

Interprocedural analysis: Sharir-Pnueli's functional approach

Deepak D'Souza

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

30 September 2024

Outline

1 Functional Approach

2 Example

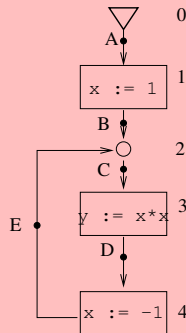
3 Iterative Approach

4 Exercises

Equation solving approach

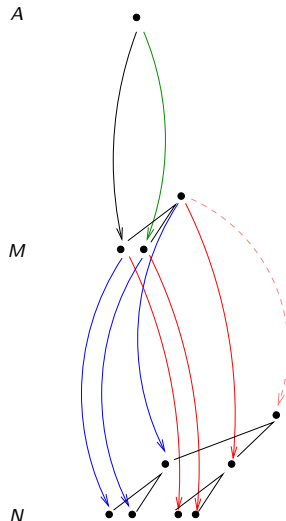
In non-procedural case, we set up equations to capture JOP assuming distributivity. Least solution to these equations gave us exact/over-approx JOP depending on distributive/monotonic framework.

$$\begin{aligned} x_A &= \emptyset \\ x_B &= f_1(x_A) \\ x_C &= x_B \sqcup x_E \\ x_D &= f_3(x_C) \\ x_E &= f_4(x_D) \end{aligned}$$



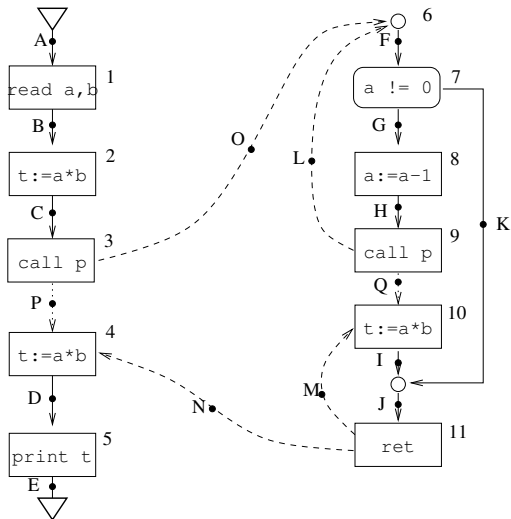
Equations to capture JOP: why it works

- We want JOP at N .
- Suppose M is an intermediate point such that **all** paths to N pass through M .
- If transfer functions are distributive, then we can take join over paths at point M , and then join over paths from M to N .



Equation solving: Problems with naive approach

- Try to set up similar equations for x_N (JVP at program point N).
- How do we describe x_N in terms of x_J ?

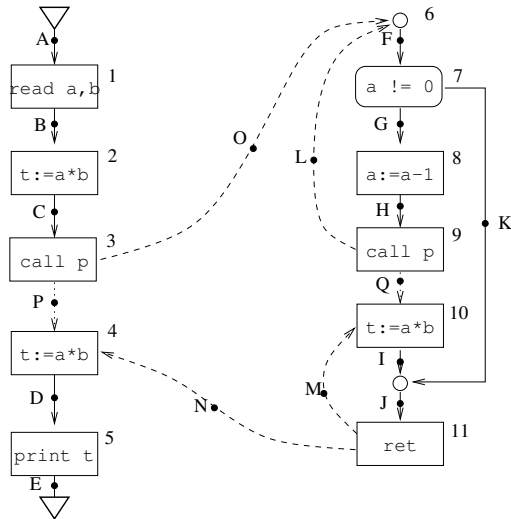


Instead try to capture join over **complete** paths first

- Set up equations to capture join over **complete** paths.
- Now set up equations to capture JVP using join over complete path values.

Notation

- Root of procedure p is denoted r_p .
- Exit (return) of procedure p is denoted e_p .
- Sometimes use r_1 for r_{main} .
- Assume WLOG that $main$ is not called.

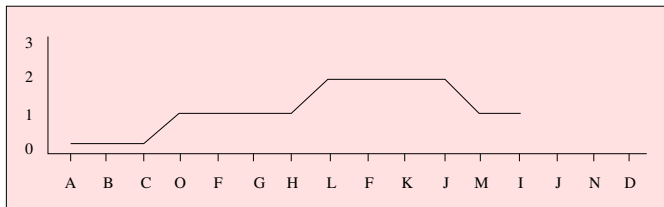


Valid and complete paths

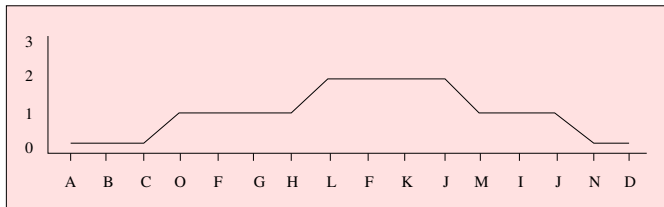
- A path ρ in G' is **valid and complete** if it is an interprocedurally valid path in G' and the associated call-string is empty.
- Denote the set of such valid and complete paths by $IVP_0(G')$
- That is $\rho \in IVP_0(G')$ iff $\rho \in IVP(G')$ and $cs(\rho) = \epsilon$.

Example paths

An example valid path in $IVP(r_1, I)$:



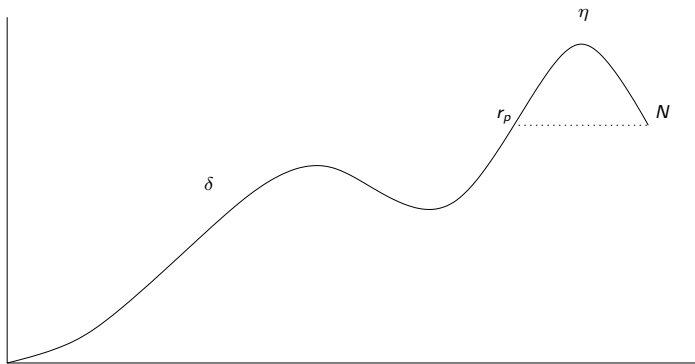
An example valid and complete path in $IVP_0(r_1, D)$:



Path “FGHLFKJMIJ” is valid and complete and is in $IVP_0(r_p, J)$.

Basic idea: Why join over complete paths help

An IVP path ρ from r_1 to N in procedure p can be written as $\delta \cdot \eta$ where δ is in $\text{IVP}(r_1, r_p)$, and η is in $\text{IVP}_0(r_p, N)$.



Path η is suffix after last pending call to procedure p was made.

Join over valid and complete paths from r_p to N

For a procedure p and node N in p , define:

$$\phi_{r_p, N} : D \rightarrow D$$

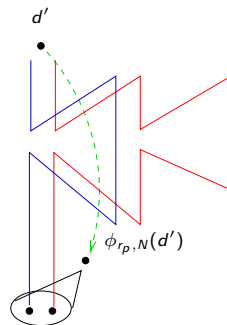
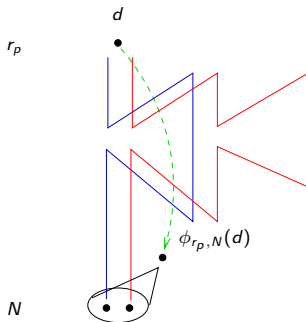
given by

$$\phi_{r_p, N}(d) = \bigsqcup_{\text{paths } \rho \in \text{IVP}_0(r_p, N)} f_\rho(d).$$

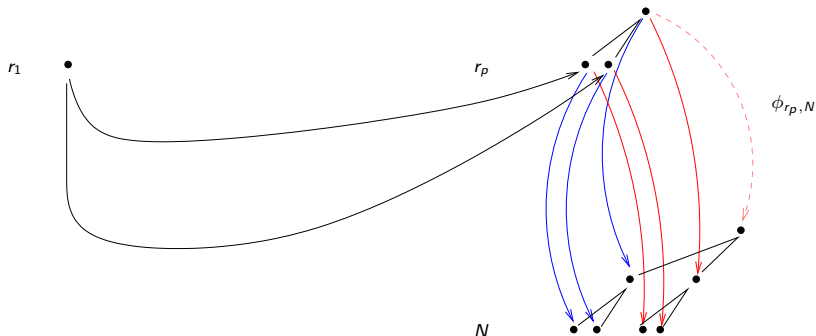
$\phi_{r_p, N}$ is thus the join of all functions f_ρ where ρ is an **interprocedurally valid and complete** path from r_p to N .

We call $\phi_{r_p, N}$ the **Join over Valid and Complete Paths** (JVCP) from r_p to N .

Visualizing $\phi_{r_p, N}$



Using $\phi_{r_p, N}$'s to get JVP values



Assuming distributivity of underlying transfer functions, JVP value at N equals $\phi_{r_p, N}$ applied to JVP value at r_p .

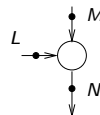
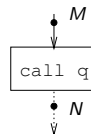
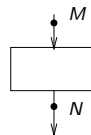
Equations (1) to capture $\phi_{r_p, N}$

$$y_{r_p, r_p} = id_D \quad (root)$$

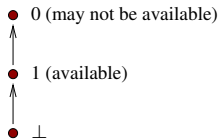
$$y_{r_p, N} = f_{MN} \circ y_{r_p, M} \quad (stmt)$$

$$y_{r_p, N} = y_{r_q, e_q} \circ y_{r_p, M} \quad (call)$$

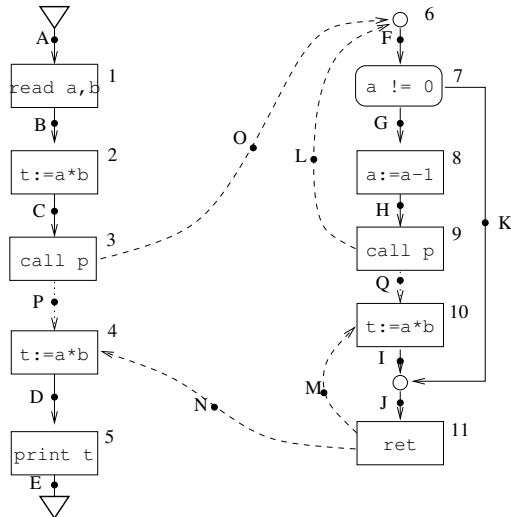
$$y_{r_p, N} = y_{r_p, L} \sqcup y_{r_p, M} \quad (join)$$



Example: Available expressions analysis



Lattice for Av-Exp analysis.



Functions we will use for example analysis

- $D = \{\perp, 1, 0\}$.

- $\mathbf{0} : D \rightarrow D$ given by

0	$\mapsto$	0
1	$\mapsto$	0
$\perp$	$\mapsto$	$\perp$

- $\mathbf{1} : D \rightarrow D$ given by

0	$\mapsto$	1
1	$\mapsto$	1
$\perp$	$\mapsto$	$\perp$

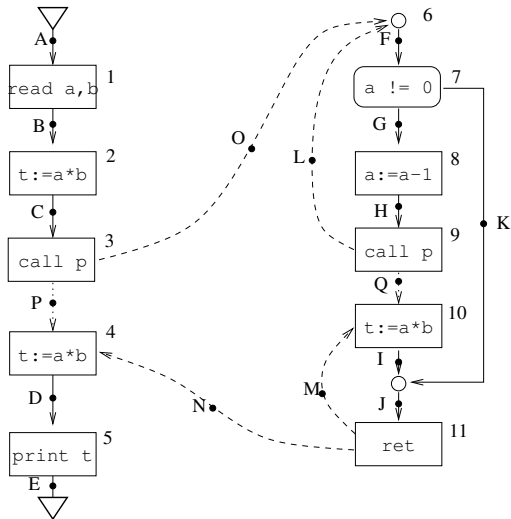
- $id : D \rightarrow D$ given by

0	$\mapsto$	0
1	$\mapsto$	1
$\perp$	$\mapsto$	$\perp$

- Ordering: $\mathbf{1} \leq id \leq \mathbf{0}$.

Example: Equations (1) for ϕ 's

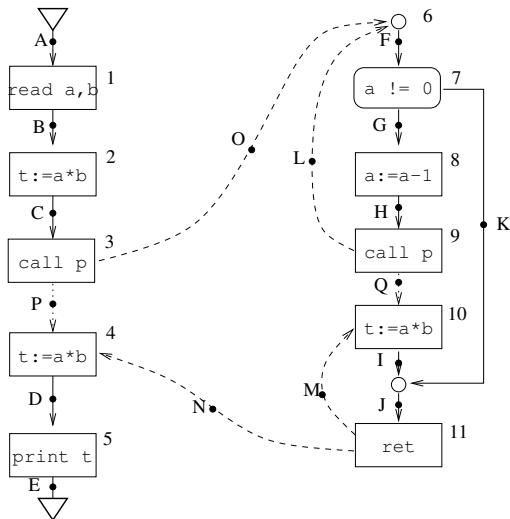
$$\begin{aligned}
 y_{A,A} &= id \\
 y_{A,B} &= \mathbf{0} \circ y_{A,A} \\
 y_{A,C} &= \mathbf{1} \circ y_{A,B} \\
 y_{A,P} &= y_{F,J} \circ y_{A,C} \\
 y_{A,D} &= \mathbf{1} \circ y_{A,P} \\
 y_{A,E} &= id \circ y_{A,D}
 \end{aligned}$$



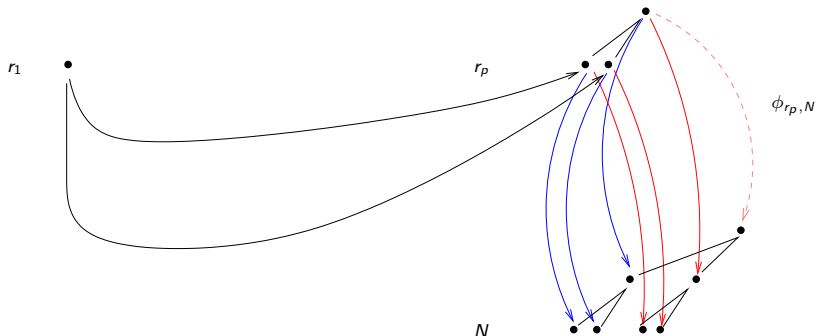
Example: Equations (1) for ϕ 's

$$\begin{aligned}
 y_{A,A} &= id \\
 y_{A,B} &= \mathbf{0} \circ y_{A,A} \\
 y_{A,C} &= \mathbf{1} \circ y_{A,B} \\
 y_{A,P} &= y_{F,J} \circ y_{A,C} \\
 y_{A,D} &= \mathbf{1} \circ y_{A,P} \\
 y_{A,E} &= id \circ y_{A,D}
 \end{aligned}$$

$$\begin{aligned}
 y_{F,F} &= id \\
 y_{F,G} &= id \circ y_{F,F} \\
 y_{F,K} &= id \circ y_{F,F} \\
 y_{F,H} &= \mathbf{0} \circ y_{F,G} \\
 y_{F,Q} &= y_{F,J} \circ y_{F,H} \\
 y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\
 y_{F,J} &= y_{F,I} \sqcup y_{F,K}
 \end{aligned}$$



Using $\phi_{r_p, N}$'s to get JVP values



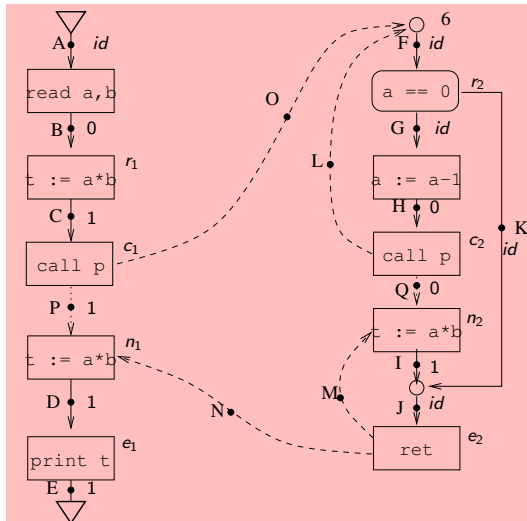
Assuming distributivity of underlying transfer functions, JVP value at N equals $\phi_{r_p, N}$ applied to JVP value at r_p .

Equations (2) to capture JVP

$$\begin{aligned}
 x_1 &= d_0 \\
 x_{r_p} &= \bigsqcup_{\text{calls } C \text{ to } p} x_C \\
 x_N &= \phi_{r_p, N}(x_{r_p}) \quad \text{for } N \in \text{ProgPts}(p) - \{r_p\}.
 \end{aligned}$$

Example: Equations for x_N 's (JVP)

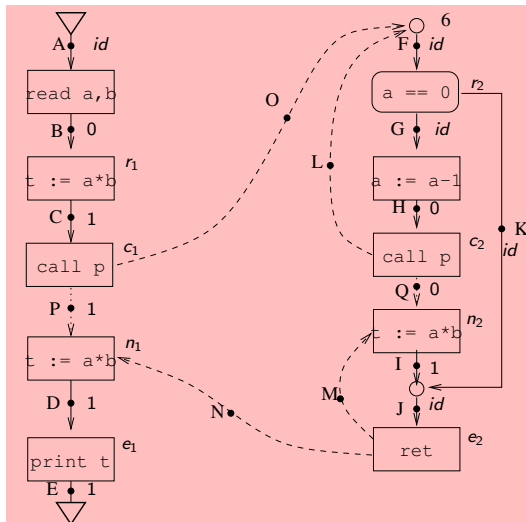
$$\begin{aligned}
 x_A &= 0 \\
 x_B &= \phi_{AB}(x_A) \\
 x_C &= \phi_{AC}(x_A) \\
 x_P &= \phi_{AP}(x_A) \\
 x_D &= \phi_{AD}(x_A) \\
 x_E &= \phi_{AE}(x_A)
 \end{aligned}$$



Example: Equations for x_N 's (JVP)

$$\begin{aligned}
 x_A &= 0 \\
 x_B &= \phi_{AB}(x_A) \\
 x_C &= \phi_{AC}(x_A) \\
 x_P &= \phi_{AP}(x_A) \\
 x_D &= \phi_{AD}(x_A) \\
 x_E &= \phi_{AE}(x_A)
 \end{aligned}$$

$$\begin{aligned}
 x_F &= x_C \sqcup x_H \\
 x_G &= \phi_{FG}(x_F) \\
 x_K &= \phi_{FK}(x_F) \\
 x_H &= \phi_{FH}(x_F) \\
 x_Q &= \phi_{FQ}(x_F) \\
 x_I &= \phi_{FI}(x_F) \\
 x_J &= \phi_{FJ}(x_F).
 \end{aligned}$$



Example: Equations for x_N 's (JVP)

$$\begin{aligned}
 x_A &= 0 \\
 x_B &= \mathbf{0}(x_A) \\
 x_C &= \mathbf{1}(x_A) \\
 x_P &= \mathbf{1}(x_A) \\
 x_D &= \mathbf{1}(x_A) \\
 x_E &= \mathbf{1}(x_A)
 \end{aligned}$$

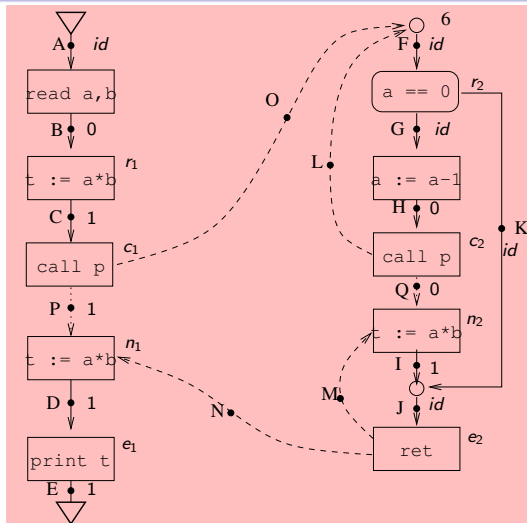


Fig. shows values of $\phi_{r_p, N}$'s in bold.

Example: Equations for x_N 's (JVP)

$$x_A = 0$$

$$x_B = \mathbf{0}(x_A)$$

$$x_C = \mathbf{1}(x_A)$$

$$x_P = \mathbf{1}(x_A)$$

$$x_D = \mathbf{1}(x_A)$$

$$x_F = \mathbf{1}(x_A)$$

$$X_F = X_C \sqcup X_H$$

$$x_G = id(x_F)$$

$$x_K = id(x_F)$$

$$x_H = \mathbf{0}(x_F)$$

$$x_Q = \mathbf{0}(x_F)$$

$$x_I = \mathbf{1}(x_F)$$

$$x_I = id(x_F).$$

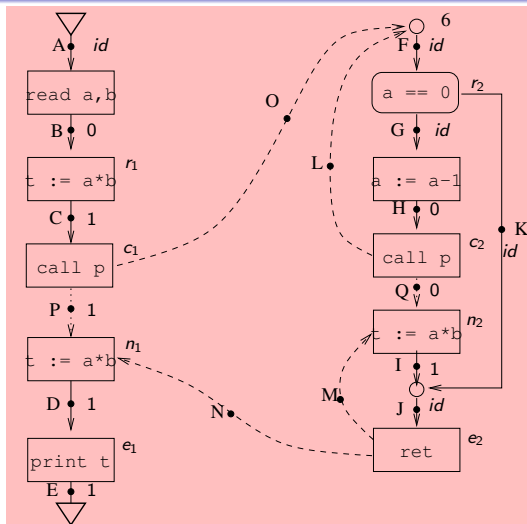


Fig. shows values of $\phi_{r_p, N}$'s in bold.

Solving a system of equations using Knaster-Tarski Theorem

Given equations (E_1)

$$\begin{aligned} y_1 &= f_1(y_1, \dots, y_n) \\ &\dots \\ y_n &= f_n(y_1, \dots, y_n) \end{aligned}$$

Consider a complete lattice $(D, \leq)$ such that:

- D is **closed** under each f_i .
- Each f_i is a **monotonic** function on this lattice: if $\langle d_1, \dots, d_n \rangle \leq \langle e_1, \dots, e_n \rangle$ then $f_i(d_1, \dots, d_n) \leq f_i(e_1, \dots, e_n)$.
- Equivalently, the function $\bar{F}$ on $(D^n, \leq)$ given by

$$\bar{F}(\langle d_1, \dots, d_n \rangle) = \langle f_1(d_1, \dots, d_n), \dots, f_n(d_1, \dots, d_n) \rangle,$$

is monotonic.

Then, by Knaster-Tarski, the function $\bar{F}$ on $(D^n, \leq)$ has a LFP, which coincides with the least solution (in D^n) to equations (E_1) .

Solving Eq (1) using Knaster-Tarski

- Consider lattice $(F, \leq)$ of **functions** from D to D , obtained by closing the transfer functions, identity, and $f_{\perp} : d \mapsto \perp$ under composition and join. (Alternatively we can take F to be all **monotone** functions on D .)
- Ordering is $f \leq g$ iff $f(d) \leq g(d)$ for each $d \in D$.
- $(F, \leq)$ is also a complete lattice.
- $\bar{f}$ induced by Eq (1) is monotone on complete lattice $(\bar{F}, \bar{\leq})$.
 - Sufficient to argue that function composition $\circ$ is monotone when applied to monotone functions.
 - Join operation $\sqcup$ is monotone.
- LFP / least solution (say $y_{r_p, N}^*$'s) exists by Knaster-Tarski.
- Each $y_{r_p, N}^*$ is necessarily monotonic.

Using Kildall to compute LFP

- We can use Kildall's algo to compute the LFP of these equations as follows.
 - Initialize the value at all program points with RHS of the constant equations (in this case id at entry of procedures), and the bottom value (in this case $f_{\perp}$).
 - Mark all values
 - Pick a marked value at point say N , and “propagate” it (i.e. for any node M in the LHS of an equation in which N occurs in the RHS, evaluate M and join it with the existing value at M). Mark as before in Kildall's algo.
 - Stop when no more marked values to propagate.
- Kildall's algo will compute $y_{r_p, N}^*$ if D is finite. Note that finite height of $(D, \leq)$ is not sufficient for termination.

Correctness claim I

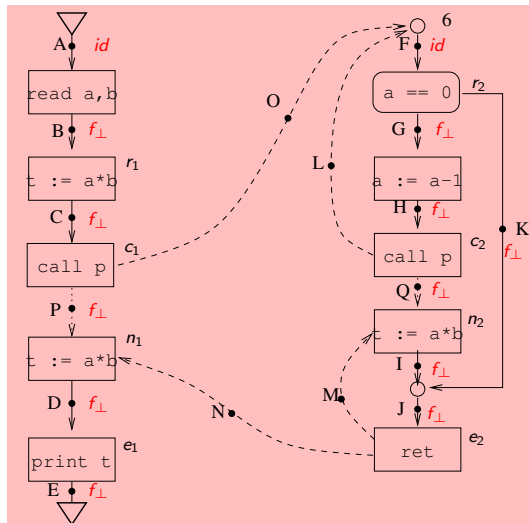
Claim

- ① The least solution to Eq (1) dominates $\phi_{r_p, N}$'s (i.e. $\phi_{r_p, N} \leq y_{r_p, N}^*$ for each N).
- ② $\phi_{r_p, N}$'s are the least solution to Eq (1) (i.e. $\phi_{r_p, N} = y_{r_p, N}^*$ for each N) when f_{MN} 's are distributive.

Example: Computing $\phi_{r_p,N}$'s ($y_{r_p,N}^*$ to be precise) using Kildall's algo

$$\begin{aligned}
 y_{A,A} &= id \\
 y_{A,B} &= \mathbf{0} \circ y_{A,A} \\
 y_{A,C} &= \mathbf{1} \circ y_{A,B} \\
 y_{A,P} &= y_{F,J} \circ y_{A,C} \\
 y_{A,D} &= \mathbf{1} \circ y_{A,P} \\
 y_{A,E} &= id \circ y_{A,D}
 \end{aligned}$$

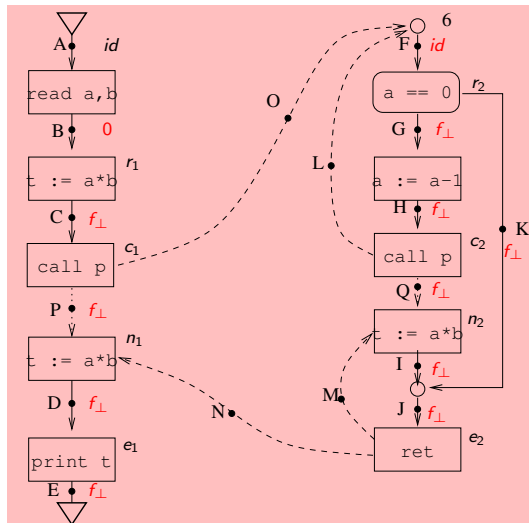
$$\begin{aligned}
 y_{F,F} &= id \\
 y_{F,G} &= id \circ y_{F,F} \\
 y_{F,K} &= id \circ y_{F,F} \\
 y_{F,H} &= \mathbf{0} \circ y_{F,G} \\
 y_{F,Q} &= y_{F,J} \circ y_{F,H} \\
 y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\
 y_{F,J} &= y_{F,I} \sqcup y_{F,K}
 \end{aligned}$$



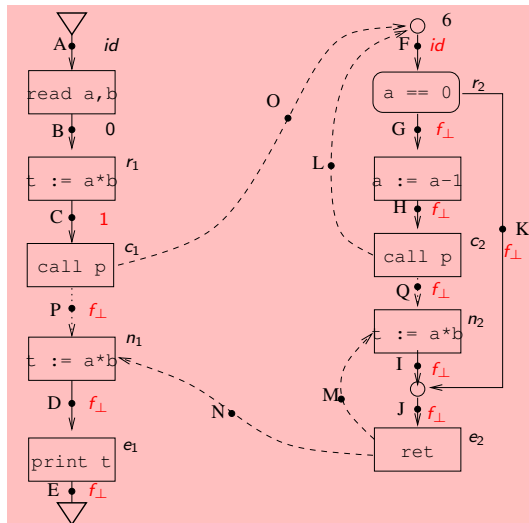
Example: Computing $\phi_{r_p,N}$'s ($y_{r_p,N}^*$ to be precise) using Kildall's algo

$y_{A,A} = id$
 $y_{A,B} = \mathbf{0} \circ y_{A,A}$
 $y_{A,C} = \mathbf{1} \circ y_{A,B}$
 $y_{A,P} = y_{F,J} \circ y_{A,C}$
 $y_{A,D} = \mathbf{1} \circ y_{A,P}$
 $y_{A,E} = id \circ y_{A,D}$

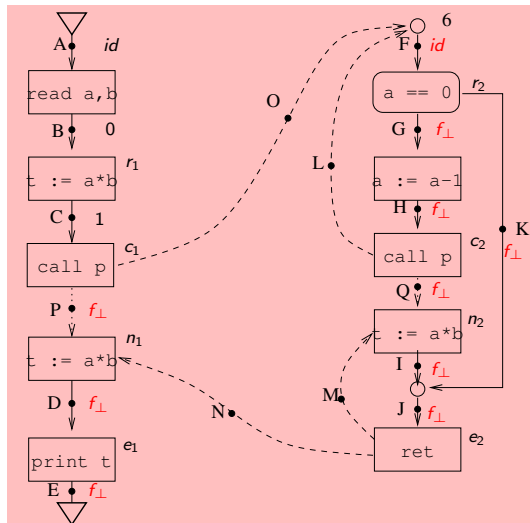
$y_{F,F} = id$
 $y_{F,G} = id \circ y_{F,F}$
 $y_{F,K} = id \circ y_{F,F}$
 $y_{F,H} = \mathbf{0} \circ y_{F,G}$
 $y_{F,Q} = y_{F,J} \circ y_{F,H}$
 $y_{F,I} = \mathbf{1} \circ y_{F,Q}$
 $y_{F,J} = y_{F,I} \sqcup y_{F,K}$



Example: Computing $\phi_{r_p, N}$'s ($y_{r_p, N}^*$ to be precise) using Kildall's algo

$$\begin{aligned} y_{A,A} &= id \\ y_{A,B} &= \mathbf{0} \circ y_{A,A} \\ y_{A,C} &= \mathbf{1} \circ y_{A,B} \\ y_{A,P} &= y_{F,J} \circ y_{A,C} \\ y_{A,D} &= \mathbf{1} \circ y_{A,P} \\ y_{A,E} &= id \circ y_{A,D} \end{aligned}$$
$$\begin{aligned} y_{F,F} &= id \\ y_{F,G} &= id \circ y_{F,F} \\ y_{F,K} &= id \circ y_{F,F} \\ y_{F,H} &= \mathbf{0} \circ y_{F,G} \\ y_{F,Q} &= y_{F,J} \circ y_{F,H} \\ y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\ y_{F,J} &= y_{F,I} \sqcup y_{F,K} \end{aligned}$$


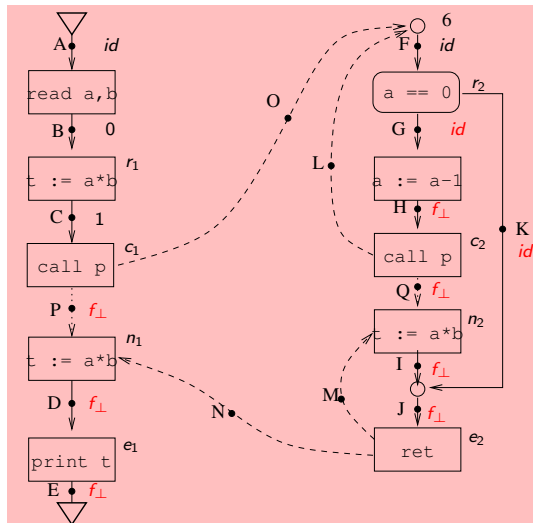
Example: Computing $\phi_{r_p, N}$'s ($y_{r_p, N}^*$ to be precise) using Kildall's algo

$$\begin{aligned} y_{A,A} &= id \\ y_{A,B} &= \mathbf{0} \circ y_{A,A} \\ y_{A,C} &= \mathbf{1} \circ y_{A,B} \\ y_{A,P} &= y_{F,J} \circ y_{A,C} \\ y_{A,D} &= \mathbf{1} \circ y_{A,P} \\ y_{A,E} &= id \circ y_{A,D} \end{aligned}$$
$$\begin{aligned} y_{F,F} &= id \\ y_{F,G} &= id \circ y_{F,F} \\ y_{F,K} &= id \circ y_{F,F} \\ y_{F,H} &= \mathbf{0} \circ y_{F,G} \\ y_{F,Q} &= y_{F,J} \circ y_{F,H} \\ y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\ y_{F,J} &= y_{F,I} \sqcup y_{F,K} \end{aligned}$$


Example: Computing $\phi_{r_p,N}$'s ($y_{r_p,N}^*$ to be precise) using Kildall's algo

$$\begin{aligned}
 y_{A,A} &= id \\
 y_{A,B} &= \mathbf{0} \circ y_{A,A} \\
 y_{A,C} &= \mathbf{1} \circ y_{A,B} \\
 y_{A,P} &= y_{F,J} \circ y_{A,C} \\
 y_{A,D} &= \mathbf{1} \circ y_{A,P} \\
 y_{A,E} &= id \circ y_{A,D}
 \end{aligned}$$

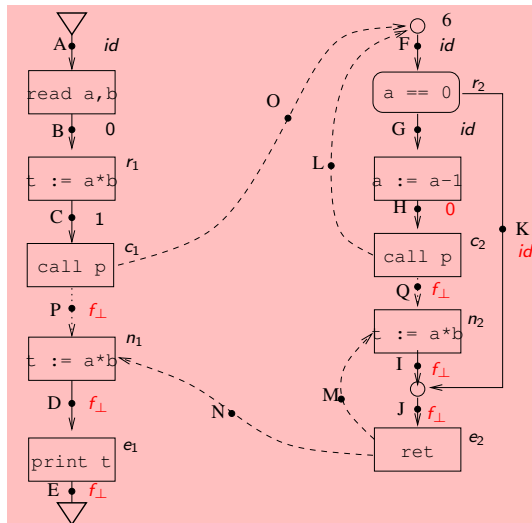
$$\begin{aligned}
 y_{F,F} &= id \\
 y_{F,G} &= id \circ y_{F,F} \\
 y_{F,K} &= id \circ y_{F,F} \\
 y_{F,H} &= \mathbf{0} \circ y_{F,G} \\
 y_{F,Q} &= y_{F,J} \circ y_{F,H} \\
 y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\
 y_{F,J} &= y_{F,I} \sqcup y_{F,K}
 \end{aligned}$$



Example: Computing $\phi_{r_p,N}$'s ($y_{r_p,N}^*$ to be precise) using Kildall's algo

$$\begin{aligned}
 y_{A,A} &= id \\
 y_{A,B} &= \mathbf{0} \circ y_{A,A} \\
 y_{A,C} &= \mathbf{1} \circ y_{A,B} \\
 y_{A,P} &= y_{F,J} \circ y_{A,C} \\
 y_{A,D} &= \mathbf{1} \circ y_{A,P} \\
 y_{A,E} &= id \circ y_{A,D}
 \end{aligned}$$

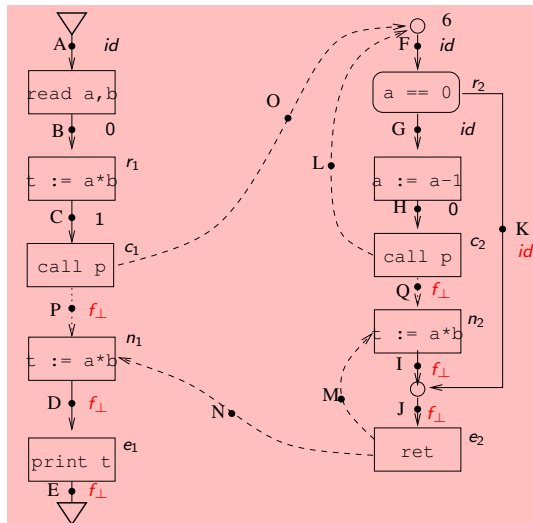
$$\begin{aligned}
 y_{F,F} &= id \\
 y_{F,G} &= id \circ y_{F,F} \\
 y_{F,K} &= id \circ y_{F,F} \\
 y_{F,H} &= \mathbf{0} \circ y_{F,G} \\
 y_{F,Q} &= y_{F,J} \circ y_{F,H} \\
 y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\
 y_{F,J} &= y_{F,I} \sqcup y_{F,K}
 \end{aligned}$$



Example: Computing $\phi_{r_p,N}$'s ($y_{r_p,N}^*$ to be precise) using Kildall's algo

$$\begin{aligned}
 y_{A,A} &= id \\
 y_{A,B} &= \mathbf{0} \circ y_{A,A} \\
 y_{A,C} &= \mathbf{1} \circ y_{A,B} \\
 y_{A,P} &= y_{F,J} \circ y_{A,C} \\
 y_{A,D} &= \mathbf{1} \circ y_{A,P} \\
 y_{A,E} &= id \circ y_{A,D}
 \end{aligned}$$

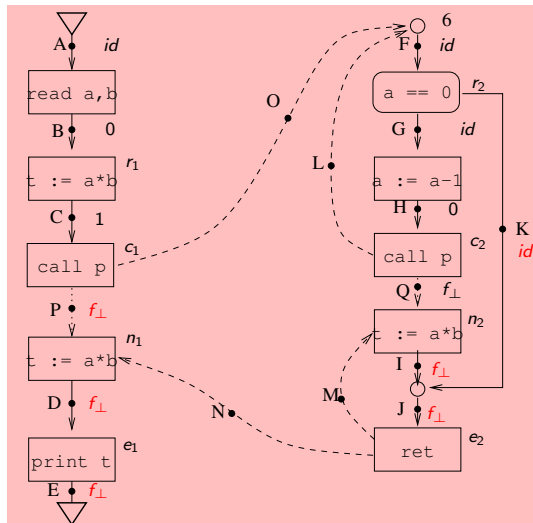
$$\begin{aligned}
 y_{F,F} &= id \\
 y_{F,G} &= id \circ y_{F,F} \\
 y_{F,K} &= id \circ y_{F,F} \\
 y_{F,H} &= \mathbf{0} \circ y_{F,G} \\
 y_{F,Q} &= y_{F,J} \circ y_{F,H} \\
 y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\
 y_{F,J} &= y_{F,I} \sqcup y_{F,K}
 \end{aligned}$$



Example: Computing $\phi_{r_p,N}$'s ($y_{r_p,N}^*$ to be precise) using Kildall's algo

$$\begin{aligned}
 y_{A,A} &= id \\
 y_{A,B} &= \mathbf{0} \circ y_{A,A} \\
 y_{A,C} &= \mathbf{1} \circ y_{A,B} \\
 y_{A,P} &= y_{F,J} \circ y_{A,C} \\
 y_{A,D} &= \mathbf{1} \circ y_{A,P} \\
 y_{A,E} &= id \circ y_{A,D}
 \end{aligned}$$

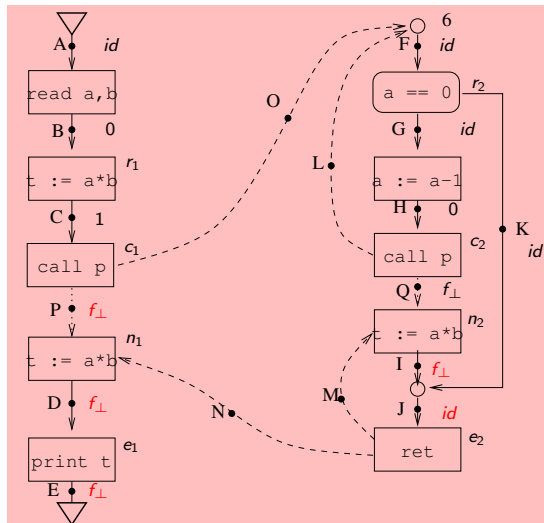
$$\begin{aligned}
 y_{F,F} &= id \\
 y_{F,G} &= id \circ y_{F,F} \\
 y_{F,K} &= id \circ y_{F,F} \\
 y_{F,H} &= \mathbf{0} \circ y_{F,G} \\
 y_{F,Q} &= y_{F,J} \circ y_{F,H} \\
 y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\
 y_{F,J} &= y_{F,I} \sqcup y_{F,K}
 \end{aligned}$$



Example: Computing $\phi_{r_p,N}$'s ($y_{r_p,N}^*$ to be precise) using Kildall's algo

$$\begin{aligned}
 y_{A,A} &= id \\
 y_{A,B} &= \mathbf{0} \circ y_{A,A} \\
 y_{A,C} &= \mathbf{1} \circ y_{A,B} \\
 y_{A,P} &= y_{F,J} \circ y_{A,C} \\
 y_{A,D} &= \mathbf{1} \circ y_{A,P} \\
 y_{A,E} &= id \circ y_{A,D}
 \end{aligned}$$

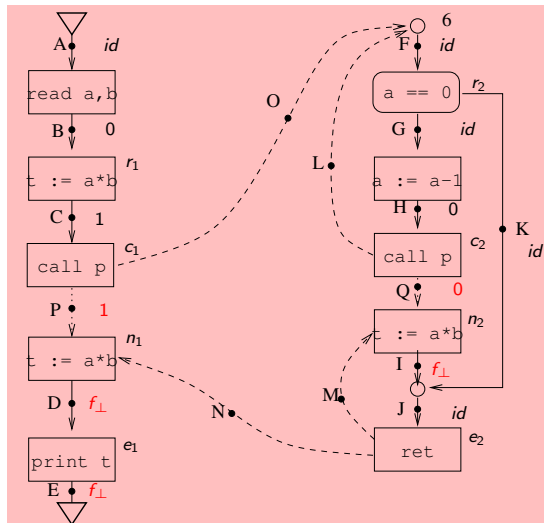
$$\begin{aligned}
 y_{F,F} &= id \\
 y_{F,G} &= id \circ y_{F,F} \\
 y_{F,K} &= id \circ y_{F,F} \\
 y_{F,H} &= \mathbf{0} \circ y_{F,G} \\
 y_{F,Q} &= y_{F,J} \circ y_{F,H} \\
 y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\
 y_{F,J} &= y_{F,I} \sqcup y_{F,K}
 \end{aligned}$$



Example: Computing $\phi_{r_p,N}$'s ($y_{r_p,N}^*$ to be precise) using Kildall's algo

$$\begin{aligned}
 y_{A,A} &= id \\
 y_{A,B} &= \mathbf{0} \circ y_{A,A} \\
 y_{A,C} &= \mathbf{1} \circ y_{A,B} \\
 y_{A,P} &= y_{F,J} \circ y_{A,C} \\
 y_{A,D} &= \mathbf{1} \circ y_{A,P} \\
 y_{A,E} &= id \circ y_{A,D}
 \end{aligned}$$

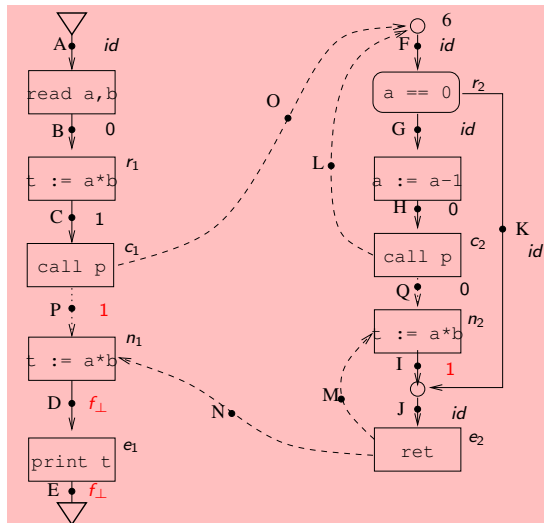
$$\begin{aligned}
 y_{F,F} &= id \\
 y_{F,G} &= id \circ y_{F,F} \\
 y_{F,K} &= id \circ y_{F,F} \\
 y_{F,H} &= \mathbf{0} \circ y_{F,G} \\
 y_{F,Q} &= y_{F,J} \circ y_{F,H} \\
 y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\
 y_{F,J} &= y_{F,I} \sqcup y_{F,K}
 \end{aligned}$$



Example: Computing $\phi_{r_p,N}$'s ($y_{r_p,N}^*$ to be precise) using Kildall's algo

$$\begin{aligned}
 y_{A,A} &= id \\
 y_{A,B} &= \mathbf{0} \circ y_{A,A} \\
 y_{A,C} &= \mathbf{1} \circ y_{A,B} \\
 y_{A,P} &= y_{F,J} \circ y_{A,C} \\
 y_{A,D} &= \mathbf{1} \circ y_{A,P} \\
 y_{A,E} &= id \circ y_{A,D}
 \end{aligned}$$

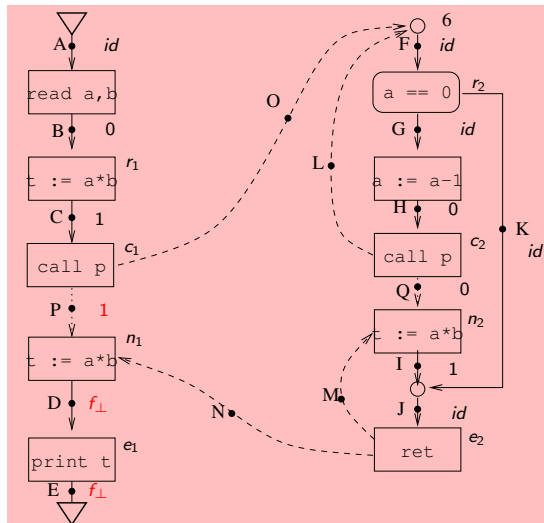
$$\begin{aligned}
 y_{F,F} &= id \\
 y_{F,G} &= id \circ y_{F,F} \\
 y_{F,K} &= id \circ y_{F,F} \\
 y_{F,H} &= \mathbf{0} \circ y_{F,G} \\
 y_{F,Q} &= y_{F,J} \circ y_{F,H} \\
 y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\
 y_{F,J} &= y_{F,I} \sqcup y_{F,K}
 \end{aligned}$$



Example: Computing $\phi_{r_p,N}$'s ($y_{r_p,N}^*$ to be precise) using Kildall's algo

$$\begin{aligned}
 y_{A,A} &= id \\
 y_{A,B} &= \mathbf{0} \circ y_{A,A} \\
 y_{A,C} &= \mathbf{1} \circ y_{A,B} \\
 y_{A,P} &= y_{F,J} \circ y_{A,C} \\
 y_{A,D} &= \mathbf{1} \circ y_{A,P} \\
 y_{A,E} &= id \circ y_{A,D}
 \end{aligned}$$

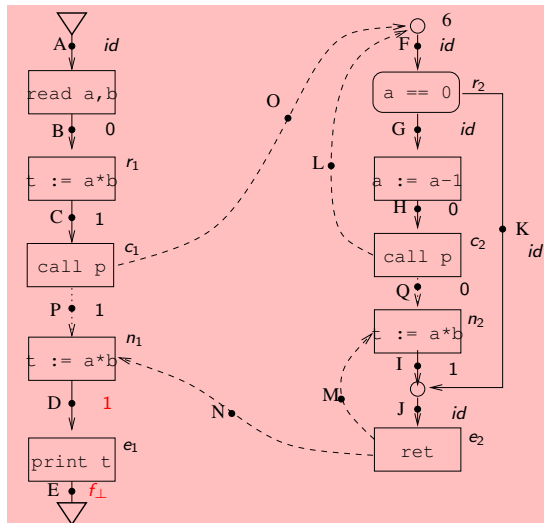
$$\begin{aligned}
 y_{F,F} &= id \\
 y_{F,G} &= id \circ y_{F,F} \\
 y_{F,K} &= id \circ y_{F,F} \\
 y_{F,H} &= \mathbf{0} \circ y_{F,G} \\
 y_{F,Q} &= y_{F,J} \circ y_{F,H} \\
 y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\
 y_{F,J} &= y_{F,I} \sqcup y_{F,K}
 \end{aligned}$$



Example: Computing $\phi_{r_p,N}$'s ($y_{r_p,N}^*$ to be precise) using Kildall's algo

$$\begin{aligned}
 y_{A,A} &= id \\
 y_{A,B} &= \mathbf{0} \circ y_{A,A} \\
 y_{A,C} &= \mathbf{1} \circ y_{A,B} \\
 y_{A,P} &= y_{F,J} \circ y_{A,C} \\
 y_{A,D} &= \mathbf{1} \circ y_{A,P} \\
 y_{A,E} &= id \circ y_{A,D}
 \end{aligned}$$

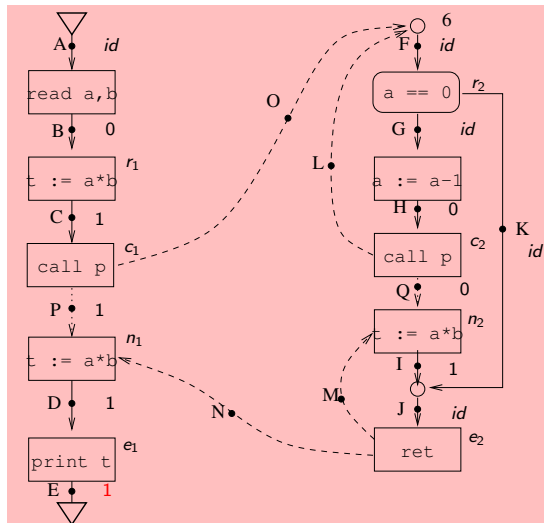
$$\begin{aligned}
 y_{F,F} &= id \\
 y_{F,G} &= id \circ y_{F,F} \\
 y_{F,K} &= id \circ y_{F,F} \\
 y_{F,H} &= \mathbf{0} \circ y_{F,G} \\
 y_{F,Q} &= y_{F,J} \circ y_{F,H} \\
 y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\
 y_{F,J} &= y_{F,I} \sqcup y_{F,K}
 \end{aligned}$$



Example: Computing $\phi_{r_p,N}$'s ($y_{r_p,N}^*$ to be precise) using Kildall's algo

$$\begin{aligned}
 y_{A,A} &= id \\
 y_{A,B} &= \mathbf{0} \circ y_{A,A} \\
 y_{A,C} &= \mathbf{1} \circ y_{A,B} \\
 y_{A,P} &= y_{F,J} \circ y_{A,C} \\
 y_{A,D} &= \mathbf{1} \circ y_{A,P} \\
 y_{A,E} &= id \circ y_{A,D}
 \end{aligned}$$

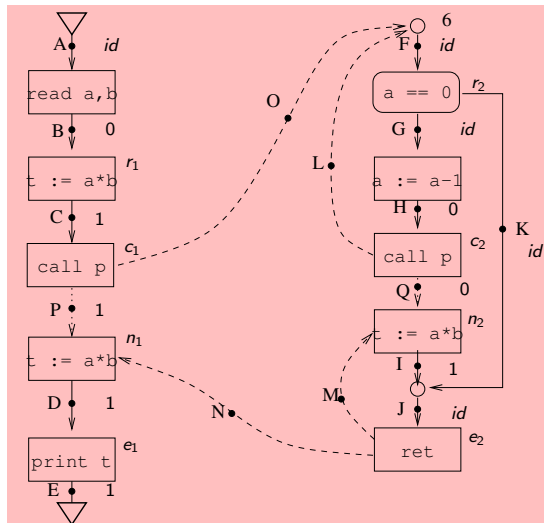
$$\begin{aligned}
 y_{F,F} &= id \\
 y_{F,G} &= id \circ y_{F,F} \\
 y_{F,K} &= id \circ y_{F,F} \\
 y_{F,H} &= \mathbf{0} \circ y_{F,G} \\
 y_{F,Q} &= y_{F,J} \circ y_{F,H} \\
 y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\
 y_{F,J} &= y_{F,I} \sqcup y_{F,K}
 \end{aligned}$$



Example: Computing $\phi_{r_p,N}$'s ($y_{r_p,N}^*$ to be precise) using Kildall's algo

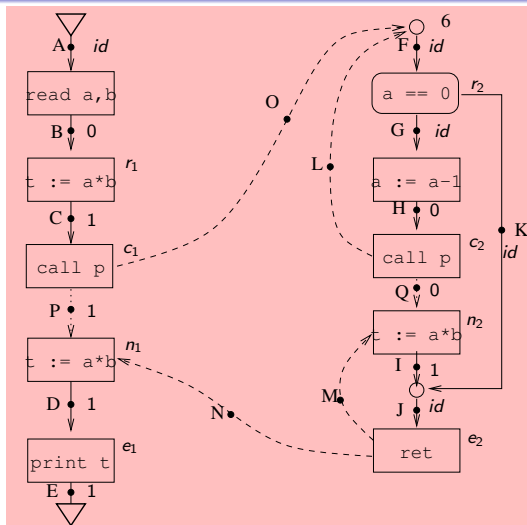
$$\begin{aligned}
 y_{A,A} &= id \\
 y_{A,B} &= \mathbf{0} \circ y_{A,A} \\
 y_{A,C} &= \mathbf{1} \circ y_{A,B} \\
 y_{A,P} &= y_{F,J} \circ y_{A,C} \\
 y_{A,D} &= \mathbf{1} \circ y_{A,P} \\
 y_{A,E} &= id \circ y_{A,D}
 \end{aligned}$$

$$\begin{aligned}
 y_{F,F} &= id \\
 y_{F,G} &= id \circ y_{F,F} \\
 y_{F,K} &= id \circ y_{F,F} \\
 y_{F,H} &= \mathbf{0} \circ y_{F,G} \\
 y_{F,Q} &= y_{F,J} \circ y_{F,H} \\
 y_{F,I} &= \mathbf{1} \circ y_{F,Q} \\
 y_{F,J} &= y_{F,I} \sqcup y_{F,K}
 \end{aligned}$$



Example: Computing JVP values (x_N^* 's to be precise)

$$\begin{aligned}
 x_A &= 0 \\
 x_B &= \mathbf{0}(x_A) \\
 x_C &= \mathbf{1}(x_A) \\
 x_P &= \mathbf{1}(x_A) \\
 x_D &= \mathbf{1}(x_A) \\
 x_E &= \mathbf{1}(x_A) \\
 \\
 x_F &= x_C \sqcup x_H \\
 x_G &= id(x_F) \\
 x_K &= id(x_F) \\
 x_H &= \mathbf{0}(x_F) \\
 x_Q &= \mathbf{0}(x_F) \\
 x_I &= \mathbf{1}(x_F) \\
 x_J &= id(x_F).
 \end{aligned}$$



Example: Computing JVP values (x_N^* 's to be precise)

$$\begin{aligned}
 x_A &= 0 \\
 x_B &= \mathbf{0}(x_A) \\
 x_C &= \mathbf{1}(x_A) \\
 x_P &= \mathbf{1}(x_A) \\
 x_D &= \mathbf{1}(x_A) \\
 x_E &= \mathbf{1}(x_A) \\
 \\
 x_F &= x_C \sqcup x_H \\
 x_G &= id(x_F) \\
 x_K &= id(x_F) \\
 x_H &= \mathbf{0}(x_F) \\
 x_Q &= \mathbf{0}(x_F) \\
 x_I &= \mathbf{1}(x_F) \\
 x_J &= id(x_F).
 \end{aligned}$$

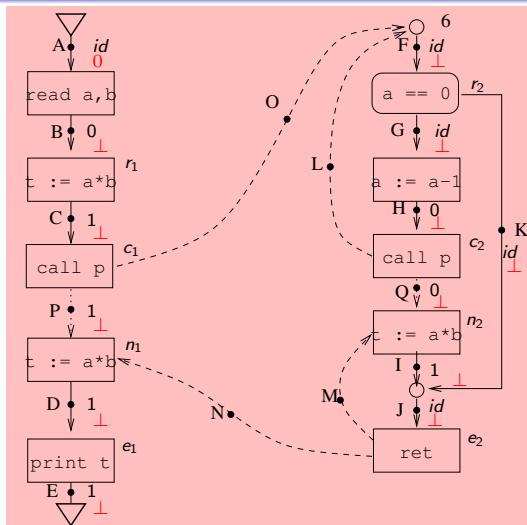


Fig shows initial (red) and final (blue) values.

Example: Computing JVP values (x_N^* 's to be precise)

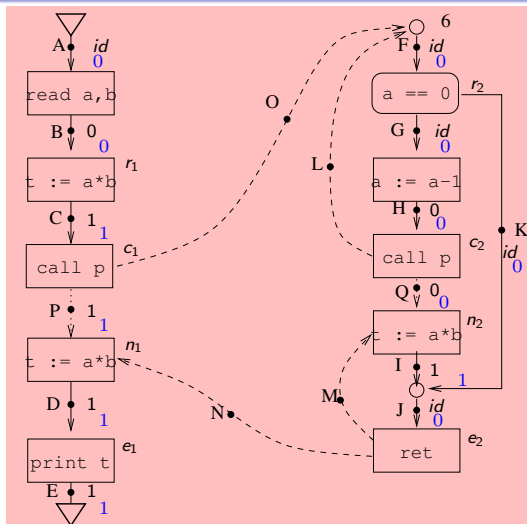
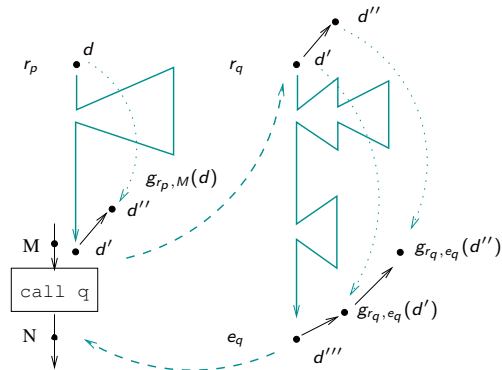
$$\begin{aligned} X_A &= 0 \\ X_B &= \mathbf{0}(x_A) \\ X_C &= \mathbf{1}(x_A) \\ X_P &= \mathbf{1}(x_A) \\ X_D &= \mathbf{1}(x_A) \\ X_E &= \mathbf{1}(x_A) \\ X_F &= x_C \sqcup x_H \\ X_G &= id(x_F) \\ X_K &= id(x_F) \\ X_H &= \mathbf{0}(x_F) \\ X_Q &= \mathbf{0}(x_F) \\ X_I &= \mathbf{1}(x_F) \\ X_J &= id(x_F). \end{aligned}$$


Fig shows initial (red) and final (blue) values.

Proof

- 1 For part 1:
 - Let $g_{r_p, N}$'s be **any** monotone solution to Eq (1).
 - Sufficient to prove:
For each proc p in P , $d \in D$, and ρ an IVP_0 path from r_p to N ,
 $f_\rho(d) \leq g_{r_p, N}(d)$.
 - Proof by induction on length of path ρ .
- 2 For part 2: Prove that $\phi_{r_p, N}$'s are a solution to Eq (1), and hence they will dominate the least solution.



Correctness and Algo II

Consider Eq (2)':

$$\begin{aligned} x_1 &= d_0 \\ x_{r_p} &= \bigsqcup_{\text{calls } C \text{ to } p} x_C \\ x_N &= y_{r_p, N}^*(x_{r_p}) \quad \text{for } N \in N_p - \{r_p\}. \end{aligned}$$

(Recall that $y_{r_p, N}^*$'s are the least solution of Eq (1).)

- $\bar{f}$ induced by Eq (2)' is a monotone function on the complete lattice $(\bar{D}, \leq)$.
- LFP / least solution (say x_N^* 's) exists by Knaster-Tarski.

Claim

JVP values are the least solution to Eq (2)' (i.e. $\text{JVP}_N = x_N^*$) when f_{MN} 's are distributive. Otherwise $\text{JVP}_N \leq x_N^*$ for each N .

Kleene/Kildall's algo will compute x_N^* 's (assuming D finite).

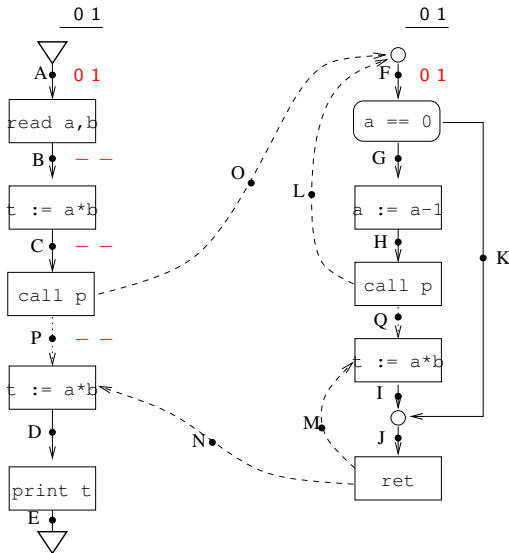
Summary of functional approach

- Uses a two step approach
 - 1 Compute $\phi_{r_p, N}$'s (JVCPs for each function).
 - 2 Compute x_n 's (JVPs) at each point.

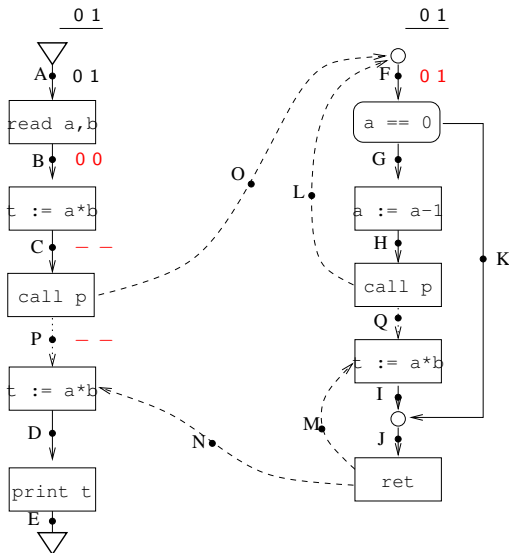
Summary of conditions: For each property (column heading), the conjunction of the ticked conditions (row headings) are sufficient to ensure the property.

	Termination	Least Sol of Eq(2) $\geq$ JVP	Least Sol of Eq(2) = JVP
f_{MN} 's monotonic	✓	✓	✓
Finite underlying lattice	✓		
f_{MN} 's distributive			✓

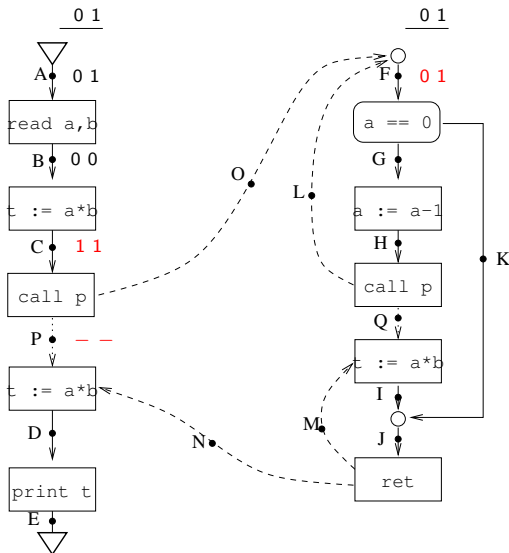
Viewing ϕ computation as a table



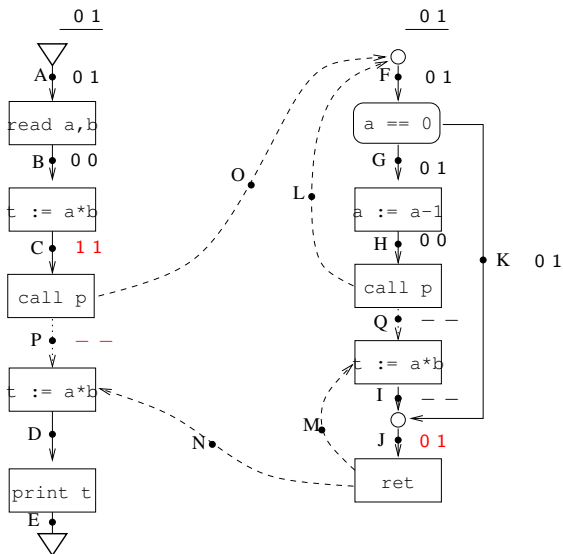
Viewing ϕ computation as a table



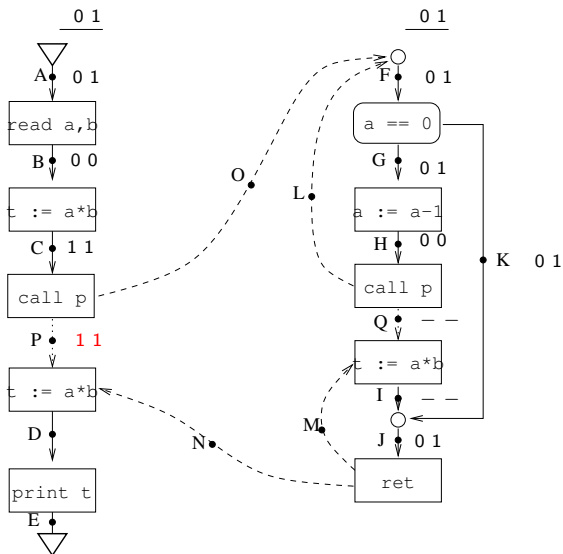
Viewing ϕ computation as a table



Viewing ϕ computation as a table



Viewing ϕ computation as a table



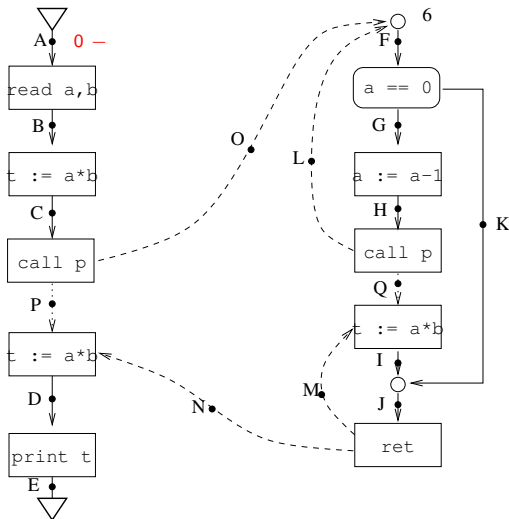
Iterative/Tabulation Approach

- Main idea: **de-couple** the propagation of function rows.
- Maintain a **table** of values representing the current value of $\phi_{r_p, N}$ for each program point N in procedure p .
- Expand column for data value d in procedure p only if d is reachable at r_p .
- Informally, at N in procedure p , the table has an entry $d \mapsto d'$ if we have seen
 - 1 valid paths ρ from r_1 to r_p with $\bigsqcup_{\rho} f_{\rho}(d_0) = d$, and
 - 2 valid and complete paths δ from r_p to N with $\bigsqcup_{\delta} f_{\delta}(d) = d'$.

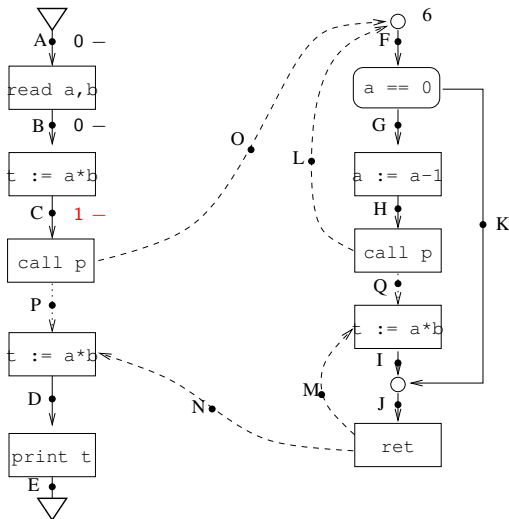
Iterative/Tabulation Approach

- Apply Kildall's algo with initial value of $d_0 \mapsto d_0$ at r_1 .
- Propagating value d across a call to procedure p : (a) begin a column for d at root of p if not already there; Also (b) if d is mapped to d' at the end of p , then propagate d' to the return site of the call.
- Propagating across return nodes from procedure p : value d' in column for d is propagated to each column at a return site of a call to procedure p that has the value d in the preceding entry.

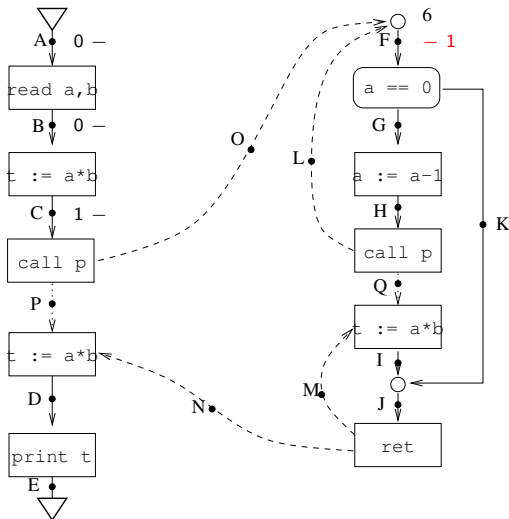
Example: Computing ϕ 's iteratively: 1



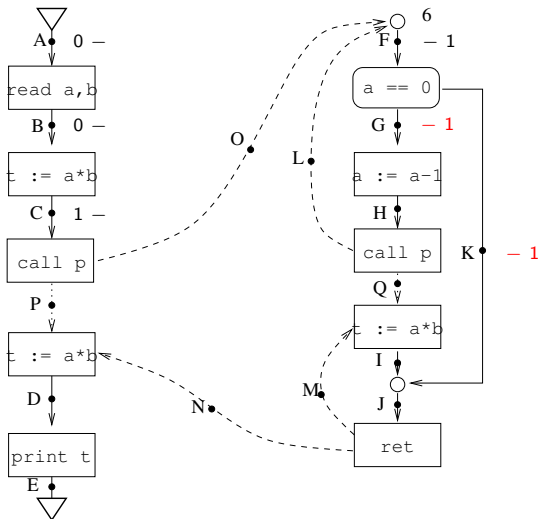
Example: Computing ϕ 's iteratively: 2



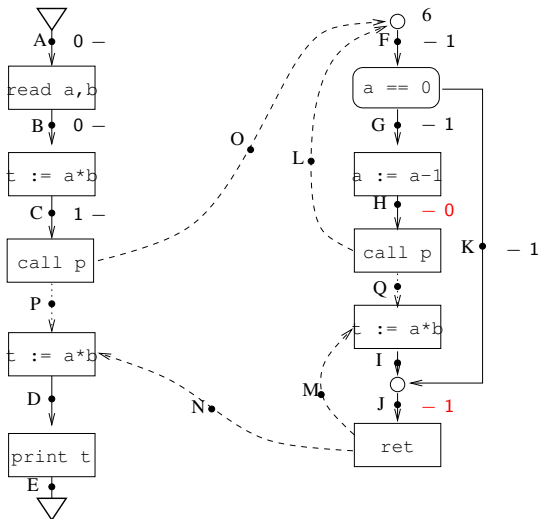
Example: Computing ϕ 's iteratively: 4



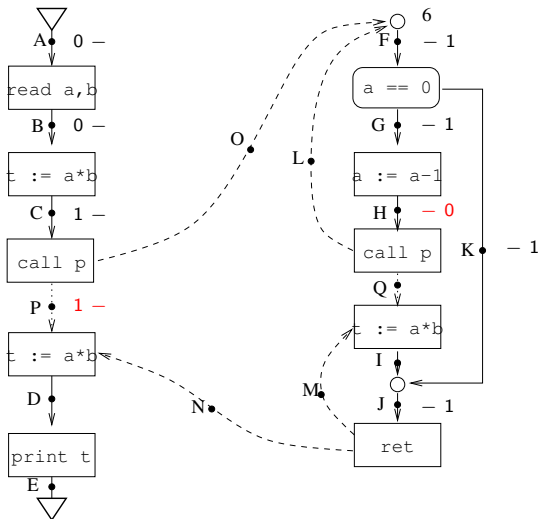
Example: Computing ϕ 's iteratively: 5



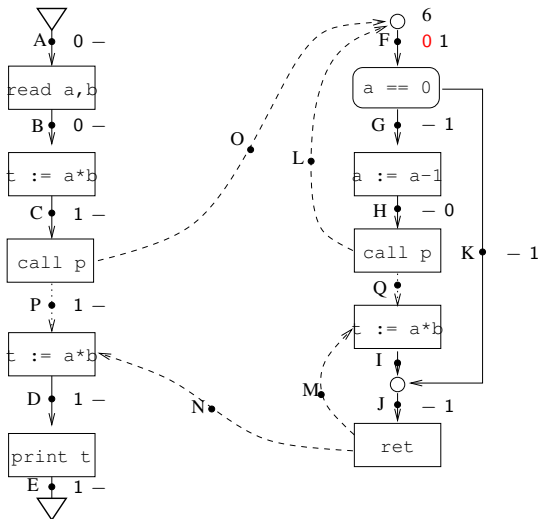
Example: Computing ϕ 's iteratively: 6



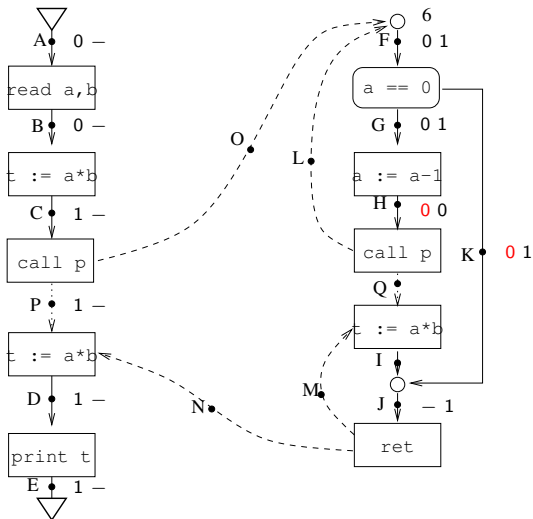
Example: Computing ϕ 's iteratively: 7



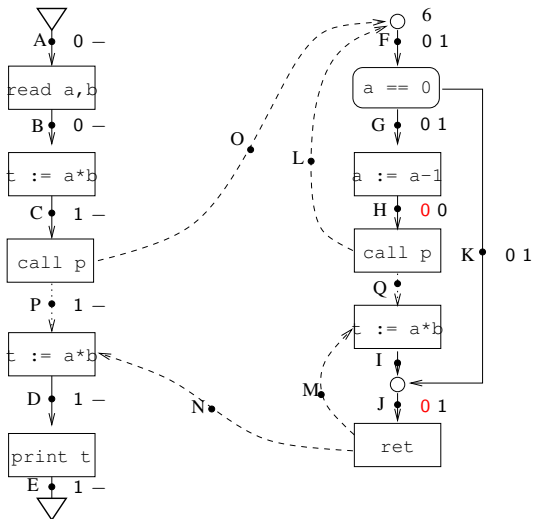
Example: Computing ϕ 's iteratively: 8



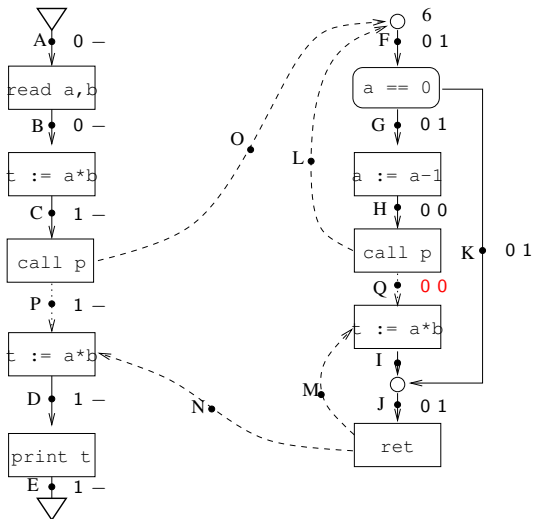
Example: Computing ϕ 's iteratively: 9



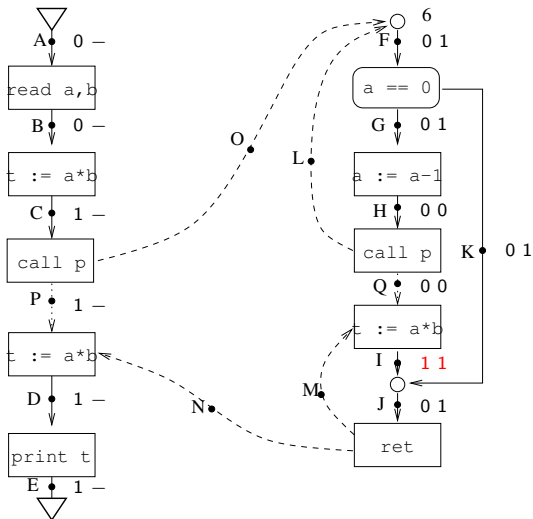
Example: Computing ϕ 's iteratively: 10



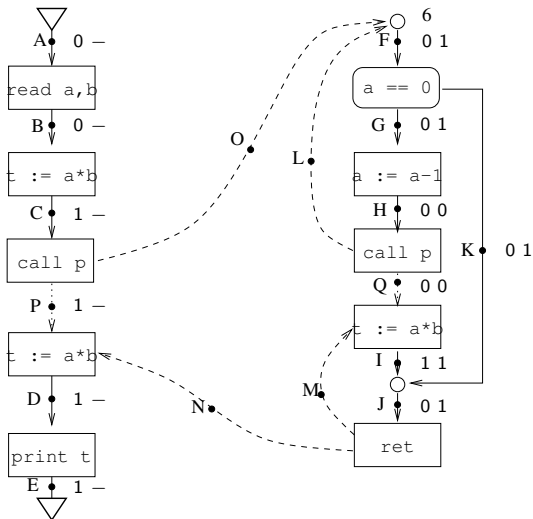
Example: Computing ϕ 's iteratively: 11



Example: Computing ϕ 's iteratively: 12

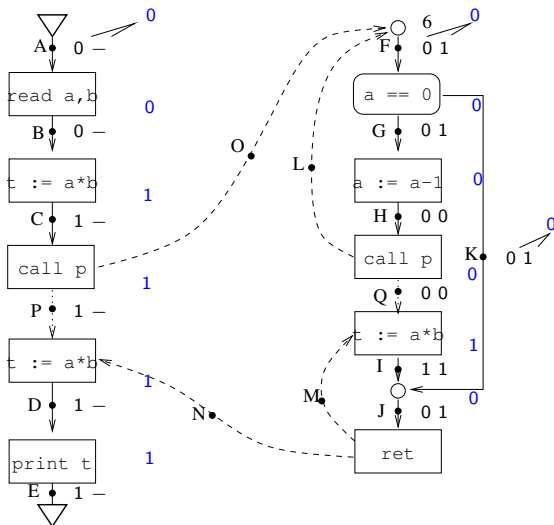


Example: Computing ϕ 's iteratively: 13



Example: Finally compute x_N 's from ϕ values

At each point N take join of reachable $\phi_{r_p, N}$ values.



Correctness of iterative algo

- Iterative algo terminates provided underlying lattice is finite.
- It computes the $y_{r_p, N}^*$'s (where $y_{r_p, N}^*$'s are the least solution to Eq (1)) "partially": If it maps d to $d' \neq \perp$ then $y_{r_p, N}^*(d) = d'$.
- The JVP values it gives (say z_N 's) are such that

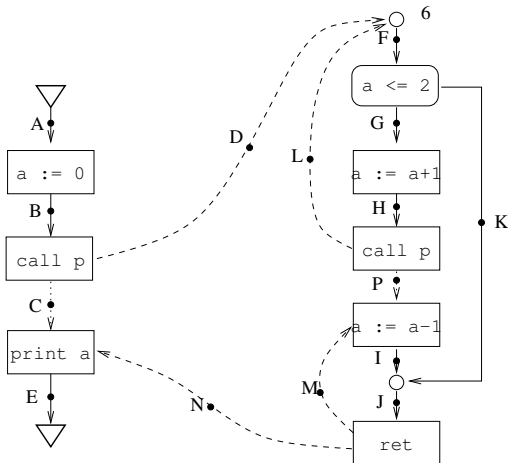
$$\text{JVP}_N \leq z_N \leq x_N^*$$

(where x_N^* 's are the solution to Eq (2')).

- If underlying transfer functions are distributive it computes $\phi_{r_p, N}$'s correctly (though partially), and the JVP values correctly.
- It thus computes an overapproximation of JVP for monotonic transfer functions, and exact JVP when transfer functions are distributive.

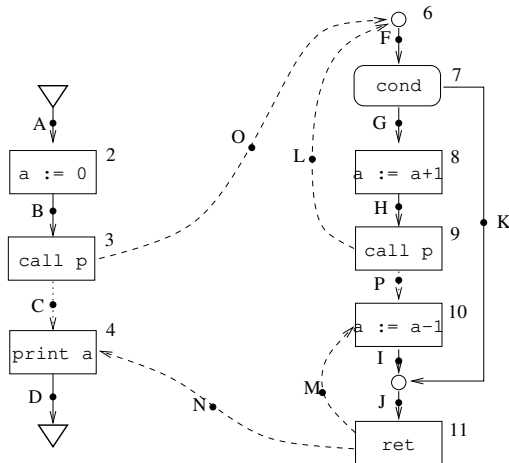
Exercise 1: Iterative algo

Run the iterative algo to do constant propagation analysis for the program below with initial value $\emptyset$.



Exercise 2: Functional vs Iterative algo

Run the functional and iterative algos to do constant propagation analysis for the program below with initial value $\emptyset$:



Comparing functional vs iterative approach

- Functional algo can terminate even when underlying lattice is infinite, provided we can represent and compose/join functions “symbolically”.
- Iterative is typically more efficient than functional since it only computes $\phi_{r_p, N}$'s for values **reachable** at start of procedure.