Interprocedural Analysis: Sharir-Pnueli’s Call-strings Approach

Deepak D’Souza

Department of Computer Science and Automation
Indian Institute of Science, Bangalore.

06 October 2010
Call strings approach

- For a given program $P$ and analysis $((D, \leq), f_{MN}, d_0)$, the join over all interprocedurally valid paths (JVP) at point $N$ is defined to be:

$$\bigcup_{\rho \in IVP(r_1,N)} f_{\rho}(d_0).$$

- Idea: collect data values that reach each point, tagged with call-string of associated path.
- This helps to say which values pass to a given return site.
- Now we can set up equations that capture JVP values.
Call-string along an interprocedurally valid path

- Call-string associated with an IVP path $\rho$, denoted $CM(\rho)$, is the sequence of pending calls in $\rho$.
- A path $\rho$ in $IVP(r_1, I)$ for example program:
  
  ![Call-string Diagram](image)

  - Associated call-string $CM(\rho)$ is $c_1$. 
Call-string along an interprocedurally valid path

- Call-string associated with an IVP path $\rho$, denoted $CM(\rho)$, is the sequence of pending calls in $\rho$.
- A path $\rho$ in $IVP(r_1, I)$ for example program:

  ![Diagram showing a sequence of letters representing a path in a program]

  Associated call-string $CM(\rho)$ is $c_1$.
- For $\rho' = ABCOFGHLF \quad CM(\rho') = c_1c_2$.
- Denote set of all call-strings for given program by $\Gamma$. 
Tagging with call-strings

- Classify paths reaching $N$ according to call-strings.
- For each call-string $\gamma$ maintain data value

$$d = \bigsqcup_{\rho \in CM^{-1}(\gamma)} f_\rho(d_0).$$

- Thus elements of $L^*$ are maps $\xi : \Gamma \to D$, and ordering $\xi_1 \leq \xi_2$ is pointwise extension of $\leq$ in $D$.
- Tagged JVP value: $\xi^*_N : \gamma \mapsto \bigsqcup_{\rho \in CM^{-1}(\gamma)} f_\rho(d_0)$.
- JVP value $d_N = \bigsqcup_{\gamma \in \Gamma} \xi^*_N(\gamma)$. 
Example: Tagging

Eg: Path ABCOFGHJK has associated callstring c₁c₂.

\[
\begin{array}{c|c|c}
\gamma & c_1 & c_1c_2 \\
\hline
\varepsilon & & \\
\xi(\gamma) & \cdot & \cdot \\
\hline
\end{array}
\]

Tagged data values at J for availability of a*b analysis

\[
\begin{array}{c|c|c|c|c}
\gamma & c_1 & c_1c_2 & c_1c_2c_2 \\
\hline
\varepsilon & & & \\
\bot & 1 & 0 & 0 \\
\xi(\gamma) & & & \\
\hline
\end{array}
\]
Let $D^* = \Gamma \rightarrow D$.

Pointwise ordering on $D^*$

- $\xi \preceq' \xi'$ iff $\xi(\gamma) \leq \xi'(\gamma)$ for each call-string $\gamma$.

$(D^*, \preceq')$ is also a complete lattice.

Initial value $\xi_0$ is given by

$$\xi_0(\gamma) = \begin{cases} 
  d_0 & \text{if } \gamma = \epsilon \\
  \perp & \text{otherwise}
\end{cases}$$

Transfer functions for non call/ret nodes: $f_{MN}^* = \lambda \xi. f_{MN} \circ \xi$.

Transfer functions $f_{MN}^*$'s are monotonic (distributive) if $f_{MN}$'s are monotonic (distributive).
Transfer functions $f_{MN}^*$ by example

- (Non-call/ret node)
  \[ \xi_C = f_{BC} \circ \xi_B. \]

- (Call node)
  \[ \xi_F(\gamma) = \begin{cases} \xi_C(\gamma') & \text{if } \gamma = \gamma' \cdot c_1 \\ \bot & \text{otherwise} \end{cases} \]

- (Return site)
  \[ \xi_P(\gamma) = \xi_J(\gamma \cdot c_1). \]
Correctness claims

Claim

Let the LFP of the analysis \(((D^*, \leq'), f_{MN}^*, \xi_0)\) be \(\xi^*\). Then

\[ x^*_N = \bigsqcup_{\gamma \in \Gamma} \xi^*_N(\gamma) \]

is an over-approximation of the JVP at \(N\). When \(f_{MN}\)'s are distributive \(x^*_N\) coincides with JVN at \(N\).
Exercise

Use Kildall’s algo to compute the $\xi$ table values for the example program, for $|\gamma| \leq 4$. Start with initial value $d_0 = 0$. 

```
a := a−1
F
G
1

11

ret

D

I

J

K

```

```
t := a*b
C

2

3

P

call p

O

L

M

```

```
t := a*b
D

4

```

```
print t
E

5

```

```
read a,b
B

1

```

```
a == 0
G

6

```

```
a := a−1
H

7

```

```
call p
Q

8

```

```
t := a*b
I

9

```

```
```
Exercise

Use Kildall’s algo to compute the $\xi$ table values for the example program, for $|\gamma| \leq 4$. Start with initial value $d_0 = 0$. 

```
a := a-1
F

G

t := a*b
1

read a,b

C

c_1

call p

c
P

I

O

L

M

J

K

N

D

P

ret

print t

10

ret

9

a := a-1

H

Q

a == 0

G

6

F

7

8

9

11
```
Exercise

Use Kildall’s algo to compute the $\xi$ table values for the example program, for $|\gamma| \leq 4$. Start with initial value $d_0 = 0$. 

```
a := a-1
F
G
t := a*b
1
read a,b
A
B

2
t := a*b
C

c_1

3
call p
P

4
t := a*b
D

5
print t
E

6
a == 0
F
G

7
a := a-1
H

8
c_2

9
call p
Q

10
t := a*b
I

11
ret
J
```
Use Kildall’s algo to compute the $\xi$ table values for the example program, for $|\gamma| \leq 4$. Start with initial value $d_0 = 0$. 
Use Kildall’s algo to compute the $\xi$ table values for the example program, for $|\gamma| \leq 4$. Start with initial value $d_0 = 0$. 

```plaintext
read a, b

1. \( t := a \times b \)

2. \( \varepsilon_0 \)

3. \( \varepsilon_1 \)

4. \( \varepsilon_1 \)

5. \( \text{print } t \)

6. \( \text{call } p \)

7. \( \text{call } p \)

8. \( \text{a := a-1} \)

9. \( \text{call } p \)

10. \( \text{t := a*b} \)

11. \( \text{ret} \)
```
Convergence of iteration

- Lattice \((D^*, \leq')\) is infinite for recursive programs.
- It is possible to bound the size of call strings \(\Gamma\) we need to consider.
- Let \(k\) be the number of call sites in \(P\).
**Convergence of iteration**

**Claim**

For any path $p$ with a prefix $q$ such that $|CM(q)| > k|D|^2 = M$ there is a path $p'$ with $|CM(q')| \leq M$ for each prefix $q'$ of $p'$, and $f_p(d_0) = f_{p'}(d_0)$.

**Paths with bounded call-strings**

Proof follows shortly.
Go over to a finite lattice.
Consider only call strings of length \( \leq M \) (Call this \( \Gamma_M \)).

\[ IVP_{\Gamma_M}(r_1, N) = \text{paths from } r_1 \text{ to } N \text{ such that for each prefix } q, CM(q) \leq M. \]
Data-flow analysis for JVP over $IVP_{\Gamma_M}$

- **(Non-call/ret node)**
  \[
  \xi_C = f_{BC} \circ \xi_B.
  \]

- **(Call node)**
  \[
  \xi_F(\gamma) = \begin{cases} 
  \xi_C(\gamma') & \text{if } \gamma = \gamma' \cdot c_1 \\
  \bot & \text{and } \gamma \in \Gamma_M \\
  \end{cases}
  \]

- **(Return site)**
  \[
  \xi_P(\gamma) = \xi_J(\gamma \cdot c_1).
  \]

```
read a, b
read a, b

call p
call p

t := a * b
t := a * b

t := a * b
print t
print t

c := 1
c := 1

c := 1
ret
ret
```

---

```
a := a - 1
```

---

```
a == 0
```
Bounding call-string size

**Claim**

For any path $p$ in $IVP(r_1, N)$ such that $|CM(q)| > M = k|D|^2$ for some prefix $q$ of $p$, there is a path $p'$ in $IVP_{\Gamma_M}(r_1, N)$ with $f_{p'}(d_0) = f_p(d_0)$.

- Sufficient to prove:

**Subclaim**

For any path $p$ in $IVP(r_1, N)$ with a prefix $q$ such that $|CM(q)| > M$, we can produce a smaller path $p'$ in $IVP(r_1, N)$ with $f_{p'}(d_0) = f_p(d_0)$.

- ...since if $|p| \leq M$ then $p \in IVP_{\Gamma_M}$.
A path $\rho$ in $IVP(r_1, n)$ can be decomposed as

$$
\rho_1 \parallel (c_1, r_{p_2}) \parallel \rho_2 \parallel (c_2, r_{p_3}) \parallel \sigma_3 \parallel \cdots \parallel (c_{j-1}, r_{p_j}) \parallel \rho_j.
$$

where each $\rho_i$ ($i < j$) is a valid and complete path from $r_{p_i}$ to $c_i$, and $\rho_j$ is a valid and complete path from $r_{p_j}$ to $n$. Thus $c_1, \ldots, c_j$ are the unfinished calls at the end of $\rho$. 

![Diagram showing path decomposition](image)
Let $p_0$ be the first prefix of $p$ where $|CM| > M$.

Let decomposition of $p_0$ be

$$\rho_1 \| (c_1, r_{p_2}) \| \rho_2 \| (c_2, r_{p_3}) \| \sigma_3 \| \cdots \| (c_{j-1}, r_{p_j}) \| \rho_j.$$

Tag each unfinished-call $c_i$ in $p_0$ by $(c_i, f_{q \cdot c_i}(d_0), f_{q \cdot c_i q' e_{i+1}}$) where $e_{i+1}$ is corresponding return of $c_i$ in $p$.

If no return for $c_i$ in $p$ tag with $(c, f_{q \cdot c_i}(d_0), \bot)$.

Number of distinct such tags is $k \cdot |D|^2$.

So there are two calls $qc$ and $qcq'c$ with same tag values.
Proving subclaim – tag values are \(\perp\)
Proving subclaim – tag values are not $\perp$
Example

A
read a, b

B

C
call p2

D
return

E
t := a * b

F

call p1

G

H

I
a := 0

J

K
call p1

L

M

N
return