

Live Variable Analysis UI

Raadhes Chandaluru
CS22B069

IIT Madras

02/11/2025

Creating Educational UI for CFG and Live Variable Analysis

- Frontend is in Python Dash which is an interactive web application framework
- C Code is input by user in frontend
 - Python uses clang on the backend to lower C code to LLVM IR
 - This is passed through our live variable LLVM Pass
- LLVM Pass to generate information per iteration
 - Changes in information populated into graphviz dot file format
- Python graphviz renders .dot files
- Provided a slot to give clang flags in case higher optimization levels want to be seen in CFG (Or other flags)

AI assistance in UI code

Challenges & Final Setup

LLVM C Code CFG & Live Variable Per Iteration

Show / Hide Instructions

Enter C Code:

```
#include <stdio.h>
int main() {
    int x = 3;
    if(x > 0){
        x = x + 1;
    }
    return x;
}
```

Clang Compile Flags:

-O0

Generate CFG

{ Prev } { Next }

```
graph TD
    B1["104213791800240  
Live:  
%1 = allocate i32, align 4  
%2 = allocate i32, align 4  
store i32 0, ptr %1, align 4  
store i32 3, ptr %2, align 4, !dbg !16  
%3 = load i32, ptr %2, align 4, !dbg !17  
%4 = icmp sgt i32 %3, 0, !dbg !19  
br i1 %4, label %5, label %8, !dbg !19"]
    B2["104213791800592  
Live:  
%6 = load i32, ptr %2, align 4, !dbg !20  
%7 = add nsw i32 %6, 1, !dbg !22  
store i32 %7, ptr %2, align 4, !dbg !23  
br label %8, !dbg !24"]
    B3["104213791818416  
Live:  
%9 = load i32, ptr %2, align 4, !dbg !25  
ret i32 %9, !dbg !26"]
    B1 --> B2
    B1 --> B3
```

- Changes such as DbgVariableRecord not present in LLVM-14, but part of LLVM-20. Upgraded to LLVM-20 by installing from llvm.sh
- Generation of CFG manually slightly painful but not too hard