On the Cost of Mining Very Large Open Source Repositories

Sean Banerjee
Carnegie Mellon University

Bojan Cukic
University of North Carolina at Charlotte

BIGDSE, Florence 2015
Introduction

- Issue tracking is a central part of software maintenance and management.
- Open source bug repositories provide rich datasets for researchers.
- Researchers use millions of reports to address questions such as:
  - Who should fix this problem?
  - Is this problem new or duplicate?
  - Which report best describes this duplicate?
### Millions of Reports

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Date Viewed</th>
<th>Total Reports</th>
</tr>
</thead>
<tbody>
<tr>
<td>RedHat</td>
<td>January 19, 2015</td>
<td>1,183,340</td>
</tr>
<tr>
<td>Mozilla</td>
<td>January 19, 2015</td>
<td>1,123,125</td>
</tr>
<tr>
<td>Novell</td>
<td>January 19, 2015</td>
<td>913,595</td>
</tr>
<tr>
<td>Eclipse</td>
<td>January 19, 2015</td>
<td>457,796</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Date Viewed</th>
<th>Total Reports</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>RedHat</td>
<td>May 4, 2015</td>
<td>1,218,445</td>
<td>35,105</td>
</tr>
<tr>
<td>Mozilla</td>
<td>May 4, 2015</td>
<td>1,161,313</td>
<td>38,188</td>
</tr>
<tr>
<td>Novell</td>
<td>May 4, 2015</td>
<td>929,583</td>
<td>15,988</td>
</tr>
<tr>
<td>Eclipse</td>
<td>May 4, 2015</td>
<td>466,360</td>
<td>8,564</td>
</tr>
</tbody>
</table>

Repositories are forever growing and evolving.
Goals

- Highlight the drawbacks of using subsets of repositories.
- A time, space and cost analysis of mining complete repositories.
- A case study of our own experience in mining some of these large repositories.
- A series of challenges that remain difficult when mining large repositories.
## Research Use of Repositories

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Duration</th>
<th>Method</th>
<th>Classification</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Firefox</td>
<td>Start to 2005</td>
<td>Word Frequency</td>
<td>50% [1]</td>
<td>Only FIXED, DUPLICATE, OPEN</td>
</tr>
<tr>
<td>Eclipse</td>
<td>Start to 2005</td>
<td>Word Frequency</td>
<td>20% [1]</td>
<td>Only FIXED, DUPLICATE, OPEN</td>
</tr>
<tr>
<td>Firefox</td>
<td>Jan 1, 2004 to April 1, 2004</td>
<td>Word Frequency</td>
<td>93% [3]</td>
<td>Utilized only 77 reports</td>
</tr>
<tr>
<td>Mozilla</td>
<td>Feb 2005 to Oct 2005</td>
<td>Word Frequency</td>
<td>50%[4]</td>
<td></td>
</tr>
<tr>
<td>Firefox</td>
<td>Apr 2002 to Jul 2007</td>
<td>Discriminative Model/SVM</td>
<td>68%[5]</td>
<td></td>
</tr>
<tr>
<td>Eclipse</td>
<td>Jan 2008 to Dec 2008</td>
<td>Discriminative Model/SVM</td>
<td>67%[5]</td>
<td></td>
</tr>
<tr>
<td>Firefox</td>
<td>Start to June 2010</td>
<td>Word Frequency</td>
<td>53%[6]</td>
<td>Used only latter 50% for testing</td>
</tr>
<tr>
<td>Mozilla</td>
<td>Jan 2010 to Dec 2010</td>
<td>Word Frequency</td>
<td>70%[7]</td>
<td></td>
</tr>
<tr>
<td>Eclipse</td>
<td>Jan 2008 to Dec 2008</td>
<td>Word Frequency</td>
<td>80%[7]</td>
<td></td>
</tr>
<tr>
<td>Large Eclipse</td>
<td>Start to Dec 2007</td>
<td>Word Frequency</td>
<td>70%[7]</td>
<td></td>
</tr>
<tr>
<td>Firefox</td>
<td>Start to March 2012</td>
<td>Word Sequence</td>
<td>68%[8]</td>
<td>Used only latter 50% for testing</td>
</tr>
<tr>
<td>Mozilla</td>
<td>Jan 2010 to Dec 2010</td>
<td>Topic Modeling</td>
<td>80%[10]</td>
<td></td>
</tr>
<tr>
<td>Eclipse</td>
<td>Jan 2008 to Dec 2008</td>
<td>Topic Modeling</td>
<td>85%[10]</td>
<td></td>
</tr>
<tr>
<td>Firefox</td>
<td>Start to March 2012</td>
<td>Multiple Methods</td>
<td>72% [9]</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Duration</th>
<th>Method</th>
<th>Results</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eclipse</td>
<td>Jan 2002 to Sept 2002</td>
<td>Machine Learning</td>
<td>30% (Recall)[11]</td>
<td>No WONTFIX, WORKSFORME, and INVALID</td>
</tr>
<tr>
<td>Eclipse</td>
<td>Sept 2004 to May 2005</td>
<td>Machine Learning</td>
<td>64% (Precision)[14]</td>
<td></td>
</tr>
<tr>
<td>Firefox</td>
<td>Sept 2004 to May 2005</td>
<td>Machine Learning</td>
<td>57% (Precision)[14]</td>
<td>No WONTFIX, WORKSFORME, and INVALID</td>
</tr>
<tr>
<td>Eclipse</td>
<td>Oct 2005 to May 2006</td>
<td>Machine Learning</td>
<td>70% (Precision)[15]</td>
<td></td>
</tr>
<tr>
<td>Firefox</td>
<td>Feb 2006 to Sept 2006</td>
<td>Machine Learning</td>
<td>75% (Precision)[15]</td>
<td></td>
</tr>
<tr>
<td>Eclipse</td>
<td>Apr 2001 to Nov 2008</td>
<td>Linguistics</td>
<td>71% (Recall)[17]</td>
<td></td>
</tr>
<tr>
<td>Eclipse</td>
<td>Oct 2001 to Mar 2010</td>
<td>Machine Learning</td>
<td>86% (Accuracy)[19]</td>
<td></td>
</tr>
<tr>
<td>Mozilla</td>
<td>May 1998 to Mar 2010</td>
<td>Machine Learning</td>
<td>84% (Accuracy)[19]</td>
<td></td>
</tr>
<tr>
<td>Eclipse</td>
<td>Dec 2006 to Jan 2007</td>
<td>Linguistics</td>
<td>89% (Accuracy) [24]</td>
<td></td>
</tr>
<tr>
<td>Mozilla</td>
<td>Dec 2006 to Jan 2007</td>
<td>Linguistics</td>
<td>59% (Accuracy) [24]</td>
<td></td>
</tr>
</tbody>
</table>
Evolving Repositories
Negative correlation (-0.66) between # one-time users and # FIXED reports. Positive correlation (0.79) between # one-time users and # INVALID reports.
Evolving Repository: Users Per Year

**Distinct Users Per Year**

- **Years**: 1998 to 2012
- **Graph** shows the number of distinct users each year for Eclipse and Mozilla.

**Average Number of Reports Per Distinct User Per Year**

- **Years**: 1998 to 2012
- **Graph** shows the average number of reports per distinct user each year for Eclipse and Mozilla.
Report Type vs. User Maturity

Eclipse: Report Type by Number of Reports Submitted

Mozilla: Report Type by Number of Reports Submitted
The “Big Data” Problem

- We should use the complete (and multiple) projects in order to create generalizable observations.

- Why do we (in research) seem to stay away from using large issue report data sets?
  - Not “cheap” to mine large repositories.
  - Little is known about the true time and cost of mining such large repositories.
Challenges in Mining Large Repositories

- Main challenges
  - Obtaining the data.
  - Storing the data.
  - Processing the data.
- Each challenge has a time component and a cost component associated with it.
Obtaining the Data

- No “one-click” download link.
- Each XML report has ~100kb of information.
  - Total of 350GB in all 4 repositories.
- 32 hours to download RedHat.
  - A million sequential wget commands was perceived as a Denial of Service (DoS) attack.
- Periodic sleep commands between each wget command helped us.
  - This increased total time required to 1 week.
Storage Cost

- If each report must be compared to all priors.
  - An $O(n^2)$ problem.
  - RedHat: $7 \times 10^{11}$ matches.
  - Mozilla: $6 \times 10^{11}$ matches.
  - Novell: $4 \times 10^{11}$ matches.
  - Eclipse: $1 \times 10^{11}$ matches.
Storage Cost

- If each match contains
  - 4 byte integer for the report ID.
  - 4 byte float for the match score.
- Then, we have the following storage requirements:
  - RedHat: 6TB of storage.
  - Mozilla: 5TB of storage.
  - Novell: 4TB of storage.
  - Eclipse: 1TB of storage.
  - 16TB of storage for a single similarity measure.
Storage Cost

- If the researcher wanted to compare similarity measures:
  - String match vs. word frequency vs. topic modeling
  - $16 \times 3 = 48$TB of storage
  - ~$2,200.00 in storage alone (using 6TB WD Drives)
- For reliability the data should be in a NAS.
  - ~$2,000.00 for a Synology (or other) 12 bay NAS.
  - Total cost goes to ~$4,200 in storage alone.
- Hard drive costs are declining, BUT repositories are also growing.
Datasets are too large for main memory.
  - Building a system with several hundred GB of RAM can cost upwards of $10,000.

Read and write time from disk is the bottleneck.
  - SSDs are needed as magnetic disks are too slow.
  - SSDs cost 4x as much as a magnetic drive.

Processing on a single machine is impossible.
  - A parallel/distributed/cloud processing.
Variable size of each problem report makes it difficult to exactly quantify the time needed.

We processed Eclipse, Firefox and OpenOffice:
  - Cluster of 25 Apple iMacs (250GB SSD, 8GB RAM, Intel i7 Quad Core 2.8GHz processor).
  - Able to process 50,000 reports per day for a single similarity measure.

For the 4 datasets using 3 similarity methods.
  - 221 days of sustained processing time.
Processing on Amazon EC2

- Amazon EC2. “Storage Optimized - Current Generation i2.8xlarge” systems
  - 104 ECUs, 244GB of RAM and 6.4TB of SSDs.
  - Each instance costs $6.820 per hour.
  - Either need 221 instances or 221 days of compute.
  - Total of $36,000 to process the data.
  - Does not account for time required to download 48TB of similarity data.
GPU Processing

- Tesla K40s provide a 12GB, 2880 CUDA core processing unit.
  - Each unit costs $4,000, however Nvidia does offer research units through a simple proposal process.

- We are currently investigating the feasibility of processing problem reports on GPUs.
  - Must address the bottlenecks associated with transferring data from main to GPU memory.
  - A GPU cluster can cost upwards of $10,000 to build and maintain.
### Summary

<table>
<thead>
<tr>
<th>Activity</th>
<th>Time</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obtaining the data</td>
<td>Upwards of 1 week</td>
<td>Marginal</td>
</tr>
<tr>
<td>Processing the data</td>
<td>Upwards of 31 weeks</td>
<td>$36,000 on Amazon EC2</td>
</tr>
<tr>
<td>Storing the data</td>
<td>Variable</td>
<td>$4,200</td>
</tr>
</tbody>
</table>

- A researcher would spend upwards of $40,000 to process 4 datasets using 3 methods.
- Or, if a local distributed solution similar to our Mac Lab is available, then over 31 weeks of local processing.
Case Study

- We developed an automated triaging system.
- Computed similarity matches for all reports from the following projects:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Reports</th>
<th>Total Reports</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eclipse</td>
<td>363,770</td>
<td>Dec 2011</td>
</tr>
<tr>
<td>Firefox</td>
<td>111,205</td>
<td>Mar 2012</td>
</tr>
<tr>
<td>Open Office</td>
<td>124,476</td>
<td>Jan 2014</td>
</tr>
</tbody>
</table>
Computed similarity scores using 24 metrics.

Total number of matches generated:
- Eclipse: $6.6 \times 10^{10}$
- Open Office: $7.7 \times 10^{9}$
- Firefox: $6.2 \times 10^{9}$
- Total: $8.0 \times 10^{10}$

Each similarity score has 100 bytes of information.
- 6 variants of each base method.

Total storage: $8.0 \times 10^{10}$ bytes $\times$ 100 $\times$ 4 = 32TB.
- $\sim$ $1,500 in storage alone.
Case Study

- Used a compute cluster consisting of:
  - 25 Apple iMacs with 250GB SSD, 2.8GHz Intel i7 quad core processors and 8GB of RAM
  - Processed 12,500 files a day for 4 base methods.
  - Approximately 50 days of sustained processing.
- The compute cluster would cost $37,500 to replicate locally, at $1,500 per machine.
Case Study

- Amazon EC2 “Storage Optimized Current Generation i2.8xlarge” provides the following:
  - 104 ECUs, 244GB RAM, 6.4TB of SSD storage
  - Costs $6.820 per hour to operate
- Would cost just under $10,000 to process all three datasets.
  - Does not factor in time required to download the 30TB of processed data.
Challenges: Obtaining Software Repositories

- No universal research dataset in use.
  - Repositories evolve, universal data would be a snapshot.

- Time consuming to download data.
  - Often perceived as a DoS attack.

- We encourage repository owners to provide archived snapshots of the data.
  - Or, PROMISE should store such snapshots.
  - Provides everyone access to the same dataset.
Challenges

Computing Resources

- Very difficult to work with complete datasets at this time.
  - Either too time consuming or too expensive.
  - Conclusions from subsets of the data may be biased.

- Private clouds at Universities.

- We encourage the open source community to offer idle compute resources for analysis.
  - Folding@home and SETI are examples of successful community based projects.
Summary

- Abundance of data in open source repositories.
- Small subsets used in research due to time and cost, but can lead to biased results.
- Universities and user communities have unused resources that should be utilized.
- Encourage open source community to provide daily, weekly and monthly data archives.
- Wait for the computational infrastructure to evolve 😊.