Coms 513 Final Project

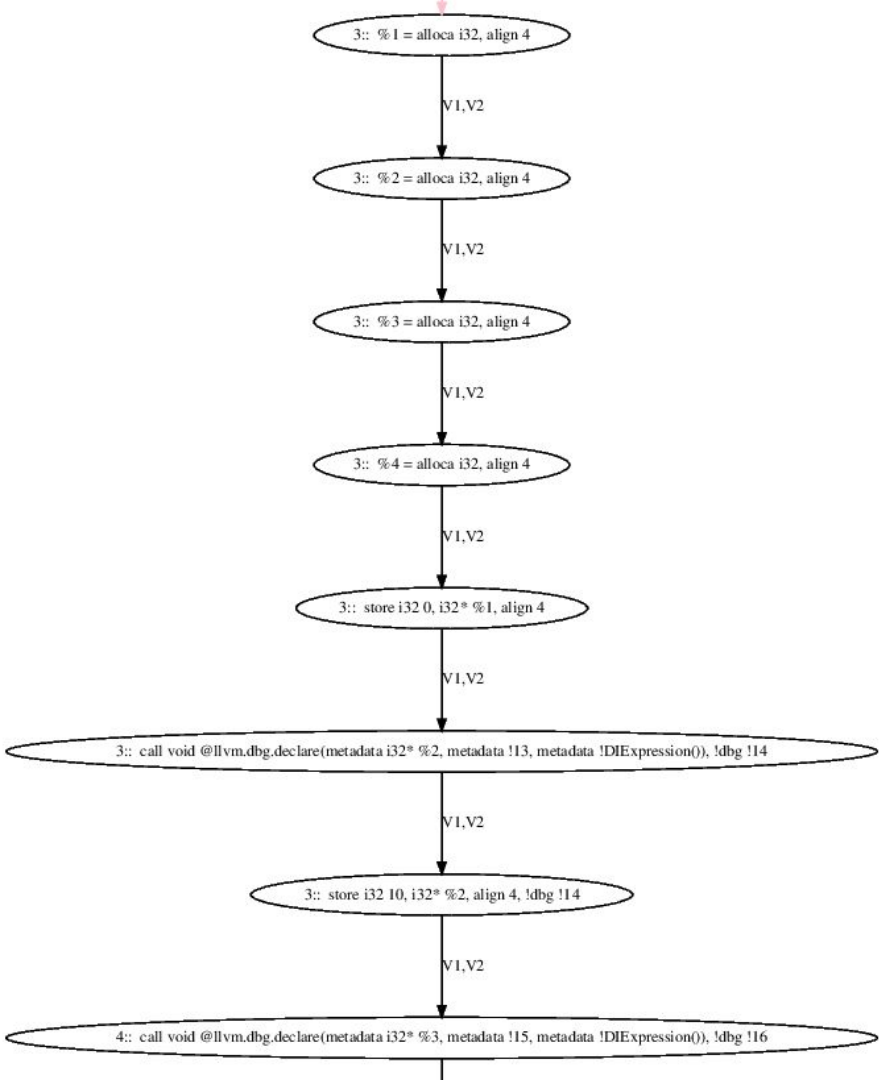
Optimization of MVICFG generated from Hydrogen

Xueyuan Chen
Jieyun Hu

Motivation

The MVICFG is complicated

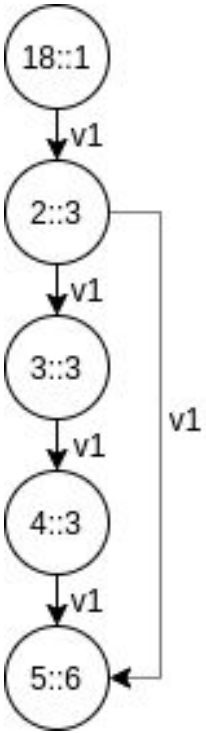
- The nodes are represented in LLVM instructions, which are hard to read.
- A simple line of code is split to many nodes which make the graph very big.
- There is no functions in Hydrogen or API which can convert instructions back to source code.

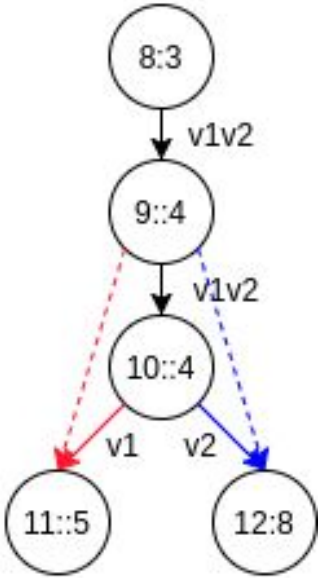


All nodes with label = “3” came from a single line
int a = 10;

Can we merge them to a single node?

Proposal





Workflow

Parse the file and save the nodes and edges

Use the adjacency matrix to represent graph.

Encode edges of different versions into weights

Merge the nodes with the same label (come from the same line)

Redirect the edges

Replace the instructions with the source code of the corresponding lines.

Optimize the program and make it work on the real world scenarios.

Test on simple test cases and real-word test cases.

Construct adjacency matrix

Adjacency matrix

- i -> j , only existed in v1, then matrix[i][j] = 1
- i -> j , only existed in v2, then matrix[i][j] = 2
- i -> j , existed in both v1 and v2, then matrix[i][j] = 3
- If there is no edge between n and m, then matrix[n][m] = 0

Reconstruct graph

Reconstruct graph

8 -> 2 v1v2
2 -> 3 v1v2
3 -> 4 v1v2
4 -> 5 v1 v2
5 -> 6 v1
5 -> 7 v2

Suppose 2, 3 ,4 has the same label. The graph can be simplified as
8 -> 2 v1v2
2 -> 6 v1
2 -> 7 v2

	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	1	2	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0
8	0	3	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0

Optimization

For real world cases, the generated MVICFG always have multiple subgraphs, each subgraph represents a function.

- Find the subgraph which contains differences
- Get the No of subgraph from dot file
- Simply this subgraph

Results

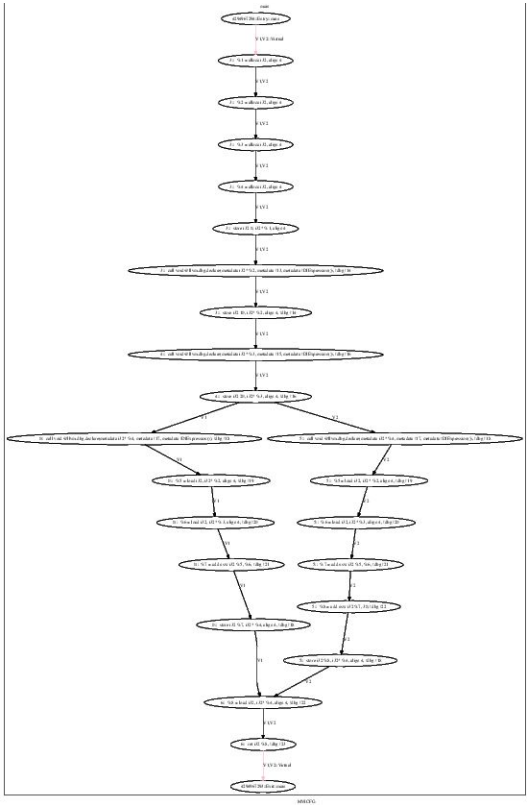
sum1.c

```
#include <stdio.h>

int main () {

   int a = 10;

   int b = 20;

   int sum = a + b;

   return sum;

}
```

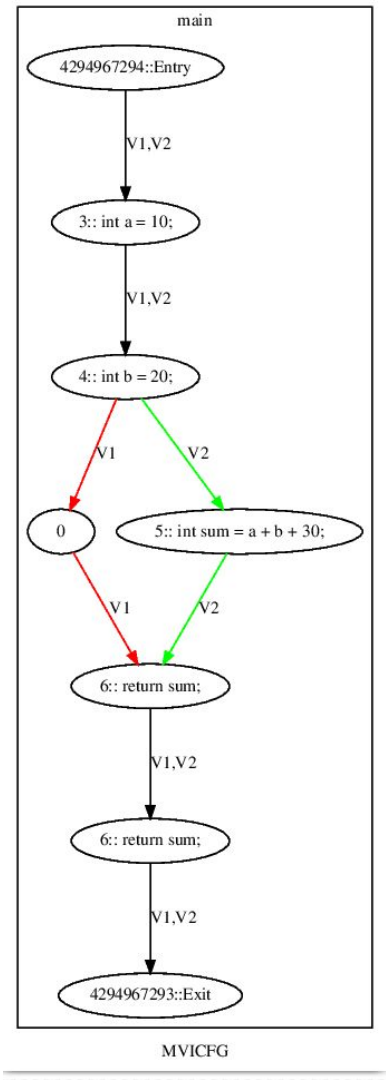
sum2.c

```
#include <stdio.h>

int main () {

   int a = 10;

   int b = 20;

   int sum = a + b +30;

   return sum;

}
```

Result

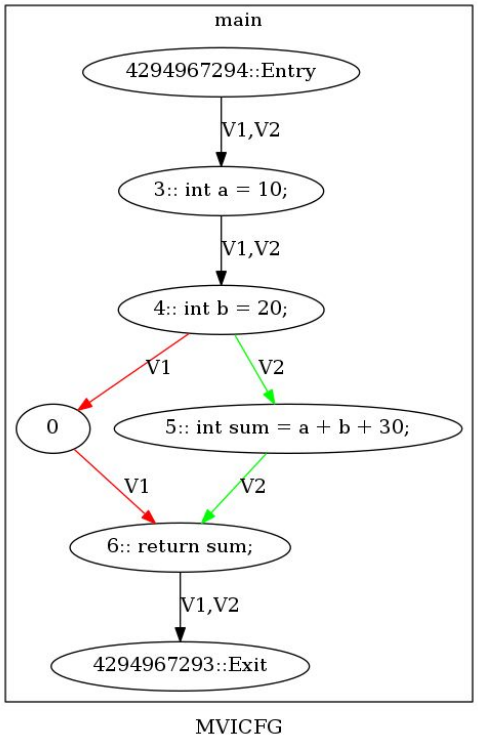


24 nodes



8 nodes

Result

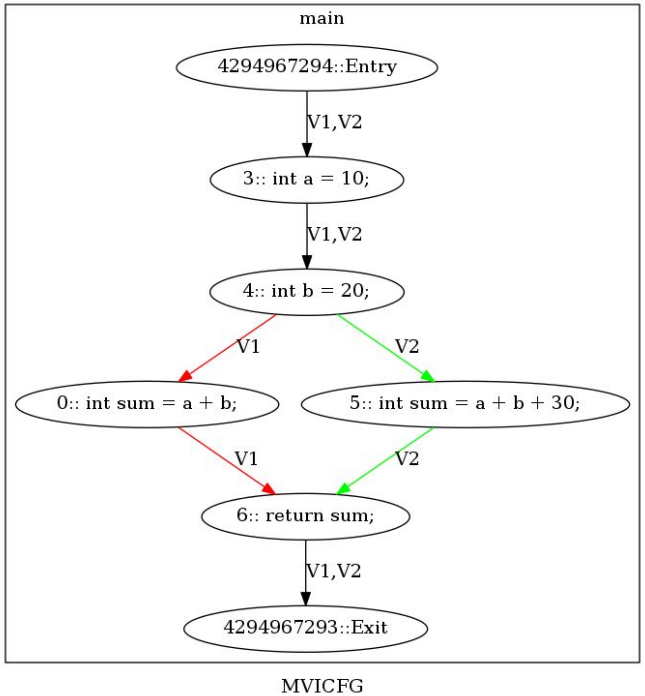


MVICFG

7 nodes

Remove repeated nodes

Try to add v1 codes into nodes.
But if number of 0 nodes is smaller than the number of removed lines, it won't add v1 codes into nodes.



MVICFG

Results (test4)

v1/get_sign.c

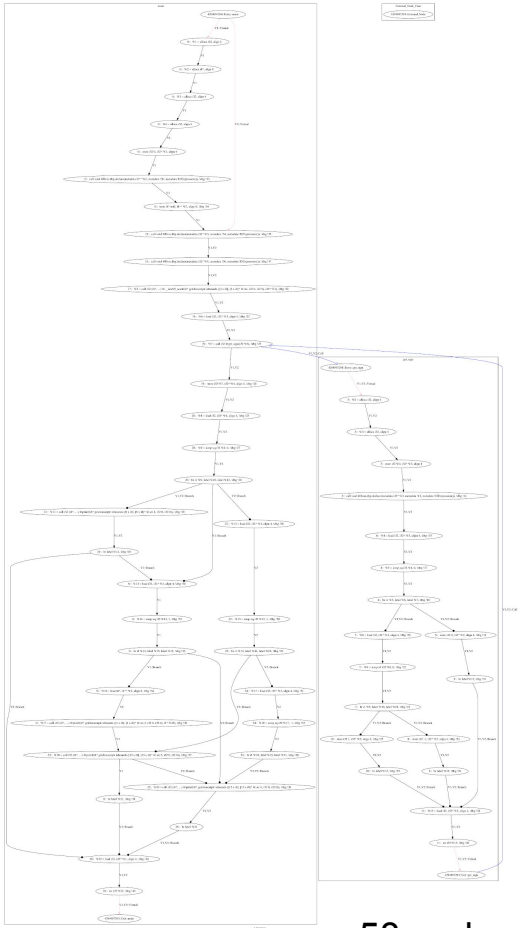
```
#include <stdio.h>

int get_sign(int x) {
  if (x == 0)
    return 0;

  if (x < 0)
    return -1;
  else
    return 1;
}
```

```
int main() {

  char* crash =0;
  int x, a;

  scanf("%d", &x);

  a = get_sign(x);
  if(a == 0)
    printf("Is zero\n");
  if(a == 1){
    printf("%s\n",crash);
    printf("Is positive\n");
  }
  else
    printf("Is negative\n");
}
```

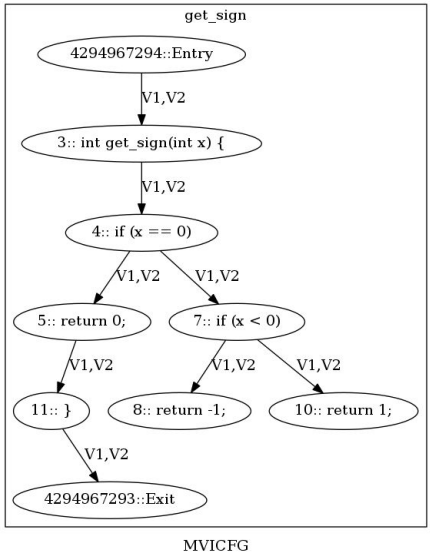
v2/get_sign.c

```
#include <stdio.h>

int get_sign(int x) {
  if (x == 0)
    return 0;

  if (x < 0)
    return -1;
  else
    return 1;
}
```

```
int main() {

  int x, a;

  scanf("%d", &x);

  a = get_sign(x);
  if(a == 0)
    printf("Is zero\n");
  else if(a == 1)
    printf("Is positive\n");
  else if(a == -1)
    printf("Is negative\n");
}
```



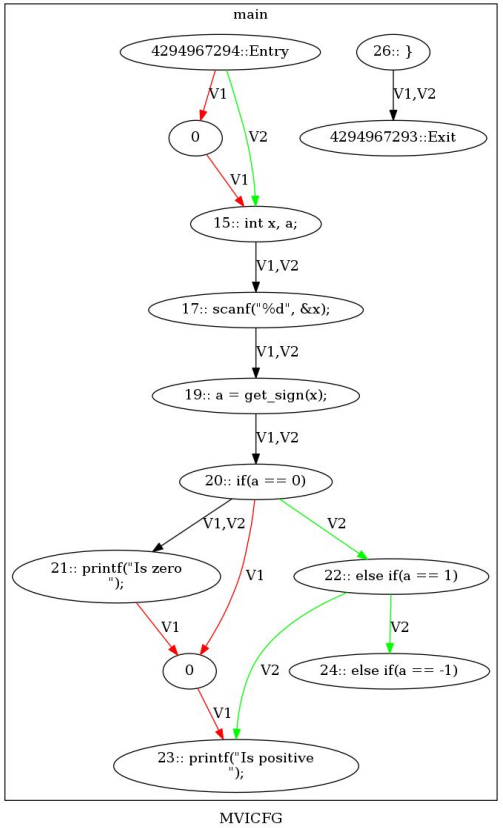
58 nodes



MVICFG

get_sign(): 9 nodes

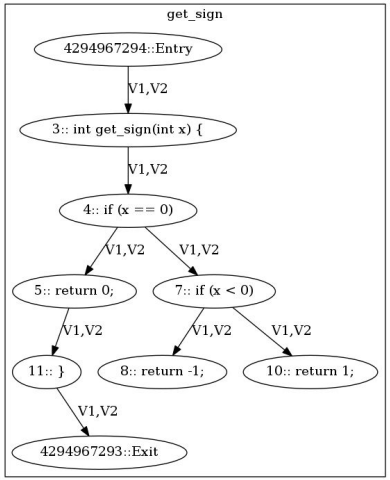
Get seperate subgraphs

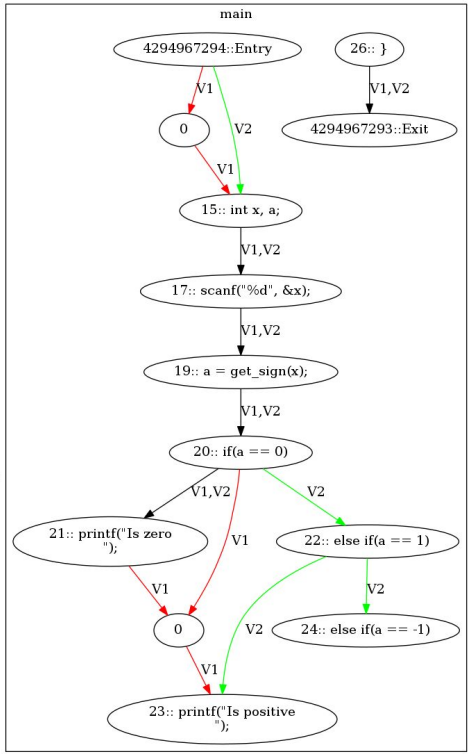


MVICFG

main(): 13 nodes

Results

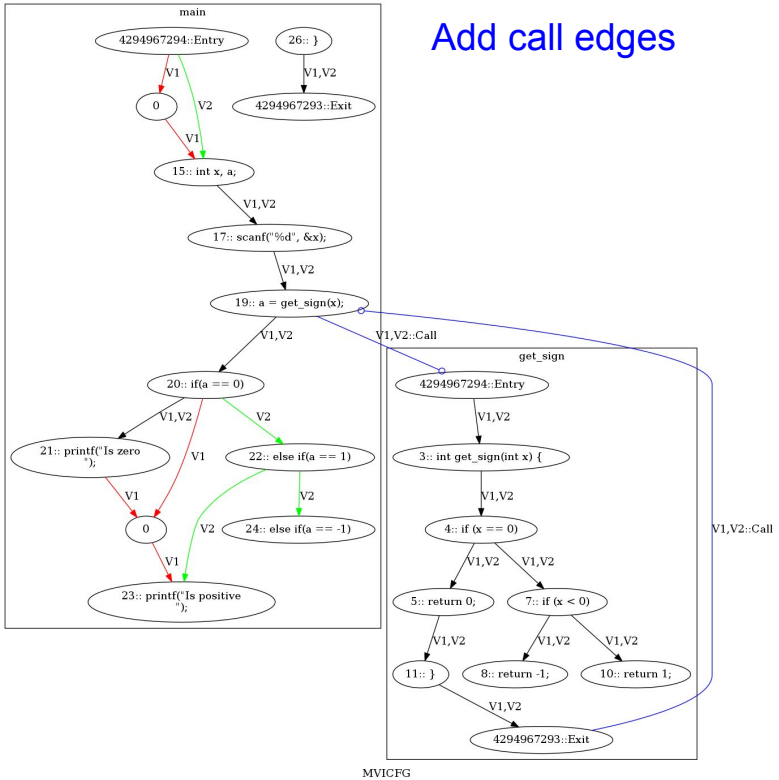




MVICFG

Combined subgraphs

Add call edges



MVICFG

Summary table

test cases	Code size	Size of code churn	No. of nodes in MVICFG	No. of nodes in simplified MVICFG
test1	18	2	28	9
test2	14	2	24	11
test3	18	2	18	9
test4	25	7	58	22
test5	11	4	74	11
test6	15	9	51	14

Future works

- Split “0” nodes and add code lines in the nodes
- Automated generate subcluster graphs
- Automated combine subcluster graphs
- Improving traverse

Thanks!