4.Overview:

Our main goal is to design and implement a tool that can produce new intermediate representation or different part compared to our original intermediate representation under the condition source program has been modified partly. In this process, changed source will be focused on and there is no need for users to rebuild their program totally after using the automated tool.

Getting the different part intermediate representation of two source code is a main object of our research. Of course, we need to develop a proper and efficient process for inputing  source programs and outputting the correct result. Then, an automated way will improve tool’s efficiency.

During  research, we modify clang, which is a front-end for the llvm compiler. To achieve this goal, we have following steps:

1.Add flags to AST node and change behavior of clang, which should ensure clang translation correct.

2.Use ChangeDisitiller, which is a tree-differencing tool for fine-grained source code change.

3.Through specific  algorithm, new intermediate representation will be generated.

4.1 Research Method

According to the description above, we have developed a tool which generate changed intermediate representation based on changed code. Figure 1 helps us illustrate the purpose of our tool, once source code has been changed, almost everyone will rebuild changed source totally. However, rebuilding source will not make sense for some researches who is dedicated to intermediate representation and desire to know the changed part of intermediate representation and it is also not very efficient.

Now for achieving this object, sections below briefly describe process of mentioned tool.

4.2 Add flags

Clang creates a new C, C++, Objective C and Objective C++ front-end for llvm compiler. It is a open source project and hence convenient for us to read and change behavior of clang.

In the our research, we propose a method adding flags in the clang compiler, which will change clang AST node data structure with attaching a integer variable. The integer variable is on behave of every statement position.Due to ChangeDistiller a tool can analyze source code at the statement level and is going to be introduced in the next section,  our tool also focus on the statement level.

Source Code

New IR

Edit Script

Changed code

Intermediate Representation

Build

Diff

Merge

After adding flags, every instruction in generated IR  will be mapped to source code at the statement level since the integer variable will be marked as a flag in the IR.