Introduction to Data Flow Analysis

Reading: NNH 1.1-1.3, 1.7-1.8

17-654/17-765
Analysis of Software Artifacts
Jonathan Aldrich

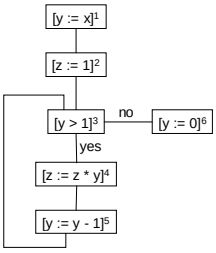
Example WHILE Program

$[y := x]^1$;
$[z := 1]^2$;
while $[y>1]^3$ do
  $[z := z * y]^4$;
  $[y := y - 1]^5$;
$[y := 0]^6$;

Computes the factorial function, with the input in x and the output in z

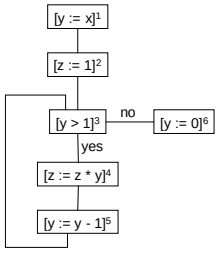
Data Flow Analysis

- View program as graph
  - Nodes are elementary blocks like assignments, if statements, etc.
  - Edges show control flow



Data Flow Equations (1)

- Transfer Functions
  - show how a statement affects data flow information

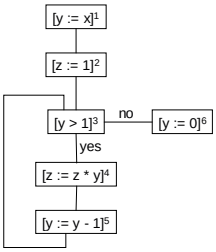


$RD_{exit}(1) = (RD_{entry}(1) \setminus \{(y,*)\}) \cup (y,1)$
$RD_{exit}(2) = (RD_{entry}(2) \setminus \{(z,*)\}) \cup (z,2)$
$RD_{exit}(3) = RD_{entry}(3)$
$RD_{exit}(4) = (RD_{entry}(4) \setminus \{(z,*)\}) \cup (z,4)$
$RD_{exit}(5) = (RD_{entry}(5) \setminus \{(y,*)\}) \cup (y,5)$
$RD_{exit}(6) = (RD_{entry}(6) \setminus \{(y,*)\}) \cup (y,6)$

Data Flow Equations (1)

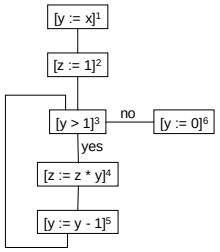
- Pattern
  - Assignments
    - **kill** reaching defs for that var
    - **generate** new reaching def
  - All others
    - no change

$RD_{exit}(1) = (RD_{entry}(1) \setminus \{(y,*)\}) \cup (y,1)$
$RD_{exit}(2) = (RD_{entry}(2) \setminus \{(z,*)\}) \cup (z,2)$
$RD_{exit}(3) = RD_{entry}(3)$
$RD_{exit}(4) = (RD_{entry}(4) \setminus \{(z,*)\}) \cup (z,4)$
$RD_{exit}(5) = (RD_{entry}(5) \setminus \{(y,*)\}) \cup (y,5)$
$RD_{exit}(6) = (RD_{entry}(6) \setminus \{(y,*)\}) \cup (y,6)$



Data Flow Equations (2)

- Flow equations
  - Show how analysis information flows from one statement to another



$RD_{entry}(1) = \{ (x,?), (y,?), (z,?) \}$
$RD_{entry}(2) = RD_{exit}(1)$
$RD_{entry}(3) = RD_{exit}(2) \cup RD_{exit}(5)$
$RD_{entry}(4) = RD_{exit}(3)$
$RD_{entry}(5) = RD_{exit}(4)$
$RD_{entry}(6) = RD_{exit}(3)$

Data Flow Solution

- Solution
  - A 12-tuple $\overline{RD}$
  - Such that $\overline{RD} = F(\overline{RD})$
  - Where F is derived from the equations at right
  - As small (precise) as possible
- *Fixed point* of f
  - A value v such that $v = f(v)$

$RD_{exit}(1) = (RD_{entry}(1) \setminus \{(y,*)\}) \cup (y,1)$
$RD_{exit}(2) = (RD_{entry}(2) \setminus \{(z,*)\}) \cup (z,2)$
$RD_{exit}(3) = RD_{entry}(3)$
$RD_{exit}(4) = (RD_{entry}(4) \setminus \{(z,*)\}) \cup (z,4)$
$RD_{exit}(5) = (RD_{entry}(5) \setminus \{(y,*)\}) \cup (y,5)$
$RD_{exit}(6) = (RD_{entry}(6) \setminus \{(y,*)\}) \cup (y,6)$

$RD_{entry}(1) = \{ (x,?), (y,?), (z,?) \}$
$RD_{entry}(2) = RD_{exit}(1)$
$RD_{entry}(3) = RD_{exit}(2) \cup RD_{exit}(5)$
$RD_{entry}(4) = RD_{exit}(3)$
$RD_{entry}(5) = RD_{exit}(4)$
$RD_{entry}(6) = RD_{exit}(3)$

Equations as Functions

$F(RD) = (F_{entry}(1)(\overline{RD}), RD_{exit}(1)(\overline{RD}),$
$\ldots,$
$F_{entry}(6)(\overline{RD}), RD_{exit}(6)(\overline{RD}))$

where, for example,

$F_{exit}(1)(\ldots, RD_{entry}(1), \ldots) = (RD_{entry}(1) \setminus \{(y,*)\}) \cup (y,1)$

$F_{entry}(3)(\ldots, RD_{exit}(2), \ldots, RD_{exit}(5), \ldots) = RD_{exit}(2) \cup RD_{exit}(5)$

Computing a Fixed Point of F

- Start with the tuple $\overline{RD_\emptyset} = (\emptyset, \emptyset, \emptyset, \ldots, \emptyset)$
- Let $F^0(x) = x$
- Let $F^{n+1}(x) = F(F^n(x))$

- Surprise!
  - Now find n such that $F^{n+1}(\overline{RD_\emptyset}) = F^n(\overline{RD_\emptyset})$
  - By definition $F^n(\overline{RD_\emptyset})$ is a fixed point of F
  - Does this really work?

Computing the Fixed Point

	0	1	2	3	4	10
$RD_{entry}(1)$	∅	x?y?z?				x?y?z?
$RD_{exit}(1)$	∅	y1	x?y1z?			x?y1z?
$RD_{entry}(2)$	∅	∅	y1	x?y1z?		x?y1z?
$RD_{exit}(2)$	∅	z2	z2	y1z2		x?y1z2
$RD_{entry}(3)$	∅	∅	y5z2	y5z2z4		x?y1y5z2z4
$RD_{exit}(3)$	∅	∅	∅	y5z2		x?y1y5z2z4
$RD_{entry}(4)$	∅	∅	∅	∅		x?y1y5z2z4
$RD_{exit}(4)$	∅	z4	z4	z4		x?y1y5z4
$RD_{entry}(5)$	∅	∅	z4	z4		x?y1y5z4
$RD_{exit}(5)$	∅	y5	y5	y5z4		x?y5z4
$RD_{entry}(6)$	∅	∅	∅	∅		x?y1y5z2z4
$RD_{exit}(6)$	∅	y6	y6	y6		x?y6z2z4

$RD_{exit}(1) = (RD_{entry}(1) \setminus \{(y,*)\}) \cup (y,1)$
$RD_{exit}(2) = (RD_{entry}(2) \setminus \{(z,*)\}) \cup (z,2)$
$RD_{exit}(3) = RD_{entry}(3)$
$RD_{exit}(4) = (RD_{entry}(4) \setminus \{(z,*)\}) \cup (z,4)$
$RD_{exit}(5) = (RD_{entry}(5) \setminus \{(y,*)\}) \cup (y,5)$
$RD_{exit}(6) = (RD_{entry}(6) \setminus \{(y,*)\}) \cup (y,6)$

$RD_{entry}(1) = \{ (x,?), (y,?), (z,?) \}$
$RD_{entry}(2) = RD_{exit}(1)$
$RD_{entry}(3) = RD_{exit}(2) \cup RD_{exit}(5)$
$RD_{entry}(4) = RD_{exit}(3)$
$RD_{entry}(5) = RD_{exit}(4)$
$RD_{entry}(6) = RD_{exit}(3)$

Finding the Fixed Point

- Why should we think we will find an n such that $F^{n+1}(\overline{RD_\emptyset}) = F^n(\overline{RD_\emptyset})$?

Monotone Functions

- f is *monotone* if $v \subseteq v'$ implies $f(v) \subseteq f(v')$
- Assertion: F is monotone
  - Intuition: preserves input/output relationship
  - Check a couple of cases
  - (1) $RD_{exit}(1) = (RD_{entry}(1) \setminus \{(y,*)\}) \cup (y,1)$
    - Assume $RD_{entry}(1) \subseteq RD'_{entry}(1)$
    - Then $(RD_{entry}(1) \setminus \{(y,*)\}) \subseteq (RD'_{entry}(1) \setminus \{(y,*)\})$
    - So $RD_{exit}(1) \subseteq RD'_{exit}(1)$
    - Would this also be true if we used $\subset$?

Monotone Functions

- f is *monotone* if v $\subseteq$ v' implies f(v) $\subseteq$ f(v')
- Assertion: *F* is monotone
  - Intuition: preserves input/output relationship
  - Check a couple of cases
  - (1) $RD_{exit}(1) = (RD_{entry}(1) \setminus \{(y,*)\}) \cup (y,1)$
  - (2) $RD_{entry}(3) = RD_{exit}(2) \cup RD_{exit}(5)$
    - Assume $RD_{exit}(2) \subseteq RD'_{exit}(2)$
    - Assume $RD_{exit}(5) \subseteq RD'_{exit}(5)$
    - Then $RD_{exit}(2) \cup RD'_{exit}(5) \subseteq RD'_{exit}(2) \cup RD'_{exit}(5)$
    - So $RD_{entry}(3) \subseteq RD'_{entry}(3)$

Finding the Fixed Point

- Why should we think we will find an *n* such that $F^{n+1}(\overline{RD_\emptyset}) = F^n(\overline{RD_\emptyset})$?
  - *F* is monotone
  - Claim: $\forall n\ F^n(\overline{RD_\emptyset}) \subseteq F^{n+1}(\overline{RD_\emptyset})$
    - Base case: $\overline{RD_\emptyset} \subseteq F(\overline{RD_\emptyset})$
      - Since no tuple has smaller sets than $RD_\emptyset$
    - Inductive case:
      - Assume $F^{n-1}(\overline{RD_\emptyset}) \subseteq F^n(\overline{RD_\emptyset})$
      - *F* is monotone, so $F(F^{n-1}(\overline{RD_\emptyset})) \subseteq F(F^n(\overline{RD_\emptyset}))$
      - Equivalently, $F^n(\overline{RD_\emptyset}) \subseteq F^{n+1}(\overline{RD_\emptyset})$
  - Therefore, every application of F either:
    - Does not change $\overline{RD}$ (and so we have a fixed point)
    - Or increases the size of a set in $\overline{RD}$
  - The set of definitions is finite so the sets in $\overline{RD}$ cannot increase in size forever
  - Therefore the algorithm terminates with a fixed point at some finite *n*

Precision

- Is $F^n(\overline{RD_\emptyset})$ the least fixed point?
  - i.e., the fixed point with the smallest sets?
- Yes. Proof:
  - Consider some other fixed point $\overline{RD_{fix}}$
  - $\overline{RD_\emptyset} \subseteq \overline{RD_{fix}}$
  - Since *F* is monotone, $F(\overline{RD_\emptyset}) \subseteq F(\overline{RD_{fix}})$
  - By induction $F^n(\overline{RD_\emptyset}) \subseteq F^n(\overline{RD_{fix}})$
  - But RD_{fix} is a fixed point so $\overline{RD_{fix}} = F(\overline{RD_{fix}}) = F^n(\overline{RD_{fix}})$
  - Therefore $F^n(\overline{RD_\emptyset}) \subseteq \overline{RD_{fix}}$
  - Therefore $F^n(\overline{RD_\emptyset})$ is the least fixed point of *F*

Efficient Algorithms

- Computing $F^n(\overline{RD_\emptyset})$ is slow
  - 10 iterations
  - Each iteration recomputes each member of $\overline{RD_\emptyset}$
  - Few members of $\overline{RD_\emptyset}$ change each iteration
- Optimization: Chaotic Iteration
  - Recompute one member of $\overline{RD_\emptyset}$ at a time
  - Guess a member that is likely to change
  - Can compute fixed point in 17 iterations, one recomputation per iteration (vs. 12 before)

Chaotic Iteration

$RD_{1..n} = \emptyset$
while $RD_j \neq F_j(RD_{1..n})$ for some *j*
  do $RD_j := F_j(RD_{1..n})$

- How to choose *j*?
  - Later!

Chaotic Iteration

$RD_{1..n} = \emptyset$
while $RD_j \neq F_j(RD_{1..n})$ for some *j*
  do $RD_j := F_j(RD_{1..n})$

- Properties
  - If chaotic iteration terminates, $RD_{1..n}$ is a fixed point of *F*
    - Proof: termination implies $\overline{RD} = F(\overline{RD})$
  - That fixed point is a least fixed point
    - Proof: $\overline{RD} \subseteq F^n(\overline{RD_\emptyset})$ is an invariant of the algorithm
  - Chaotic iteration terminates for monotone *F* and finite sets $RD_{1..n}$
    - Proof: F is monotone and so $\overline{RD}$ is increasing

Chaotic Iteration Example

Iter	Position	Value
0	--	$\overline{\varnothing}$
1	entry(1)	x?y?z?
2	exit(1)	x?y1z?
3	entry(2)	x?y1z?
4	exit(2)	x?y1z2
5	entry(3)	x?y1z2
6	exit(3)	x?y1z2
7	entry(4)	x?y1z2
8	exit(4)	x?y1z4
9	entry(5)	x?y1z4
10	exit(5)	x?y5z4
11	entry(3)	x?y1y5z2z4
12	exit(3)	x?y1y5z2z4
13	entry(4)	x?y1y5z2z4
14	exit(4)	x?y1y5z4
15	entry(5)	x?y1y5z4
16	entry(6)	x?y1y5z2z4
17	exit(6)	x?y6z2z4

$RD_{exit}(1) = (RD_{entry}(1) \setminus \{(y,*)\}) \cup (y,1)$
$RD_{exit}(2) = (RD_{entry}(2) \setminus \{(z,*)\}) \cup (z,2)$
$RD_{exit}(3) = RD_{entry}(3)$
$RD_{exit}(4) = (RD_{entry}(4) \setminus \{(z,*)\}) \cup (z,4)$
$RD_{exit}(5) = (RD_{entry}(5) \setminus \{(y,*)\}) \cup (y,5)$
$RD_{exit}(6) = (RD_{entry}(6) \setminus \{(y,*)\}) \cup (y,6)$

$RD_{entry}(1) = \{ (x,?), (y,?), (z,?) \}$
$RD_{entry}(2) = RD_{exit}(1)$
$RD_{entry}(3) = RD_{exit}(2) \cup RD_{exit}(5)$
$RD_{entry}(4) = RD_{exit}(3)$
$RD_{entry}(5) = RD_{exit}(4)$
$RD_{entry}(6) = RD_{exit}(3)$

Comparison to Naïve Algorithm

	0	1	2	3	4	10
$RD_{entry}(1)$	∅	x?y?z?	———	———	———	x?y?z?
$RD_{exit}(1)$	∅	y1	x?y1z?	———	———	x?y1z?
$RD_{entry}(2)$	∅	∅	y1	x?y1z?	———	x?y1z?
$RD_{exit}(2)$	∅	z2	z2	y1z2		x?y1z2
$RD_{entry}(3)$	∅	∅	y5z2	y5z2z4		x?y1y5z2z4
$RD_{exit}(3)$	∅	∅	∅	y5z2		x?y1y5z2z4
$RD_{entry}(4)$	∅	∅	∅	∅		x?y1y5z2z4
$RD_{exit}(4)$	∅	z4	z4	z4		x?y1y5z4
$RD_{entry}(5)$	∅	∅	z4	z4		x?y1y5z4
$RD_{exit}(5)$	∅	y5	y5	y5z4		x?y5z4
$RD_{entry}(6)$	∅	∅	∅	∅		x?y1y5z2z4
$RD_{exit}(6)$	∅	y6	y6	y6		x?y6z2z4

$RD_{exit}(1) = (RD_{entry}(1) \setminus \{(y,*)\}) \cup (y,1)$
$RD_{exit}(2) = (RD_{entry}(2) \setminus \{(z,*)\}) \cup (z,2)$
$RD_{exit}(3) = RD_{entry}(3)$
$RD_{exit}(4) = (RD_{entry}(4) \setminus \{(z,*)\}) \cup (z,4)$
$RD_{exit}(5) = (RD_{entry}(5) \setminus \{(y,*)\}) \cup (y,5)$
$RD_{exit}(6) = (RD_{entry}(6) \setminus \{(y,*)\}) \cup (y,6)$

$RD_{entry}(1) = \{ (x,?), (y,?), (z,?) \}$
$RD_{entry}(2) = RD_{exit}(1)$
$RD_{entry}(3) = RD_{exit}(2) \cup RD_{exit}(5)$
$RD_{entry}(4) = RD_{exit}(3)$
$RD_{entry}(5) = RD_{exit}(4)$
$RD_{entry}(6) = RD_{exit}(3)$

Constant Folding

- A program optimization
  - Replaces computation with constants
  - Can use reaching definitions
- Notation
  - RD ⊢ S ▷ S'
    - "Given reaching definitions RD, statement S can be transformed into S' "
  - $\frac{C}{T}$
    - Transformation T is legal if condition(s) C hold
  - FV(exp)
    - The variables mentioned in exp
  - exp[y↦n]
    - Replace all occurrences of y in exp with n

Constant Folding Rules

$[ass_1]$ $\quad RD \vdash [x := a]^\ell \rhd [x := a[y \mapsto n]]^\ell$
$\quad$ if $\begin{cases} y \in FV(a) \wedge (y,?) \notin RD_{entry}(\ell) \wedge \\ \forall (x,\ell') \in RD_{entry}(\ell) : (x = y \Rightarrow [\cdots]^{\ell'} \text{ is } [y := n]^{\ell'}) \end{cases}$

$[ass_2]$ $\quad RD \vdash [x := a]^\ell \rhd [x := n]^\ell$
$\quad$ if $FV(a) = \emptyset \wedge a \notin \mathbf{Num} \wedge a$ evaluates to n

$[seq_1]$ $\quad \dfrac{RD \vdash S_1 \rhd S_1'}{RD \vdash S_1; S_2 \rhd S_1'; S_2}$

$[seq_2]$ $\quad \dfrac{RD \vdash S_2 \rhd S_2'}{RD \vdash S_1; S_2 \rhd S_1; S_2'}$

$[if_1]$ $\quad \dfrac{RD \vdash S_1 \rhd S_1'}{RD \vdash \texttt{if } [b]^\ell \texttt{ then } S_1 \texttt{ else } S_2 \rhd \texttt{if } [b]^\ell \texttt{ then } S_1' \texttt{ else } S_2}$

$[if_2]$ $\quad \dfrac{RD \vdash S_2 \rhd S_2'}{RD \vdash \texttt{if } [b]^\ell \texttt{ then } S_1 \texttt{ else } S_2 \rhd \texttt{if } [b]^\ell \texttt{ then } S_1 \texttt{ else } S_2'}$

$[wh]$ $\quad \dfrac{RD \vdash S \rhd S'}{RD \vdash \texttt{while } [b]^\ell \texttt{ do } S \rhd \texttt{while } [b]^\ell \texttt{ do } S'}$

Taken from Nielson, Nielson, and Hankin, page 27

Example

$[x:=10]^1; [y:=x+10]^2; [z:=y+10]^3$

- $RD_{enter}(2) = \{ (x,1), (y,?), (z,?) \}$
- $RD \vdash [y:=x+10]^2 \rhd [y:=10+10]^2$ *by [ass$_1$]*
- Thus:
  - $RD \vdash [x:=10]^1; [y:=x+10]^2; [z:=y+10]^3 \rhd$ $[x:=10]^1; [y:=10+10]^2; [z:=y+10]^3$ *by [seq] rules*

Example

RD $\vdash [x:=10]^1; [y:=x+10]^2; [z:=y+10]^3$
$\rhd [x:=10]^1; [y:=10+10]^2; [z:=y+10]^3$ *by [ass$_1$]*
$\rhd [x:=10]^1; [y:=20]^2; [z:=y+10]^3$ *by [ass$_2$]*
$\rhd [x:=10]^1; [y:=20]^2; [z:=20+10]^3$ *by [ass$_1$]*
$\rhd [x:=10]^1; [y:=20]^2; [z:=30]^3$ *by [ass$_2$]*