C|ISPA Center for IT-Security, Privacy and Accountability

UNIVERSITÄT DES SAARLANDES

Interprocedural Data Flow Analysis

Static Program Analysis

Christian Hammer

18. Juni 2014

SPONSORED BY THE

Federal Ministry of Education and Research

max planck institut informatik

German Research Center for Artificial Intelligence

Max Planck Institute for Software Systems

Interprocedural Reaching Definitions

```
int a, b, c;

void q () {
  int z=1;
  a=2;
  b=3;
  p(4, z);
  z=a;
  c=5;
  p(6, c);
}
```

```
void p(int x,int &y) {
  static int d = 6;
  a=c;
  if(x) {
    d=7;
    p(8, x);
  } else {
    b=9;
  }
  y =0;
}
```

Interprocedural Reaching Definitions

```
int a, b, c;

void q () {
  int z=1;
  a=2;
  b=3;
  p(4, z);
  z=a;
  c=5;
  p(6, c);
}
void p(int x,int &y) {
  static int d = 6;
  a=c;
  if(x) {
    d=7;
    p(8, x);
  } else {
    b=9;
  }
  y =0;
}
```

Interprocedural Reaching Definitions

CISPA Center for IT-Security, Privacy and Accountability

```
(1)  int a, b, c;

(3)  void q () {
(4)    int z=1;
(5)    a=2;
(6)    b=3;
(7)    p(4, z);
(8)    z=a;
(9)    c=5;
(10)   p(6, c);
(11) }
```

```
(12) void p(int x,int &y) {
(13)   static int d = 6;
(14)   a=c;
(15)   if(x) {
(16)     d=7;
(17)     p(8, x);
(18)   } else {
(19)     b=9;
(20)   }
(21)   y =0;
(22) }
```

call-by-value

call-by-reference

Interprocedural Reaching Definitions

Global Variables

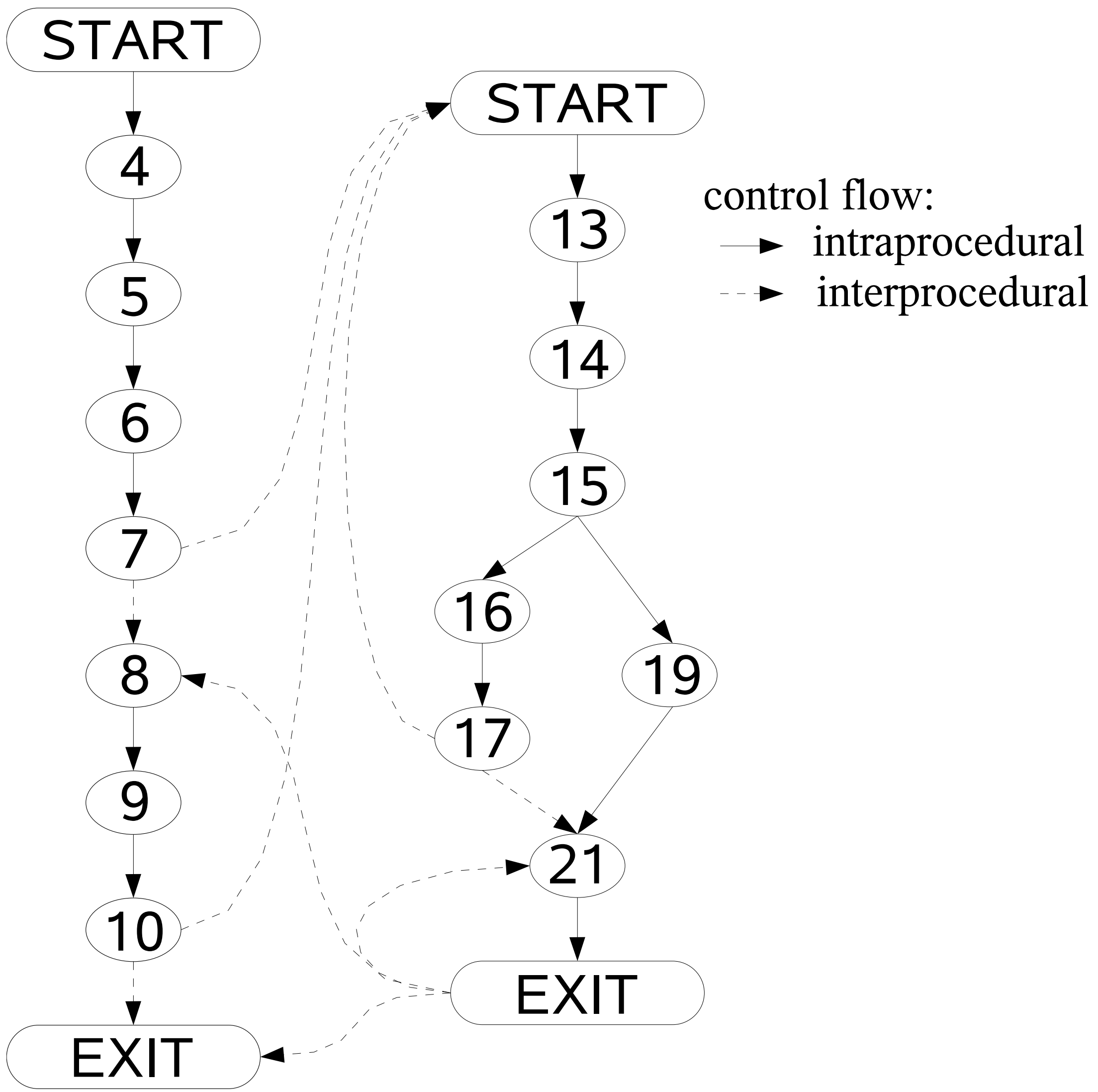
```
(1)  int a, b, c;

(3)  void q () {
(4)    int z=1;
(5)    a=2;
(6)    b=3;
(7)    p(4, z);
(8)    z=a;
(9)    c=5;
(10)   p(6, c);
(11) }
```

```
(12) void p(int x,int &y) {
(13)   static int d = 6;
(14)   a=c;
(15)   if(x) {
(16)     d=7;
(17)     p(8, x);
(18)   } else {
(19)     b=9;
(20)   }
(21)   y =0;
(22) }
```

- definition of a (line 5) reaches 6–7 but not 8–11 (killed by 14 through call in 7), also reaches 13–14 but not 15–22
- definition of c in line 9 reaches 13–22 through call in line 10.

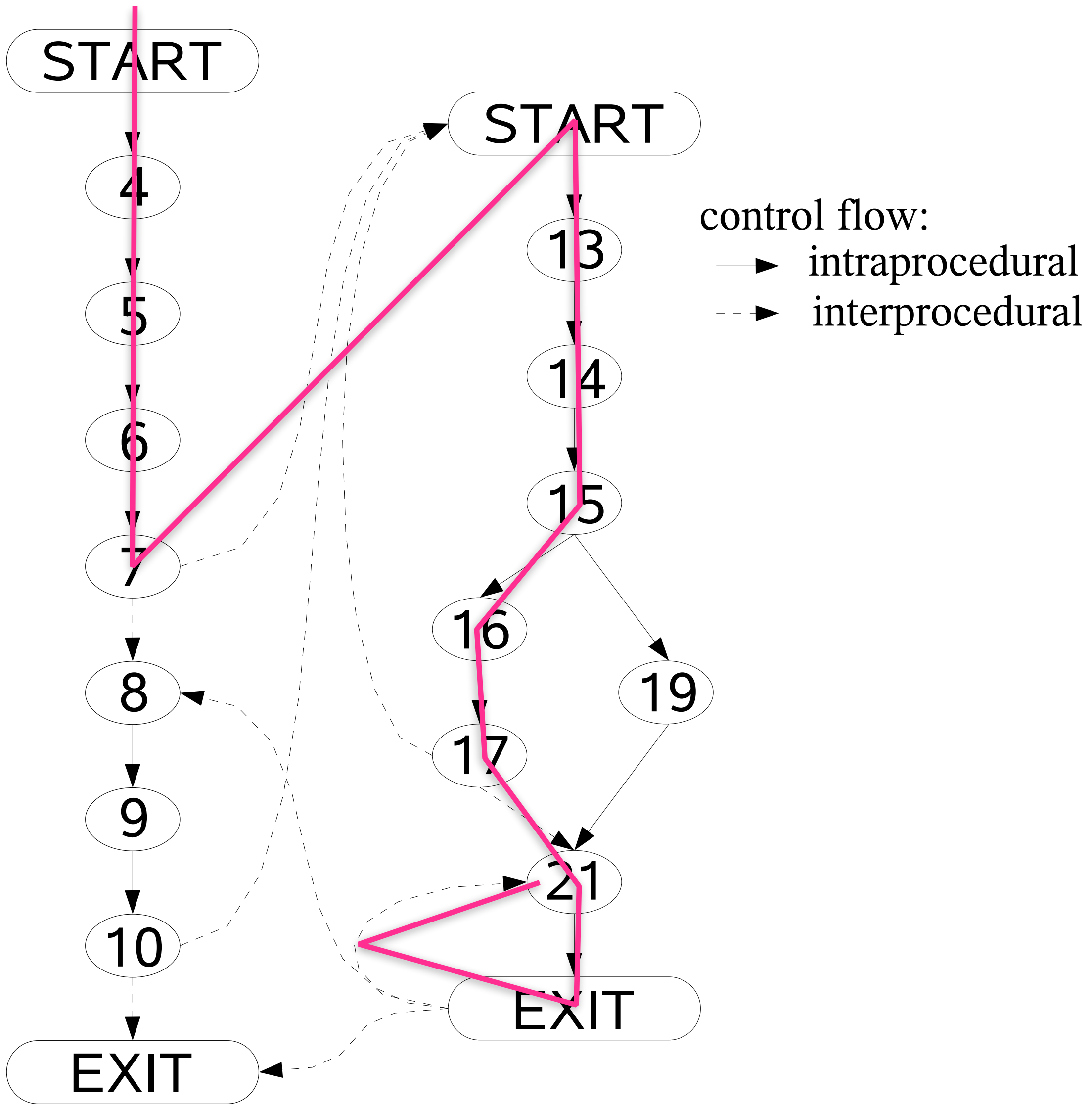
- More complex are the definitions of global b: the definition in line 6 can- not reach lines 8–10 or 21, as line 19 kills it —any call of p must execute line 19 to terminate the recursion. Also, the definition in line 19 reaches line 13–19, as it might reach the call in line 10 by procedure p returning from the call in line 7.
- The variable d is global and only visible inside procedure p. the definition in line 16 may reach lines 13–16 because of the call in line 17. Through procedure p returning from the call in line 7, both definitions (line 13 and 16) may reach lines 8–10 and therefore also line 13–16 and 18–22.
- Locals like z are (usually) only visible in procedures they are defined in. Call-by-value parameters are like locals, with a definition at the proce- dure entry: x is defined in line 12.
- Call-by-reference introduces a simple form of aliasing and make other- wise invisible variables available in called procedures.

Interprocedural Control Flow Graph



- *Control Flow Graph* $G_p = (N_p, E_p, n^s{}_p, n^e{}_p)$ for each procedure *p*. An interprocedural control flow graph (ICFG) is a directed graph $G = (N^*, E^*, n^s{}_0, n^e{}_0)$, where $N^* = \bigcup_p N_p$ and $E^* = E^C \cup \bigcup_p E_p$
- *call* and *return edges* in E^C: A call edge e $\in E^C$ is going from a call node $n \in N_p$ to the START node $n^s{}_q$ of the called procedure q. A return edge e $\in E^C$ is going from the EXIT node $n^e{}_q$ of the called procedure q back to the immediate successor of the call node $n \in N_p$
- *unrealizable paths* possible if leaving a function on a different node than the call's successor

Interprocedural Control Flow Graph



Inter-procedurally Realizable Path

CISPA
Center for IT-Security, Privacy and Accountability

Edges from E_p are marked with the empty word ε and edges from E^C are marked according to their source and target nodes

return edge

entry node

$$
\begin{aligned}
M &\rightarrow MM \\
&\mid\ (^{n_p^s}_{m}\, M\,)^{n_p^s}_{m} \qquad \forall (^{n_p^s}_{m} \in \Sigma \\
&\mid\ \epsilon
\end{aligned}
$$

call edge

call node

$$
\begin{aligned}
R &\rightarrow MR \\
&\mid\ (^{n_p^s}_{m}\, R \qquad \forall (^{n_p^s}_{m} \in \Sigma \\
&\mid\ \epsilon
\end{aligned}
$$

$$
\begin{aligned}
L &\rightarrow LM \\
&\mid\ L\,)^{n_p^s}_{m} \qquad \forall (^{n_p^s}_{m} \in \Sigma \\
&\mid\ \epsilon
\end{aligned}
$$

$$
\begin{aligned}
I &\rightarrow L \\
&\mid\ R
\end{aligned}
$$

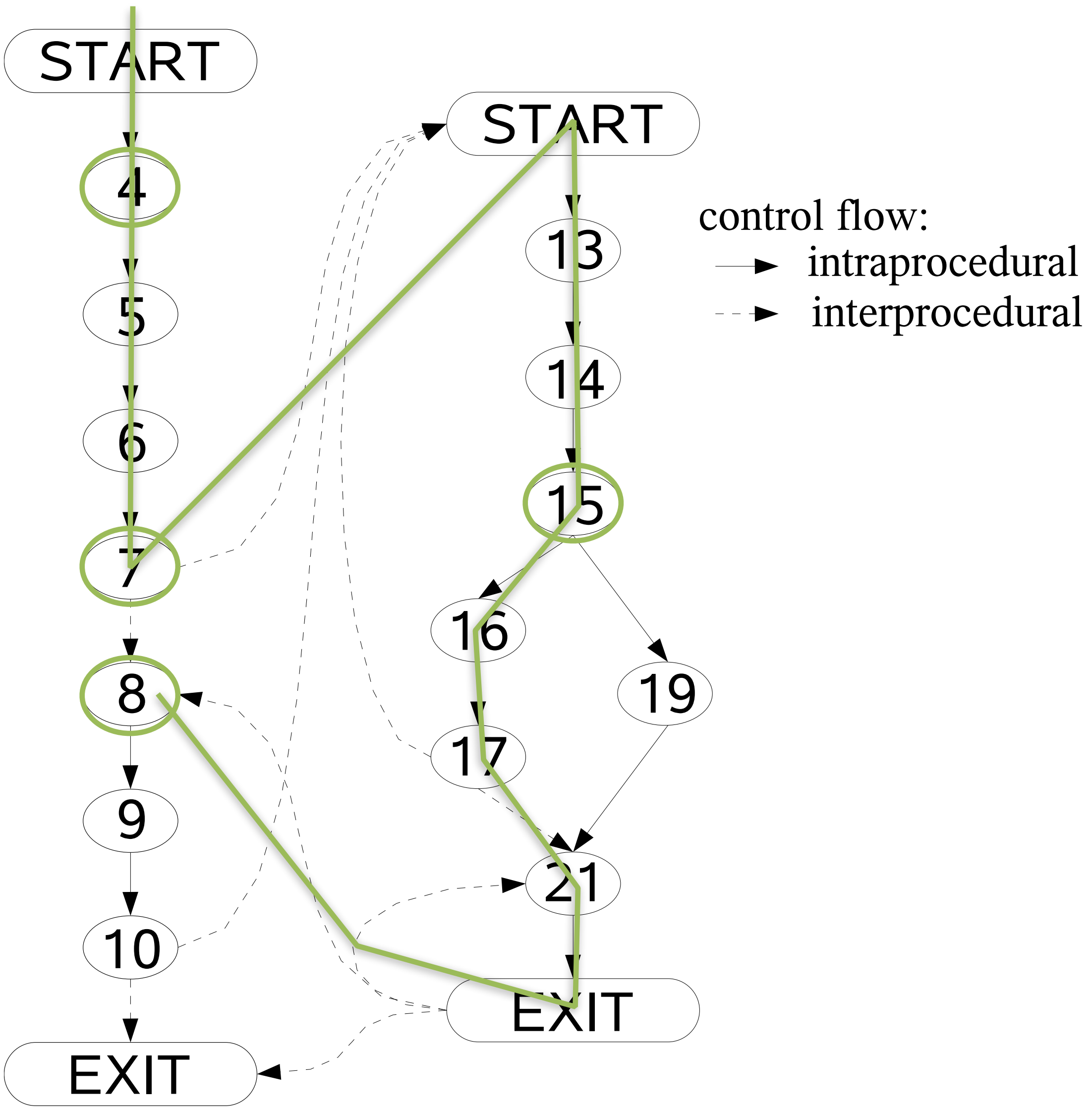
An interprocedurally realizable path I is an interprocedurally right- or left-balanced path.

Interprocedural Reachability, Witness

CISPA
Center for IT-Security, Privacy and Accountability

- A node n is interprocedurally reachable from node m, iff an interprocedurally realizable path from m to n in the ICFG exists, written as $m \rightharpoonup^*_R n$.
- A sequence $\langle n_1, \ldots, n_k \rangle$ of nodes is called an interprocedurally (realizable) witness, iff n_k is interprocedurally reachable from n_1 via an interprocedurally realizable path $p = \langle m_1, \ldots, m_l \rangle$ with:
  - $m_1 = n_1$, $m_l = n_k$, and
  - $\forall 1 \leq i < k\ \exists x,y\colon x<y \wedge m_x = n_i \wedge m_y = n_i + 1$.

Interprocedural Control Flow Graph



Interprocedural Control Flow Graph