Dissertation Project - Final View

<table>
<tr><th>ID</th><th>Subject</th></tr>
<tr><td>44</td><td>Bootstrap</td></tr>
<tr><td colspan="2">Initial studying to learn about different decompilation approaches carried out with neural networks</td></tr>
<tr><td>46</td><td>Writing tool for extracting bytecodes</td></tr>
<tr><td colspan="2">Creation of the tool to extract bytecode. In order to deliver this tool, the student needed to study the structure of .class files and related bytecode. Moreover, bytecode needed to be handled programmatically. To answer this need, the student needed to study, understand and manage two libraries: ASM and javassist. The latter was used to implement the tool as it offers a very powerful APIs to handle and analyse bytecode in .class files.</td></tr>
<tr><td>45</td><td>Extraction of Bytecode</td></tr>
<tr><td colspan="2">Creation of a tool able to extract bytecode from java methods in .class files. To collect a big enough number of .class files, Maven was used. Through the usage of this software, 144 open-source libraries (packed in JAR) were collected, both as compiled and source.</td></tr>
<tr><td>57</td><td>Create Schedule</td></tr>
<tr><td colspan="2"></td></tr>
<tr><td>47</td><td>Summarise the knowledge learnt during the bootstrap</td></tr>
<tr><td colspan="2">It is very important to summarise the knowledge learnt during the Bootstrap phase. This will pay off during the writing phase</td></tr>
<tr><td>48</td><td>Writing tool for extracting ASTs</td></tr>
<tr><td colspan="2">This tool is used to extract ASTs (Abstract syntax tree) from the source code: instead of using the source code as is, ASTs help to focus on the logic structure of the code. This is particularly important when the entire method is used as a sentence. In fact, methods have comments and instructions on multiple lines (alongside other issues) that can make the cleaning process of the source code quite a challenge.
To write this tool, Jonathan suggested to study the library javaparser to extract ASTs from the source code. Moreover, a dictionary must be created to organise the information extracted from the source code</td></tr>
<tr><td>49</td><td>Extraction of ASTs</td></tr>
<tr><td colspan="2">Tool to extract ASTs from source code.</td></tr>
</table>

<table>
<tr><td>56</td><td>First version of the NN decompiler</td></tr>
<tr><td colspan="2">Put together everything developed so far. The result should be a working solution able to translate bytecode sentences back to java source code.</td></tr>
<tr><td>55</td><td>Use OpenNMT to get results</td></tr>
<tr><td colspan="2">Once the extraction of bytecode and ASTs from java methods is completed, pairs should be processed through OpenNMT to create a NN model able to translate bytecode sentences back to java source code.</td></tr>
<tr><td>54</td><td>Performance metrics</td></tr>
<tr><td colspan="2">Choose metrics to evaluate performances of the tool. As the output of decompilation using neural network models won't be exactly equal to the ground truth, maybe a distance metric would work better than checking the score of the model</td></tr>
<tr><td>70</td><td>Work out stats</td></tr>
<tr><td colspan="2">Calculate some statistics about the two datasets (both bytecode and source code)</td></tr>
<tr><td>59</td><td>Summarise motivations for chosen metrics</td></tr>
<tr><td colspan="2">Describe the motivations for choosing particular performance metrics</td></tr>
<tr><td>60</td><td>Improvements</td></tr>
<tr><td colspan="2">Once the first version of the java NN decompiler is ready, improvements can start. Following the literature on NN decompilers, there are two important improvements that have proved to enhance the result of the decompilation:
1. Instead of considering methods as sentences, computer language knowledge can be introduced in the extraction of bytecode and ASTs. Doing so, methods are split in smaller blocks that. On the bytecode side, this could be done using the Control Flow Graph (CFG) analysis
2. In the decompilation process, the result coming out of the NN model can be post-processed to correct potential errors introduced by the model itself. This approach can be used to produce source code that is actually compilable. This idea is based on the latest result in this fascinating field.</td></tr>
<tr><td>64</td><td>Extract CFG Blocks</td></tr>
<tr><td colspan="2">Write a tool able to extract Control Flow Graph blocks and dominator trees. Javassist offers thorough analytic APIs to extract blocks from .class files. Moreover, line-number information can be extracted from blocks to create a connection between CFG blocks and AST sub-trees</td></tr>
<tr><td>65</td><td>Consider to use a sequence-to-sequence NN</td></tr>
<tr><td colspan="2">There is the possibility to use directly a sequence-to-sequence NN instead of OpenNMT.</td></tr>
</table>

<table>
<tr><td>67</td><td>Experiments</td></tr>
<tr><td colspan="2">Experiments must be run to collect results to show how the system performs</td></tr>
<tr><td>50</td><td>Ethics form</td></tr>
<tr><td colspan="2">Compile and Submit the ethics form</td></tr>
<tr><td>61</td><td>BETA version</td></tr>
<tr><td colspan="2">Deliver a working software that is able to decompile bytecode back to java source code.</td></tr>
<tr><td>62</td><td>Writing Interim Report</td></tr>
<tr><td colspan="2">Summarise what has been developed so far</td></tr>
<tr><td>51</td><td>Interim Report 7%</td></tr>
<tr><td colspan="2">The interim report is a short document (maximum 4 A4 pages plus title page and references) which provides a brief overview of your proposed project. It should contain:
• A short introduction to the project (this should set the scene for the project and motivate why the project is interesting)
• A clear statement of the high-level aim of the project and a list of the scientific objectives of the work
• A brief description of the progress so far made (i.e. areas researched, analysis undertaken, software or model development undertaken)
• A timed work plan for completion of the project.</td></tr>
<tr><td>58</td><td>Writing</td></tr>
<tr><td colspan="2">The dissertation takes the form of a scientific paper describing the research that you have undertaken, and its outcomes. You will be provided with author instructions (via <b>Blackboard</b>) that detail the specific requirements for formatting your dissertation. The following briefly provides some advice on writing your dissertation.
• An introduction to the problem addressed by your research project, including the motivation for the project and its wider significance.
• A clear statement of the aim and scientific objectives of your work.
• A discussion of background and related work relevant to your problem.
• A description of the research undertaken and its conclusions.
• An appraisal of the contribution that your project makes to the state of the art.
• An assessment of the scope and limits of your work and relevant future work.
Assessment:
• Introduction (10%)</td></tr>
</table>

- Background (10%)
- What was done, and how (40%)
- Results and Evaluation (20%)
- Conclusions (10%)
- References (5%)
- Form (5%)

63 Create Presentation

52 Presentations 8%

Your presentation is a 45 minute slot (30 minute presentation and 15 minutes for questions) in which you will give a public presentation on your project, including a description of the scientific background, the work undertaken, and the results.

53 Demonstration and Dissertation - 85%

The final dissertation takes the form of an extended scientific research paper (40 pages maximum, excluding appendices) of the style and quality produced for the main conferences in the field.

Format. You are free to choose your own format that best fits the style of your dissertation work. As a default, you may wish to format it in the manner of a conference or journal paper. If so, we recommend the style of Springer’s “Lecture Notes in Computer Science” (LNCS).

Length. The dissertation must not exceed 45 pages in length (excluding Appendix). This is around three times the size of a regular conference paper. The reason for the limitation is to ensure that you take time to write a well-crafted paper, and that your supervisor has time to read it and comment in detail.

Supplementary material may be made available on a web site.

Structure and Style. The detailed structure of the paper must be decided between you and the supervisor. Section 5 below gives more detail on the content and style of the final dissertation.

Submission and Binding. The dissertation must be submitted electronically via NESS in PDF. However, this deadline applies only to those who have no re-sits; others can have extended deadlines by presenting their case through PEC forms.

Demonstration. You are expected to give a 30-minute demonstration on your project to your supervisor and second examiner. It is up to you to arrange this meeting at a mutually convenient time but all demonstrations should have taken place by the dissertation submission deadline. The demonstration is a chance to describe in detail the work done in your project and to show any models or software that you developed. While no formal mark is given for the demonstration it will be used by the markers to help inform them when marking your dissertation.

68 Read new papers

Read new papers about NN applied to translation problems. Try to understand if there are alternatives to the sequence-to-sequence approach and if other techniques can be applied to improve the NN decompiler

69	Consider to migrate the solution to AWS Server
Remind Jonathan to work out a budget to rent an AWS Server. Experiments should be run on a powerful machine that can give results sonner. In this case, migrate the solution online	