Framework for plagiarism detection in Java code

Anastas Misev
Institute of Informatics
Faculty of Natural Science and Mathematics
University Ss Cyril and Methodius
Skopje, Macedonia
anastas@ii.edu.mk

Agenda
- Introduction
- Basic idea
- Open framework
- Implementation
- Future work
- Questions and discussion
Introduction

- Increased number of assignments according to current trends (Bologna declaration, …)
- Increased number of students
  - 100% increase in our Institute in this academic year
- Accessibility of artifacts over the Internet
- Little or zero effort in plagiarism, especially in source code

A few words on plagiarism

- Simple plagiarism
  - Copy-paste (with some spacing and comments modification)
  - Plagiarism with renaming
    - Methods, fields, classes
  - Reordering of the code (that does not affect the final state)
  - Addition of redundant lines of code
A few words on plagiarism (2)

- Advanced plagiarism
  - Changing of the control structures
  - Mixing of several sources
  - Mixing of own and others’ code
- Drawing the line !!!!
  - It can be very hard
  - Objective vs. subjective

Detection methods

- Attribute counting
  - Used in the earliest tools
  - Counting operators and operands
- Structure metrics
  - Compare the structure
  - Usage of tokens
Available tools

- Sim
  - Using dynamic programming compare tokens from the source

- Yap
  - Using only specific tokens that reflect the structure
  - Longest common subsequence

Available tools (2)

- MOSS
  - Available as service to the teachers over the Internet
  - Important features include
    - Unsceptible to spaces and tabs
    - Noise suppression
    - Location independency

- SID
  - Simple system
Open framework

- An implementation done as diploma thesis by D. Aleksovski
- Java based, open framework
- Initial purpose: analyze Java code
- Allows easy extension
  - New analyzers
  - New comparators

The architecture

- Two basic elements
  - Analyzer
  - Comparator
- Analyzer – lexical and syntactical analysis of the code
  - Language specific
  - Produce the syntax tree and stores it into the database
  - Based on ANTRL
- Comparator – compare elements
  - Can be used to compare code, trees, fingerprints, …
The database

- **Comparisons**
  - If the database contains Fingerprint for file 1, go to 4
  - Call computeFingerprint (file1)
  - Store the fingerprint f1 into the database
  - If the database contains Fingerprint for file2, go to 7
  - Call computeFingerprint (file2)
  - Store the fingerprint f2 into the database
  - Forward the fingerprints to the comparator
  - Call computeSimilarity(f1, f2)
  - Store the values into the database

Operations

- **Comparing sources**

1. If the database contains Fingerprint for file 1, go to 4
2. Call computeFingerprint (file1)
3. Store the fingerprint f1 into the database
4. If the database contains Fingerprint for file2, go to 7
5. Call computeFingerprint (file2)
6. Store the fingerprint f2 into the database
7. Forward the fingerprints to the comparator
8. Call computeSimilarity(f1, f2)
9. Store the values into the database
Extensions

- Two different modules developed to test the framework
  - Simple module, basic features
    - Can only detect basic plagiarism
    - Compares the structure of the syntax tree
  - Advanced module
    - Produces a fingerprint of the syntax tree
    - Measures the longest common subsequence of the two fingerprints

Screen shots
Screen shots (2)

Initial results

<table>
<thead>
<tr>
<th>File Id</th>
<th>37</th>
<th>38</th>
<th>39</th>
<th>40</th>
<th>41</th>
<th>42</th>
<th>43</th>
<th>44</th>
<th>45</th>
<th>46</th>
<th>47</th>
<th>48</th>
<th>49</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.642254</td>
<td>0.647894</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.602347</td>
<td>0.608972</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.838028</td>
<td>0.635993</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.581221</td>
<td>0.587794</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.602347</td>
<td>0.608972</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.581221</td>
<td>0.587794</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.615493</td>
<td>0.615712</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.558075</td>
<td>0.572848</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.938949</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.938949</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Future work

- Support for additional languages
- New and advanced comparators and analyzers
- Web and web service interfaces
- Integration into
  - Moodle
  - Eclipse