Static Analysis Toolset with Clang

Bence Babati, Gábor Horváth, Viktor Májer, Norbert Pataki

Dept. of Programming Languages and Compilers,
Fac. of Informatics, Eötvös Loránd University, Budapest
babati@caesar.elte.hu, xazax.hun@gmail.com, viktor.majer1@gmail.com, patakino@elte.hu

Abstract

Static analysis is an emerging method to analyse programs without execute them only based on their source code. Static analysis can be used for various tasks, e.g. searching bugs. Clang is an open source compiler for C/C++/Objective-C. It is built on the top of the LLVM compiler infrastructure. It is a rapidly evolving project which is supported by Apple, Google, and Microsoft. Nowadays Clang is a popular tool to develop new static analyzers. One of the advantages of Clang is its modular architecture. One can use the parser as a library to build tools. The Static Analyzer is the part of the Clang compiler. It is utilizing the same AST that is generated by the compiler.

In this poster we present our tools that are based on Clang. The tools cover wide spectrum of static analysis: detecting portability issues in include dependencies of C++ Standard Template Library, refactoring for parallelized constructors and code comprehension. The C++ Standard does not define which standard header files include an other standard header file. It is easy to take advantage of an implemention-specific dependency because the code compiles. It can result in portability issue when an other STL implementation works in different way. One of our tool detects these problems. A tool of ours is able to make copying operations concurrent in specific cases. The aim of our code comprehension tool is finding specific C++ code snippets based on queries.

Keywords: C++, static analysis, Clang

MSC: 68N15 Programming languages