Vol. 1, by Jules Claretie	0.3292568203
Moon of Israel, by H. Rider Haggard	0.3009157590

Run my experiment

On the GENI portal we'll use Jacks to reserve several Exogeni VMs. In a new slice, click "Add Resources", scroll down to the "Choose RSpec" section and select the "URL" option. Copy the URL of the RSpec (https://git.io/vXcEJ) and enter it into the text box for the URL field, then click "Select".

It will load the following topology:



Click on "Site 1" on the canvas and choose an available ExoGENI aggregate. Then reserve your resources. Return to your slice page and wait for your nodes to be ready to log in.

SSH into the master VM you created. You are now logged into the VM that coordinates the processing that goes on in the cluster.

First make sure all the machines on the cluster can communicate. Test for connectivity with the following ping commands. (If you are unable

to ping to any of your worker machines from master, delete your resources on Jacks and get another.)

```
sudo ping -c 5 worker-0
sudo ping -c 5 worker-1
sudo ping -c 5 worker-2
```

Get the files you need to run the experiment by typing the following:

```
git clone https://github.com/rollingcoconut/hadoopExprOnGeni.git
```

Navigate to the "hadoopExprOnGeni" folder and execute the script "install.sh" to install the linux tools bc, unzip, and httpd, and also configure your experiment environment. Type the following to change directory and execute "install.sh":

```
cd hadoopExprOnGeni
sudo ./install.sh
```

Now start the Hadoop application and place the 1GB Gutenberg dataset you just downloaded into the Hadoop's distributed filesystem with the following commands:

```
sudo su hadoop -
cd /home/hadoop
./startHadoop.sh
```

If you haven't changed directories you should still be in /home/hadoop and logged in as the hadoop user.

To get a book recommendation, run the "exprStart.sh" script with a list of terms of interest, e.g.:

```
./exprStart.sh 1 frog prince
```

(The first argument, here "1", represents the factor by which to increase the logical block size a map job works on.)

To see your book recommendation, find out the public IP address of your master node by running:

```
wget http://ipinfo.io/ip -qO -
```

Then visit that IP address in a browser to see your recommendations.

When you're finished, don't forget to delete your resources to free them up for other experimenters!

Notes

Acknowledgements

This project was supported by the National Science Foundation and by GENI as a Research Experience for Undergraduate (REU) project.

The Rspecs in this experiment are based on the ones in this GENI tutorial. A more recent Hadoop tutorial is available on the ExoBlog.

Software versions

- CentOS Linux release 7.2.1511 (Core)

- Python 2.7.5
- Github repository: https://github.com/rollingcoconut/hadoopExprOnGeni

Possible extensions

If you're interested in modifying our algorithm for calculating relevance scores, you may want to edit the reducer.py file. That's where the relevance scores are computed.

Known issues

If you interrupt your MapReduce run, the next time you run "exprStart.sh" your job may be stuck at 0%. This can be resolved by following the steps outlined in "restartYARN.sh"

Kimberly Devi Milner

Read more posts by this author.

Did you reproduce this experiment? Have useful information to share with other intrepid researchers? Post it here! Comments are posted following moderation.