A NEW METRIC FOR PREDICTING SOFTWARE CHANGE USING GENE EXPRESSION PROGRAMMING

By:

Dr. Ruchika Malhotra  
Department of Software Engineering  
Delhi Technological University

Ms. Megha Khanna  
Acharya Narendra Dev College  
University of Delhi
“You can’t control, what you can’t measure” – Tom DeMarco

It is crucial to measure various Object-Oriented (OO) attributes of a software like coupling, cohesion, size etc. in order to continuously assess and control a software product effectively.

OO metrics can be used to develop quality models which predict various software quality attributes like fault proneness and maintainability.
CHANGE PRONENESS

- Change proneness is an important software quality attribute.
- It is the probability of a class of a software to change in the forthcoming release of a software i.e. after it goes into operation.
Identification of Change Prone (CP) classes is crucial:

- To manage and allocate adequate resources like cost, effort and time to CP classes.
- To take preventive design measures in early phases of software development life cycle in CP classes to prevent introduction of defects.
- To perform rigorous verification activities in CP classes to detect changes in initial phases of software development.
To develop a new metric through which we could ascertain whether a class is CP.

If a metric can determine the occurrence of change in a class in the initial phases of a software product life cycle, we could manage testing and maintenance resources effectively.
In the past, researchers have applied traditional statistical techniques for predicting CP classes while certain others have evaluated machine learning techniques.

No study till date examines the applicability of evolutionary algorithms for predicting change.

This study evaluates the applicability of Gene Expression Programming (GEP) for prediction of CP classes.
WHY GEP??

- According to Sherrod, GEP is 60,000 times faster than traditional genetic algorithms.
- Further GEP chromosomes are
  - simple and easy to interpret with a number of operations and transitions.
  - and formulate precise expression trees even after undergoing operations like selection and reproduction with modification.
OBJECTIVES OF THE STUDY

- Validation of the Chidamber and Kemerer (CK) metrics suite using the GEP algorithm to ascertain CP classes.
- Development of a new metric for classifying CP classes.
- Validation of the proposed metric to assess its effectiveness.
Dependent Variable: Change Proneness
- Change is calculated by counting the Source Lines of Code (SLOC) added, deleted or modified in a class.

Independent Variables: Chidamber & Kemerer metrics suite and SLOC.
- Coupling Between Objects (CBO)
- Weighted Methods of a Class (WMC)
- Number of Children (NOC)
- Depth of Inheritance Tree (DIT)
- Response for a Class (RFC)
- Lack of Cohesion amongst Methods (LCOM)
SOFTWARE DETAILS

- Two open source software developed using C++ language.
- Simutrans: A transport simulation game.
- Glest: 3D strategy game.
- Source Code available at http://sourceforge.net/

<table>
<thead>
<tr>
<th>Name</th>
<th>Ver.</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simutrans</td>
<td>1 1 1-3</td>
<td>99,331 SLOC</td>
</tr>
<tr>
<td></td>
<td>1 1 2-3</td>
<td>1,17,568 SLOC</td>
</tr>
<tr>
<td>Glest</td>
<td>1.0.10</td>
<td>13,764 SLOC</td>
</tr>
<tr>
<td></td>
<td>3.2.2</td>
<td>17,453 SLOC</td>
</tr>
</tbody>
</table>
DATA COLLECTION

- Compute OO metrics for all classes of the previous version.
- Extract common classes from both the versions of the software.
- Compute change statistics (SLOC added, SLOC deleted, SLOC modified) for all the common classes after line by line comparison.
- Determine ALTER variable.
- Data Point = OO metrics + Change Statistics

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Points</th>
<th>Changed Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simutrans</td>
<td>510</td>
<td>45%</td>
</tr>
<tr>
<td>Glest</td>
<td>108</td>
<td>66%</td>
</tr>
</tbody>
</table>
DATA COLLECTION PROCEDURE

1. **Prepare**
   - Previous version
   - Software
   - Recent Version

2. **Pre-process**
   - Pre-process

3. **Comparable**
   - Comparable Previous Version
   - Comparable Recent Version

4. **Compare**
   - Compare

5. **Metrics Collection**
   - Metrics Collection
   - Combine

6. **Change Statistics**
   - Change Statistics
   - Data Points
Ten-fold Cross Validation

- Data set divided into ten subsets randomly.
- Numerous iterations are performed by using one of the ten subsets as the test set and the other nine are used for training.
PERFORMANCE EVALUATORS

- **Sensitivity (Sens.)**: It is the ratio of correctly predicted CP classes to the actual number of CP classes.
- **Specificity (Spec.)**: It is the ratio of correctly predicted not change prone (NCP) classes to the actual number of NCP classes.
- **Accuracy (Acc.)**: It is the percentage of number of classes which were correctly predicted to be CP or NCP.
- **Precision (Prec.)**: It is the ratio of correctly predicted CP classes to the total number of CP classes.
- **f-measure (f)**: It is computed as the harmonic mean of precision and sensitivity.
- An **Receiver Operating Characteristic (ROC)** curve is plotted between sensitivity and 1-specificity values. The model’s effectiveness is measured by calculating the Area Under the ROC Curve (AUC).
VARIABLE IMPORTANCE

- The importance of independent variables is computed across each validation fold and the geometric mean of the variable importance is calculated.
- The method minimizes the importance of those variables which are unimportant across any fold.
- After analyzing the variable importance, the SLOC metric (100%), the LCOM metric (16.86%), the WMC metric (11.07%) and the RFC metric (0.63%) were computed as important variables.
## GEP PARAMETER SETTINGS

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Parameter Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population Size</td>
<td>50</td>
</tr>
<tr>
<td>Gene per chromosome</td>
<td>4</td>
</tr>
<tr>
<td>Gene head length</td>
<td>8</td>
</tr>
<tr>
<td>Simplification Generations</td>
<td>500</td>
</tr>
<tr>
<td>Linking Function</td>
<td>Addition</td>
</tr>
<tr>
<td>Fitness Function</td>
<td>Number of correct hits (Accuracy) with penalty</td>
</tr>
<tr>
<td>Maximum Generations</td>
<td>2000</td>
</tr>
<tr>
<td>Stop if fitness reaches</td>
<td>1.00</td>
</tr>
<tr>
<td>Mutation Rate</td>
<td>0.044</td>
</tr>
<tr>
<td>Transposition Rates (IS, RIS, Gene)</td>
<td>0.1 each</td>
</tr>
<tr>
<td>Recombination Rate (One-point, Two-point)</td>
<td>0.3 each</td>
</tr>
<tr>
<td>Gene Recombination Rate</td>
<td>0.1</td>
</tr>
</tbody>
</table>
The model for predicting CP classes was generated with 510 data points of the Simutrans data set.

We also calculated the univariate results of each individual metric using ten-fold cross validation.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Acc.</th>
<th>Sens.</th>
<th>Spec.</th>
<th>Prec.</th>
<th>f</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBO</td>
<td>70.00%</td>
<td>58.87%</td>
<td>79.21%</td>
<td>70.10%</td>
<td>64.00%</td>
<td>0.691</td>
</tr>
<tr>
<td>RFC</td>
<td>66.08%</td>
<td>51.95%</td>
<td>77.78%</td>
<td>65.93%</td>
<td>58.11%</td>
<td>0.650</td>
</tr>
<tr>
<td>SLOC</td>
<td>70.59%</td>
<td>62.34%</td>
<td>77.42%</td>
<td>69.57%</td>
<td>65.75%</td>
<td>0.700</td>
</tr>
<tr>
<td>NOC</td>
<td>56.67%</td>
<td>11.69%</td>
<td>93.91%</td>
<td>61.36%</td>
<td>19.64%</td>
<td>0.531</td>
</tr>
<tr>
<td>LCOM</td>
<td>54.51%</td>
<td>49.78%</td>
<td>58.42%</td>
<td>49.78%</td>
<td>49.78%</td>
<td>0.541</td>
</tr>
<tr>
<td>WMC</td>
<td>65.29%</td>
<td>46.75%</td>
<td>80.65%</td>
<td>66.67%</td>
<td>54.96%</td>
<td>0.632</td>
</tr>
<tr>
<td>DIT</td>
<td>57.45%</td>
<td>32.03%</td>
<td>78.49%</td>
<td>55.22%</td>
<td>40.55%</td>
<td>0.545</td>
</tr>
<tr>
<td>GEP Model</td>
<td>72.94%</td>
<td>72.73%</td>
<td>73.12%</td>
<td>69.14%</td>
<td>70.89%</td>
<td>0.732</td>
</tr>
</tbody>
</table>
Change Index (CI) - A new metric for ascertaining CP nature of a class.

Based on the results of the Simutrans data set using the GEP algorithm the CI metric was proposed as follows:

For a given class \( C_j \), the CI for the given class can be computed by the following equation:

\[
CI = (((((0.24 \times RFC) + (2 \times SLOC)) - WMC) - 59.62) - LCOM
\]

A class \( C_j \) is designated CP status according to the following function:

\[
ALTER(C_j) = \begin{cases} 
\text{yes, if } CI > 0 \\
\text{no, if } CI \leq 0 
\end{cases}
\]
We validate the CI metric on Glest data set which has 108 data points. The validation results are as follows:

- Accuracy: 72.22 %
- Sensitivity: 60.56 %
- Specificity: 94.00 %
- Precision: 95.55 %
- f measure: 74.59 %
- AUC: 0.776
APPLICATION OF CHANGE INDEX METRIC

- To identify CP classes in early phases of software development life cycle.
- Can be used during both maintenance and testing phases to distribute limited resources effectively.
- Can be used to form quality standards for producing good quality software as developers should ensure that proper design measures should be taken while coding CP classes so that they can be efficiently refactored in the forthcoming releases of the software.
CONCLUSIONS

- Important indicators of change: SLOC, LCOM, WMC and RFC.
- GEP algorithm can be efficiently used for developing models which predict software change. The results of the GEP model were better than univariate results of individual metrics.
- A metric called Change Index (CI) is proposed which can be used by software practitioners to identify CP classes. The CI metric can be used for efficient quality control and resource allocation.
Thank You