An Algorithm for Temporal Analysis of Social Positions

Scott Christley, Greg Madey
Dept. of Computer Science and Engineering
University of Notre Dame

Supported in part by National Science Foundation, CISE/IIS-Digital Society & Technology, under Grant No. 0222829
Motivation

- Successful software development requires various positions to be filled; developers, testers, administrators, management, end-users, etc.
- People in Open Source Software community self-select into a social position on a software project.
- We don’t know what these positions are; emerged from the self-organization of the community.
- Do people stay in same social position, or does there position change over time?
- Positional analysis seeks to group actors into disjoint subsets according to their social position in the network.
Structural Equivalence

- Actors who are similarly embedded occupy similar social position.
- C ~ D have same relationships with same other actors.
- Exact equivalence is too strict so use an approximate measure, like Euclidean distance.
- Weighted relationships
Clustering

• Standard data mining algorithms
  – K-means, Expectation-Minimization (EM)

• What’s wrong with Euclidean distance?
  – Data mapped to points in an N-dimensional space.
  – Points “close” in space are in same cluster.
  – Normalization techniques very important.
  – Not comparing the underlying distributions.

• Assume Gaussian (normal) distribution

• What can we use instead of a distance metric?
  – Statistical test
Clustering with a Statistical Test

- Fisher’s contingency-table test (non-parametric)
  - Chi-square family of goodness-of-fit tests

- Given two independent samples
  - First sample, $S_1$, with $n_1$ random variables
  - Second sample, $S_2$, with $n_2$ random variables
  - Where $n_1$ not necessarily equal to $n_2$, each r.v. in each samples placed in one of $C$ categories.

- $H_0$: The distributions of $S_1$ and $S_2$ do not differ.
- $H_A$: The distributions $S_1$ and $S_2$ differ.
- Structural In-equivalence
Algorithm (Intersection)

While (still unclustered samples)
   Put all unclustered samples into one cluster.
While (some samples not yet pairwise compared)
   A = Pick sample from cluster
      For each other sample, B, in cluster
         Run statistical test on A and B.
         If significant result
            Remove B from cluster.

• Rejection of null hypothesis means A and B **must** be in different clusters.
• Confidence level tightens/broadens cluster inclusion.
• Any statistical test for a two-sided test problem.
OSS Activity

• User performs an activity for a project.
• 21 activities; submit bug, submit feature request, assign bug, post forum message, create file release, create project task, etc.
• Multi-relational, weighted, bipartite network.
  – Activity = relation, weight = activity count
• Activity distribution for user/project pair defines a sample for our statistical test.
• That is, the activity a user performs on a project defines their social position for that project.
Social Positions of OSS

<table>
<thead>
<tr>
<th>Social Position</th>
<th>Size</th>
<th># of clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brief Flame</td>
<td>122654</td>
<td>1</td>
</tr>
<tr>
<td>Message Posting</td>
<td>50067</td>
<td>4</td>
</tr>
<tr>
<td>Task Management</td>
<td>2762</td>
<td>5</td>
</tr>
<tr>
<td>Release Management</td>
<td>6509</td>
<td>5</td>
</tr>
<tr>
<td>Documentation</td>
<td>1266</td>
<td>4</td>
</tr>
<tr>
<td>Job Posting</td>
<td>899</td>
<td>2</td>
</tr>
<tr>
<td>Artifact Management</td>
<td>1674</td>
<td>6</td>
</tr>
<tr>
<td>Administrators</td>
<td>10377</td>
<td>4</td>
</tr>
<tr>
<td>Not Categorized</td>
<td>13786</td>
<td>1546</td>
</tr>
</tbody>
</table>

Total User/Project Pairs: 209994
Temporal Analysis

- Previous analysis, activity over 10 years, lose knowledge of evolution of positions.
- How to deal with time (data)?
  - Global time; snapshot of the whole network at points in time: node/edge add/remove, attribute change, tends to get aggregate measures.
  - Local time; user/project’s first activity is time 0, aligns actors in a time-relative way to the network, egocentric viewpoint.
- Chunk data into monthly activity, run clustering algorithm for data for each time period.
## Temporal Social Positions of OSS

<table>
<thead>
<tr>
<th>Social Position</th>
<th>Period 1</th>
<th>Period 2</th>
<th>Period 3</th>
<th>Period 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brief Flame</td>
<td>127302</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Message Posting</td>
<td>49754</td>
<td>1418</td>
<td>828</td>
<td>151</td>
</tr>
<tr>
<td>Administrators</td>
<td>10356</td>
<td>5415</td>
<td>905</td>
<td>496</td>
</tr>
<tr>
<td>Release Management</td>
<td>6304</td>
<td>1001</td>
<td>796</td>
<td>869</td>
</tr>
<tr>
<td>Task Management</td>
<td>3466</td>
<td>625</td>
<td>254</td>
<td>401</td>
</tr>
<tr>
<td>Artifact Management</td>
<td>1967</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Documentation</td>
<td>1130</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Job Posting</td>
<td>1125</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Not Categorized</td>
<td>4904</td>
<td>2002</td>
<td>1313</td>
<td>1105</td>
</tr>
<tr>
<td>Handyperson</td>
<td>7282</td>
<td>8280</td>
<td>6664</td>
<td></td>
</tr>
<tr>
<td><strong>Total User/Project</strong></td>
<td><strong>206308</strong></td>
<td><strong>17743</strong></td>
<td><strong>12376</strong></td>
<td><strong>9686</strong></td>
</tr>
<tr>
<td><strong>Total Clusters</strong></td>
<td><strong>397</strong></td>
<td><strong>183</strong></td>
<td><strong>143</strong></td>
<td><strong>139</strong></td>
</tr>
</tbody>
</table>
Summary

• Clustering algorithm using a statistical test.
  – Don’t have to specify # of clusters a priori.
  – No assumption of underlying distribution.
  – Must be appropriate statistical test.
• Temporal Analysis
  – How you organize/view your data is important.
  – Global metrics --> global time
  – Egocentric measures --> local time
Iterative Classification

- Order of comparison matters.
- Clustering is NP-complete so intractable to check all combinations to find the optimal.
- Iterative approach
  - Perform initial clustering
  - Calculate cluster center