MASTER THESIS
READER'S REVIEW

Title: *Verification of Memory Safety with Predator and Symbiotic*

Author: *Tomáš Jašek*

Supervisor: *Jan Strejček*

Reader: *Vladimír Štill*

The goal of this work was to improve the performance of the Symbiotic tool with integrated Predator in the memory safety analysis. This was to be achieved by means of improving the performance of Predator when analysing LLVM intermediate representation. This goal of the thesis was met.

The thesis is written in English and has a good language quality and adequate length for a master's thesis. The typesetting of text is decent, but the figures are sometimes crude, with graph labels overlying one another or the bounding box. I also find some of the figures a bit hard to read as their description is not in their labels but in the main text.

The text structure is good, but I would prefer a longer and more comprehensive introduction and more details overall. The thesis expects the reader to have quite a lot of domain-specific knowledge – for example, it does not explain what LLVM and GIMPLE are or even why it Symbiotic needs to use Predator working with LLVM (which is not obvious). Furthermore, I find it peculiar that the thesis does not directly state if the integration of Predator into Symbiotic was done by the author and if it was implemented as a part of the thesis or not (consulting the thesis assignment, it becomes clear the integration was done by the author before the thesis and only its description is considered part of the thesis). On top of that, I would appreciate if there was a clear side-by-side comparison of the performance of Predator and Symbiotic before and after the modifications, so the reader does not need to cross-reference graphs from chapter 5 with graphs from chapter 2. I would also appreciate if the performance of optimisations evaluated in chapter 5.3 was presented in the form of a table which would facilitate comparison of different optimisations.

The main body of the work was to identify, fix and evaluate issues that cause Predator that analyses LLVM intermediate representation of a program to perform worse than Predator using GIMPLE intermediate representation of the same program. The issues and solutions are presented in chapter 3 and the evaluation is in chapter 5. In this area, the author identified seven problems, six of which he resolved himself and the last one was resolved by the authors of Predator based on bug reports by the thesis author. According to the evaluation, these improvements bring the LLVM-based Predator on par with the GIMPLE-based version and improve the performance of GIMPLE-based Predator as well, which is quite a good result. However, I find some of the solutions to take a suboptimal approach to the fixes.

- In "3.2 Function Pointer Casting" the author introduces a modification of the translation from LLVM to Predator's intermediate representation that ignores some cases of function pointer casts (if one of the types is a variadic version of the other). However, I believe this could introduce some imprecision and, considering the analyser should have access to the whole code of the analysed program it should be possible to detect cases when these casts are not actually needed.

- In “3.5 Redefinition of Block Label”, the author side-steps a problem in the translation of LLVM into Predator’s intermediate representation by running an LLVM preprocessing step. It would be probably better to fix the translation itself.
- In “3.7 Undefined Values in Union Initializers”, there appears to be a bug in the `global-vars` transformation – the initialisers should not be inserted at the beginning of the `main` function, but before the C global initialisers, which run before the `main` function.
- In “3.8 Liveness Analysis With Pointers” I find it peculiar that optimisation in Predator introduces spurious memory leak reports and the author have chosen to work around this problem instead of fixing it. I also miss the description of the cause of this problem. In particular, I do not see why the variable `p` is not live inside the `if` in Figure 3.5. Finally, even if it is not possible or practical to fix the bug in the live variable analysis, it should be quite easy to check if a basic block dominates a call to `abort` instead of merely checking that the `abort` is present in the next basic block.

While chapter 3 lacks some details, chapter 4 “Integration of Predator into Symbiotic” contains details that are, in my option, not suited for the text of the thesis. In particular, section 4.4 resembles more an English-language rewrite of a code than a high-level description of a (rather simple) problem.

The evaluation (chapter 5) provides data about the performance of the modified Predator and Symbiotic with Predator integration. While it provides sufficient data about the performance on safe programs, I would appreciate more details about the performance on unsafe programs too. Even more important, I miss information whether or not all decided benchmarks were decided correctly. Furthermore, I would appreciate it if the evaluation contained some analysis of the reasons for the remaining differences between GIMPLE- and LLVM-based Predator.

Implementation The implementation was done in the context of two existing projects, Predator and Symbiotic and was integrated into their code bases. Especially the integration into Predator is a good indication of the quality of the proposed changes as it required collaboration with developers external to the research group in which the author and his advisor are.

On the other hand, evaluating the actual amount of work and quality of the new code is complicated by missing information in the thesis – the author does not describe in any way the submitted archive and the changes are scattered among multiple repositories, some of which are not described in the thesis. From the changes I have tracked in the Predator and ProStatA repositories (the second of which is not mentioned by the work at all) it appears that the actual amount of changes was rather small. This does not necessarily mean the overall amount of work was small as the author had to also identify the problems and familiarise himself with the new projects. However, considering this apparent small amount of work, I would expect a more detailed description of the work done in the text of the thesis.

Questions for Presentation

1. Are there any situations where the omission of pointer casts introduced in 3.2 could cause wrong results of the verification (e.g. for variadic functions like `printf` or for argument-taking functions forward-declared without arguments)?

2. Why have you decided to transform the LLVM IR to side-steps problems described in sections 3.5 and 3.8 instead of fixing the underlying problem in the translation of LLVM into Predator's internal representation?

3. Are there any wrong answers for the evaluated benchmarks in some of the evaluated versions? If so, how did they change with the improvements introduced in the thesis?

Conclusion

Overall, the thesis has met the requirements stated in the thesis assignment and presents an important improvement to Predator and Symbiotic. Therefore, I recommend accepting this work. However, considering the shortcoming described in this report, I suggest awarding it with mark **C**.

Brno 30th January 2021 Vladimír Štill