Assignment-4

Yulei Sui

University of Technology Sydney, Australia

Assignment 4: Quiz + A Coding Task

- One quiz (10 points)
  - Static symbolic execution
  - Automatic translation from code to Z3 formulas/constraints

Assignment 4: Quiz + A Coding Task

- One quiz (10 points)
  - Static symbolic execution
  - Automatic translation from code to Z3 formulas/constraints
- One coding task (15 points)
  - **Goal:** automatically perform assertion-based verification for code using static symbolic execution.
  - **Specification and code template:** https://github.com/SVF-tools/Teaching-Software-Verification/wiki/Assignment-4
  - **SVF Z3 APIs:** https://github.com/SVF-tools/Teaching-Software-Verification/wiki/Z3-API

You are encouraged to finish the quizzes before starting your coding task.

Methods to Be Implemented

You need to implement the following four functions in `Assignment-4.cpp`:

- `SSE::handleNonBranch`
- `SSE::handleCall`
- `SSE::handleRet`
- `SSE::handleBranch`
- The required implementation parts are indicated with TODO comments and you only need to fill up the code template if a method is partially implemented.

In the following slides, we provide several examples to assist your understanding of SSE.

Interprocedural Example

```
void foo(int* p) {
  *p = 1;
}
int main() {
    int a = 0;
    foo(&a);
    svf_assert(a == 1);
}
```

↓ compile

```
void @foo(i32* %p) {
entry:
  store i32 1, i32* %p
  ret void
}
i32 @main() {
entry:
  %a = alloca i32
  store i32 0, i32* %a
  call void @foo(i32* %a)
  %0 = load i32, i32* %a
  %cmp = icmp eq i32 %0, 1
  call void @svf_assert(i1 zeroext %cmp)
  ret i32 0
}
```
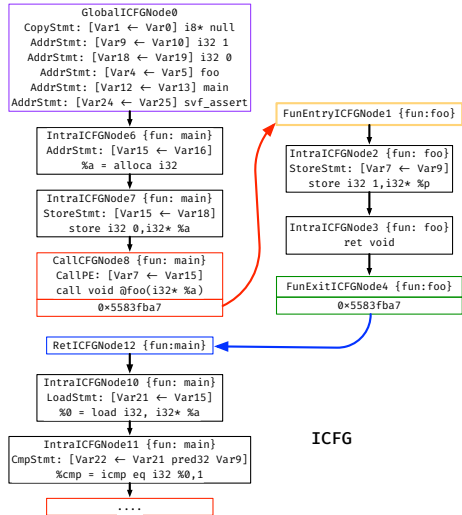
Interprocedural Example

```
void foo(int* p) {
    *p = 1;
}
int main() {
    int a = 0;
    foo(&a);
    svf_assert(a == 1);
}
```

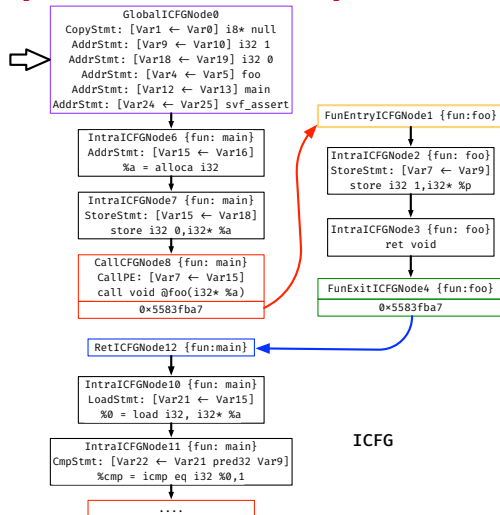↓ **compile**

```
void @foo(i32* %p) {
entry:
    store i32 1, i32* %p
    ret void
}
i32 @main() {
entry:
    %a = alloca i32
    store i32 0, i32* %a
    call void @foo(i32* %a)
    %0 = load i32, i32* %a
    %cmp = icmp eq i32 %0, 1
    call void @svf_assert(i1 zeroext %cmp)
    ret i32 0
}
```

SVF →
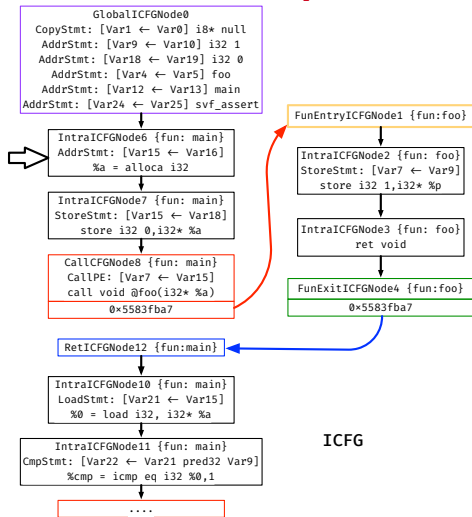

ICFG

Interprocedural Example

The values of Z3 expressions for each SVFVar after analyzing GlobalICFGNode0 (use printExprValues() to print SVFVars and their Values)

Interprocedural Example

ICFG

Interprocedural Example

Interprocedural Example

ICFG

Algorithm 3 `handleIntra(intraEdge)`

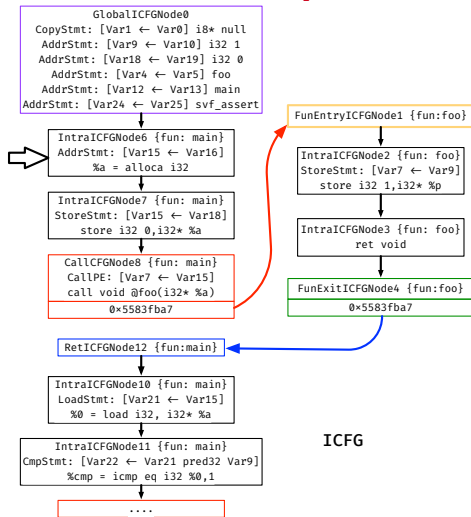1 **if** intraEdge.getCondition() && !handleBranch(intraEdge) **then**
2   return false
3 **else**
4   handleNonBranch(edge)

HandleNonBranch(intraEdge)

    dst ← intraEdge.getDstNode(); src ← intraEdge.getSrcNode()
    foreach stmt ∈ dst.getSVFStmts() **do**
       ...
11   **else if** load ∈ dyn_cast⟨LoadStmt⟩(stmt) **then**
12   lhs ← getZ3Expr(load.getLHSVarID())
13   rhs ← getZ3Expr(load.getRHSVarID())
14   addToSolver(lhs == z3Mgr.loadValue(rhs))
15   **else if** store ∈ dyn_cast⟨StoreStmt⟩(stmt) **then**
16   lhs ← getZ3Expr(store.getLHSVarID())
17   rhs ← getZ3Expr(store.getRHSVarID())
18   z3Mgr.storeValue(lhs, rhs)
19   **else if** gep ∈ dyn_cast⟨GepStmt⟩(stmt) **then**
20   lhs ← getZ3Expr(gep.getLHSVarID())
21   rhs ← getZ3Expr(gep.getRHSVarID())
22   offset = z3Mgr.getGepOffset(gep)
23   gepAddress = z3Mgr.getGepObjAddress(rhs, offset)
24   addToSolver(lhs == gepAddress)
       ...

Interprocedural Example

Analyzing IntraICFGNode6 {fun: main}

StoreStmt: [Var15 ← Var18]

`store i32 0, i32 * %a`

Interprocedural Example
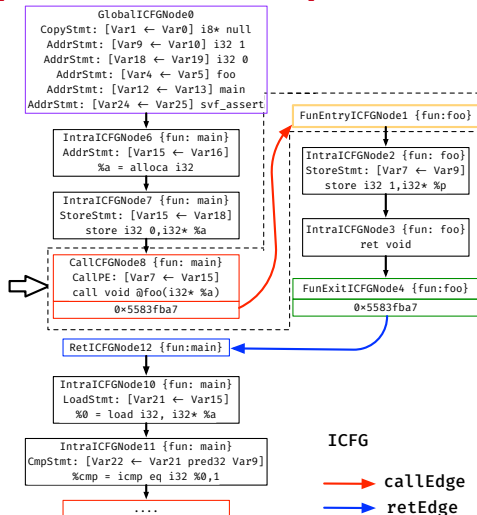
Algorithm 4 handleCall(callEdge)

1 callNode ← callEdge.getSrcNode();
2 FunEntryNode ← callEdge.getDstNode();
3 getSolver().push;
4 **foreach** callPE ∈ calledge.getCallPEs() **do**
5   lhs ← getZ3Expr(callPE.getLHSVarID());
6   rhs ← getZ3Expr(callPE.getRHSVarID());
7   addToSolver(lhs == rhs);

8 return true;

Interprocedural Example

Interprocedural Example

ICFG

Algorithm 3 `handleIntra(intraEdge)`

1 **if** intraEdge.getCondition() && !handleBranch(intraEdge) **then**
2   return false
3 **else**
4   handleNonBranch(edge)

HandleNonBranch(intraEdge)

   dst ← intraEdge.getDstNode(); src ← intraEdge.getSrcNode()
   foreach stmt ∈ dst.getSVFStmts() **do**
     ...
11   **else if** load ∈ dyn_cast⟨LoadStmt⟩(stmt) **then**
12   lhs ← getZ3Expr(load.getLHSVarID())
13   rhs ← getZ3Expr(load.getRHSVarID())
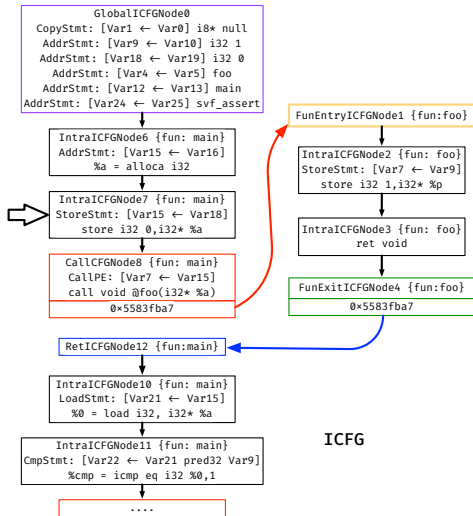14   addToSolver(lhs == z3Mgr.loadValue(rhs))
15   **else if** store ∈ dyn_cast⟨StoreStmt⟩(stmt) **then**
16   lhs ← getZ3Expr(store.getLHSVarID())
17   rhs ← getZ3Expr(store.getRHSVarID())
18   z3Mgr.storeValue(lhs, rhs)
19   **else if** gep ∈ dyn_cast⟨GepStmt⟩(stmt) **then**
20   lhs ← getZ3Expr(gep.getLHSVarID())
21   rhs ← getZ3Expr(gep.getRHSVarID())
22   offset = z3Mgr.getGepOffset(gep)
23   gepAddress = z3Mgr.getGepObjAddress(rhs, offset)
24   addToSolver(lhs == gepAddress)
     ...

Interprocedural Example

ret void instruction.
Nothing needs to be done.
Continue.

ICFG

Interprocedural Example

ICFG
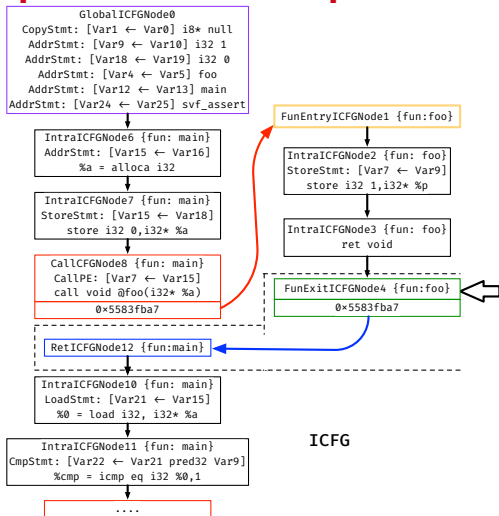
Algorithm 5 handleRet(retEdge)

1.   rhs(getCtx());
2.   **if** retPE ← retEdge.getRetPE() **then**
3. | rhs ← getEvalExpr(getZ3Expr(retPE.getRHSVarID()));
4.   getSolver().pop();
5.   **if** retPE ← retEdge.getRetPE() **then**
6. | lhs ← getZ3Expr(retPE.getLHSVarID());
7. | addToSolver(lhs == rhs);
8.   return true;

Interprocedural Example
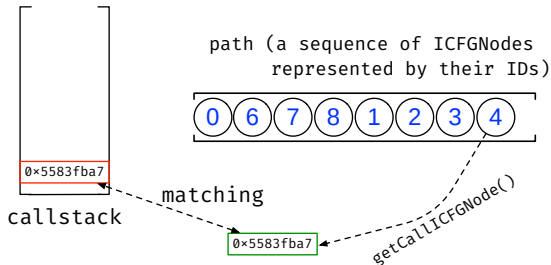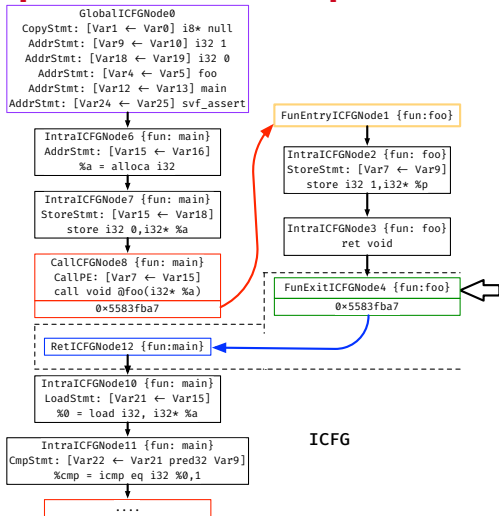
State of callstack while processing return edge from FunExitICFGNode4 to RetICFGNode12

path (a sequence of ICFGNodes represented by their IDs)

Interprocedural Example

ICFG

Algorithm 3 `handleIntra(intraEdge)`

1 **if** `intraEdge.getCondition()` && `!handleBranch(intraEdge)` **then**

2   `return false`
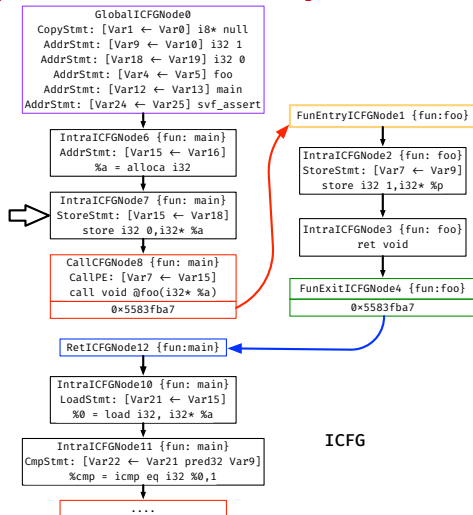
3 **else**

4   `handleNonBranch(edge)`

HandleNonBranch(intraEdge)

   dst ← intraEdge.getDstNode(); src ← intraEdge.getSrcNode()
   foreach stmt ∈ dst.getSVFStmts() **do**
     ...

11   **else if** load ∈ dyn_cast⟨LoadStmt⟩(stmt) **then**

12   lhs ← getZ3Expr(load.getLHSVarID())

13   rhs ← getZ3Expr(load.getRHSVarID())

14   addToSolver(lhs == z3Mgr.loadValue(rhs))

15   **else if** store ∈ dyn_cast⟨StoreStmt⟩(stmt) **then**

16   lhs ← getZ3Expr(store.getLHSVarID())

17   rhs ← getZ3Expr(store.getRHSVarID())

18   z3Mgr.storeValue(lhs, rhs)

19   **else if** gep ∈ dyn_cast⟨GepStmt⟩(stmt) **then**

20   lhs ← getZ3Expr(gep.getLHSVarID())

21   rhs ← getZ3Expr(gep.getRHSVarID())

22   offset = z3Mgr.getGepOffset(gep)

23   gepAddress = z3Mgr.getGepObjAddress(rhs, offset)

24   addToSolver(lhs == gepAddress)
     ...

Interprocedural Example

Software Verification   https://github.com/SVF-tools/Teaching-Software-Verification

Branch Example

```c
int main(){
    int x = 1, y = 1;
    int a = 1, b = 2;
    if (a > b) {
        y++;
    } else {
        x++;
        svf_assert (x == 2);
    }
    return 0;
}
```

compile

```llvm
i32 @main() {
entry:
  %cmp = icmp sgt i32 1, 2
  br i1 %cmp, label %if.then, label %if.else
if.then:
  %inc = add nsw i32 1, 1
  br label %if.end
if.else:
  %inc1 = add nsw i32 1, 1
  %cmp2 = icmp eq i32 %inc1, 2
  call void @svf_assert(i1 zeroext %cmp2)
  br label %if.end
if.end:
ret i32 0
}
```

Branch Example

```
int main(){
    int x = 1, y = 1;
    int a = 1, b = 2;
    if (a > b) {
        y++;
    } else {
        x++;
        svf_assert (x == 2);
    }
    return 0;
}
```

compile

```
i32 @main() {
entry:
  %cmp = icmp sgt i32 1, 2
  br i1 %cmp, label %if.then, label %if.else
if.then:
  %inc = add nsw i32 1, 1
  br label %if.end
if.else:
  %inc1 = add nsw i32 1, 1
  %cmp2 = icmp eq i32 %inc1, 2
  call void @svf_assert(i1 zeroext %cmp2)
  br label %if.end
if.end:
ret i32 0
}
```

SVF →

ICFG
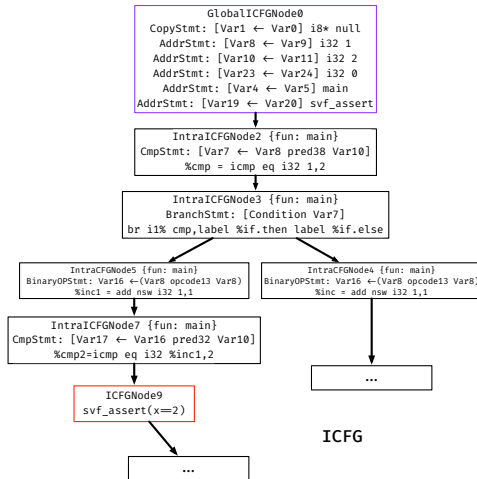
Branch Example


```
-------------SVFVar and Value-------------
ObjVar20 (0x7f000014)    Value: NULL
ObjVar24 (0x7f000018)    Value: 0
ObjVar11 (0x7f00000b)    Value: 2
ObjVar9 (0x7f000009)     Value: 1
ObjVar5 (0x7f000005)     Value: NULL
ValVar19                 Value: 0x7f000014
ValVar23                 Value: 0
ObjVar2 (0x7f000002)     Value: NULL
ObjVar3 (0x7f000003)     Value: NULL
ValVar1                  Value: 3
ValVar0                  Value: 3
ValVar4                  Value: 0x7f000005
ValVar8                  Value: 1
ValVar10                 Value: 2
           ...
-------------------------------------------
```

The values of Z3 expressions for each SVFVar

after analyzing `GlobalICFGNode0`

Branch Example


```
GlobalICFGNode0
CopyStmt: [Var1 ← Var0] i8* null
AddrStmt: [Var8 ← Var9] i32 1
AddrStmt: [Var10 ← Var11] i32 2
AddrStmt: [Var23 ← Var24] i32 0
AddrStmt: [Var4 ← Var5] main
AddrStmt: [Var19 ← Var20] svf_assert
```

```
IntraICFGNode2 {fun: main}
CmpStmt: [Var7 ← Var8 pred38 Var10]
   %cmp = icmp eq i32 1,2
```

```
IntraICFGNode3 {fun: main}
BranchStmt: [Condition Var7]
br i1% cmp,label %if.then label %if.else
```

```
IntraCFGNode5 {fun: main}
BinaryOPStmt: Var16 ←(Var8 opcode13 Var8)
   %inc1 = add nsw i32 1,1
```

```
IntraCFGNode4 {fun: main}
BinaryOPStmt: Var16 ←(Var8 opcode13 Var8)
   %inc = add nsw i32 1,1
```

```
IntraICFGNode7 {fun: main}
CmpStmt: [Var17 ← Var16 pred32 Var10]
   %cmp2=icmp eq i32 %inc1,2
```

```
...
```

```
ICFGNode9
svf_assert(x==2)
```

```
...
```

ICFG

```
## Analyzing IntraICFGNode2 {fun: main}
CmpStmt: [Var7 <-- (Var8 predicate38 Var10)]
   %cmp = icmp sgt i32 1, 2
==> (not (<= ValVar8 ValVar10))
==> (= ValVar7 0)
                    ...
```

Code for handling `CmpStmt` has been implemented in the HandleNonBranch() function.

Branch Example

GlobalICFGNode0
CopyStmt: [Var1 ← Var0] i8* null
AddrStmt: [Var8 ← Var9] i32 1
AddrStmt: [Var10 ← Var11] i32 2
AddrStmt: [Var23 ← Var24] i32 0
AddrStmt: [Var4 ← Var5] main
AddrStmt: [Var19 ← Var20] svf_assert

IntraICFGNode2 {fun: main}
CmpStmt: [Var7 ← Var8 pred38 Var10]
%cmp = icmp eq i32 1,2

IntraICFGNode3 {fun: main}
BranchStmt: [Condition Var7]
br i1% cmp,label %if.then label %if.else

IntraCFGNode5 {fun: main}
BinaryOPStmt: Var16 ←(Var8 opcode13 Var8)
%inc1 = add nsw i32 1,1

IntraCFGNode4 {fun: main}
BinaryOPStmt: Var16 ←(Var8 opcode13 Var8)
%inc = add nsw i32 1,1

IntraICFGNode7 {fun: main}
CmpStmt: [Var17 ← Var16 pred32 Var10]
%cmp2=icmp eq i32 %inc1,2

ICFGNode9
svf_assert(x=2)

ICFG

Algorithm 3 `handleIntra(intraEdge)`

1 **if** `intraEdge.getCondition()` _&&_ `!handleBranch(intraEdge)` **then**

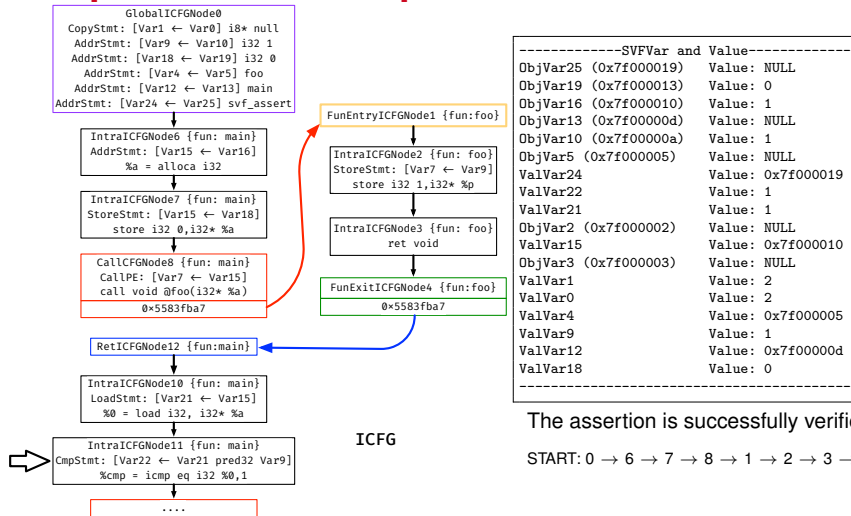2 | return false

3 **else**

4 | handleNonBranch(edge)

`handleBranch(intraEdge)`

1 $\quad$ cond = intraEdge.getCondition()

2 $\quad$ successorVal = intraEdge.getSuccessorCondValue()

3 $\quad$ res = getEvalExpr(cond == suc)

4 **if** res.is_false() **then**

5 | addToSolver(cond! = suc)

6 | return false

7 **else if** res.is_true() **then**

8 | addToSolver(cond == suc)

9 | return true

10 **else**

11 | return true

Branch Example


```
                 GlobalICFGNode0
CopyStmt: [Var1 ← Var0] i8* null
 AddrStmt: [Var8 ← Var9] i32 1
 AddrStmt: [Var10 ← Var11] i32 2
 AddrStmt: [Var23 ← Var24] i32 0
 AddrStmt: [Var4 ← Var5] main
AddrStmt: [Var19 ← Var20] svf_assert
```

```
     IntraICFGNode2 {fun: main}
CmpStmt: [Var7 ← Var8 pred38 Var10]
     %cmp = icmp eq i32 1,2
```

```
     IntraICFGNode3 {fun: main}
     BranchStmt: [Condition Var7]
br i1% cmp,label %if.then label %if.else
```

```
        IntraCFGNode5 {fun: main}
BinaryOPStmt: Var16 ←(Var8 opcode13 Var8)
        %inc1 = add nsw i32 1,1
```

```
        IntraCFGNode4 {fun: main}
BinaryOPStmt: Var16 ←(Var8 opcode13 Var8)
        %inc = add nsw i32 1,1
```

```
     IntraICFGNode7 {fun: main}
CmpStmt: [Var17 ← Var16 pred32 Var10]
     %cmp2=icmp eq i32 %inc1,2
```

```
   ICFGNode9
svf_assert(x==2)
```

```
ICFG
```

```
-----------SVFVar and Value-----------
ObjVar20 (0x7f000014)   Value: NULL
ObjVar24 (0x7f000018)   Value: 0
ObjVar11 (0x7f00000b)   Value: 2
ObjVar9 (0x7f000009)    Value: 1
ObjVar5 (0x7f000005)    Value: NULL
ValVar19                Value: 0x7f000014
ValVar23                Value: 0
ObjVar2 (0x7f000002)    Value: NULL
ObjVar3 (0x7f000003)    Value: NULL
ValVar1                 Value: 3
ValVar0                 Value: 3
ValVar4                 Value: 0x7f000005
ValVar8                 Value: 1
ValVar10                Value: 2
ValVar7                 Value: 0
            ...
-----------------------------------------
```
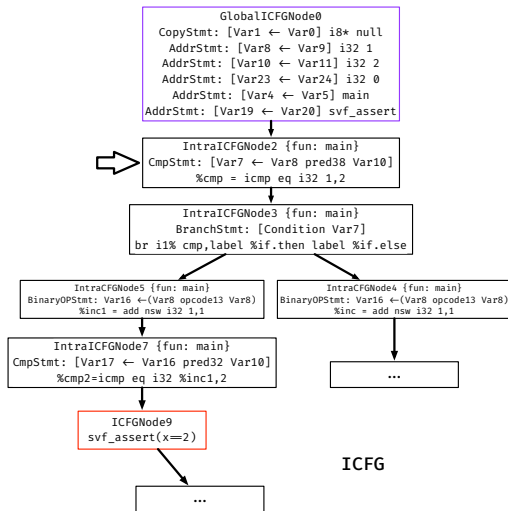
Branch IntraCFGEdge: [ICFGNode5 ← ICFGNode3]

branchCondition: $\%\text{cmp} = \text{icmp sgt i32 } 1, 2$

$(= \text{ValVar7 } 0)$

This conditional ICFGEdge is **feasible**!!

Branch Example


```
            GlobalICFGNode0
CopyStmt: [Var1 ← Var0] i8* null
  AddrStmt: [Var8 ← Var9] i32 1
 AddrStmt: [Var10 ← Var11] i32 2
 AddrStmt: [Var23 ← Var24] i32 0
 AddrStmt: [Var4 ← Var5] main
AddrStmt: [Var19 ← Var20] svf_assert
```

```
        IntraICFGNode2 {fun: main}
CmpStmt: [Var7 ← Var8 pred38 Var10]
      %cmp = icmp eq i32 1,2
```

```
          IntraICFGNode3 {fun: main}
             BranchStmt: [Condition Var7]
br i1% cmp,label %if.then label %if.else
```

```
IntraCFGNode5 {fun: main}
BinaryOPStmt: Var16 ←(Var8 opcode13 Var8)
     %inc1 = add nsw i32 1,1
```

```
IntraCFGNode4 {fun: main}
BinaryOPStmt: Var16 ←(Var8 opcode13 Var8)
     %inc = add nsw i32 1,1
```

```
        IntraICFGNode7 {fun: main}
CmpStmt: [Var17 ← Var16 pred32 Var10]
    %cmp2=icmp eq i32 %inc1,2
```

```
     ICFGNode9
  svf_assert(x==2)
```

```
...
```

```
...
```

ICFG

```
------------SVFVar and Value------------
ObjVar20 (0x7f000014)   Value: NULL
ObjVar24 (0x7f000018)    Value: 0
ObjVar11 (0x7f00000b)   Value: 2
ObjVar9 (0x7f000009)    Value: 1
ObjVar5 (0x7f000005)    Value: NULL
ValVar19               Value: 0x7f000014
ValVar23               Value: 0
ValVar17               Value: 1
ObjVar2 (0x7f000002)    Value: NULL
ObjVar3 (0x7f000003)    Value: NULL
ValVar16               Value: 2
ValVar1                Value: 3
ValVar0                Value: 3
ValVar4                Value: 0x7f000005
ValVar8                Value: 1
ValVar10               Value: 2
ValVar7                Value: 0
                 . . .
----------------------------------------
```
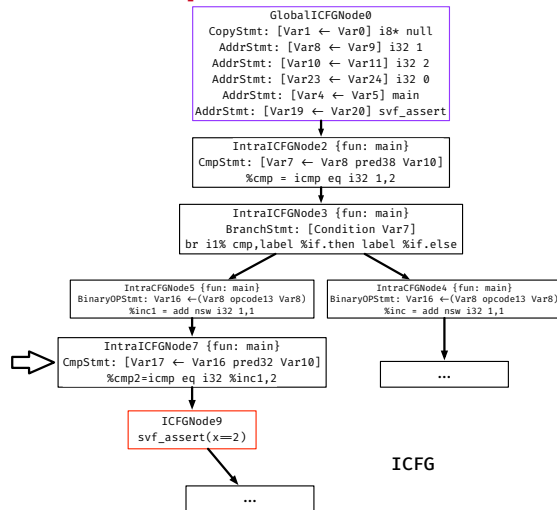
Analyzing IntraICFGNode7 fun: main

CmpStmt: [Var17 ← (Var16 predicate32 Var10)]

Branch Example

The assertion is successfully verified!!

START: $0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 5 \rightarrow 7 \rightarrow 9 \rightarrow$ *END*

Branch Example


```
               GlobalICFGNode0
   CopyStmt: [Var1 ← Var0] i8* null
    AddrStmt: [Var8 ← Var9] i32 1
    AddrStmt: [Var10 ← Var11] i32 2
    AddrStmt: [Var23 ← Var24] i32 0
     AddrStmt: [Var4 ← Var5] main
  AddrStmt: [Var19 ← Var20] svf_assert
```

```
     IntraICFGNode2 {fun: main}
CmpStmt: [Var7 ← Var8 pred38 Var10]
      %cmp = icmp eq i32 1,2
```

```
        IntraICFGNode3 {fun: main}
        BranchStmt: [Condition Var7]
br i1% cmp,label %if.then label %if.else
```

```
      IntraICFGNode5 {fun: main}
BinaryOPStmt: Var16 ←(Var8 opcode13 Var8)
      %inc1 = add nsw i32 1,1
```

```
      IntraICFGNode4 {fun: main}
BinaryOPStmt: Var16 ←(Var8 opcode13 Var8)
      %inc = add nsw i32 1,1
```

```
       IntraICFGNode7 {fun: main}
CmpStmt: [Var17 ← Var16 pred32 Var10]
     %cmp2=icmp eq i32 %inc1,2
```

```
     ICFGNode9
   svf_assert(x==2)
```
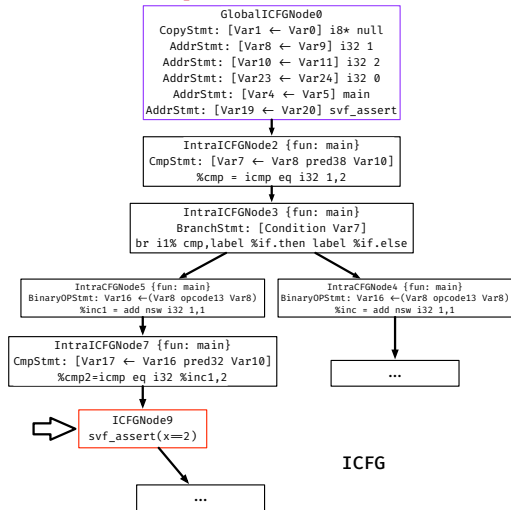
ICFG

```
-------------SVFVar and Value-------------
ObjVar20 (0x7f000014)   Value: NULL
ObjVar24 (0x7f000018)   Value: 0
ObjVar11 (0x7f00000b)   Value: 2
ObjVar9 (0x7f000009)    Value: 1
ObjVar5 (0x7f000005)    Value: NULL
ValVar19                Value: 0x7f000014
ValVar23                Value: 0
ObjVar2 (0x7f000002)    Value: NULL
ObjVar3 (0x7f000003)    Value: NULL
ValVar1                 Value: 3
ValVar0                 Value: 3
ValVar4                 Value: 0x7f000005
ValVar8                 Value: 1
ValVar10                Value: 2
ValVar7                 Value: 0
              ...
-------------------------------------------
```

Branch IntraCFGEdge: [ICFGNode4 ← ICFGNode3]

branchCondition: %cmp = icmp sgt i32 1, 2

(= ValVar7 1)

This conditional ICFGEdge is **infeasible**!!