CS738: Advanced Compiler Optimizations

INTER-PROCEDURAL DATA FLOW ANALYSIS IN LLVM USING VALUE CONTEXTS

Interprocedural data flow analysis in Soot using value contexts - Rohan Padhye, Uday P. Khedkar (SOAP'13)

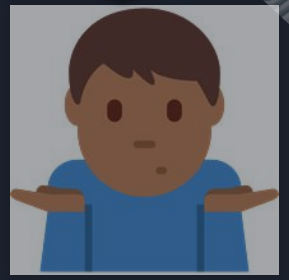
Group 1- Deeksha Arora (20111017)

Sambhrant Maurya (20111054)

Shruti Sharma (20111061)

What has been implemented

- A general forward intra/interprocedural framework
- Currently available analyses:
  - Sign Analysis
  - Copy Constant Propagation
    - Problems with pass execution
- Can be extended to other analyses:
  - Heap reference analysis
  - Points-to analysis
  - ...and much more



Flow of Control

Introduction

Project Objective

The Approach

Implementation

Algorithm

Sign Analysis: An example

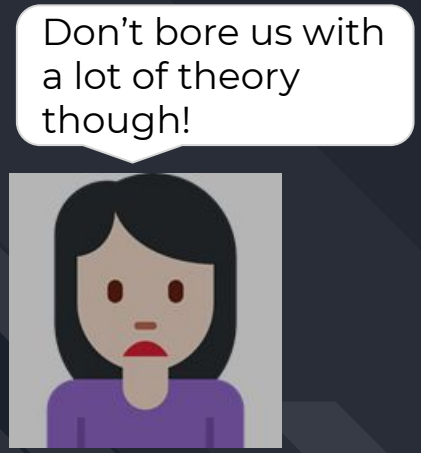
Files in our Framework

Workflow

Demo

Test Results for Sign Analysis

Future Work



Introduction - What is IPA?

- Interprocedural analysis spans across procedures and incorporates the effects of procedure calls in the caller procedures and calling contexts in the callee procedures.

- Two approaches
  - Functional - Construct summary flow functions for each procedure.
    - Graph Reachability : special case
  - Call-Strings - Remember unfinished calls as strings of procedure names.

How Precise?

Only valid paths should be covered.

01 Flow sensitivity

- Whether or not an analysis takes order of code into account.
- Considers intraprocedurally valid paths

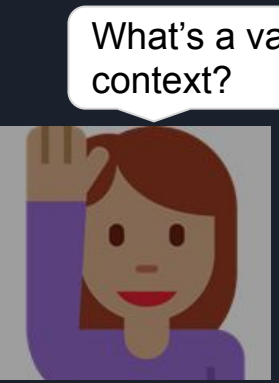
02 Context sensitivity

- Considers the calling context when analyzing the target of a function call.
- Considers interprocedurally valid paths

For maximum statically attainable precision , analysis must be both flow and context sensitive.

Project Objective

Implement a framework in LLVM to perform a fully context-sensitive forward IPA that uses value context, `context <methodName, entryValue>` for analysis.



Value Context

Use data flow values as context of analysis (to avoid reanalysis of procedure bodies). *entryValue* is the dataflow value at entry of procedure.

Value Context

- X = ⟨method, entryValue⟩
- exitValue(X)
- Data Flow Analysis is performed using traditional work-list method
- Work-list contains ⟨context, node⟩ pairs
- Call-sites: Find value context X = ⟨method, entryValue⟩
  - Found: Re-use exitValue(X)
  - Not found: Create new X and add all nodes to work-list
  - Record transition from this call-site to X
- Exit-sites: Set exitValue(X) and add callers to work-list

Value Context (continued)

Context<M,N,A>: A value-based context for a context-sensitive inter-procedural data flow analysis.

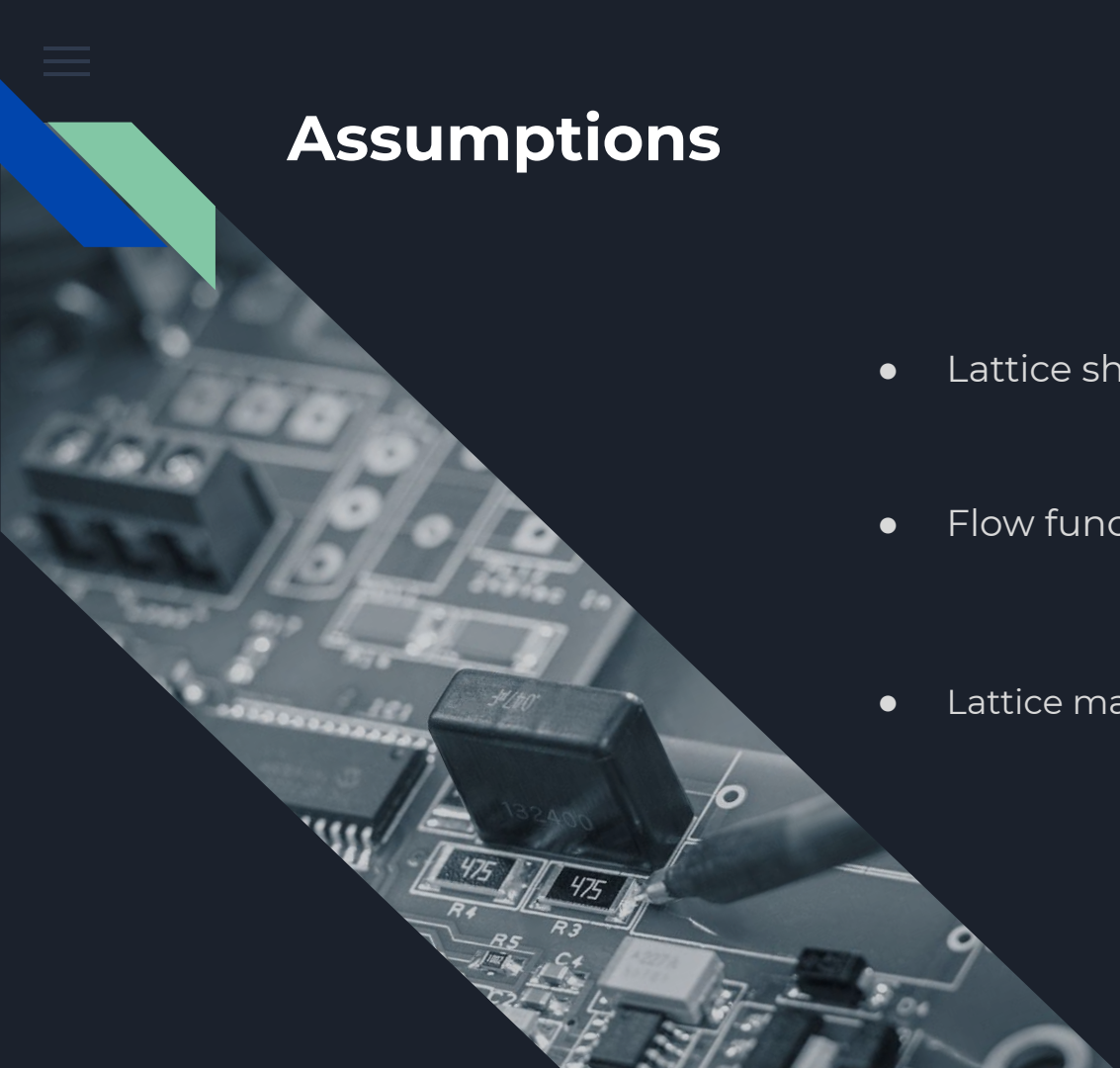
Type Parameters:

M - the type of a method

N - the type of a node in the CFG

A - the type of a data flow value

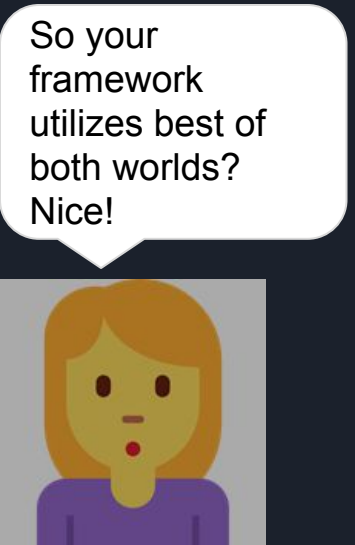
Assumptions



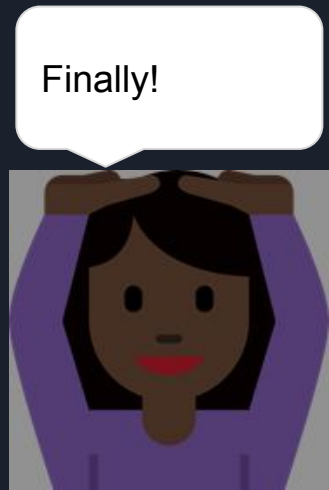
- Lattice should be finite.
- Flow functions are monotonic.
- Lattice may or may not be distributive.

The Approach

- Tabulation method of functional approach- Enumeration of functions as <*input*, *output*> pairs.
- Value based termination of call-strings approach - Use data flow values to restrict the explosion of call strings.
- If two or more calls to a procedure *P* have same dataflow value at entry of *P* , then all calls will have same value at exit of *P*.



Implementation



Global Data Structures

- `map <M, List<Context>> contexts:` set of all created contexts for a method M.
- `Map <CallSite,map <M,Context>> transitions`: map from call-sites to contexts, parameterised by the called method M.
- `list <pair<Context,list<<N>>> worklist`: CFG nodes parameterized with contexts that still need to be processed.

Algorithm

-procedure INITCONTEXT(X):

-ADD(contexts,X)

- Set ExitValue(X) <- T

- for all nodes n in the body of METHOD(X) do:

- ADD(worklist, <X, n>)

- Set IN(X, n) <- T and OUT(X, n) <- T

- Set IN(X, ENTRYNODE(m)) <- ENTRYVALUE(X)

-procedure DOANALYSIS:

- INITCONTEXT(<main,BI>)

- while ! worklist. empty() do:

- <X, n> <- REMOVENEXT(worklist)

Algorithm (Contd.)

- if n is not the entry node then:
    - IN(n) <- $\sqcap_{p \in pred(n)} OUT(p)$
- a <- IN(X, n)
- if n contains a method call then:
    - m <- TARGETMETHOD(n)
    - x <- CALLENTRYFLOWFUNCTION(X, m,n, a)
    - *X'* <- <m, x>
    - Add transition (*X'* <- *<X, n>)*
    - if X' ∈ contexts:
        - y <- EXITVALUE(X')
        - b1 <- CALLEXITFLOWFUNCTION(X, m, n, y)

Algorithm (Contd.)

- b2 <- CALLLOCALFLOWFUNCTION(X, n, a)

- OUT(X, n) <- b1 ⊓ b2

-else:

-INITCONTEXT(X')

-else:

- OUT(X, n) <- NORMALFLOWFUNCTION(X, n, a)

- if OUT(X, n) has changed :

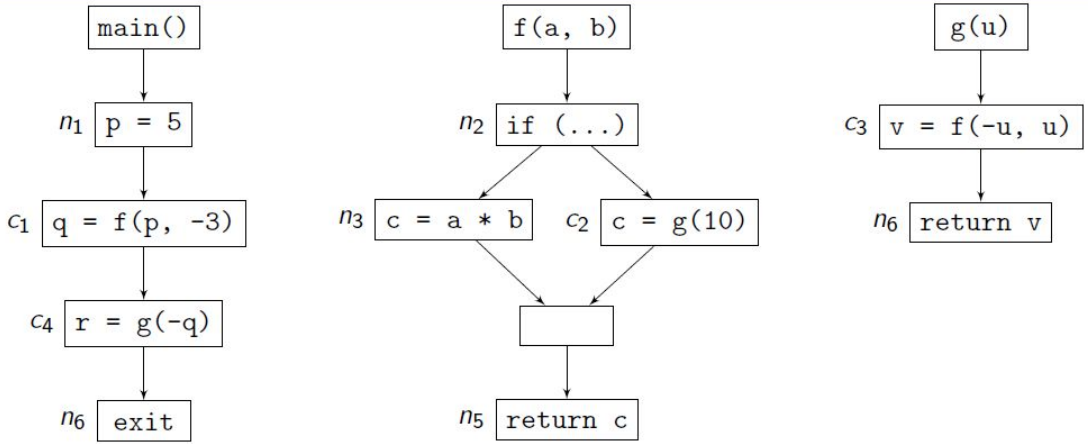
- add succ(n) to worklist[X]

- if n is the exit node then:

- EXITVALUE(X) <- OUT(X, n)

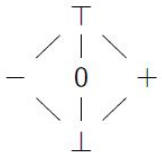
- ADD(worklist, <X', c>) ∀ <*X'* , *c*> ∈ transition[X]

Sign analysis : An example

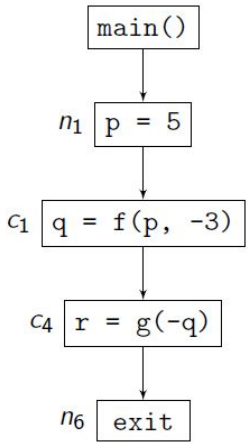


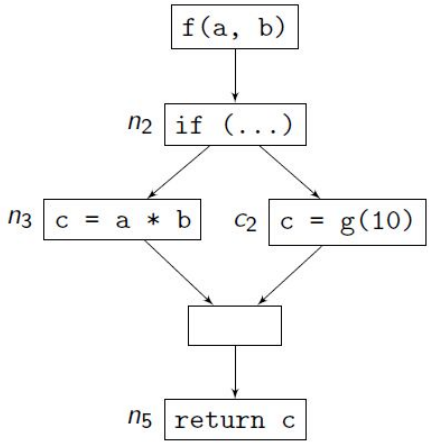
Context	Proc.	Entry	Exit

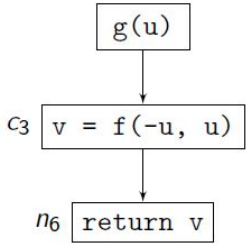
Value Contexts



Component Lattice

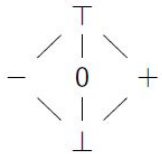




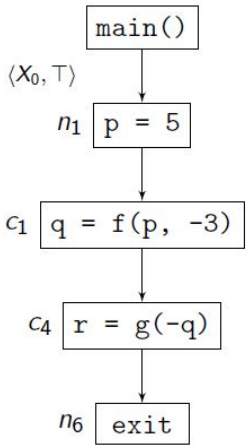


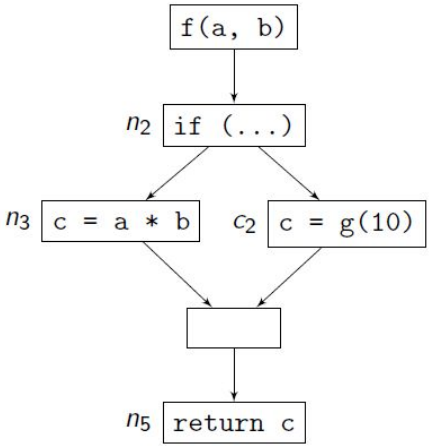
Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$

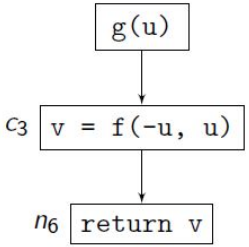
Value Contexts



Component Lattice

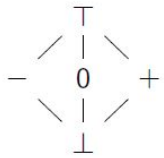




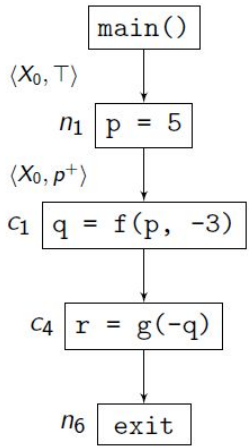


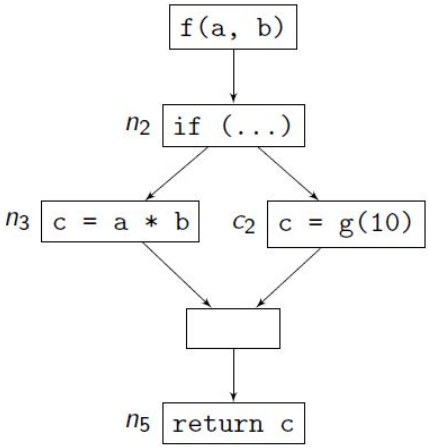
Context	Proc.	Entry	Exit
X_0	main	⊤	⊤

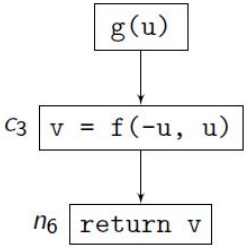
Value Contexts



Component Lattice

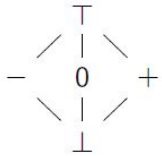




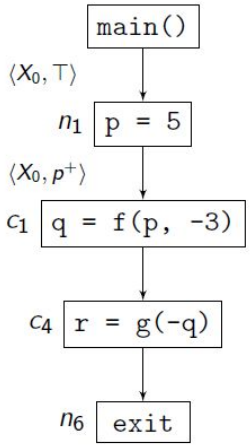


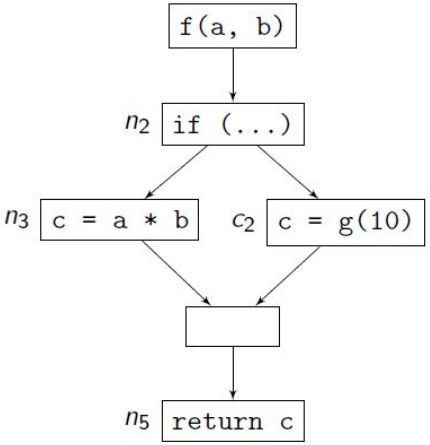
Context	Proc.	Entry	Exit
X_0	main	⊤	⊤

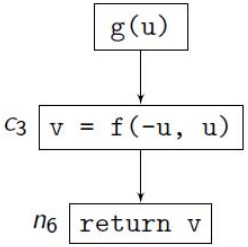
Value Contexts



Component Lattice

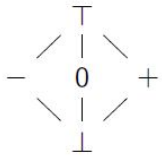




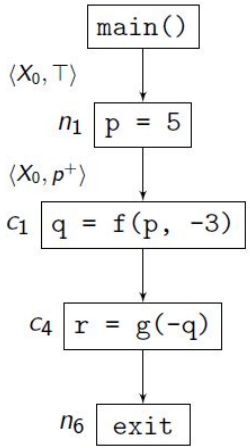


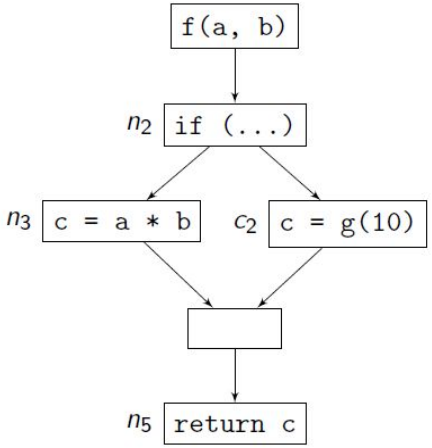
Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$

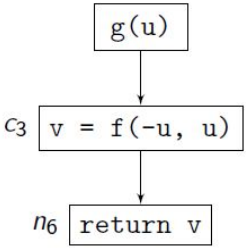
Value Contexts



Component Lattice

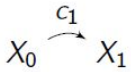




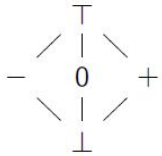


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$

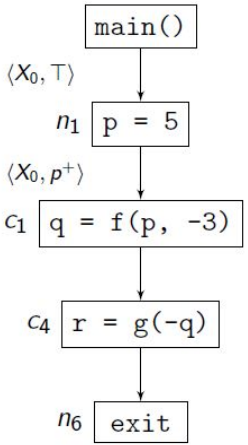
Value Contexts

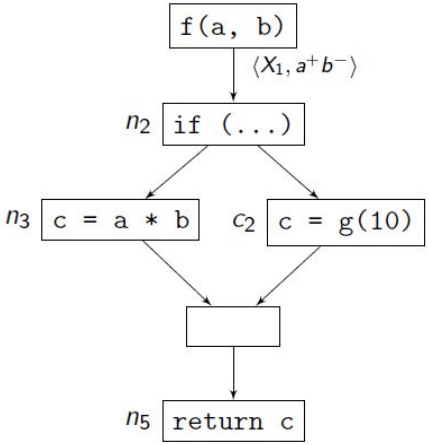


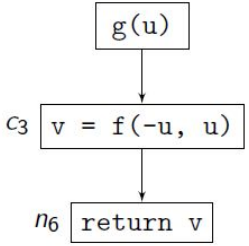
Context Transitions



Component Lattice

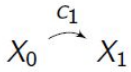




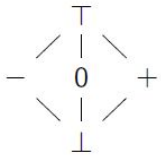


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$

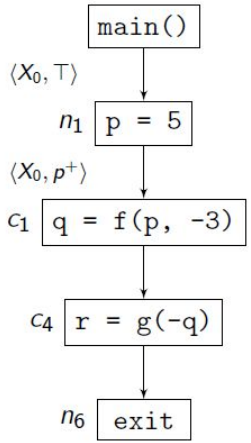
Value Contexts

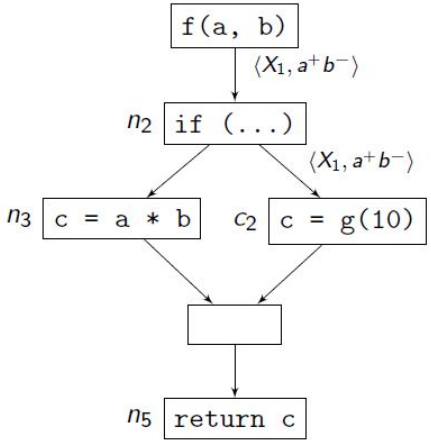


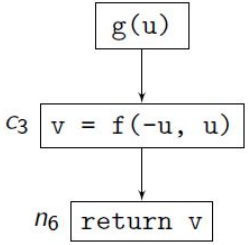
Context Transitions



Component Lattice

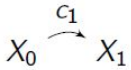




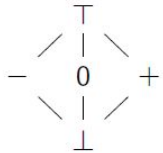


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$

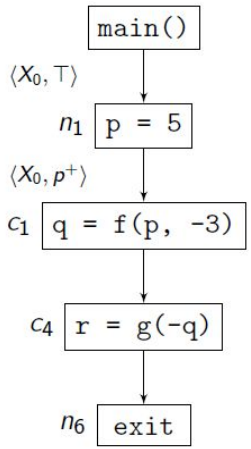
Value Contexts

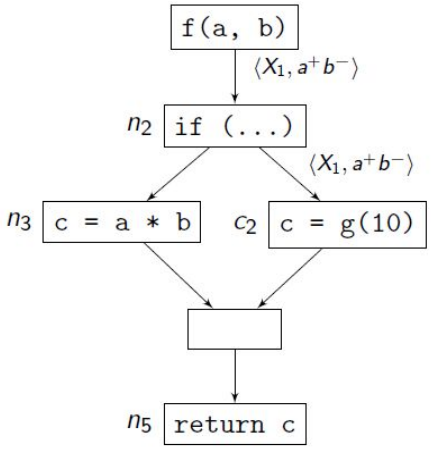


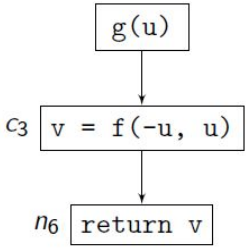
Context Transitions



Component Lattice

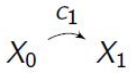




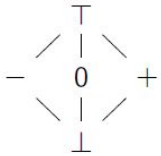


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	$\top$

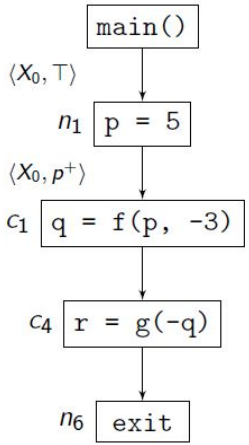
Value Contexts

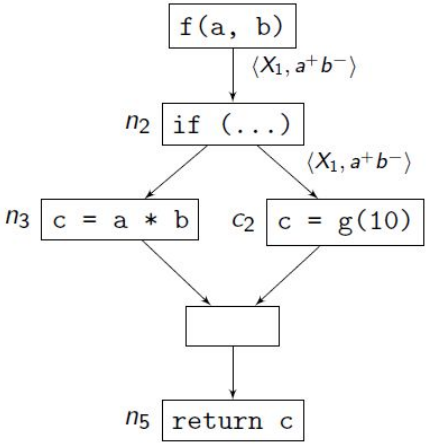


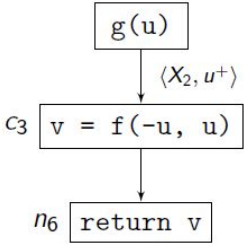
Context Transitions



Component Lattice

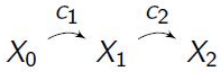




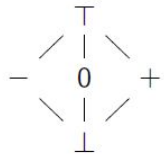


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	$\top$

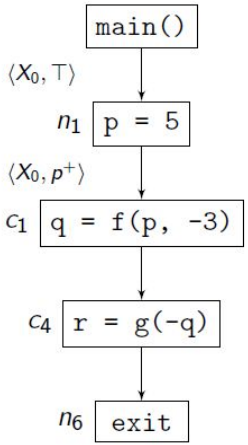
Value Contexts

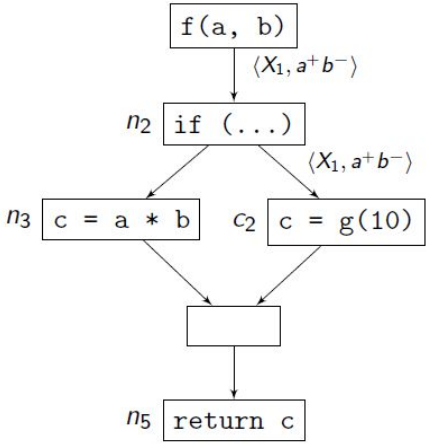


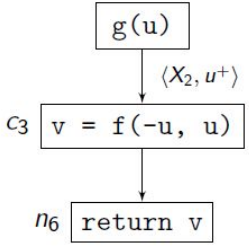
Context Transitions



Component Lattice

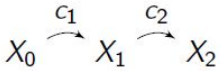




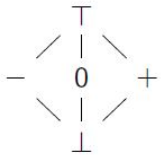


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	$\top$
X_3	f	a^-b^+	$\top$

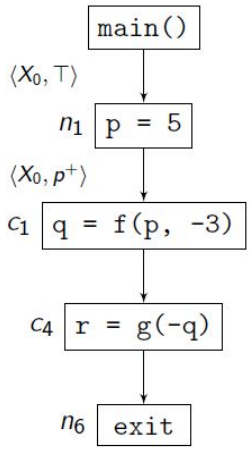
Value Contexts

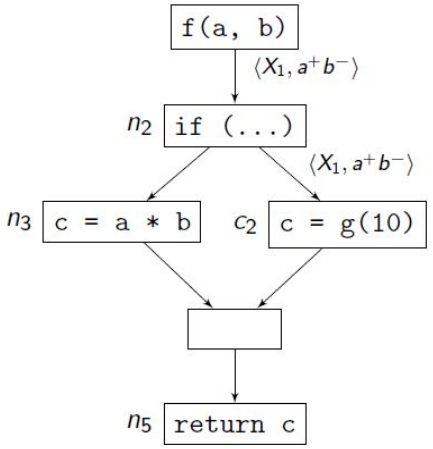


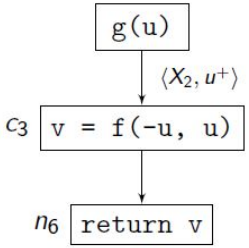
Context Transitions



Component Lattice

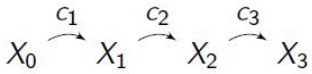




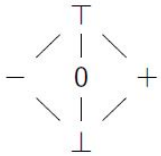


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	$\top$
X_3	f	a^-b^+	$\top$

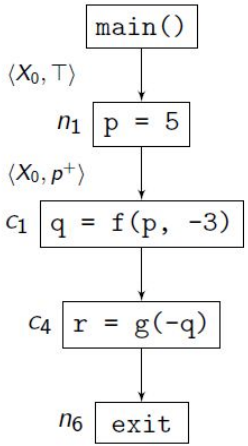
Value Contexts

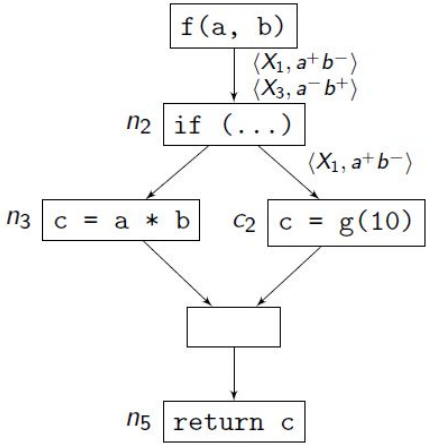


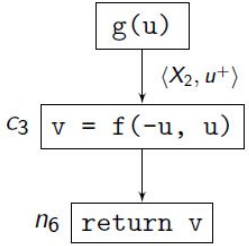
Context Transitions



Component Lattice

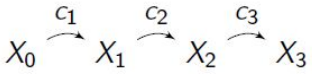




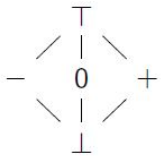


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	$\top$
X_3	f	a^-b^+	$\top$

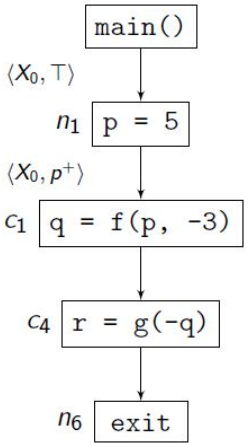
Value Contexts

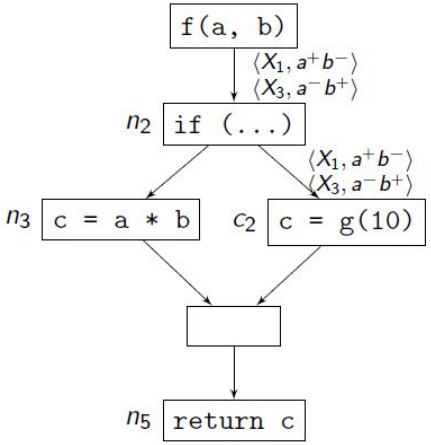


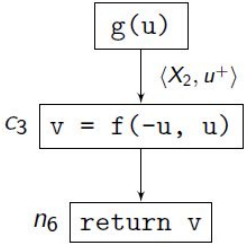
Context Transitions



Component Lattice

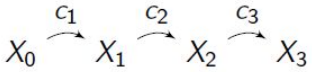




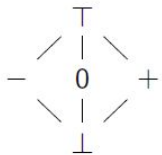


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	$\top$
X_3	f	a^-b^+	$\top$

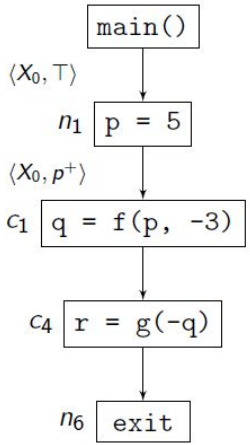
Value Contexts

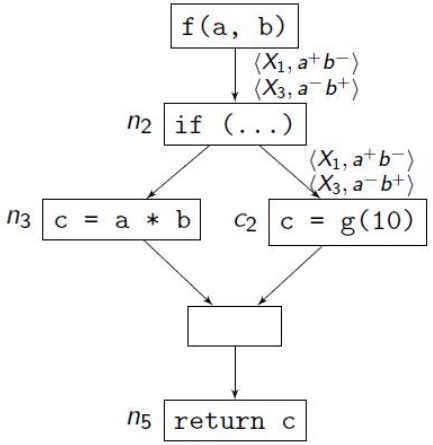


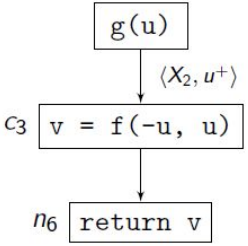
Context Transitions



Component Lattice

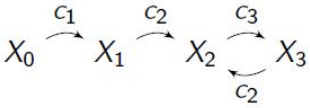




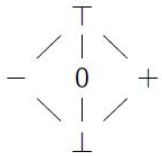


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	$\top$
X_3	f	a^-b^+	$\top$

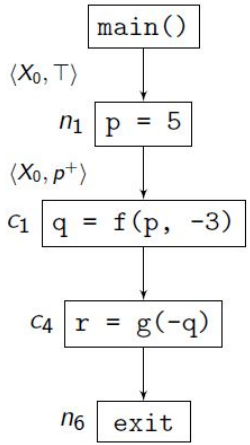
Value Contexts

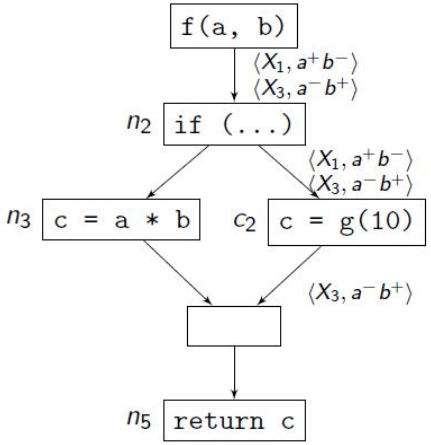


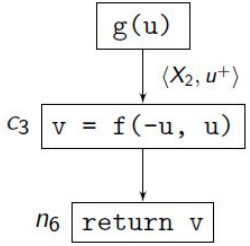
Context Transitions



Component Lattice

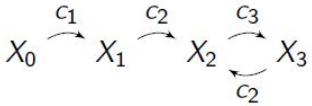




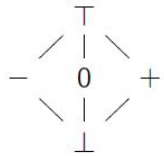


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	$\top$
X_3	f	a^-b^+	$\top$

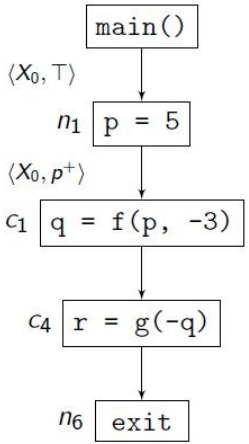
Value Contexts

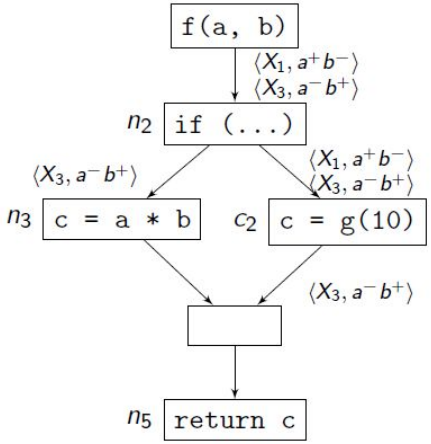


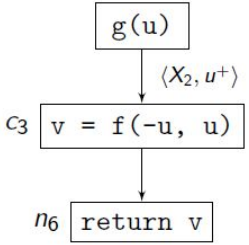
Context Transitions



Component Lattice

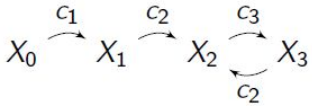




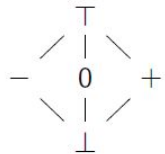


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	$\top$
X_3	f	a^-b^+	$\top$

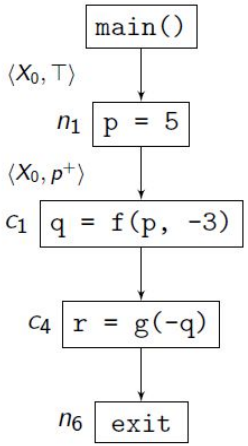
Value Contexts

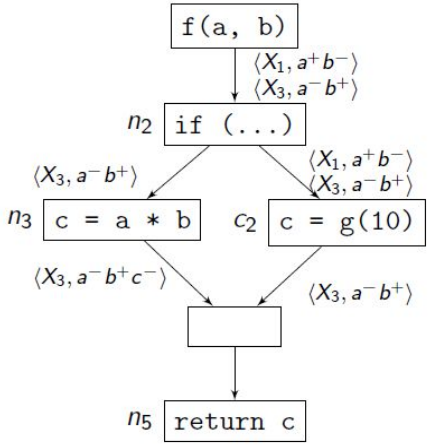


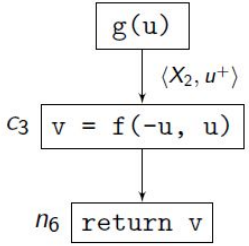
Context Transitions



Component Lattice

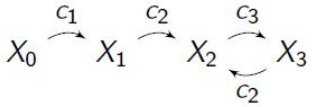




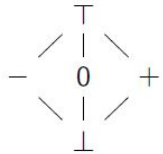


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	$\top$
X_3	f	a^-b^+	$\top$

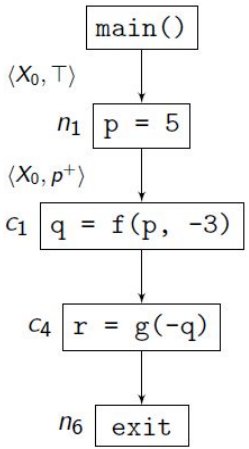
Value Contexts

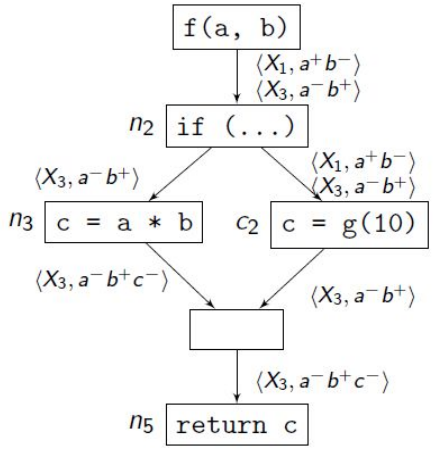


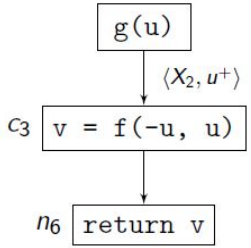
Context Transitions



Component Lattice

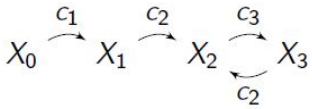




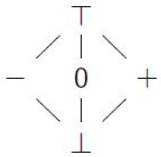


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	$\top$
X_3	f	a^-b^+	$\top$

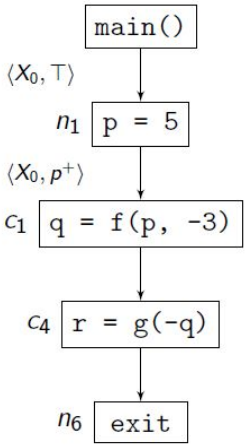
Value Contexts

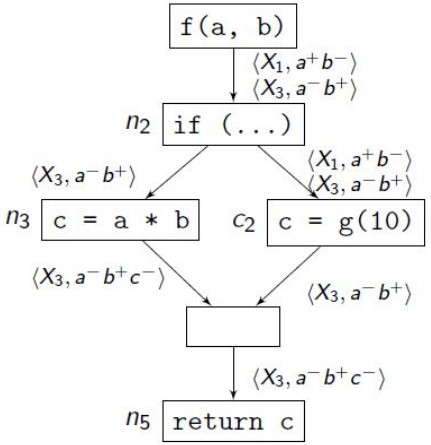


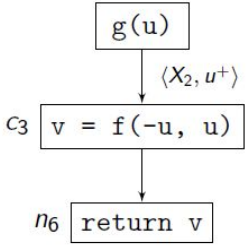
Context Transitions



Component Lattice

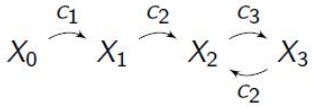




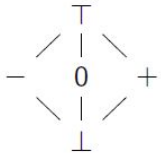


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	$\top$
X_3	f	a^-b^+	$a^-b^+c^-$

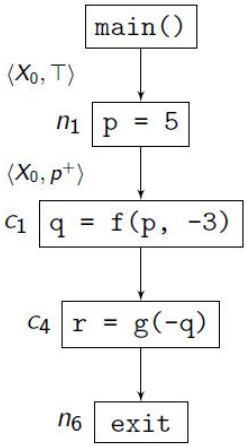
Value Contexts

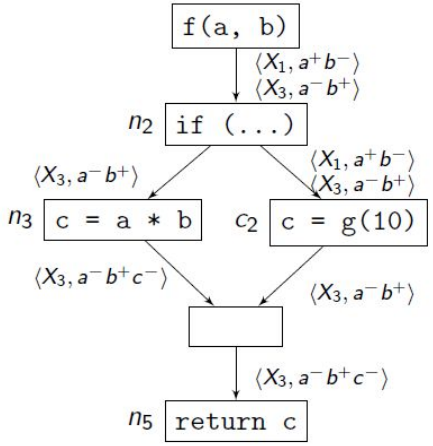


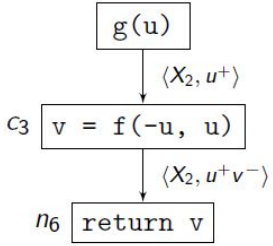
Context Transitions



Component Lattice

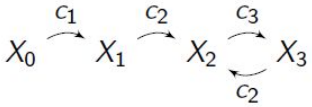




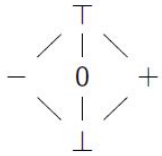


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	$\top$
X_3	f	a^-b^+	$a^-b^+c^-$

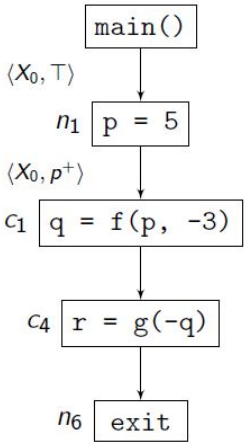
Value Contexts

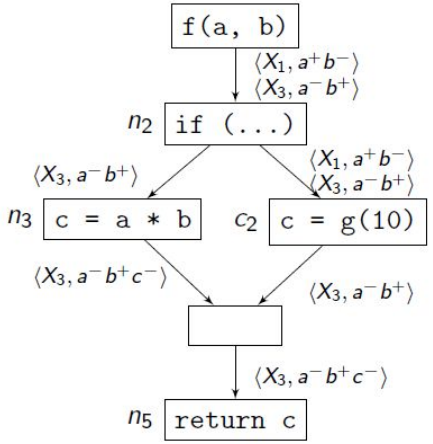


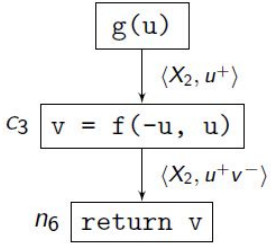
Context Transitions



Component Lattice

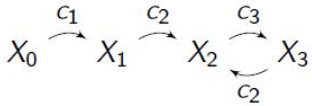




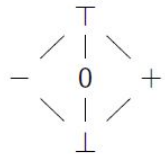


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	u^+v^-
X_3	f	a^-b^+	$a^-b^+c^-$

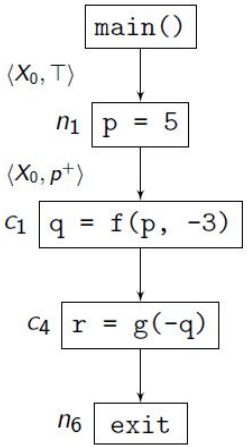
Value Contexts

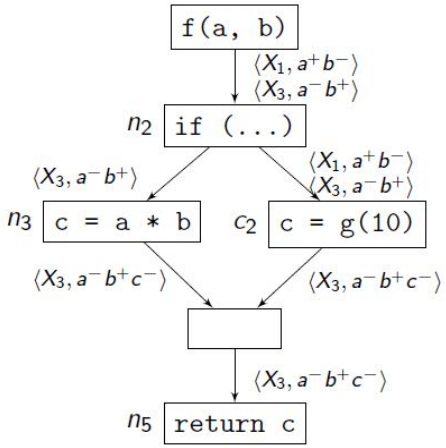


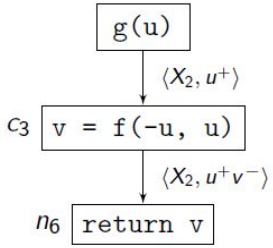
Context Transitions



Component Lattice

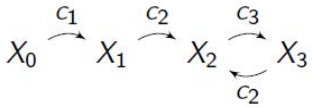




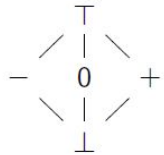


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	u^+v^-
X_3	f	a^-b^+	$a^-b^+c^-$

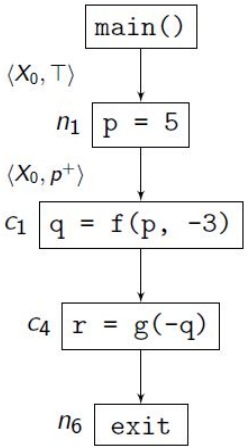
Value Contexts

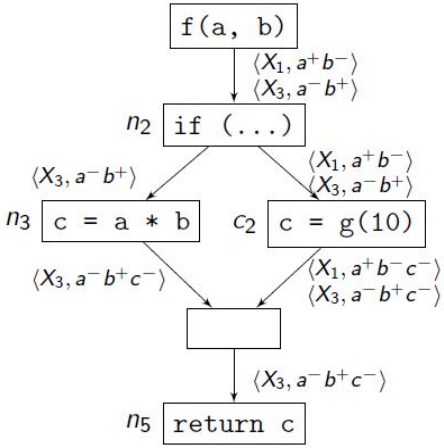


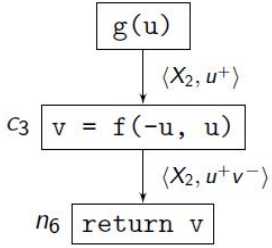
Context Transitions



Component Lattice

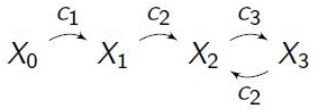




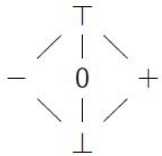


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	u^+v^-
X_3	f	a^-b^+	$a^-b^+c^-$

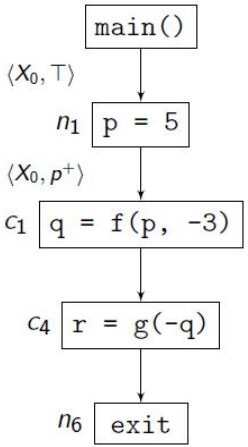
Value Contexts

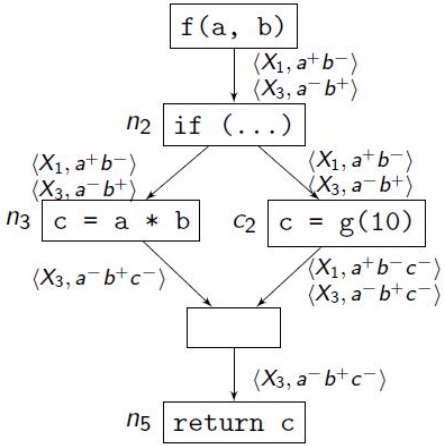


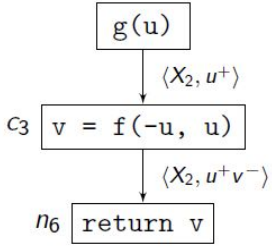
Context Transitions



Component Lattice

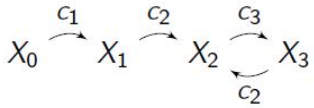




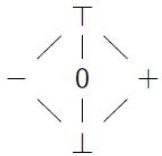


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	u^+v^-
X_3	f	a^-b^+	$a^-b^+c^-$

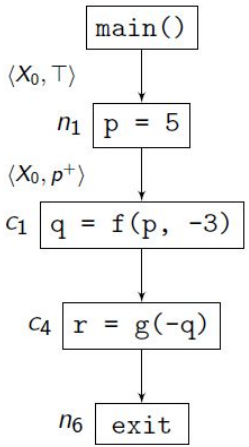
Value Contexts

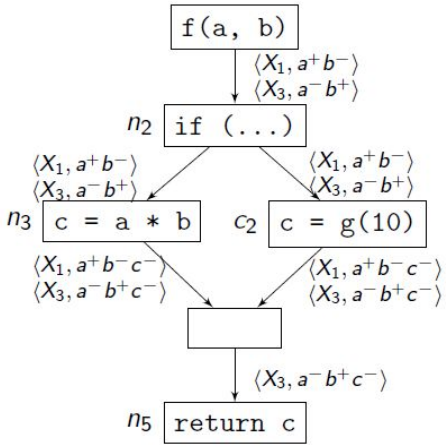


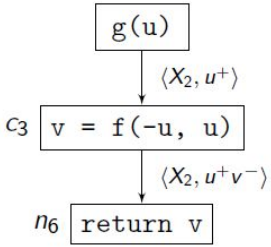
Context Transitions



Component Lattice

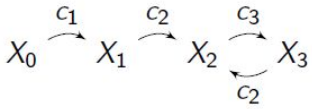




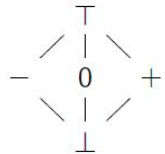


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	u^+v^-
X_3	f	a^-b^+	$a^-b^+c^-$

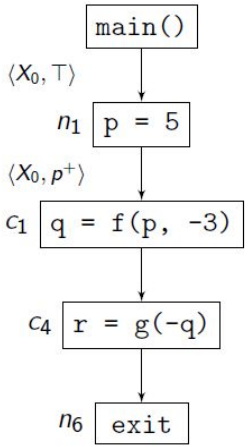
Value Contexts

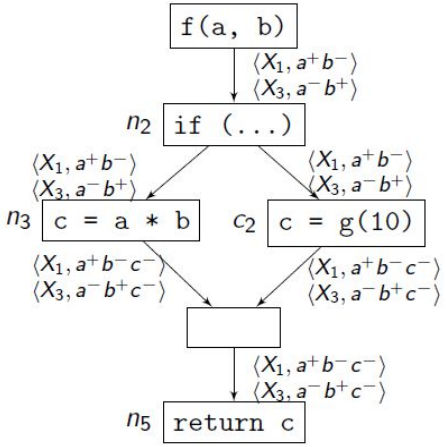


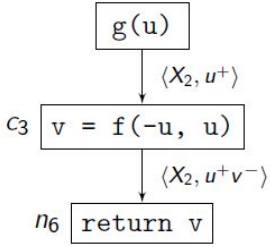
Context Transitions



Component Lattice

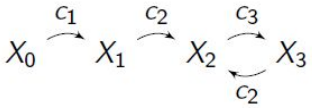




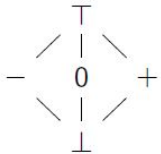


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$\top$
X_2	g	u^+	u^+v^-
X_3	f	a^-b^+	$a^-b^+c^-$

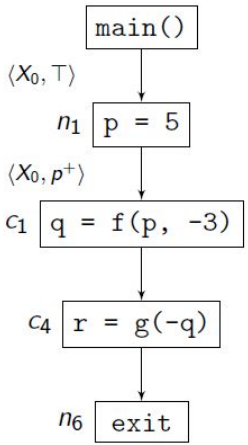
Value Contexts

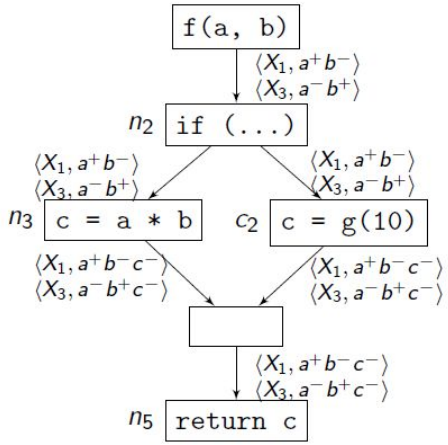


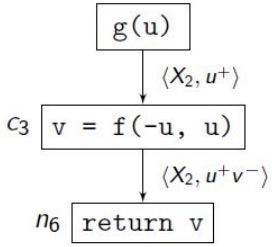
Context Transitions



Component Lattice

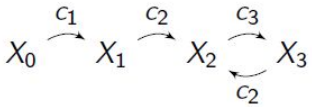




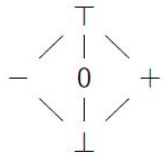


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$a^+b^-c^-$
X_2	g	u^+	u^+v^-
X_3	f	a^-b^+	$a^-b^+c^-$

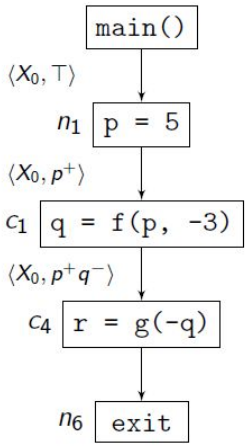
Value Contexts

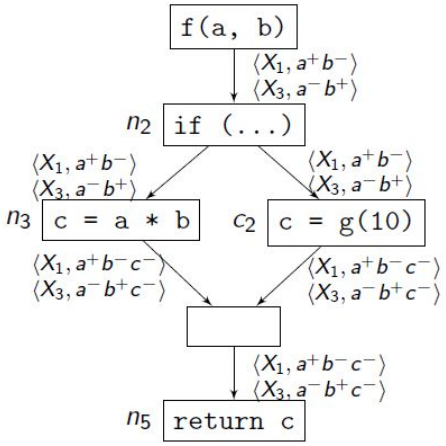


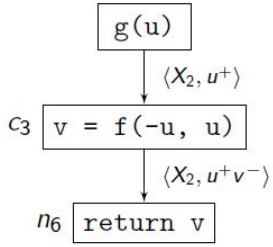
Context Transitions



Component Lattice

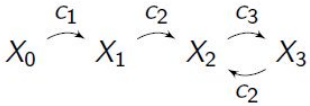




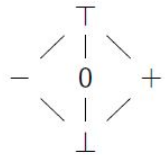


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$a^+b^-c^-$
X_2	g	u^+	u^+v^-
X_3	f	a^-b^+	$a^-b^+c^-$

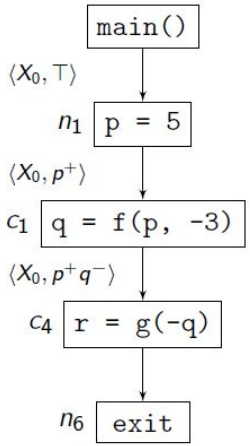
Value Contexts

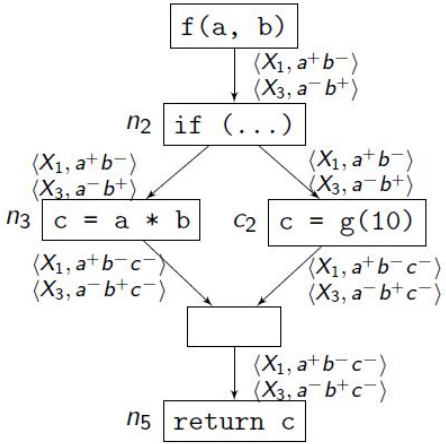


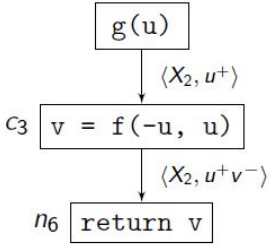
Context Transitions



Component Lattice

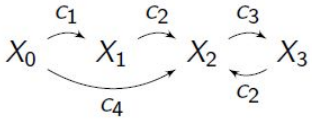




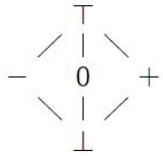


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$a^+b^-c^-$
X_2	g	u^+	u^+v^-
X_3	f	a^-b^+	$a^-b^+c^-$

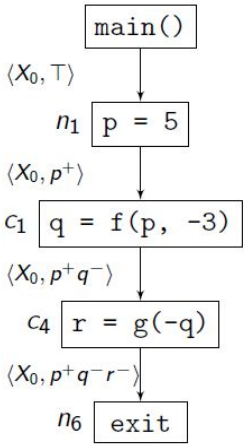
Value Contexts

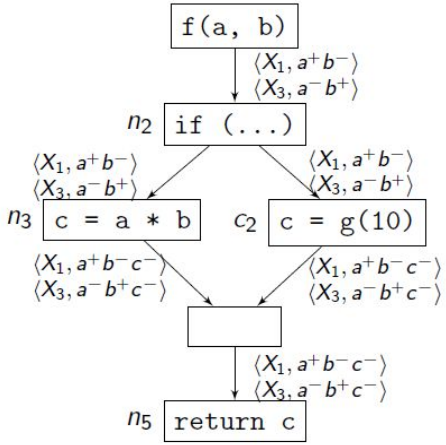


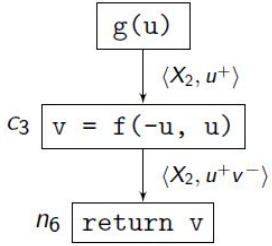
Context Transitions



Component Lattice

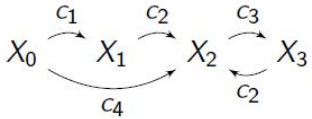




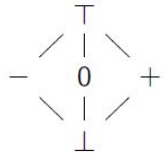


Context	Proc.	Entry	Exit
X_0	main	$\top$	$\top$
X_1	f	a^+b^-	$a^+b^-c^-$
X_2	g	u^+	u^+v^-
X_3	f	a^-b^+	$a^-b^+c^-$

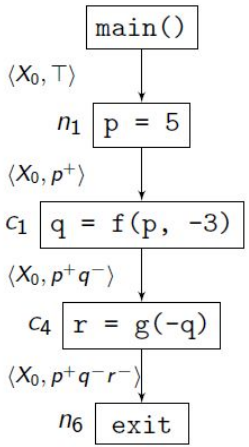
Value Contexts

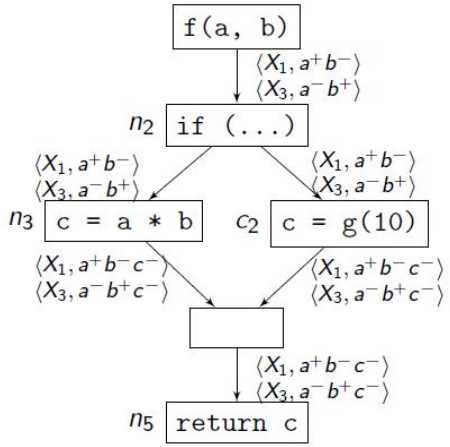


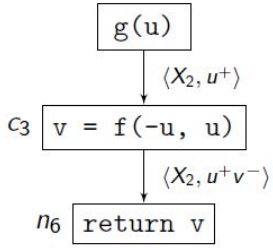
Context Transitions



Component Lattice

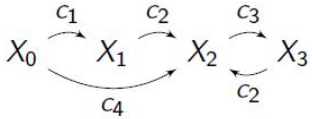




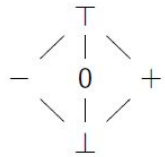


Context	Proc.	Entry	Exit
X_0	main	$\top$	$p^+q^-r^-$
X_1	f	a^+b^-	$a^+b^-c^-$
X_2	g	u^+	u^+v^-
X_3	f	a^-b^+	$a^-b^+c^-$

Value Contexts



Context Transitions



Component Lattice

Files in our framework (and what they do)

- Context.cpp
- ContextTransitionTable.cpp
- SignAnalysis.Sign.cpp
- SignAnalysis.cpp
- InterproceduralAnalysis.cpp
- ForwardInterproceduralAnalysis.cpp
- main.cpp

Context.cpp

- Defines a value based context Context <M,N,A>
  - M - the type of a method
  - N - the type of a node in the CFG
  - A - the type of a data flow value
- Each value context has its own work-list of CFG nodes to analyse, and the results of analysis are stored in a map from nodes to the data flow values before/after the node.

ContextTransitionTable.cpp

- Keeps a record of transitions between contexts at call-sites.
- The context transition table records a bidirectional one-to-many mapping of call-sites to called contexts parameterised by their methods. Defines a map from call-sites to contexts, parameterised by the called method.
  `std::unordered_map<CallSite<M, N, A>, std::unordered_map<M, Context<M, N, A>>> transitions`
  and a map of contexts to call-sites present within their method bodies is maintained.
  `std::unordered_map<Context<M, N, A>, std::vector<CallSite<M, N, A>>> call_sites_of_contexts`
- If a call-site transition is not traversed in an analysis (e.g. a call to a native method) then it is listed as a "default site" which this table also records.

InterProceduralAnalysis.cpp

- A generic inter-procedural analysis which is fully context-sensitive.
- Fully context sensitive even in the presence of recursion and uses data flow values reaching a method to distinguish contexts.
- Base file for forward_interprocedural_analysis.cpp

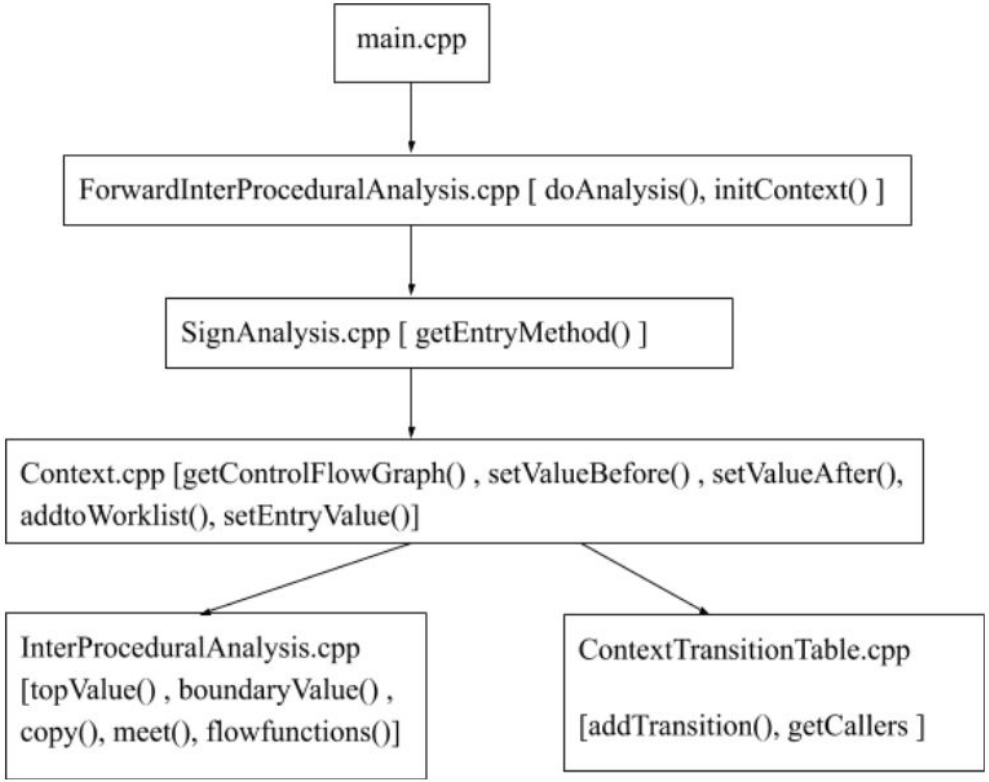
ForwardInterProceduralAnalysis.cpp

- This file essentially captures a forward data flow problem which can be solved using the context-sensitive inter-procedural analysis framework as described in Inter_procedural_analysis.cpp
- Client analyses will extend this file in order to perform forward-flow inter-procedural analysis.

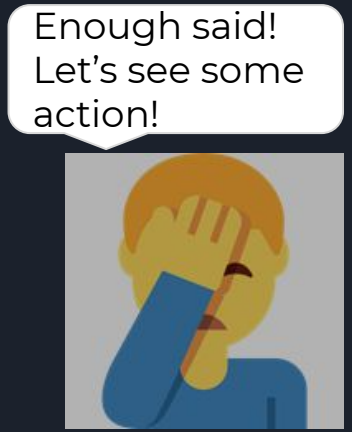
SignAnalysis.cpp

- An inter-procedural simplified sign analysis.
- Maps numeric variables to a sign (negative, positive or zero), if it is statically determined to be singular, or else bottom.
- Flow functions are non-distributive for statements involving sums or products of two variables.

Workflow


main.cpp
ForwardInterProceduralAnalysis.cpp [doAnalysis(), initContext()]
SignAnalysis.cpp [getEntryMethod()]
Context.cpp [getControlFlowGraph() , setValueBefore() , setValueAfter(), addtoWorklist(), setEntryValue()]
InterProceduralAnalysis.cpp [topValue() , boundaryValue() , copy(), meet(), flowfunctions()]
ContextTransitionTable.cpp [addTransition(), getCallers]

Demo

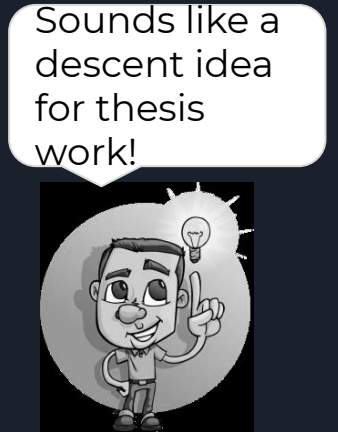


Test Results for Sign Analysis

```
===-------------------------------------------------------------------------===
                      ... Pass execution timing report ...
===-------------------------------------------------------------------------===
  Total Execution Time: 0.0116 seconds (0.0153 wall clock)

   ---User Time---   --System Time--   --User+System--   ---Wall Time---  --- Name ---
   0.0092 ( 96.7%)   0.0016 ( 75.2%)   0.0108 ( 92.8%)   0.0121 ( 79.1%)  Sign Analysis Pass
   0.0003 (  3.0%)   0.0005 ( 22.4%)   0.0008 (  6.5%)   0.0031 ( 20.4%)  Bitcode Writer
   0.0000 (  0.2%)   0.0000 (  1.8%)   0.0001 (  0.5%)   0.0001 (  0.4%)  Module Verifier
   0.0000 (  0.1%)   0.0000 (  0.7%)   0.0000 (  0.2%)   0.0000 (  0.1%)  Assign names to anonymous instructions
   0.0095 (100.0%)   0.0021 (100.0%)   0.0116 (100.0%)   0.0153 (100.0%)  Total

===-------------------------------------------------------------------------===
                                LLVM IR Parsing
===-------------------------------------------------------------------------===
  Total Execution Time: 0.0012 seconds (0.0059 wall clock)

   ---User Time---   --System Time--   --User+System--   ---Wall Time---  --- Name ---
   0.0003 (100.0%)   0.0009 (100.0%)   0.0012 (100.0%)   0.0059 (100.0%)  Parse IR
   0.0003 (100.0%)   0.0009 (100.0%)   0.0012 (100.0%)   0.0059 (100.0%)  Total
```

Future work

- Backward interprocedural analysis
- Fully context-sensitive points-to analysis
- Interprocedural heap analysis



Thank You

(For being an amazing instructor)

We really enjoyed the course.🙂

- Deeksha Arora (20111017)
- Sambhrant Maurya (20111054)
- Shruti Sharma (20111061)