

Context-Sensitive Interprocedural Analysis Algorithm

17-355/17-665/17-819: Program Analysis (Spring 2025)

Rohan Padhye

```
type Context
  val fn : Function                                ▷ the function being called
  val input :  $\sigma$                                 ▷ input for this set of calls

type Summary                                       ▷ the input/output summary for a context
  val input :  $\sigma$ 
  val output :  $\sigma$ 

val worklist : Set[Context]    ▷ contexts we must revisit due to updated analysis information
val analyzing : Set[Context]    ▷ the contexts we are currently analyzing
val results : Map[Context, Summary]    ▷ the analysis results
val callers : Map[Context, Set[Context]]    ▷ the call graph - used for change propagation

function GETCTX(f, callingCtx, n,  $\sigma_{in}$ )
  return Context(f,  $\sigma_{in}$ )                                ▷ constructs a new Context with f and  $\sigma_{in}$ 
end function

function ANALYZEPROGRAM                                ▷ starting point for interprocedural analysis
  initCtx  $\leftarrow$  GETCTX(main, nil, 0,  $\top$ )
  worklist  $\leftarrow$  {initCtx}
  results[initCtx]  $\leftarrow$  Summary( $\top$ ,  $\perp$ )
  while NOTEMPTY(worklist) do
    ctx  $\leftarrow$  REMOVE(worklist)
    ANALYZE(ctx, results[ctx].input)
  end while
end function

function ANALYZE(ctx,  $\sigma_{in}$ )
   $\sigma_{out}$   $\leftarrow$  results[ctx].output
  ADD(analyzing, ctx)
   $\sigma'_{out}$   $\leftarrow$  INTRAPROCEDURAL(ctx,  $\sigma_{in}$ )
  REMOVE(analyzing, ctx)
  if  $\sigma'_{out} \not\sqsubseteq \sigma_{out}$  then
    results[ctx]  $\leftarrow$  Summary( $\sigma_{in}$ ,  $\sigma_{out} \sqcup \sigma'_{out}$ )
    for c  $\in$  callers[ctx] do
      ADD(worklist, c)
    end for
  end if
  return  $\sigma'_{out}$ 
end function
```

function FLOW($\llbracket n: x := f(y) \rrbracket, ctx, \sigma_n$) ▷ called by intraprocedural analysis
 $\sigma_{in} \leftarrow [formal(f) \mapsto \sigma_n(y)]$ ▷ map f 's formal parameter to info on actual from σ_n
 $calleeCtx \leftarrow GETCTX(f, ctx, n, \sigma_{in})$
 $\sigma_{out} \leftarrow RESULTSFOR(calleeCtx, \sigma_{in})$
 $ADD(callers[calleeCtx], ctx)$
return $\sigma_n[x \mapsto \sigma_{out}[result]]$ ▷ update dataflow with the function's result
end function

function RESULTSFOR(ctx, σ_{in})
if $ctx \in \text{dom}(results)$ **then**
 if $\sigma_{in} \sqsubseteq results[ctx].input$ **then**
 return $results[ctx].output$ ▷ existing results are good
 else
 $results[ctx].input \leftarrow results[ctx].input \sqcup \sigma_{in}$ ▷ keep track of more general input
 end if
else
 $results[ctx] = Summary(\sigma_{in}, \perp)$ ▷ initially optimistic assumption
end if
if $ctx \in analyzing$ **then**
 return $results[ctx].output$ ▷ \perp if it hasn't been analyzed yet; otherwise last known
else
 return ANALYZE($ctx, results[ctx].input$)
end if
end function