FACULTY OF INFORMATION TECHNOLOGY CTU IN PRAGUE

Review report of a final thesis

Student:	Bc. Aleš Saska
Reviewer:	Ing. Konrad Siek, Ph.D.
Thesis title:	GNU-R Debugger Bytecode Support
Branch of the study:	System Programming

Date: 28. 12. 2018

Evaluation criterion:	*The evaluation scale: 1 to 4.*
1. Fulfilment of the assignment	***1 = assignment fulfilled,*** *2 = assignment fulfilled with minor objections,* *3 = assignment fulfilled with major objections,* *4 = assignment not fulfilled*

Criteria description:
Assess whether the submitted FT defines the objectives sufficiently and in line with the assignment; whether the objectives are formulated correctly and fulfilled sufficiently. In the comment, specify the points of the assignment that have not been met, assess the severity, impact, and, if appropriate, also the cause of the deficiencies. If the assignment differs substantially from the standards for the FT or if the student has developed the FT beyond the assignment, describe the way it got reflected on the quality of the assignment's fulfilment and the way it affected your final evaluation.

Comments:

The thesis formulates the objectives as designing and implementing a system for debugging GNU R's bytecode interpreter and specifically presents two sub-goals: (a) creating a bytecode disassembler and (b) creating a debugger within the GNU R bytecode interpreter. This accords with my understanding of the goals of the assignment.

The thesis motivates the goals by explaining that the current GNU R interpreter's debugger can only debug R's ASTs, but not its bytecode. When the debugger has to deal with bytecode, it resolves the bytecode into the original source code and uses the AST debugger on that. The author explains that this makes the debugger imprecise and unergonomic.

Evaluation criterion:	*The evaluation scale: 0 to 100 points (grade A to F).*
2. Main written part	*50 (E)*

Criteria description:
Evaluate whether the extent of the FT is adequate to its content and scope: are all the parts of the FT contentful and necessary? Next, consider whether the submitted FT is actually correct – are there factual errors or inaccuracies? Evaluate the logical structure of the FT, the thematic flow between chapters and whether the text is comprehensible to the reader. Assess whether the formal notations in the FT are used correctly. Assess the typographic and language aspects of the FT, follow the Dean's Directive No. 26/2017, Art. 3. Evaluate whether the relevant sources are properly used, quoted and cited. Verify that all quotes are properly distinguished from the results achieved in the FT, thus, that the citation ethics has not been violated and that the citations are complete and in accordance with citation practices and standards. Finally, evaluate whether the software and other copyrighted works have been used in accordance with their license terms.

Comments:

The written text of the thesis leaves a lot to be desired. It reads like a first draft. I will enumerate the various types of problems with the text below. I will limit myself to giving just one example of each.

Parts of the text are unnecessary and tedious. Specifically, we have several sections devoted to explaining commandline arguments of unrelated commands. For instance Chapter 1 begins by explaining the commands that can be used to install, build, and check packages in R, a complete non-sequitur. To add insult to injury, some of these unnecessary sections seem to be copied verbatim from manuals and source code.

Important information is missing. For instance, when the author describes a choice he had to make between two pretty print functions for R source code, dput and deparse1, he gives us some information about how dput works, but only gives us the path of the file where deparse1 is defined, and then informs us that deparse1.c was selected with very little information about why.

Small factual inaccuracies. For instance, when listing types of R's internal representations of objects, the author lists ANYSXP and FUNSXP. However objects of types ANYSXP and FUNSXP do not actually exist, but rather these are placeholder values used in conditions when referring to heterogeneous groups of other types of objects.

Typographical errors. For instance, I found the word "condiguration" on page 35.

Small typesetting errors. For example, there are plenty of references in the second half of the book that are rendered as double question marks.

Large typesetting problems. There is a source listing on page 26 that does not fit on the page and therefore run off from the page , leaving parts of the listing missing. There is an almost empty page on page 24 for no good reason.

Imprecise language. The language in a master thesis in a STEM field should be exact and technical. The author occasionally describes concepts in a very imprecise manner. For instance, on page 6 the author explains that "[SEXP] types hold values which can be printed (...)" My contention is that SEXP types are integers and therefore hold nothing. SEXP types can only be decoded. What the author likely meant was that SEXP types inform us about the contents of the objects they describe and these contents can be examined and printed.

Presentation of benchmark results. The author presents performance testing results at the end of Chapter 3. The methodology is almost completely unexplained. We do not know much about the benchmarks themselves, as we are only given the names of files and it is hard to discern what lapply.R or vectors.R, say, do exactly. The data does not show the distribution of the measurements, which either means that the figure shows only averages, or that there is only one data point per benchmark. The axes are not marked, so it is up to the reader to divine their meaning. The experiment compares execution with the bytecode debugger turned on and without it, but the reader is not given information about how the code was compiled (or indeed, that it was compiled at all). Was it precompiled or JIT compiled? If the latter, what were the settings of the JIT?

The author also sometimes refers to published works which refer to what he is describing only tangentially. For instance, on page 12 the author talks about laziness in R and provides a blog post about performance written by Hadley Wickham as a reference. A much better document would be the 1996 paper by Ross Ihaka and Robert Gentleman introducing the R language.

All in all, these problems make it very difficult to extract any insight into the design and implementation of the debugger framework and the accompanying bytecode deparser.

Evaluation criterion: *The evaluation scale: 0 to 100 points (grade A to F).*

3. Non-written part, attachments

100 (A)

Criteria description:
Depending on the nature of the FT, comment on the non-written part of the thesis. For example: SW work – the overall quality of the program. Is the technology used (from the development to deployment) suitable and adequate? HW – functional sample. Evaluate the technology and tools used. Research and experimental work – repeatability of the experiment.

Comments:

There are two significant artifacts submitted with the thesis. One is the repository containing the bytecode disassembler including the source code, installation instructions, and user documentation. The other is the repository containing GNU R patched with a bytecode debugger, including the source code, installation instructions, and user documentation, as well as the text of the thesis.

I installed the byte code disassembler in a fresh install of GNU R using the instructions provided in the repository. I used the attached vignette to familiarize myself with the API. I then proceeded to test the byte-code compiler using custom byte-compiled pieces of R code. I found the output of the disassembler to be correct and formatted in a very human-readable way. I inspected the R source code, and found it to be well written, well documented with comments, and generally clean. The design of the disassembler is straightforward and does the job right. The only drawback I see to the implementation of the disassembler is that, since it is implemented in R, it is difficult to use it to disassemble byte code from the level of GDB, so it is less useful for interpreter hacking. While I make the comment, I do not think this should count against the work, since debugging R code using GDB is a niche activity.

I built the patched version of R with the bytecode debugger included. I followed the instructions attached to the repository. I repeated this on three machines and on one of them, the compilation process yielded an error that proved inscrutable to me. I managed to build the project successfully on the other two machines. I followed the vignette to familiarize myself with the user interface, which works just like the R's AST debugger. Making the new debugger fit into the existing framework is a good decision. I inspected the source code of the debugger (in conjunction with reading the appropriate sections of the thesis). The design of the debugger follows the design of V8, where the debugger inserts new opcodes into the bytecode whose semantics allow them to stop the execution of the byte code in specific places. I found the design to be well reasoned and efficient. I also found the source code clean and well written.

Evaluation criterion: *The evaluation scale: 0 to 100 points (grade A to F).*

4. Evaluation of results, publication outputs and awards

100 (A)

Criteria description:
Depending on the nature of the thesis, estimate whether the thesis results could be deployed in practice; alternatively, evaluate whether the results of the FT extend the already published/known results or whether they bring in completely new findings.

Comments:

The work certainly has a solid real-world application and was clearly designed to be integrated into GNU R to work alongside existing systems. The implementation of the debugger is definitely something that the author can submit for inclusion in the GNU R interpreter and has a decent chance of being accepted. The implementation of the disassembler meets the CRAN standards for packages, so it can be uploaded to that package repository and it can be used as-is by R developers.

Evaluation criterion: *No evaluation scale.*

5. Questions for the defence

Criteria description:
Formulate questions that the student should answer during the Presentation and defence of the FT in front of the SFE Committee (use a bullet list).

Questions:

1. Given the state of the thesis and the significance of experiments in CS, I would like to ask the student to explain the entirety of Chapter 3 ("Testing") with a particular emphasis on section 3.2 on performance. Specifically I'd like him to explain the point of the experiment, the exact methodology, the evaluation environment, the specifics of the benchmarks, to present the data gathered, and to draw the conclusions.

2. How would the student recommend going about using the bytecode disassembler from a debugger like GDB? How difficult would it be to extend the disassembler to make this easier?

Evaluation criterion:	*The evaluation scale: 0 to 100 points (grade A to F).*
6. The overall evaluation	*80 (B)*

Criteria description:
Summarize which of the aspects of the FT affected your grading process the most. The overall grade does not need to be an arithmetic mean (or other value) calculated from the evaluation in the previous criteria. Generally, a well-fulfilled assignment is assessed by grade A.

Comments:

The work presented in the thesis and implemented in the accompanying systems solves a practical problem and does it well, to the point where I can see it being used in the future by R developers. The only problems with the implementations are minor. I think this should be reflected in the final grade.

In the light of this good implementation, it is a shame that the written part of the thesis is so badly constructed as to be almost completely useless. Written communication of technical ideas is something that should be required from an M.Sc., thus I am of the opinion that these shortcomings should also have a significant impact on the final grade.

Signature of the reviewer: