

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::translatePath`
- `SSE::handleNonBranch`
- `SSE::handleCall`
- `SSE::handleRet`
- `SSE::handleBranch`
- Remember to put your previously implemented `Assignment-2.cpp` in place (under the `Assignment-2` folder).
- 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  
}
```

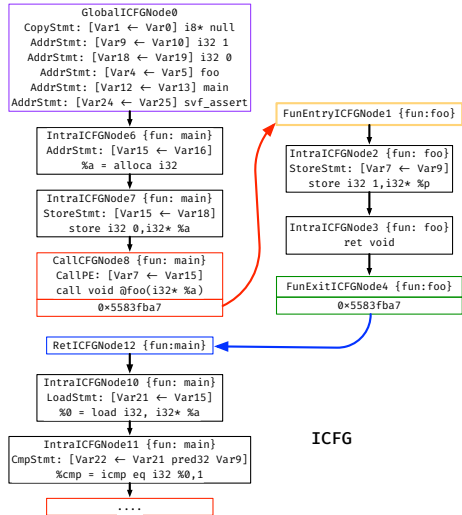
Interprocedural Example

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

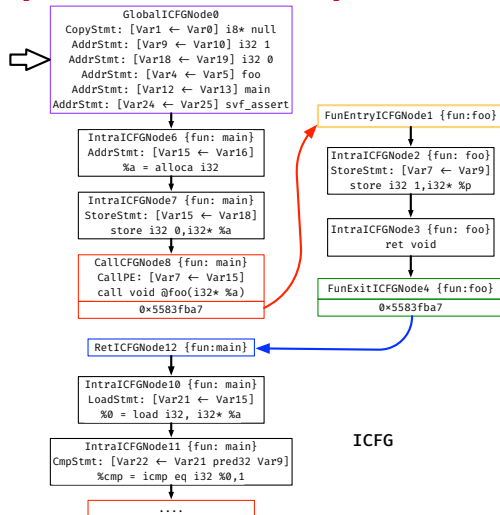
↓ compile

```
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 →



Interprocedural Example

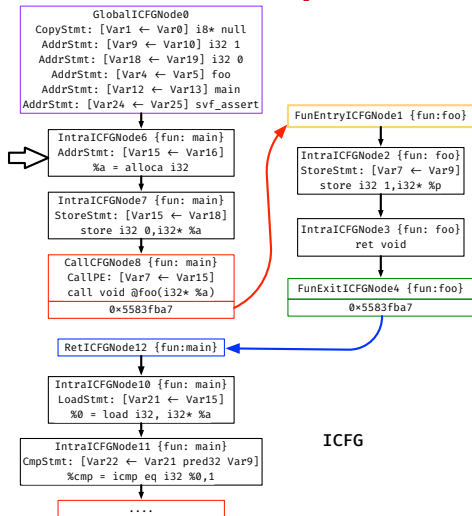


ICFG

-----SVFVar and Value-----		
ObjVar25 (0x7f000019)		Value: NULL
ObjVar19 (0x7f000013)		Value: 0
ObjVar16 (0x7f000010)		Value: NULL
ObjVar13 (0x7f00000d)		Value: NULL
ObjVar10 (0x7f00000a)		Value: 1
ObjVar5 (0x7f000005)		Value: NULL
ValVar24		Value: 0x7f000019
ObjVar2 (0x7f000002)		Value: NULL
ObjVar3 (0x7f000003)		Value: NULL
ValVar1		Value: 2
ValVar0		Value: 2
ValVar4		Value: 0x7f000005
ValVar9		Value: 1
ValVar12		Value: 0x7f00000d
ValVar18		Value: 0
...		

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

Interprocedural Example



Algorithm 3 `handleIntra(intraEdge)`

```

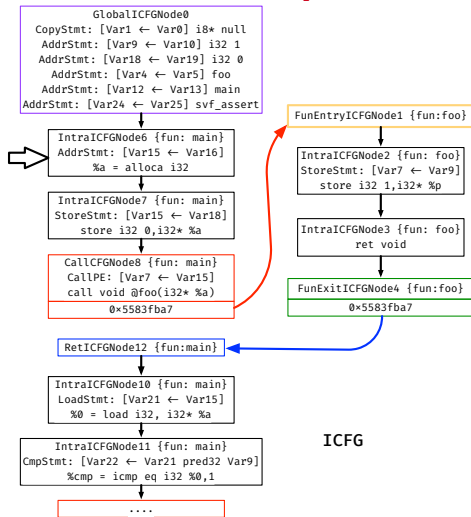
1 if intraEdge.getCondition() && !handleBranch(intraEdge)
  then
2   return false
3 else
4   handleNonBranch(edge)
  
```

HandleNonBranch(intraEdge)

```

1 dst ← intraEdge.getDstNode(); src ← intraEdge.getSrcNode()
2 foreach stmt ∈ dst.getSVFStmts() do
3   if addr ∈ dyn_cast<AddrStmt>(stmt) then
4     obj ← getMemObjAddress(addr.getRHSVarID())
5     lhs ← getZ3Expr(addr.getLHSVarID())
6     addToSolver(obj == lhs)
7   else if copy ∈ dyn_cast<CopyStmt>(stmt) then
8     lhs ← getZ3Expr(copy.getLHSVarID())
9     rhs ← getZ3Expr(copy.getRHSVarID())
10    addToSolver(rhs == lhs)
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))
  ...
  
```


Interprocedural Example



ICFG

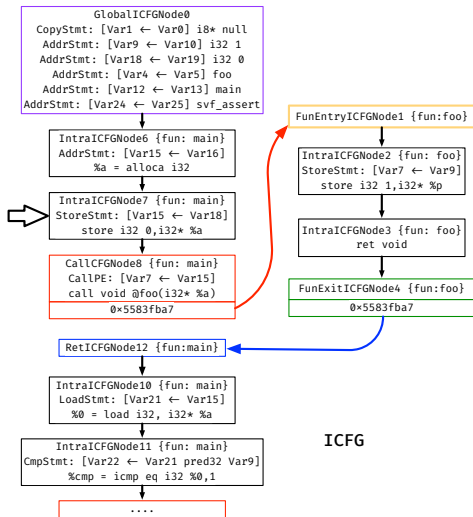
-----SVFVar and Value-----	
ObjVar25 (0x7f000019)	Value: NULL
ObjVar19 (0x7f000013)	Value: 0
ObjVar16 (0x7f000010)	Value: NULL
ObjVar13 (0x7f00000d)	Value: NULL
ObjVar10 (0x7f00000a)	Value: 1
ObjVar5 (0x7f000005)	Value: NULL
ValVar24	Value: 0x7f000019
ObjVar2 (0x7f000002)	Value: NULL
ObjVar3 (0x7f000003)	Value: NULL
ValVar1	Value: 2
ValVar0	Value: 2
ValVar4	Value: 0x7f000005
ValVar9	Value: 1
ValVar12	Value: 0x7f00000d
ValVar18	Value: 0
+ValVar15	Value: 0x7f000010
...	

Analyzing IntraICFGNode6 {fun: main}

AddrStmt: [Var14 ← Var15]

%a = alloca i32

Interprocedural Example



ICFG

Algorithm 3 `handleIntra(intraEdge)`

```

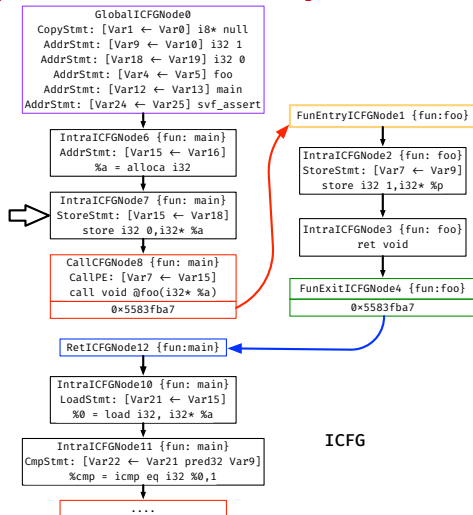
1 if intraEdge.getCondition() && !handleBranch(intraEdge)
  then
2   return false
3 else
4   handleNonBranch(edge)
    
```

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



ICFG

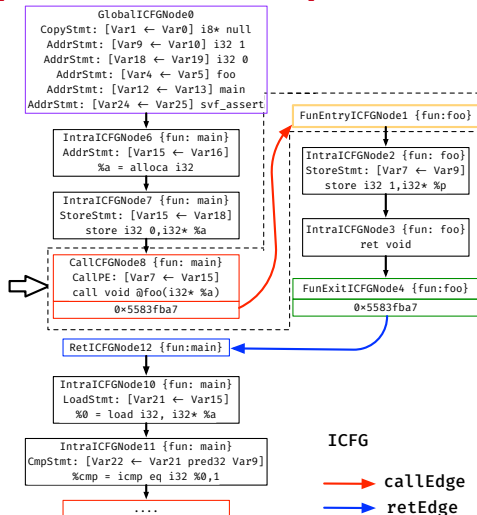
-----SVFVar and Value-----	
ObjVar25 (0x7f000019)	Value: NULL
ObjVar19 (0x7f000013)	Value: 0
ObjVar16 (0x7f000010)	Value: 0
ObjVar13 (0x7f00000d)	Value: NULL
ObjVar10 (0x7f00000a)	Value: 1
ObjVar5 (0x7f000005)	Value: NULL
ValVar24	Value: 0x7f000019
ObjVar2 (0x7f000002)	Value: NULL
ValVar15	Value: 0x7f000010
ObjVar3 (0x7f000003)	Value: NULL
ValVar1	Value: 2
ValVar0	Value: 2
ValVar4	Value: 0x7f000005
ValVar9	Value: 1
ValVar12	Value: 0x7f00000d
+ValVar18	Value: 0
...	

Analyzing IntraICFGNode6 {fun: main}

StoreStmt: [Var15 ← Var18]

store i32 0, i32 * %a

Interprocedural Example

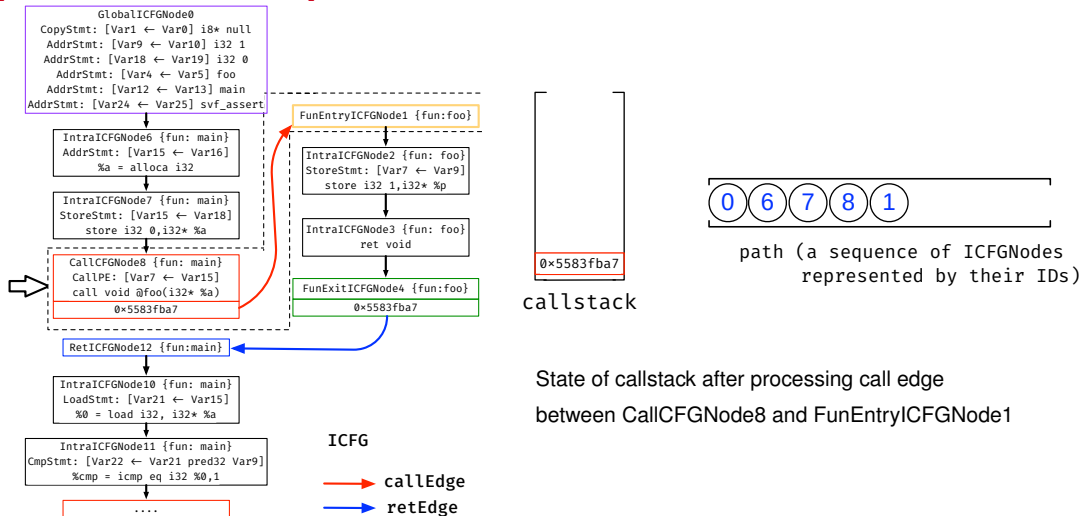


Algorithm 4 `handleCall`(callEdge)

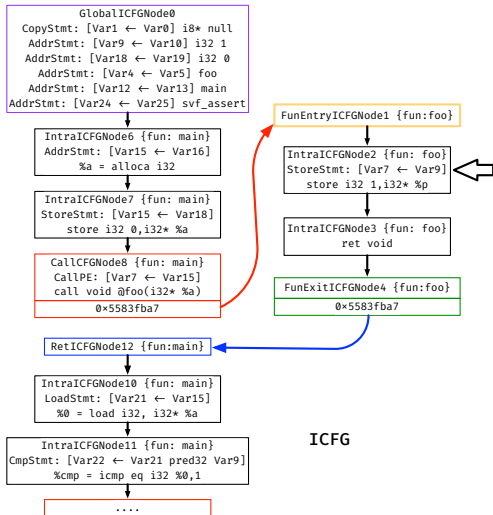
```

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



Algorithm 3 `handleIntra(intraEdge)`

```

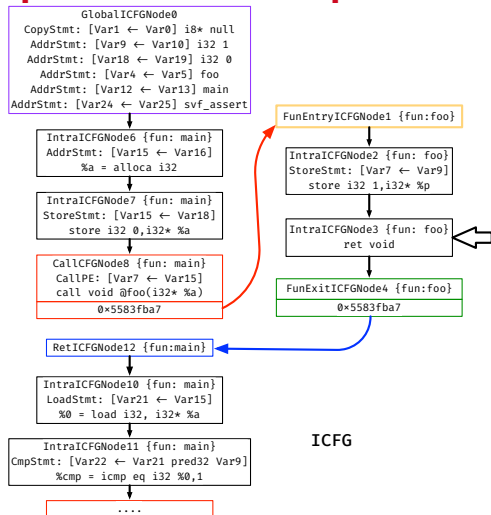
1 if intraEdge.getCondition() && !handleBranch(intraEdge)
  then
2   return false
3 else
4   handleNonBranch(edge)
  
```

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)
...
  
```

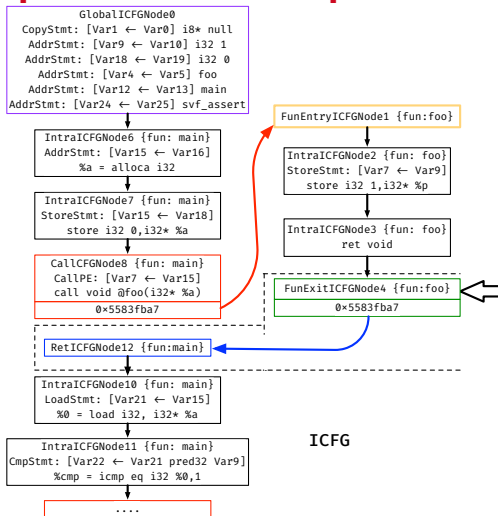
Interprocedural Example



ICFG

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

Interprocedural Example



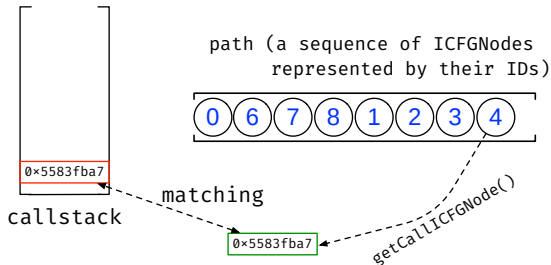
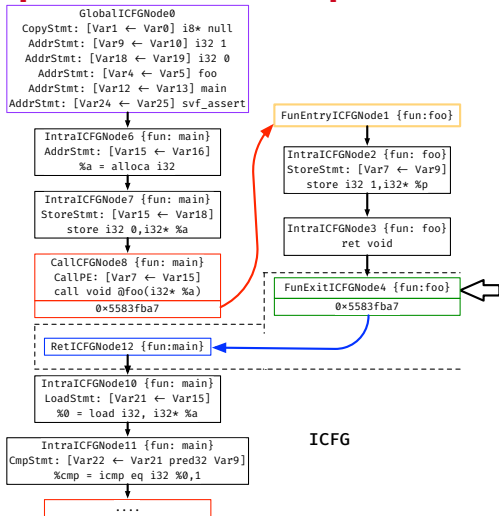
Algorithm 5 `handleRet`(retEdge)

```

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;

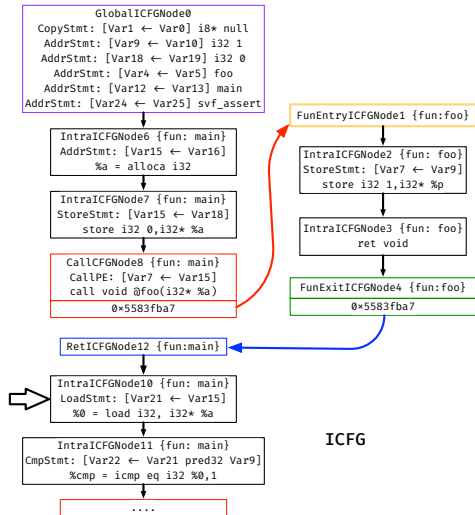
```


Interprocedural Example



State of callstack while processing return edge
from FunExitICFGNode4 to RetICFGNode12

Interprocedural Example



Algorithm 3 `handleIntra(intraEdge)`

```

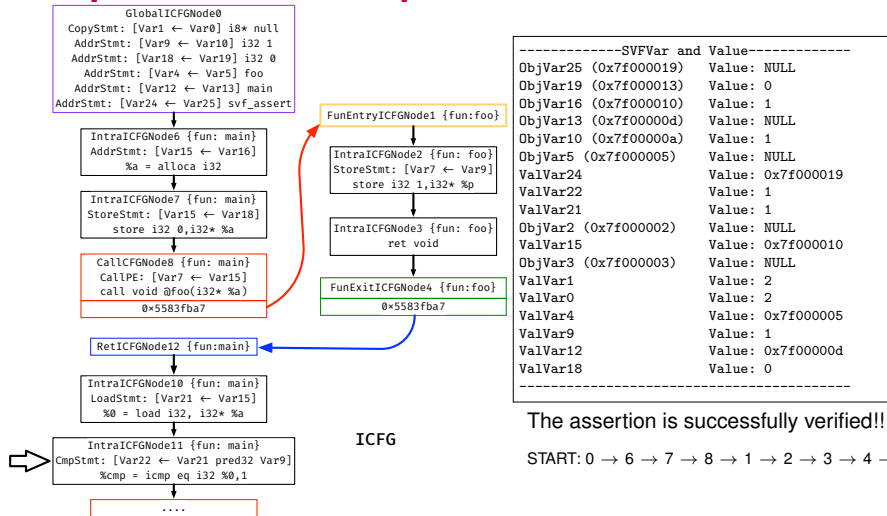
1 if intraEdge.getCondition() && !handleBranch(intraEdge)
  then
2   return false
3 else
4   handleNonBranch(edge)
  
```

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



The assertion is successfully verified!!

START: 0 → 6 → 7 → 8 → 1 → 2 → 3 → 4 → 12 → 10 → 11 → ... → END

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  
}
```

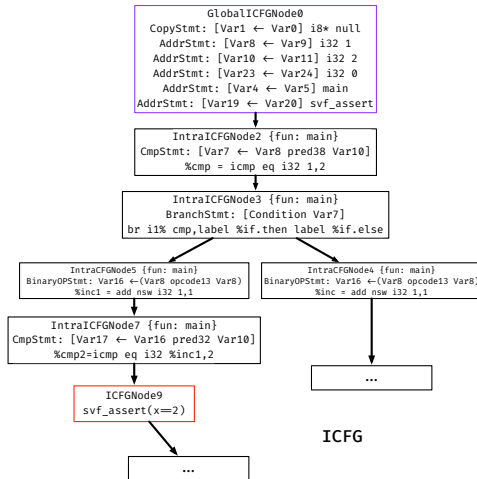
Branch Example

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

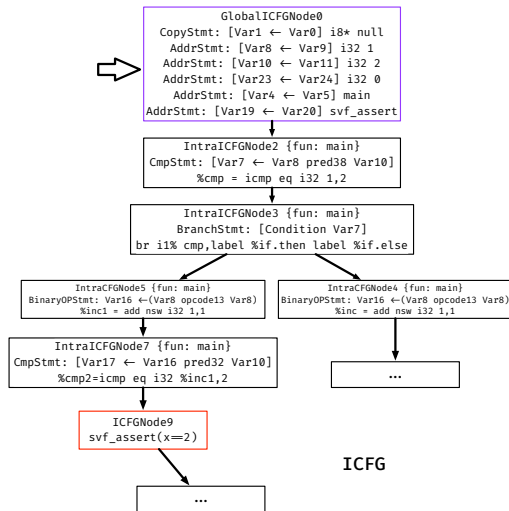
↓ compile

```
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



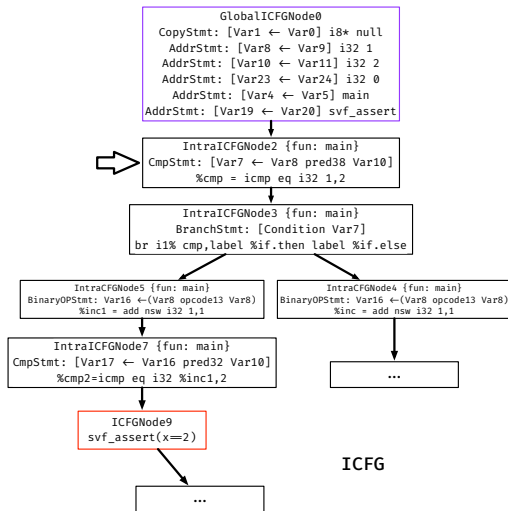
Branch Example



-----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

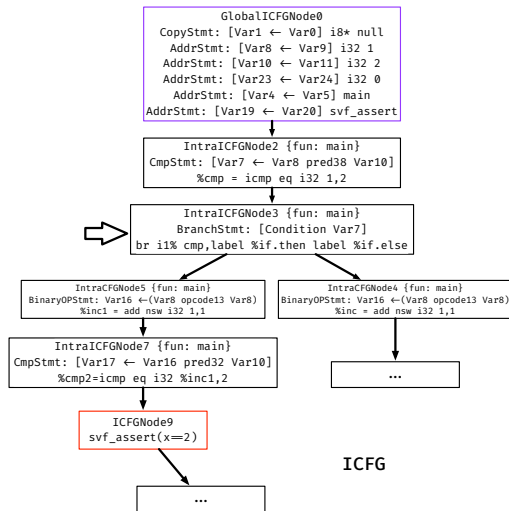
Branch Example



```
## 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



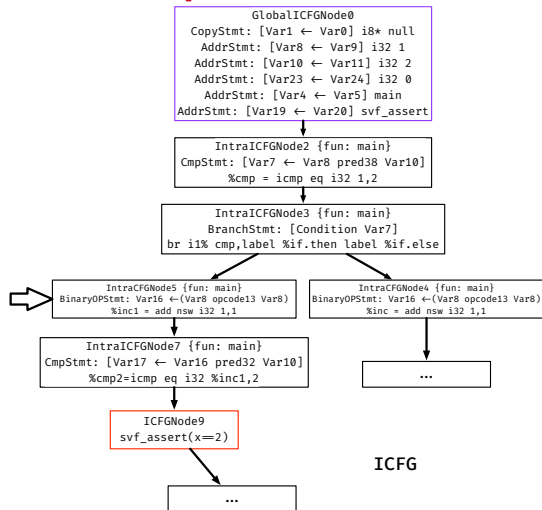
Algorithm 3 `handleIntra(intraEdge)`

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

`handleBranch(intraEdge)`

```
1 cond = intraEdge.getCondition()
2 successorVal = intraEdge.getSuccessorCondValue()
3 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



-----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
...	

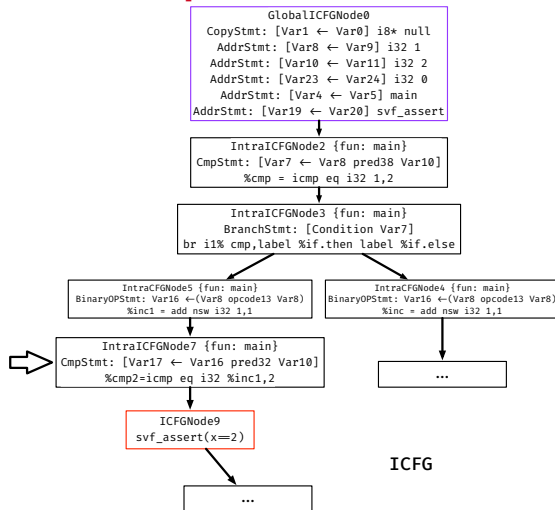
Branch IntraCFGEde: [ICFGNode5 ← ICFGNode3]

branchCondition: %cmp = icmp sgt i32 1,2

(= ValVar7 0)

This conditional ICFGEde is **feasible!!**

Branch Example

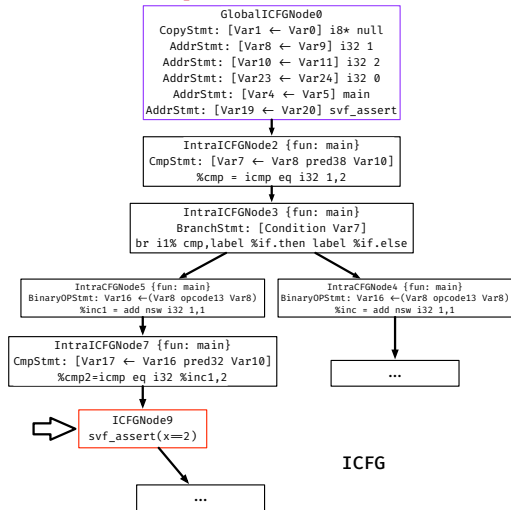


-----SVFVar and Value-----	
ObjVar20 (0x7f000014)	Value: NULL
ObjVar24 (0x7f000018)	Value: 0
ObjVar11 (0x7f00000b)	Value: 2
ObjVar9 (0x7f000009)	Value: 1
ObjVar5 (0x7f000005)	Value: NULL
ValVar19	Value: 0x7f000014
ValVar23	Value: 0
ValVar17	Value: 1
ObjVar2 (0x7f000002)	Value: NULL
ObjVar3 (0x7f000003)	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)]

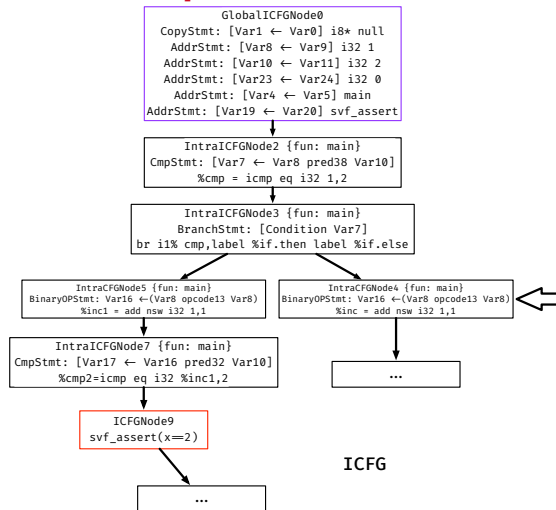
Branch Example



The assertion is successfully verified!!

START: 0 → 1 → 2 → 3 → 5 → 7 → 9 → *END*

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
ValVar7	Value: 0
...	

Branch IntraCFGEde: [ICFGNode4 ← ICFGNode3]

branchCondition: %cmp = icmp sgt i32 1,2
(= ValVar7 1)

This conditional ICFGEde is **infeasible!!**