Contribution ID: **12** Type: **Talk (15min + 5min)**

flowR: A Program Slicer for the R Programming Language

Wednesday, March 6, 2024 4:00 PM (20 minutes)

Context:
Program slicing [1] is an important technique to assist program comprehension. A program slicer identifies the parts of a program that are relevant to a given variable reference (i.e., all statements that can influence the resulting value). The resulting slice can then help R programmers and researchers to understand the program by reducing the amount of code to be considered. Furthermore, the slice can help speed up subsequent analyses, by reducing the amount of code to be analyzed.

Objective:
We present flowR, a novel program slicer and dataflow analyzer for the R programming language.
Given R code and a variable of interest, flowR can return the resulting slice as a subset of the program or highlight the relevant parts directly in the input.
Currently, flowR provides a read-eval-print loop, a server connection, and a rudimentary integration into the R language extension for Visual Studio Code which allows to interactively generate and investigate slices. flowR is available as a docker image.

We focus on the R programming language because even though it has a huge, active community, especially in the area of statistical analysis (ranking 7th on the PYPL and 17th on the TIOBE index), the set of existing tools to support R users is relatively small. To our knowledge, there is no preexisting program slicer for R.
Besides the RStudio IDE and the R language server, the {lintr} package and the {CodeDepends} package perform static analysis on R code.
However, all these tools rely on simple heuristics like XPath expressions on the abstract syntax tree (AST), causing their results to be imprecise, limited, and sometimes wrong.
Therefore, we consider flowR's dataflow analysis to be a valuable contribution to these tools by improving their accuracy.

Method:
flowR uses a five-step pipeline architecture, starting with the parser of the R interpreter to build an AST of the program. After normalizing the AST, the dataflow extraction works as a stateful fold over the AST, incrementally constructing the graph of each subtree. The calculation of the program slice reduces to a reachability traversal of the dataflow graph which contains the uses and definitions of all variables. Finally, the slice is either reconstructed as R code or highlighted in the input.

Limitations:
Currently, flowR neither handles R's reflective capabilities (e.g., modifying arguments, bodies, and environments at run-time), nor its run-time evaluation with eval. Moreover, flowR does not perform pointer analysis, which causes vectors and other objects to be treated as atomic (i.e., flowR does not differentiate access to individual elements).

Results:
Using real-world R code (written by social scientists and R package authors) shows that flowR can calculate the dataflow graph (and the respective slice for a given variable reference) in around 500ms. Moreover, averaging over every possible slice in the dataset, we achieve an average line reduction of 86.17%[±9%] (i.e., when slicing for a variable, the relevant program parts are only around 14% of the original lines).

[1] Weiser. Program Slices. Michigan, 1979.

Slot length

Primary author: SIHLER, Florian (Ulm University)

Co-author: Prof. TICHY, Matthias (Ulm University)

Presenter: SIHLER, Florian (Ulm University)

Session Classification: RSE Practices

Track Classification: Research Software: Research Software