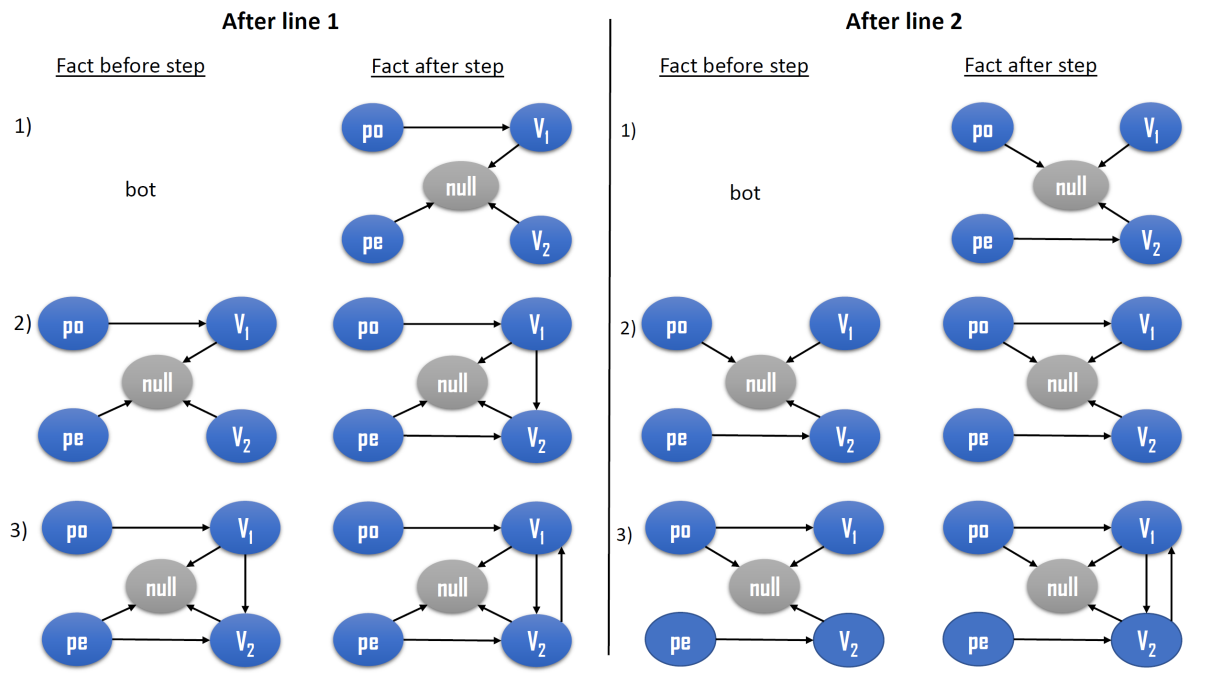
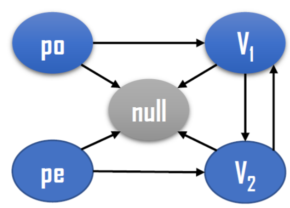
Aman Choudhary 17920 aman@iisc.ac.in

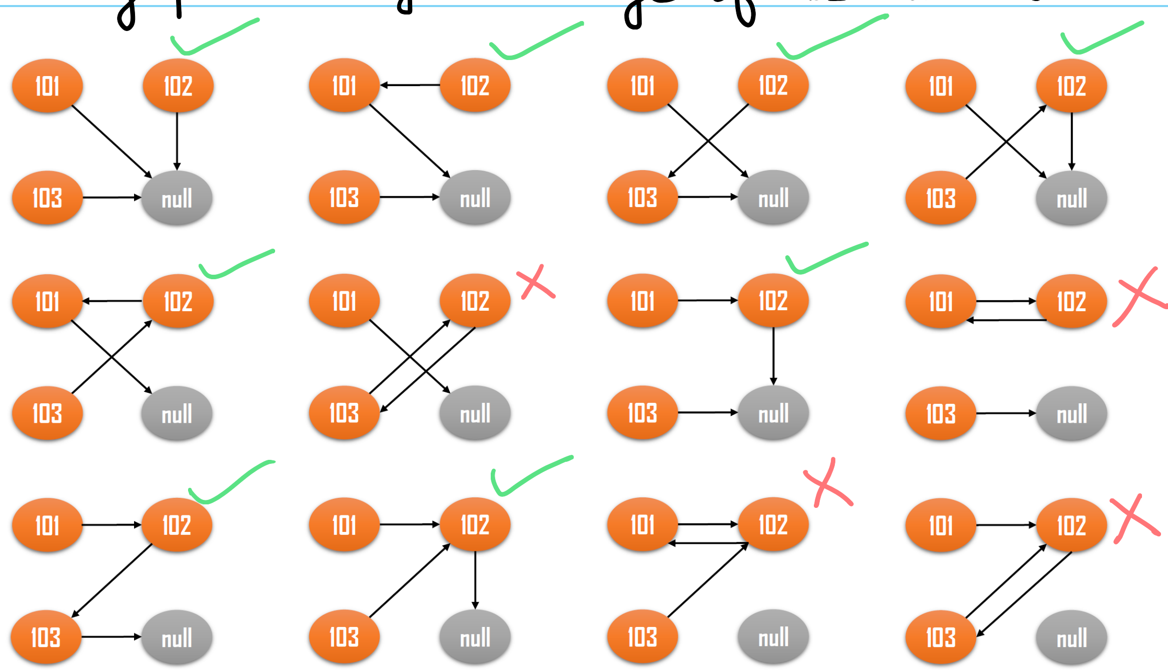
1 (a)



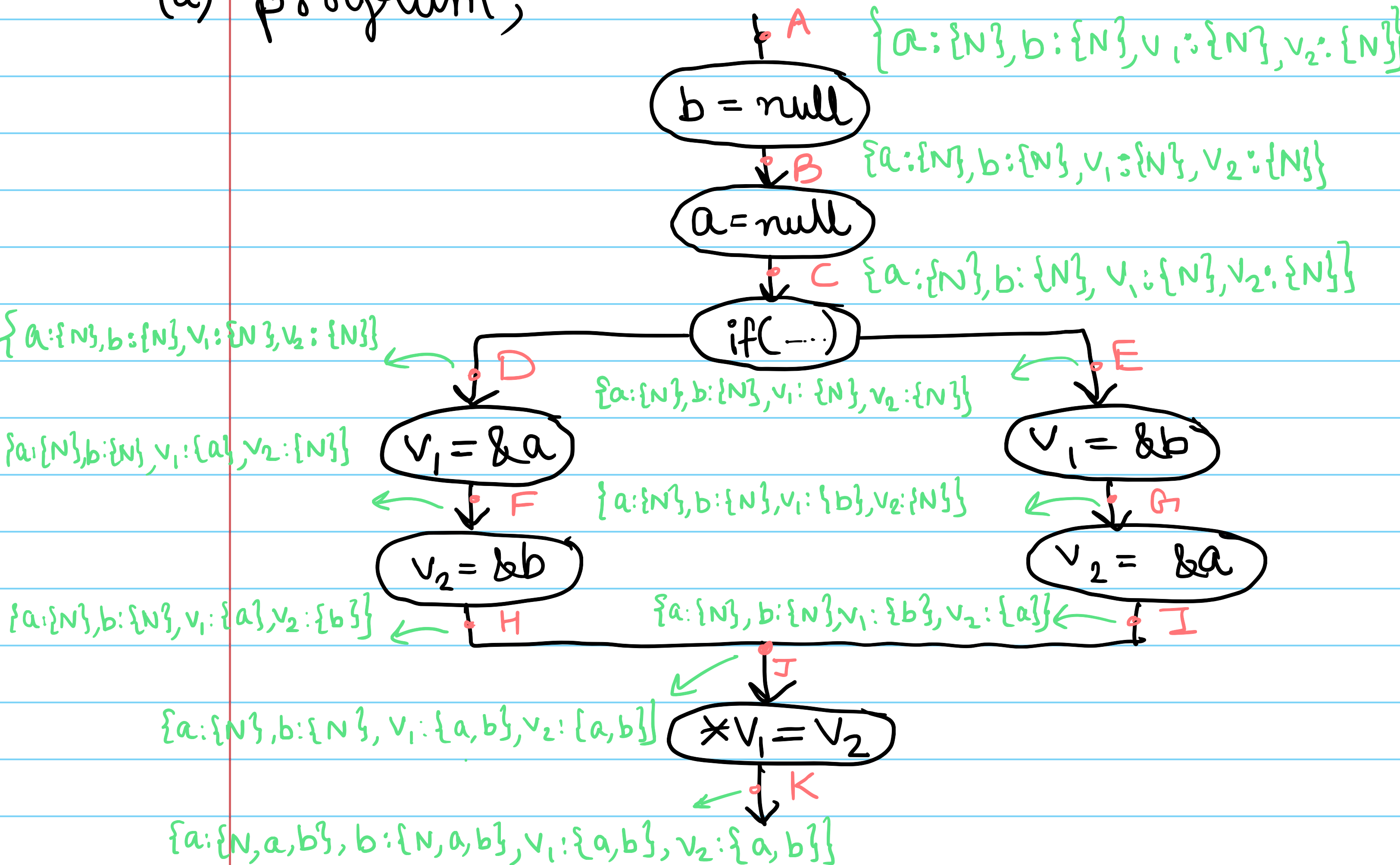
(b) Points-to graph at end of program:



(c) Concrete graphs for gamma image of abstract state



(d) Result of running Kildall on CFG of example program,



AbsDataState at H = d_1 = $\{a:\{N\}, b:\{N\}, v_1:\{a\}, v_2:\{b\}\}$

AbsDataState at I = d_2 = $\{a:\{N\}, b:\{N\}, v_1:\{b\}, v_2:\{a\}\}$

AbsDataState at J = $d_1 \sqcup d_2$ = $\{a:\{N\}, b:\{N\}, v_1:\{a,b\}, v_2:\{a,b\}\}$

Transferring across $(*v_1 = v_2)$,

$$f(d_1) = \{a:\{b\}, b:\{N\}, v_1:\{a\}, v_2:\{b\}\}$$

$$f(d_2) = \{a:\{N\}, b:\{a\}, v_1:\{b\}, v_2:\{a\}\}$$

$$f(d_1 \sqcup d_2) = \{a:\{N,a,b\}, b:\{N,a,b\}, v_1:\{a,b\}, v_2:\{a,b\}\} \quad \text{—— (1)}$$

$$f(d_1) \sqcup f(d_2) = \{a:\{N,b\}, b:\{N,a\}, v_1:\{a,b\}, v_2:\{a,b\}\} \quad \text{—— (2)}$$

From (1) & (2), $f(d_1 \sqcup d_2) \neq f(d_1) \sqcup f(d_2)$

So, transfer function 'f', for ($*v_1 = v_2$) is non-distributive

Abstract JOP at K $= f(d_1) \sqcup f(d_2)$

$= \{a:\{N,b\},\ b:\{N,a\},\ v_1:\{a,b\},\ v_2:\{a,b\}\}$

Kildall computed JOP at K $= f(d_1 \sqcup d_2)$

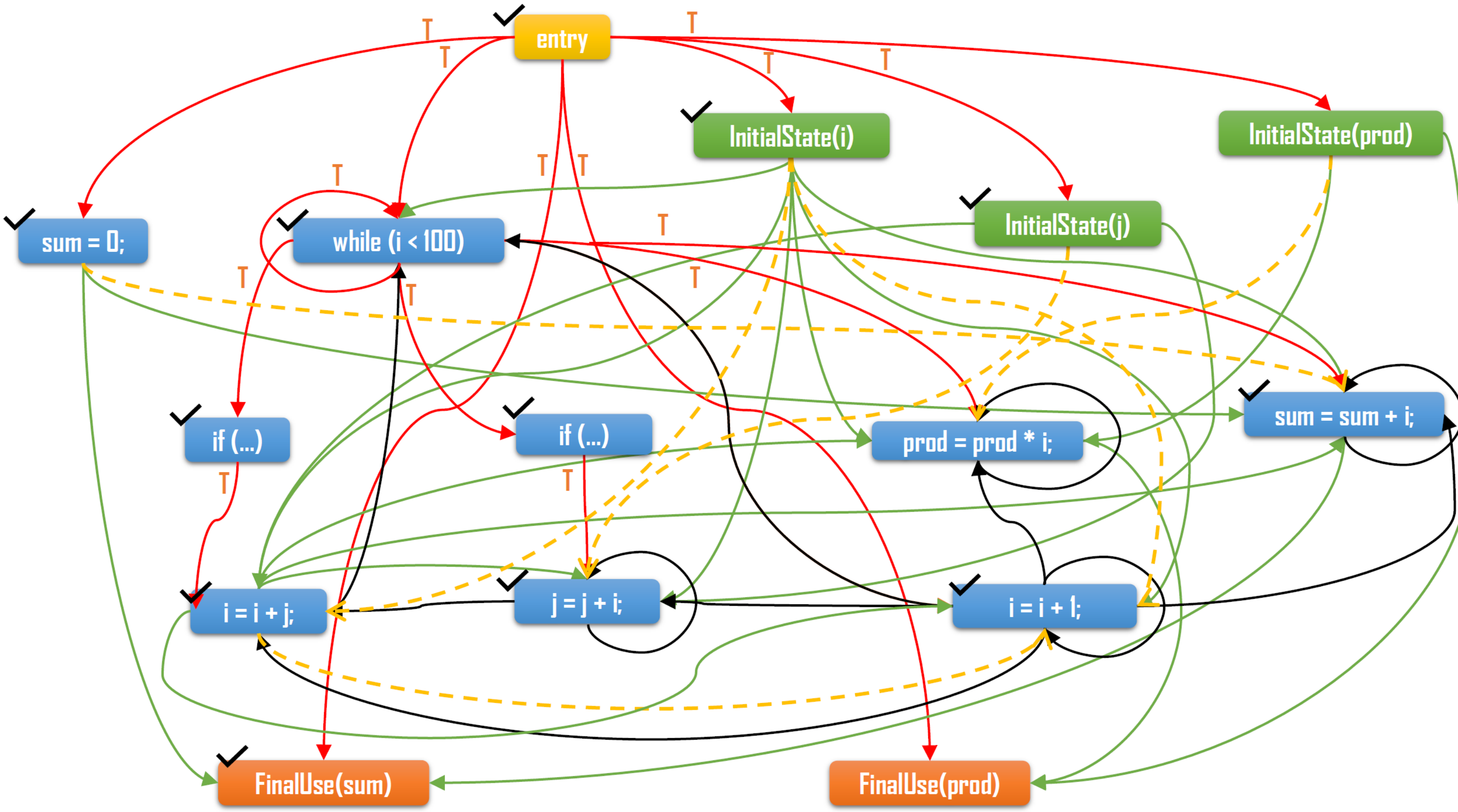
$= \{a:\{N,a,b\},\ b:\{N,a,b\},\ v_1:\{a,b\},\ v_2:\{a,b\}\}$

$$f(d_1 \sqcup d_2) \sqsupset f(d_1) \sqcup f(d_2)$$

②

— Control Edges
— Loop independent Edges
— Loop carried edges
- - - Def order edges

Ticked vertices represent the slice



Given: PDG(P) is isomorphic to PDG(P′).

③ (a) Let us assume P′ is not a permutation of P. If $P = b_1 b_2 \cdots b_n$, then for P′ to not be a permutation of P,

(i) P′ can have unequal no. of blocks. In this case no. of vertices in PDG(P) ≠ vertices in PDG(P′). This contradicts the fact that PDG(P) is isomorphic to PDG(P′).

(ii) P′ has at least one block b′ such that $b' \notin \{b_1, b_2, \ldots, b_n\}$. In this case, we cannot perform a bijection between vertices and edges of P and P′, which again violates the isomorphic property.

From (i) and (ii), we conclude that P′ and P must be permutations of each other.

(b) Programs whose PDG are isomorphic to PDG of given program: $b_1 b_2 b_4 b_3$, $b_2 b_1 b_3 b_4$, $b_2 b_1 b_4 b_3$, $b_1 b_4 b_2 b_3$

(c) Let, $S_v^P = \{b \mid b$ defines 'v' and comes before b_j in $P\}$
$S_v^{P'} = \{b \mid b$ defines 'v' and comes before b_j in $P'\}$

We need to show that: (i) $S_v^P = S_v^{P'}$, (ii) Order of blocks belonging to S_v^P in P is same as that of blocks $\in S_v^{P'}$ in P′.

Part (i) ① Let there exist $b_1 \in S_v^P$ such that $b_1 \notin S_v^{P'}$. Thus, exist there exists a loop independent flow edge from an assignment node in b_1, v_1 to an assignment / conditional node in b_j, v_2 i.e. $v_1 \xrightarrow[l_i]{} v_2$, in PDG(P). However since $b_1 \notin S_v^{P'}$, this edge is not present in PDG(P′). This is in contradiction with the isomorphic property, hence not possible.

② Similarly we can prove that there cannot any block b_2 such that $b_2 \in S_v^{P'}$ but $b_2 \notin S_v^{P}$.

From ① & ②, $S_v^P = S_v^{P'}$.

Part (ii) Let there exist blocks $b_1, b_2 \in S_v^P (= S_v^{P'})$ such that b_1 occurs before b_2 in P, but b_1 occurs after b_2 in P'. Since $b_1, b_2 \in S_v^P$, they both define 'v', and that definition would reach b_j. Hence, we would have flow edges $b_1 \xrightarrow[li]{} b_j$ and $b_2 \xrightarrow[li]{} b_j$. As a result we would have $b_1 \xrightarrow[do(b_j)]{} b_2$ in PDG(P) and $b_2 \xrightarrow[do(b_j)]{} b_1$ in PDG(P'). This is not possible as it violates the isomorphic condition. Hence, order of blocks belonging to S_v^P in P would be identical to that of blocks from $S_v^{P'}$ in P'.

(d) We can perform induction on the number of blocks.

Inductive Hypothesis: For any block b_i, the 'if' condition evaluates to the same value, and the assignment statement in the block assigns the same value to the LHS in both P and P', here $1 \leq i \leq n$, where n is total number of blocks in P.

Base Case: For $i=1$, i.e. the first block b_1, there are no defining blocks, it recieves values for its variables from initial states. Since, both P and P' have same initial state, b_1 will recieve same inputs in P and P'. As a result, the result of both conditional and assignment would be identical in both P and P'.

Inductive Case: Let the values computed by conditionals & all assignments be identical for P and P', for all b_k, $1 \leq k \leq i-1$.

→ Let us define a criterion, $S = \{v_c, v_a\}$ where v_c and v_a are the conditional and assignment node in b_i respectively. Let, P_1 be a slice of P with criterion S. Let, P_2 be a slice of P' with criterion S.

→ Now, $PDG(P_1) = \left\{ \bigcup_v S_v^P \mid \text{variable 'v' occurs in } v_c \text{ or } v_a \right\}$

Similarly,

$$PDG(P_2) = \left\{ \bigcup_v S_v^{P'} \mid \text{variable 'v' occurs in } v_c \text{ or } v_a \right\}$$

From 3(c), we can conclude that $PDG(P_1)$ is isomorphic to $PDG(P_2)$ as $S_v^P = S_v^{P'}$ for all variables 'v' in v_a or v_c occuring in b_i.

→ Using theorem 1 (slide 35/41), P_1 is a slice of P_2 wrt S, as $PDG(P_1)$ is isomorphic to $PDG(P_2) \equiv (PDG(P_2)/S)$. Similarly, P_2 is a slice of P_1.

→ From definition of slice (slide 28/41), for each node 'v' in P_1, P_2 computes the same sequence of values at its copy of 'v' as P_1 does at 'v', and vice versa. Hence, inductive case is true.

Thus, the hypothesis is true.