Part 5

IPDFA Using Value Contexts

Value Contexts: Key Ideas

Consider call chains σ_1 and σ_2 reaching S_p

- Data flow value invariant:

  If the data flow reaching S_p along σ_1 and σ_2 are identical, then

Value Contexts: Key Ideas

Consider call chains σ_1 and σ_2 reaching S_p

- Data flow value invariant:

  If the data flow reaching S_p along σ_1 and σ_2 are identical, then

  - the data flow values reaching E_p for the two contexts will also be identical

Value Contexts: Key Ideas

Consider call chains σ_1 and σ_2 reaching S_p

- Data flow value invariant:

  If the data flow reaching S_p along σ_1 and σ_2 are identical, then

  - the data flow values reaching E_p for the two contexts will also be identical

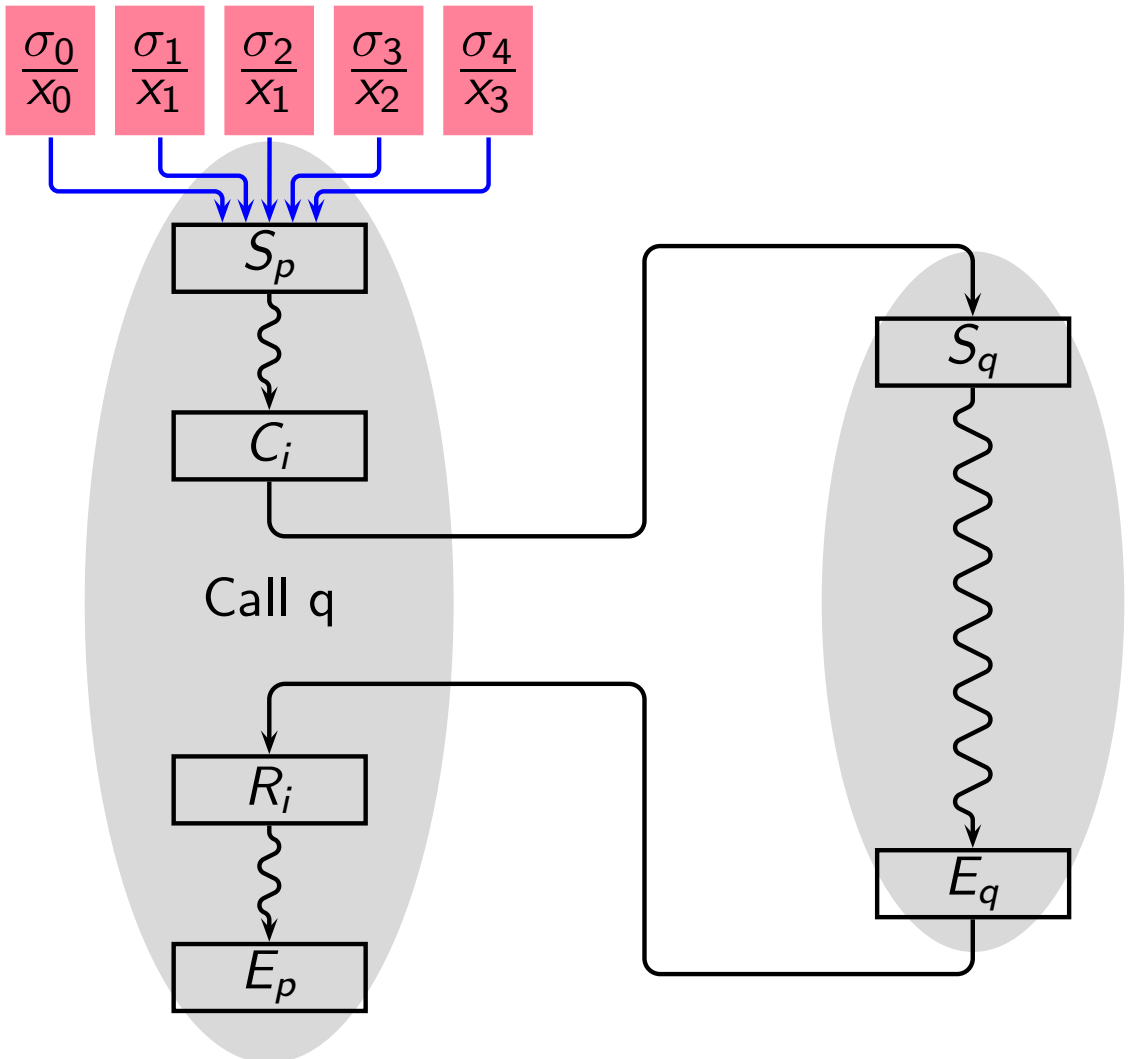
- We can reduce the amount of effort by using

  - Data flow values at S_p as value contexts
  - Maintaining distinct data flow values in p for each value context

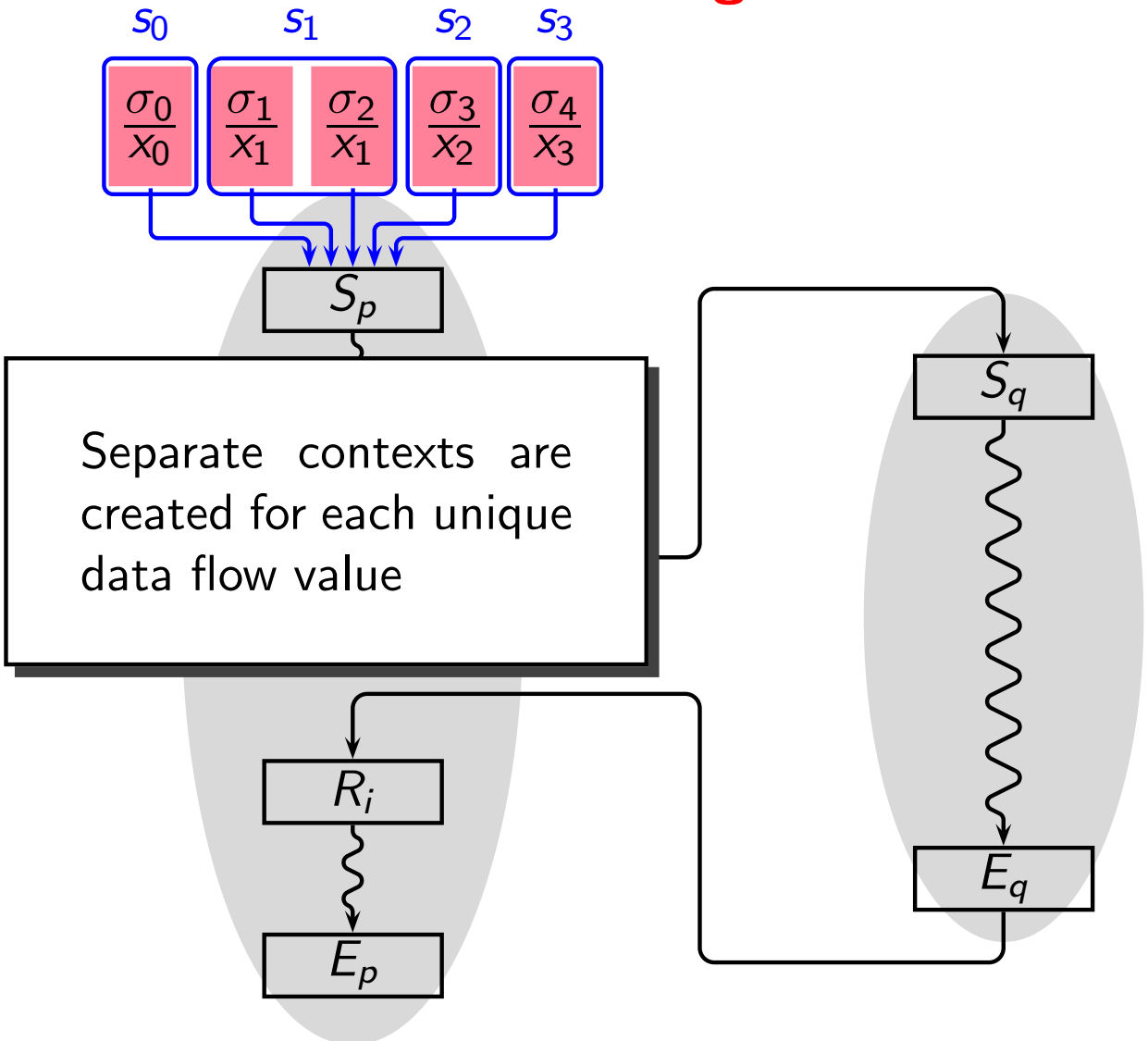
Interprocedural Data Flow Analysis Using Value Contexts

- A value context is defined by a particular input data flow value reaching a procedure
- It is used to enumerate the summary flow functions in terms of (input $\mapsto$ output) pairs
- In order to compute these pairs, data flow analysis within a procedure is performed separately for each context (i.e. input data flow value)
- When a new call to a procedure is encounterd, the pairs are consulted do decide if the procedure needs to be analysed again
  - If it was already analysed once for the input value, output can be directly processed
  - Otherwise, a new context is created and the procedure is analysed for this new context

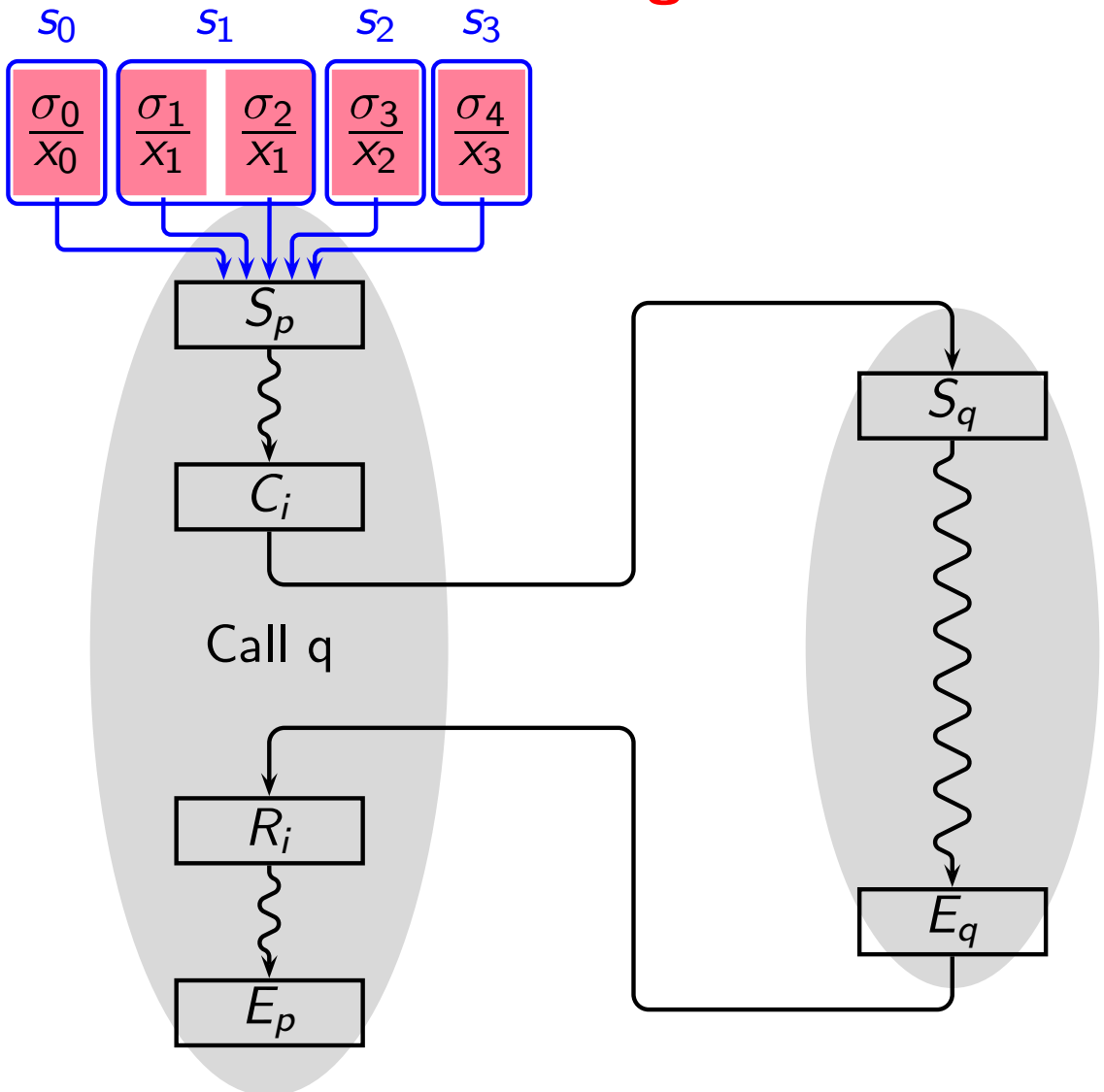
Understanding Value Contexts



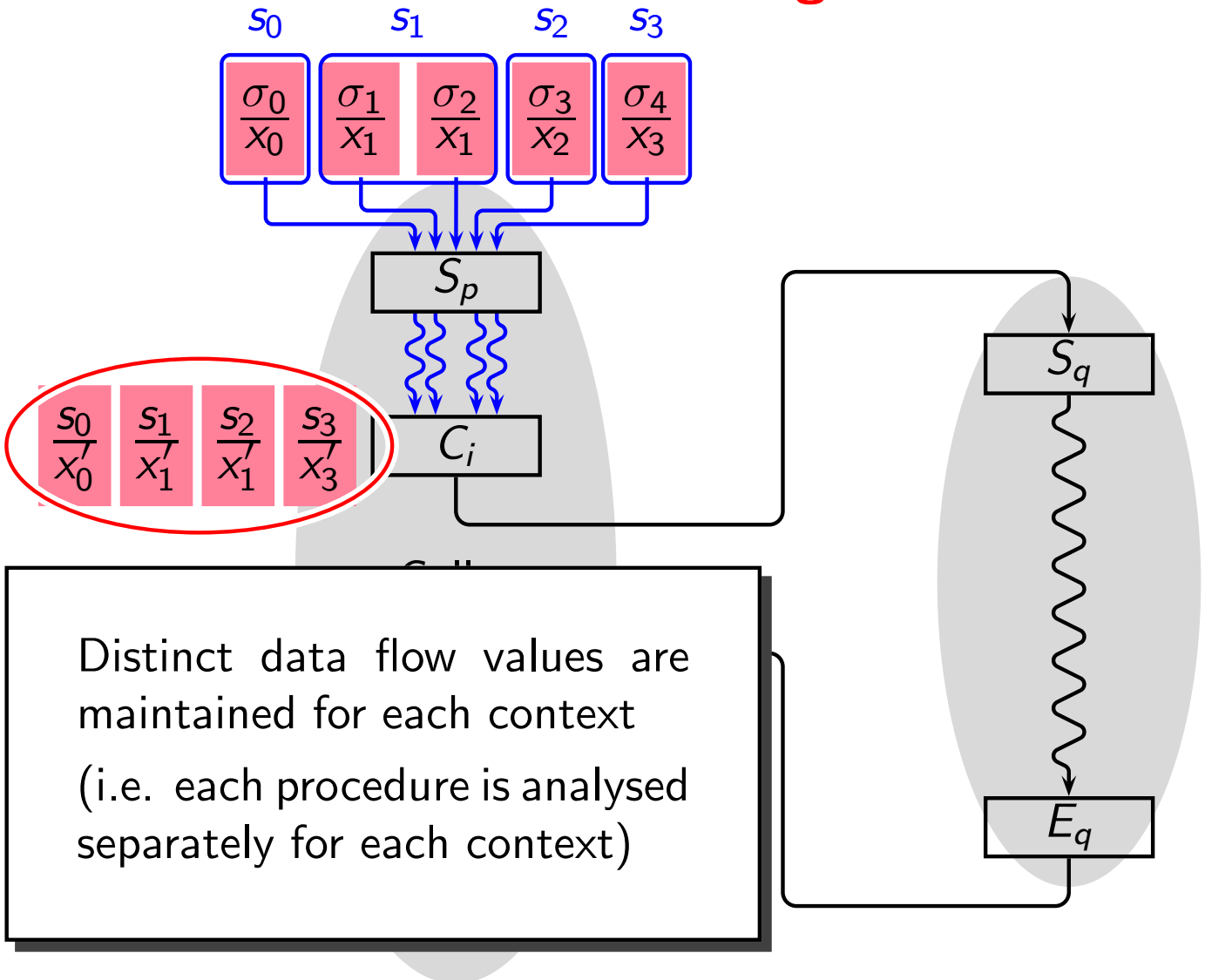
Understanding Value Contexts



Understanding Value Contexts



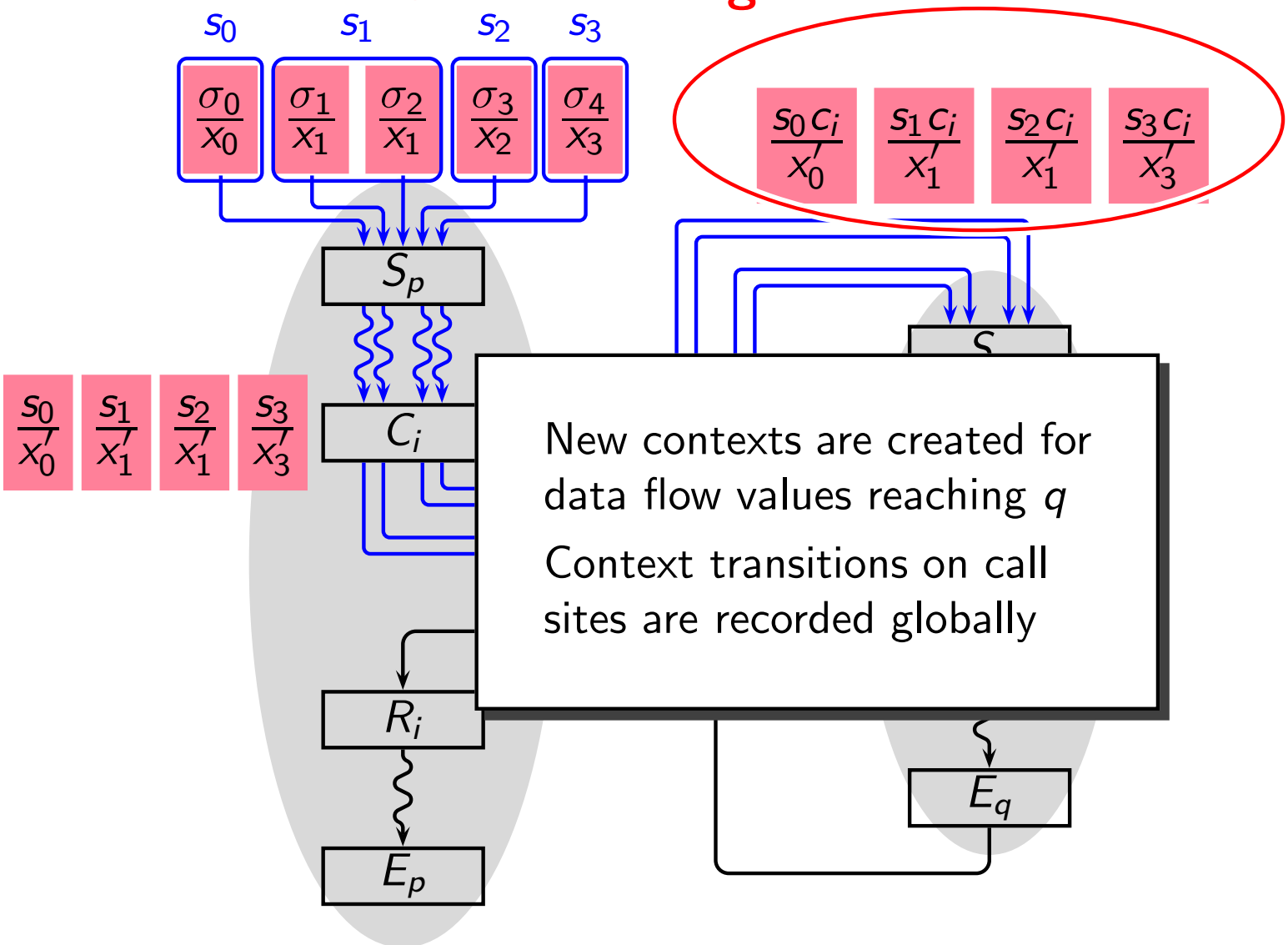
Understanding Value Contexts



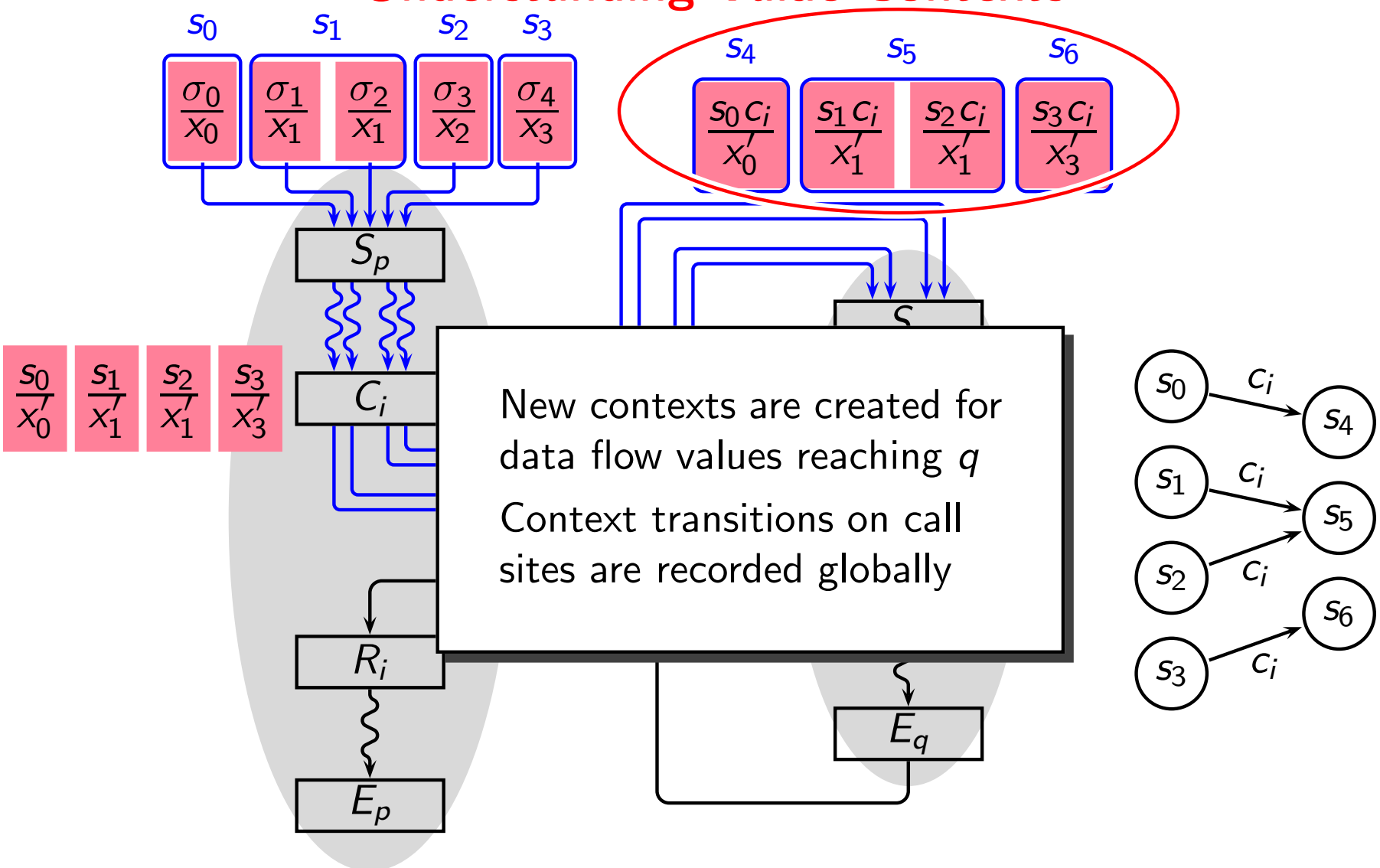
Distinct data flow values are maintained for each context

(i.e. each procedure is analysed separately for each context)

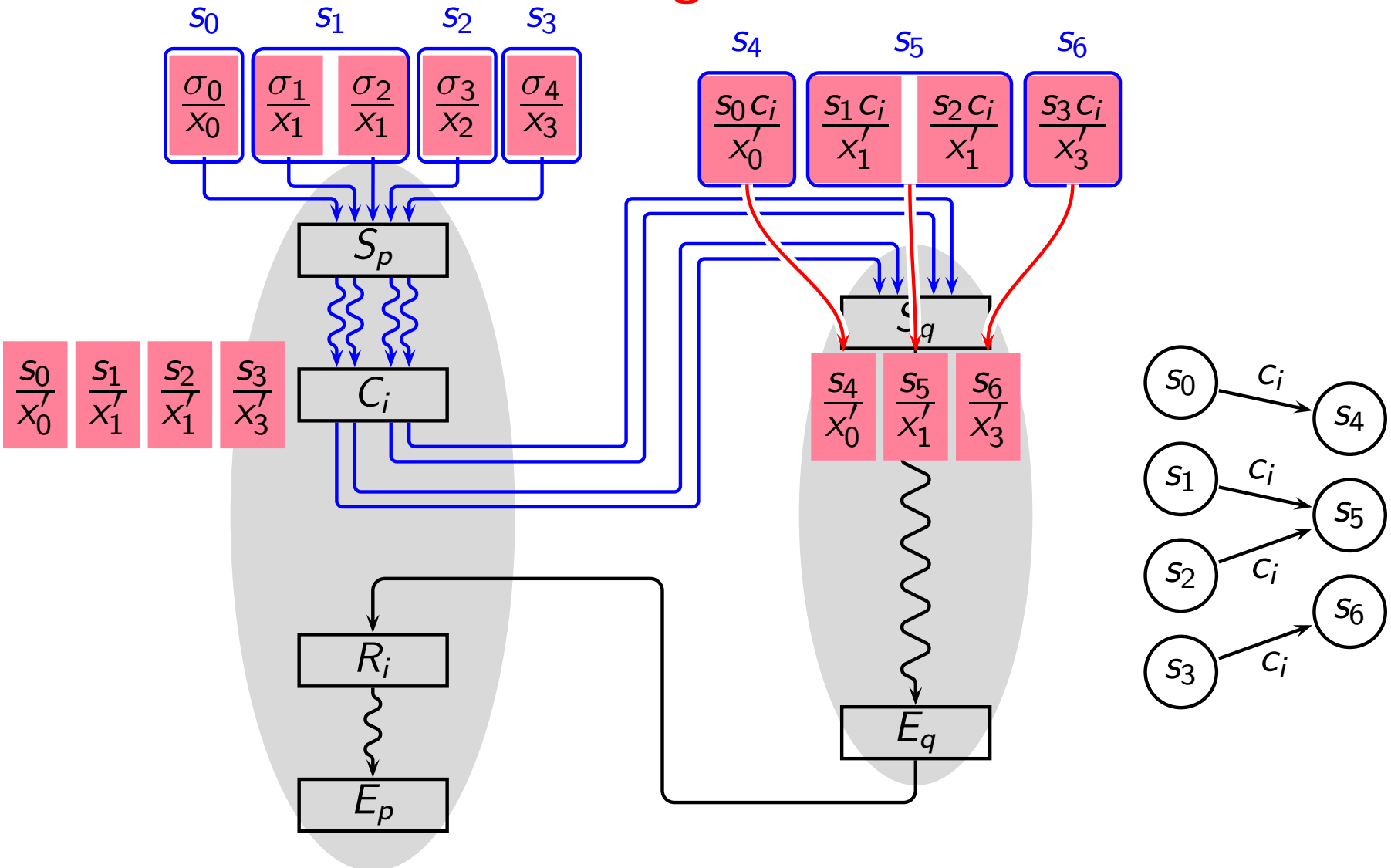
Understanding Value Contexts



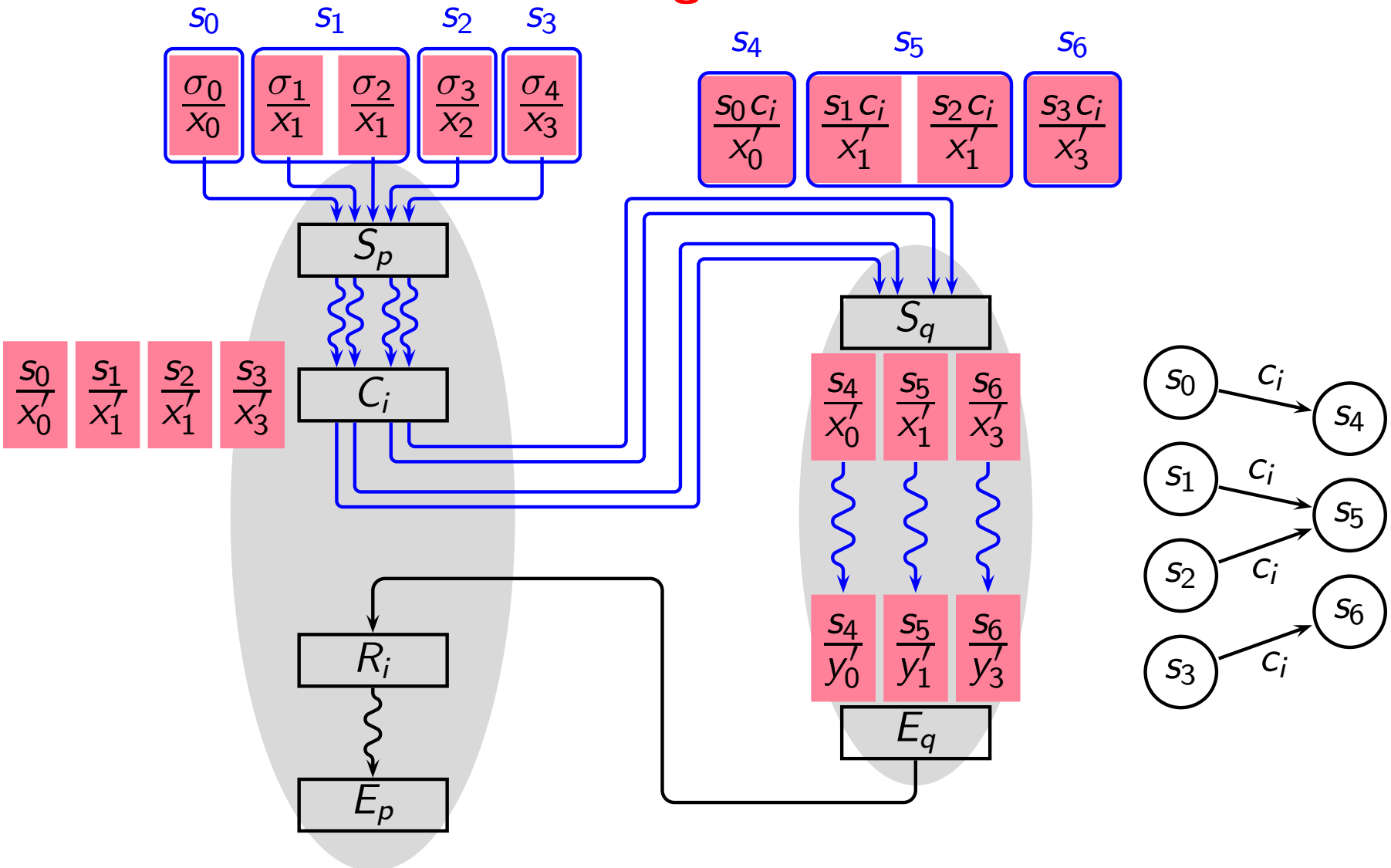
Understanding Value Contexts



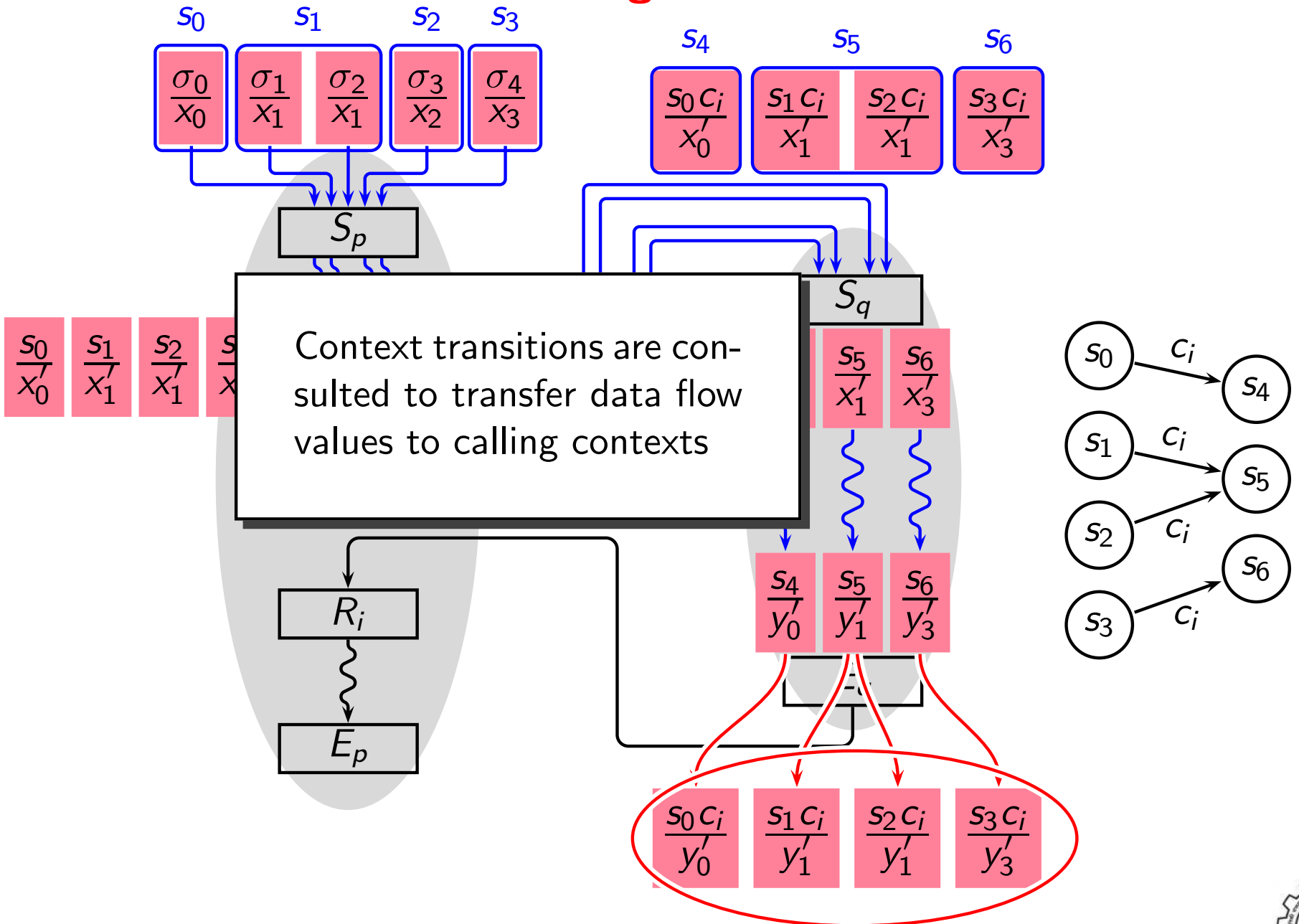
Understanding Value Contexts



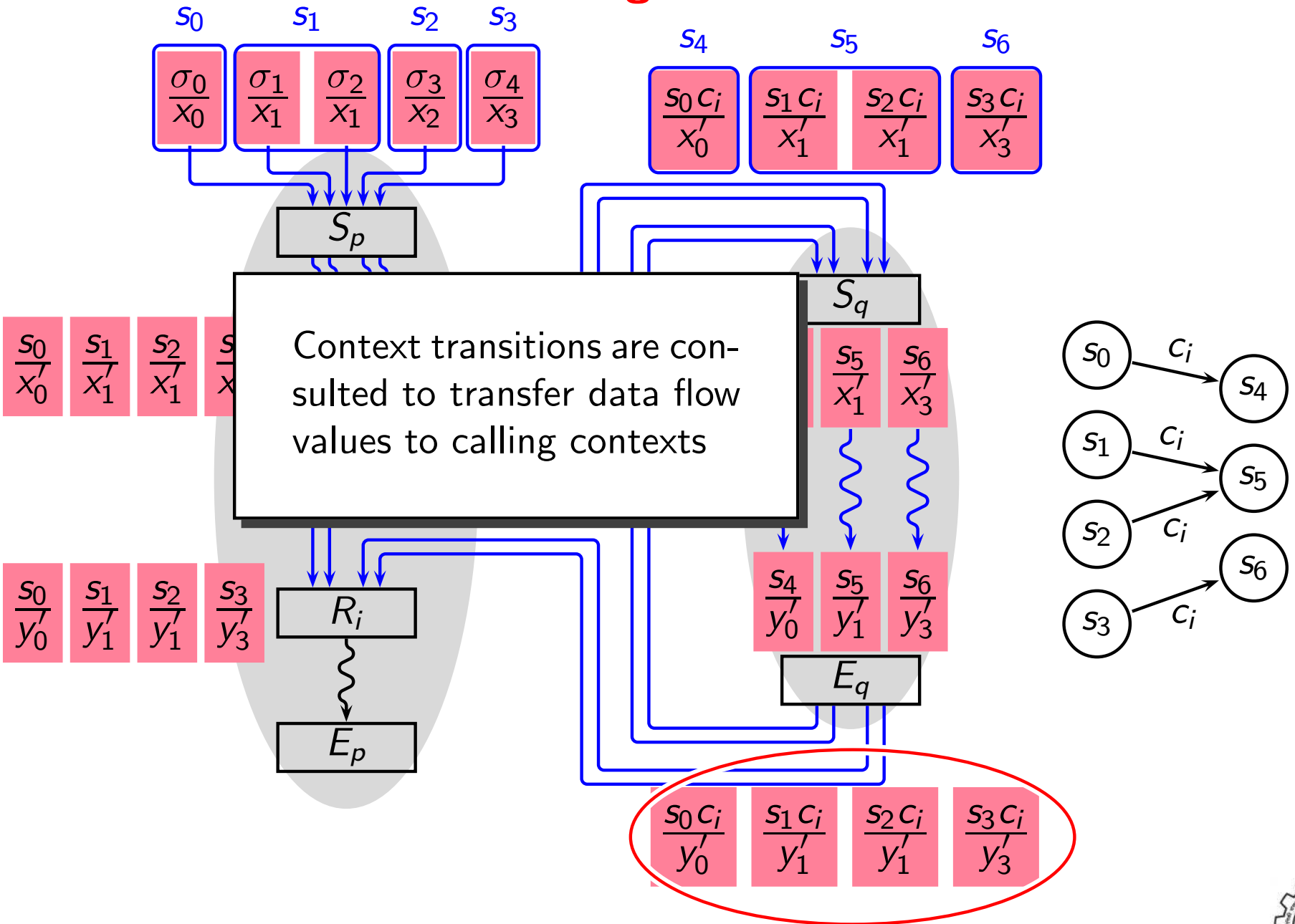
Understanding Value Contexts



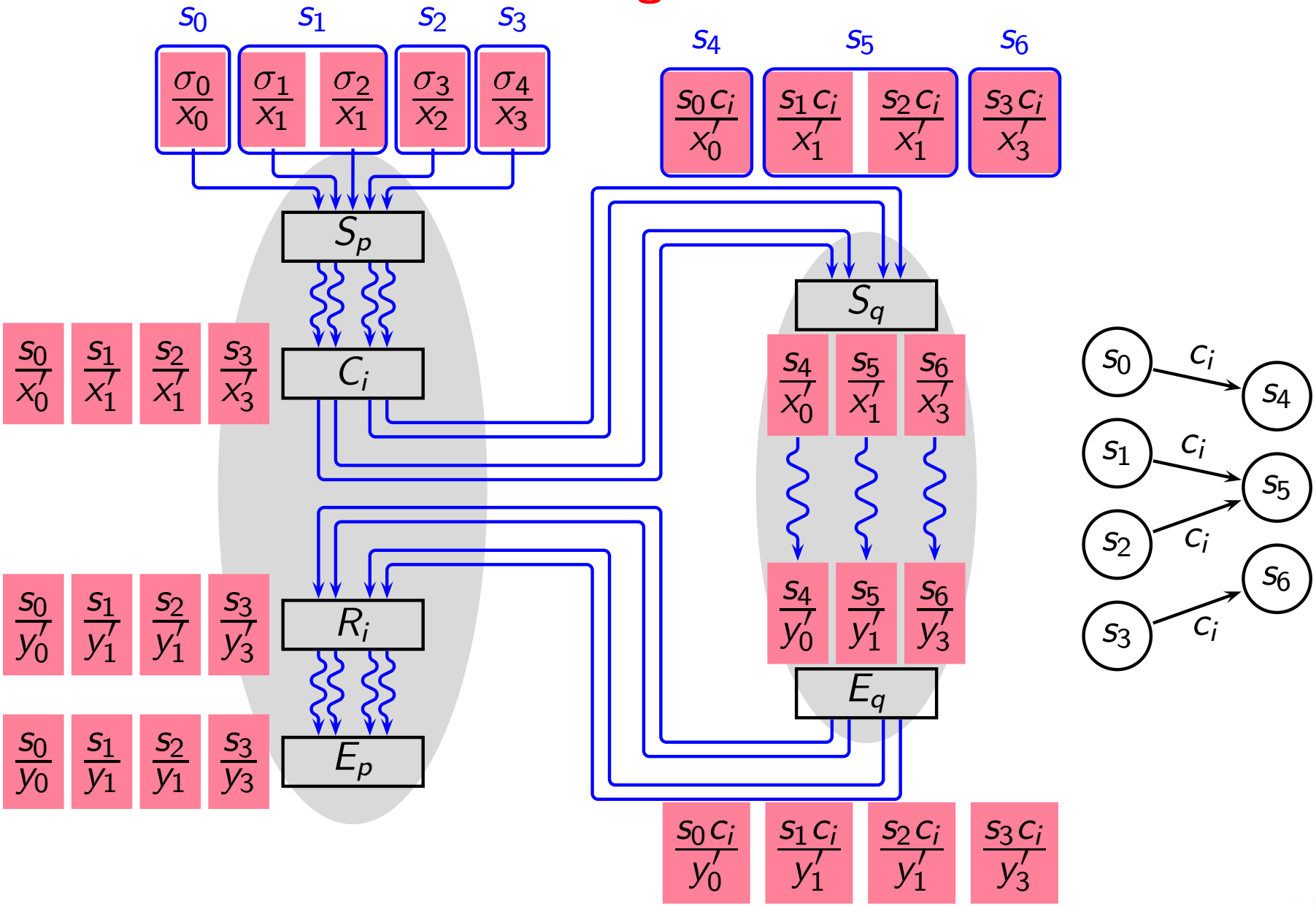
Understanding Value Contexts



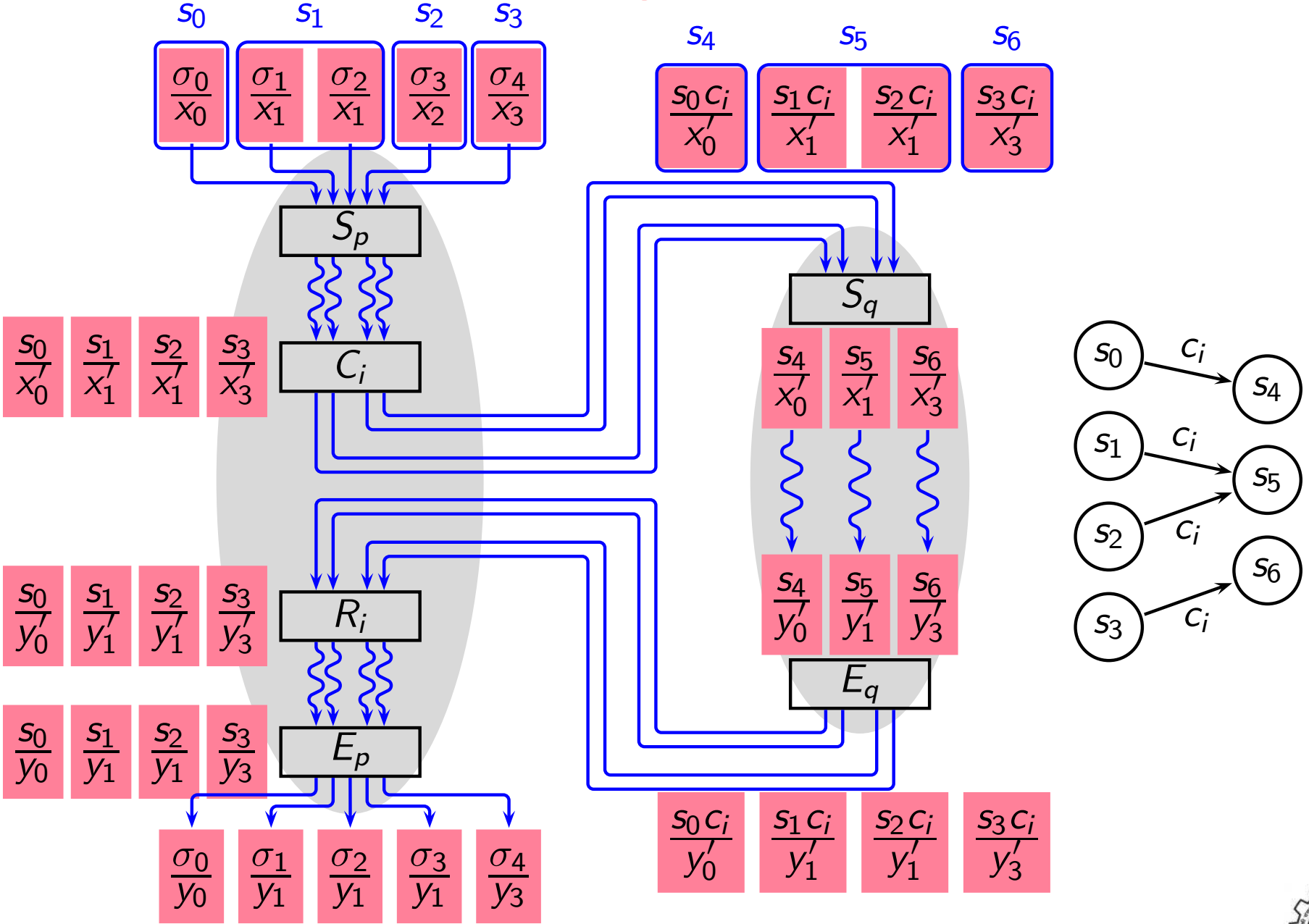
Understanding Value Contexts



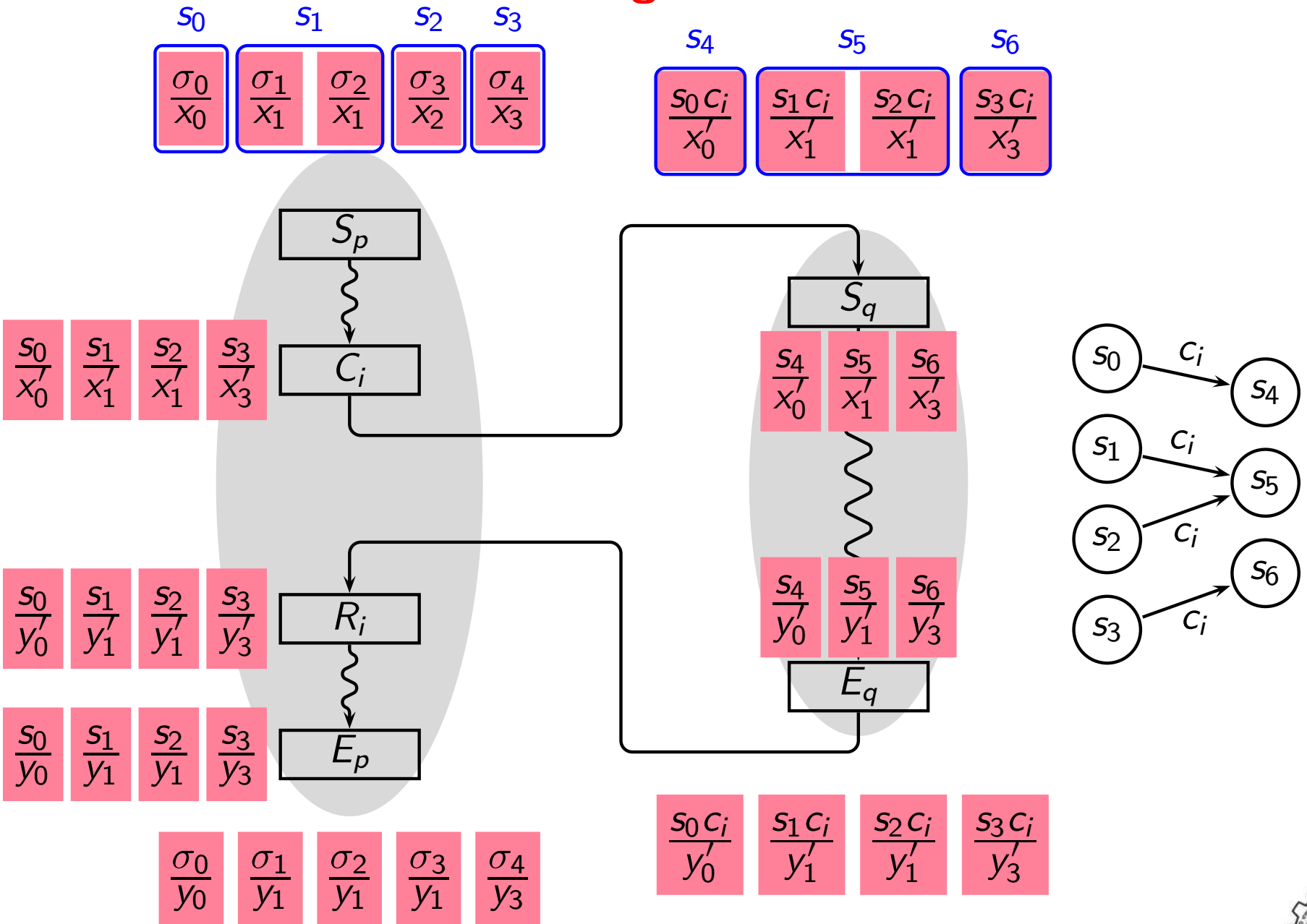
Understanding Value Contexts



Understanding Value Contexts



Understanding Value Contexts



Defining Value Contexts

- The set of value contexts is $VC = Procs \times L$

  A value context $X = \langle proc, entryValue \rangle \in VC$
  where $proc \in Procs$ and $entryValue \in L$

Defining Value Contexts

- The set of value contexts is $VC = Procs \times L$

  A value context $X = \langle proc, entryValue \rangle \in VC$
  where $proc \in Procs$ and $entryValue \in L$

- Supporting functions (CS is the set of call sites)
  - $exitValue : VC \mapsto L$
  - $transitions : (VC \times CS) \mapsto VC$

Defining Value Contexts

- The set of value contexts is $VC = Procs \times L$

  A value context $X = \langle proc, entryValue \rangle \in VC$
  where $proc \in Procs$ and $entryValue \in L$

- Supporting functions (CS is the set of call sites)
  - $exitValue : VC \mapsto L$ eg. $exitValue(X) = v$
  - $transitions : (VC \times CS) \mapsto VC$ eg. $X \xrightarrow{c_i} Y$

Interprocedural Data Flow Analysis Using Value Contexts

- The method works with a collection of control flow graphs

  No need of supergraph

  - No need to distinguish between C_i and R_i
  - No need of call $(C_i \rightarrow S_p)$ and return $(E_p \rightarrow E_i)$ edges

- Maintain a work list WL of entries $\langle context, node \rangle$

  (in reverse post order of nodes within a procedure for forward flows)

- Notation:

$\langle p,v \rangle$	Context for procedure p with data flow value v
$X \vert m$	Work list entry for context X for node m
$X.v$	Data flow value in context X is v
$Out_m[X]$	Data flow value of context X in Out_m
$X \xrightarrow{C_i} Y$	Transition from context X to context Y at call site C_i

Interprocedural Data Flow Analysis Using Value Contexts: An Overview

- Select $X|n$ from WL. Compute In_n.

Interprocedural Data Flow Analysis Using Value Contexts: An Overview

- Select $X|n$ from WL. Compute In_n.
  - If $n = C_i$ calling procedure p
  - If $n = E_p$
  - If n is some other node

Interprocedural Data Flow Analysis Using Value Contexts: An Overview

- Select $X|n$ from WL. Compute In_n.
  - If $n = C_i$ calling procedure p
    Propagate In_n to appropriate value context of the callee procedure p
  - If $n = E_p$
  - If n is some other node

Interprocedural Data Flow Analysis Using Value Contexts: An Overview

- Select $X|n$ from WL. Compute In_n.
  - If $n = C_i$ calling procedure p
  - If $n = E_p$
    Propagate In_n to appropriate value contexts of the callers of p
  - If n is some other node

Interprocedural Data Flow Analysis Using Value Contexts: An Overview

- Select $X|n$ from WL. Compute In_n.
  - If $n = C_i$ calling procedure p
  - If $n = E_p$
  - If n is some other node
    Compute Out_n

Interprocedural Data Flow Analysis Using Value Contexts: An Overview

- Select $X|n$ from WL. Compute In_n.
  - If $n = C_i$ calling procedure p
  - If $n = E_p$
  - If n is some other node

  Update WL

Interprocedural Data Flow Analysis Using Value Contexts: An Overview

- Select $X|n$ from WL. Compute In_n.
  - If $n = C_i$ calling procedure p
    Propagate In_n to appropriate value context of the callee procedure p
  - If $n = E_p$
    Propagate In_n to appropriate value contexts of the callers of p
  - If n is some other node
    Compute Out_n

  Update WL

Interprocedural Data Flow Analysis Using Value Contexts: An Overview

- Select $X|n$ from WL. Compute In_n.
  - If $n = C_i$ calling procedure p
    Propagate In_n to appropriate value context of the callee procedure p
  - If $n = E_p$
    Propagate In_n to appropriate value contexts of the callers of p
  - If n is some other node
    Compute Out_n

  Update WL
- Repeat until WL is empty

Interprocedural Data Flow Analysis Using Value Contexts (2)

Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

Interprocedural Data Flow Analysis Using Value Contexts (2)

Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

- If $n = C_i$ calling procedure p

Interprocedural Data Flow Analysis Using Value Contexts (2)

Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

- If $n = C_i$ calling procedure p
  - If some context $\langle p,v \rangle$ exists (say Y) /* p is the callee */
  - If it does not exist

Interprocedural Data Flow Analysis Using Value Contexts (2)

Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

- If $n = C_i$ calling procedure p
  - If some context $\langle p,v \rangle$ exists (say Y) /* p is the callee */
    - record the transition $X \xrightarrow{C_i} Y$
    - $Out_{C_i}[X] = Out_{C_i}[X] \sqcap exitValue(Y)$
    - if there is a change, add $X|m$, $\forall m \in succ(C_i)$ to WL
  - If it does not exist

Interprocedural Data Flow Analysis Using Value Contexts (2)

Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

- If $n = C_i$ calling procedure p
  - If some context $\langle p,v \rangle$ exists (say Y) /* p is the callee */
  - If it does not exist
    - create a new context $Y = \langle p,v \rangle$ /* p is the callee */
    - initialize $exitValue(Y) = \top$
    - record the transition $X \xrightarrow{C_i} Y$
    - initialize $Out_m[Y] = \top$ for all nodes m of procedure p
    - add entries $Y|m$ for all nodes m of procedure p to WL

Interprocedural Data Flow Analysis Using Value Contexts (2)

Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

- If $n = C_i$ calling procedure p
  - If some context $\langle p,v \rangle$ exists (say Y) /* p is the callee */
    - record the transition $X \xrightarrow{C_i} Y$
    - $Out_{C_i}[X] = Out_{C_i}[X] \sqcap exitValue(Y)$
    - if there is a change, add $X|m$, $\forall m \in succ(C_i)$ to WL
  - If it does not exist
    - create a new context $Y = \langle p,v \rangle$ /* p is the callee */
    - initialize $exitValue(Y) = \top$
    - record the transition $X \xrightarrow{C_i} Y$
    - initialize $Out_m[Y] = \top$ for all nodes m of procedure p
    - add entries $Y|m$ for all nodes m of procedure p to WL

Interprocedural Data Flow Analysis Using Value Contexts (3)

Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

Interprocedural Data Flow Analysis Using Value Contexts (3)

Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

- If $n = E_p$

- For all other nodes

Interprocedural Data Flow Analysis Using Value Contexts (3)

Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

- If $n = E_p$
  - Set $exitValue(X) = v$ /* E_p is an empty block */
- For all other nodes

Interprocedural Data Flow Analysis Using Value Contexts (3)

Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

- If $n = E_p$
  - Set $exitValue(X) = v$ /* E_p is an empty block */
  - Find out all transitions $Z \xrightarrow{C_j} X$
    - Set $Out_{C_j}[Z] = Out_{C_j}[Z] \sqcap v$
    - If there is a change, add $Z|m$, $\forall m \in succ(C_j)$ to WL
- For all other nodes

Interprocedural Data Flow Analysis Using Value Contexts (3)

Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

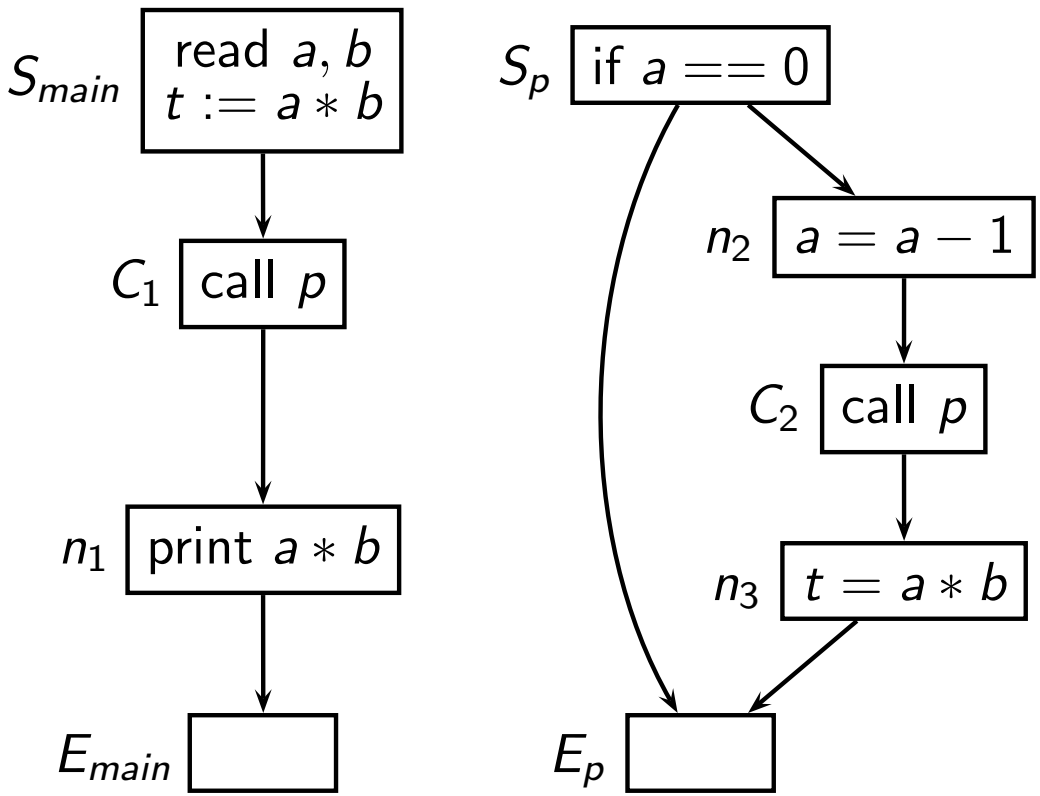
- If $n = E_p$
- For all other nodes
  - Set $Out_n[X] = f_n(v)$

Interprocedural Data Flow Analysis Using Value Contexts (3)

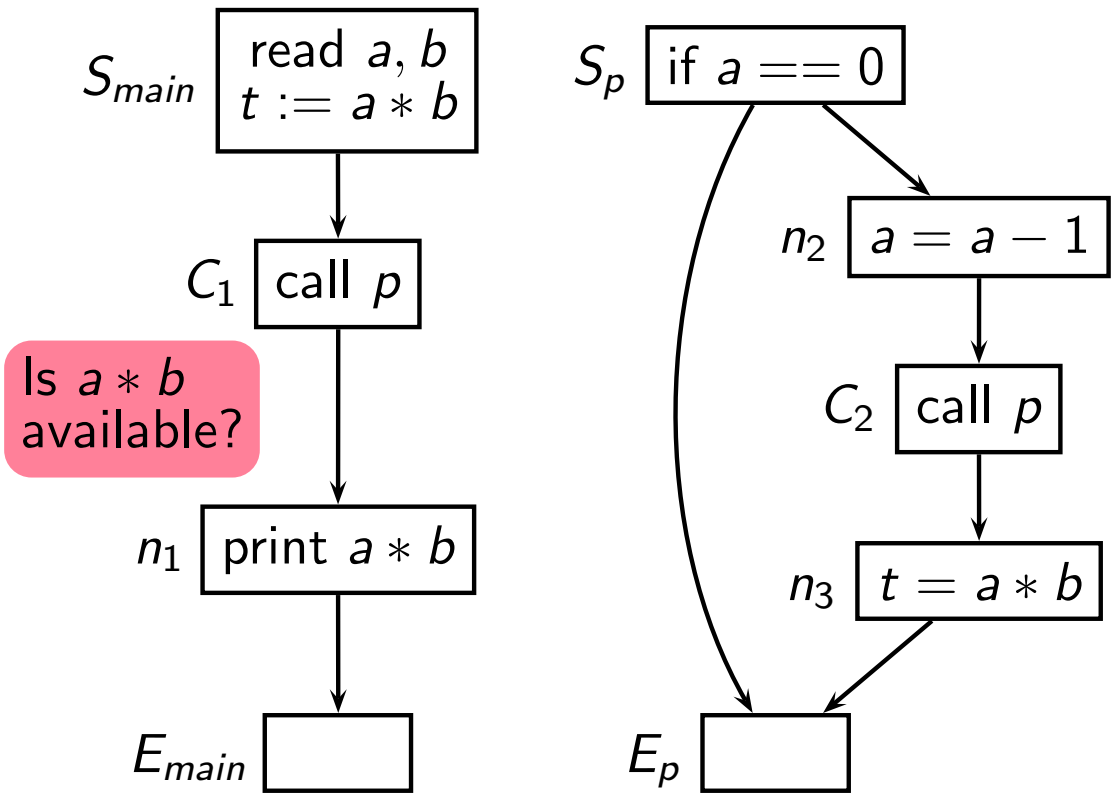
Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

- If $n = E_p$
  - Set $exitValue(X) = v$ /* E_p is an empty block */
  - Find out all transitions $Z \xrightarrow{C_j} X$
    - Set $Out_{C_j}[Z] = Out_{C_j}[Z] \sqcap v$
    - If there is a change, add $Z|m$, $\forall m \in succ(C_j)$ to WL
- For all other nodes
  - Set $Out_n[X] = f_n(v)$
  - If there is a change, add $X|m$, $\forall m \in succ(n)$ to WL

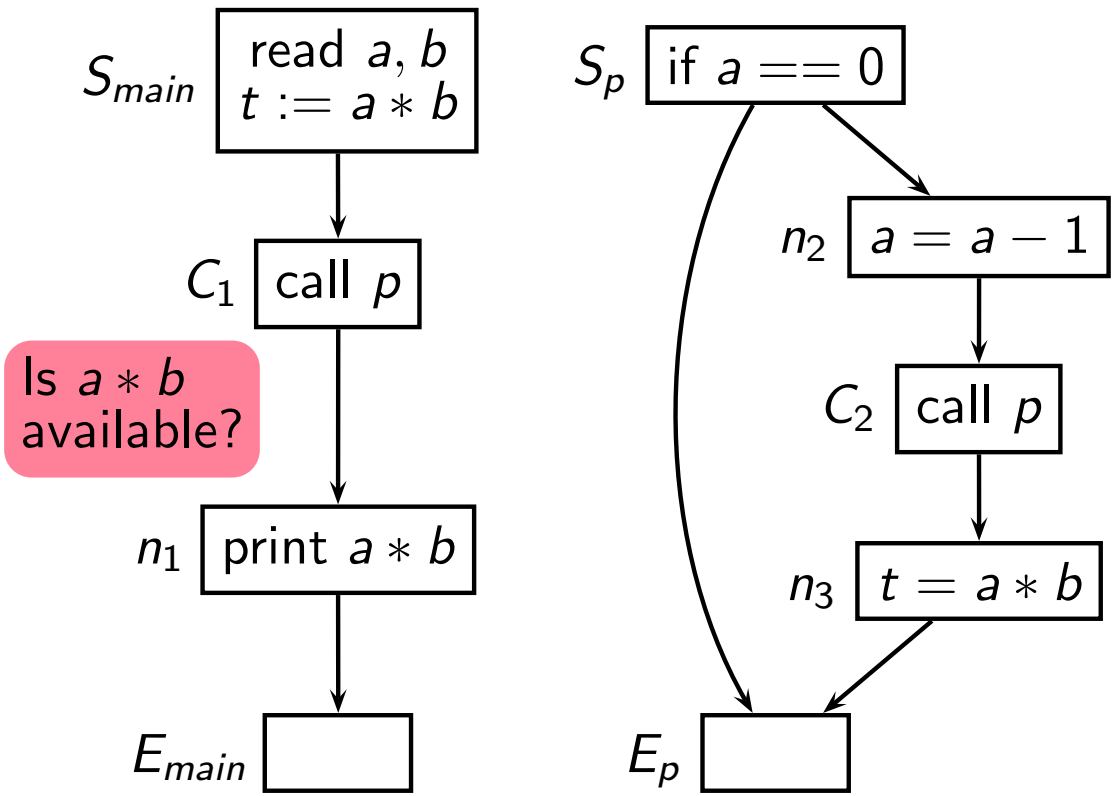
Available Expressions Analysis Using Value Contexts



Available Expressions Analysis Using Value Contexts

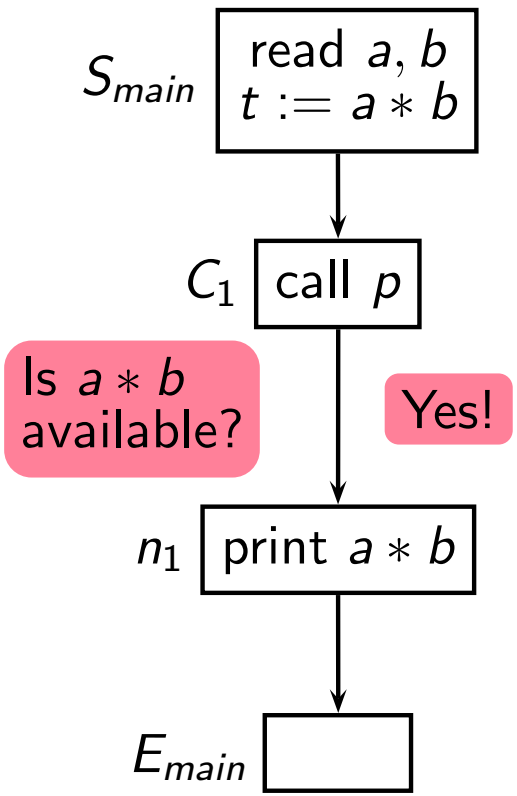


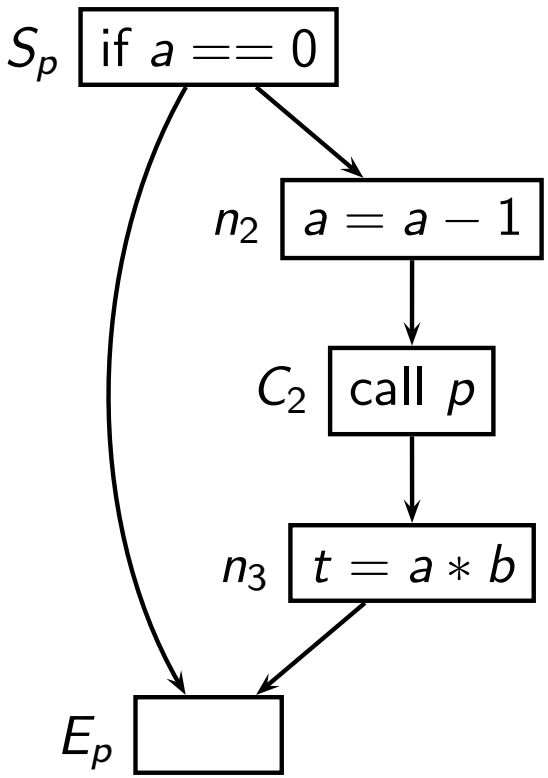
Available Expressions Analysis Using Value Contexts




```
int a, b, t;
void p()
{  if (a == 0)
   {  a = a-1;
      p();
      t = a*b;
   }
}
```

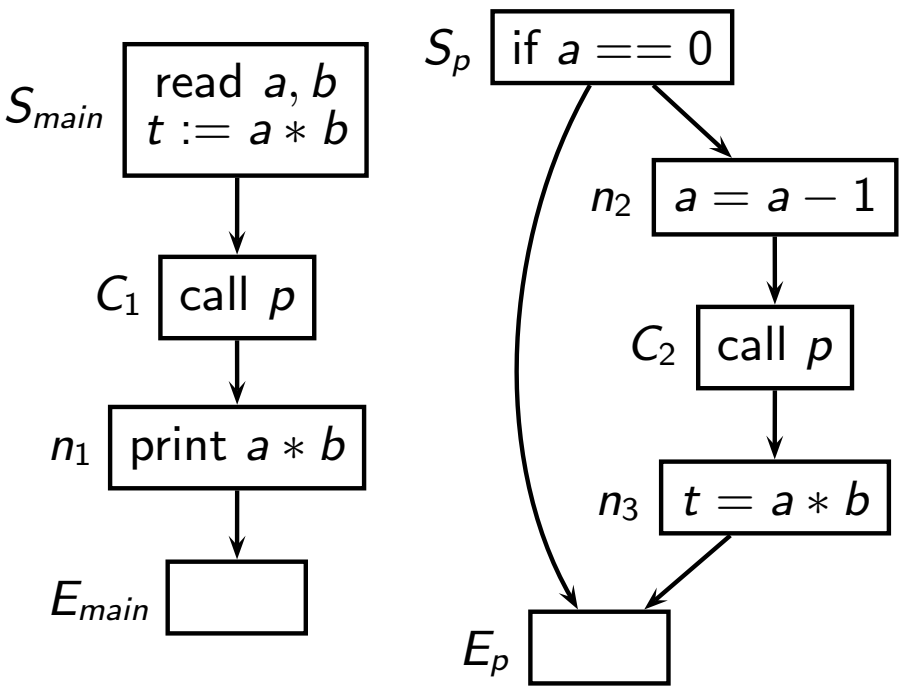

Available Expressions Analysis Using Value Contexts



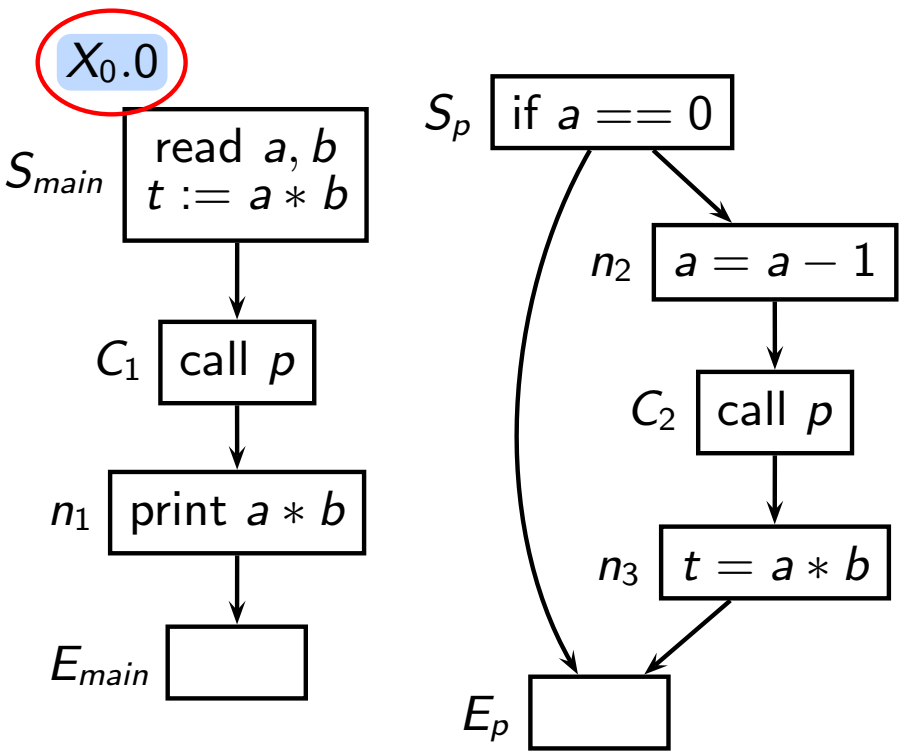



```
int a, b, t;
void p()
{  if (a == 0)
   {  a = a-1;
      p();
      t = a*b;
   }
}
```

Available Expressions Analysis Using Value Contexts



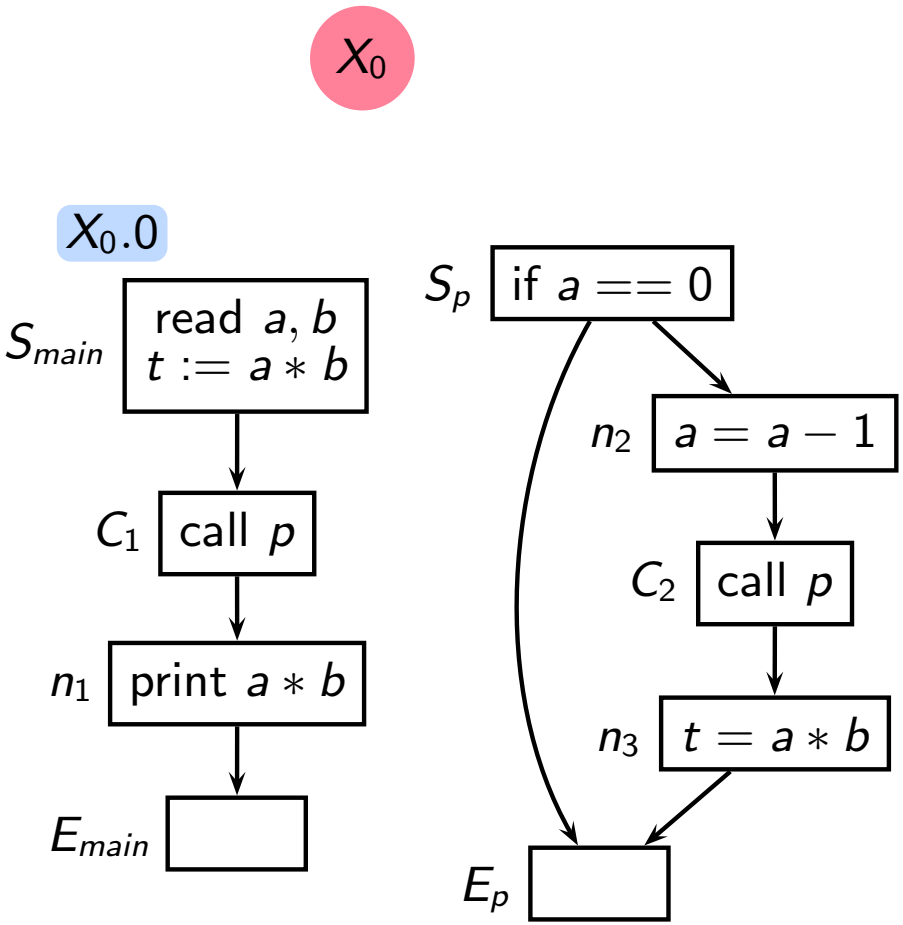
Available Expressions Analysis Using Value Contexts



Create a new context X_0 with BI which is 0 for available expressions analysis

Available Expressions Analysis Using Value Contexts

$WL = [X_0|S_m, X_0|C_1, X_0|n_1, X_0|E_m]$



Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1

Create a new context X_0 with *BI* which is 0 for available expressions analysis

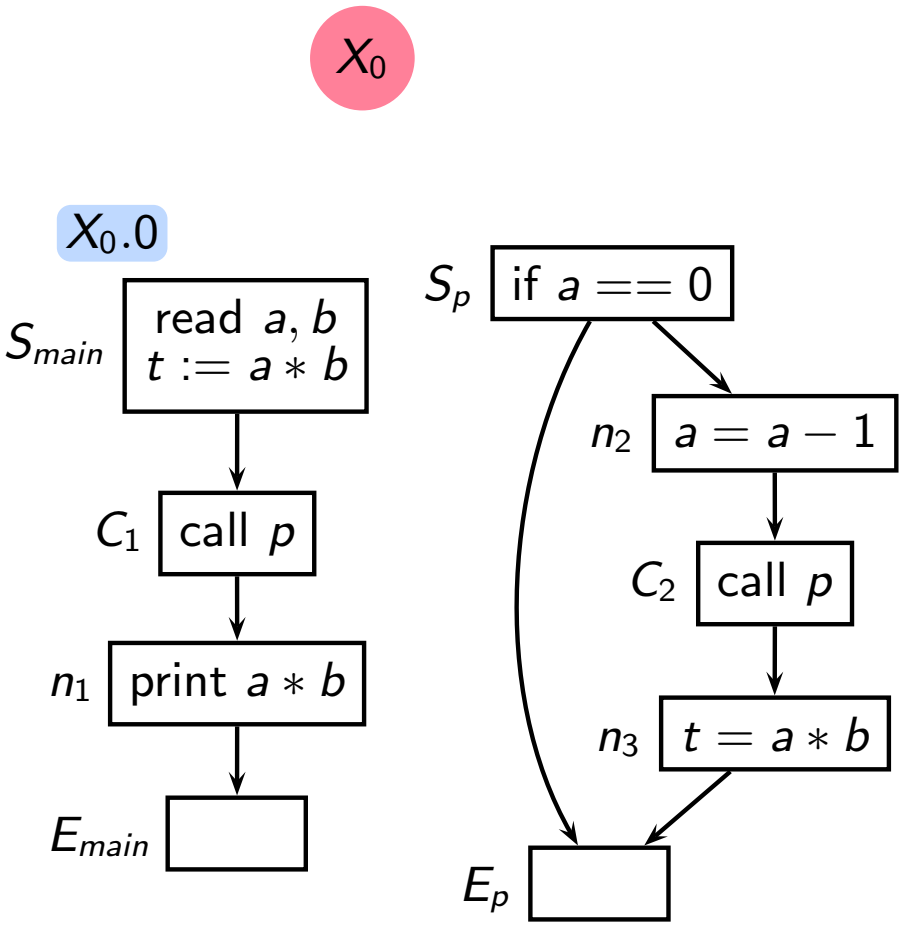
Initialize $exitValue(X_0)$ to $\top = 1$

Initialize the work list with all nodes in procedure main for X_0

Initialize $Out_n[X_0]$ for all n in main to $\top$

Available Expressions Analysis Using Value Contexts

$WL = [X_0|S_m, X_0|C_1, X_0|n_1, X_0|E_m]$

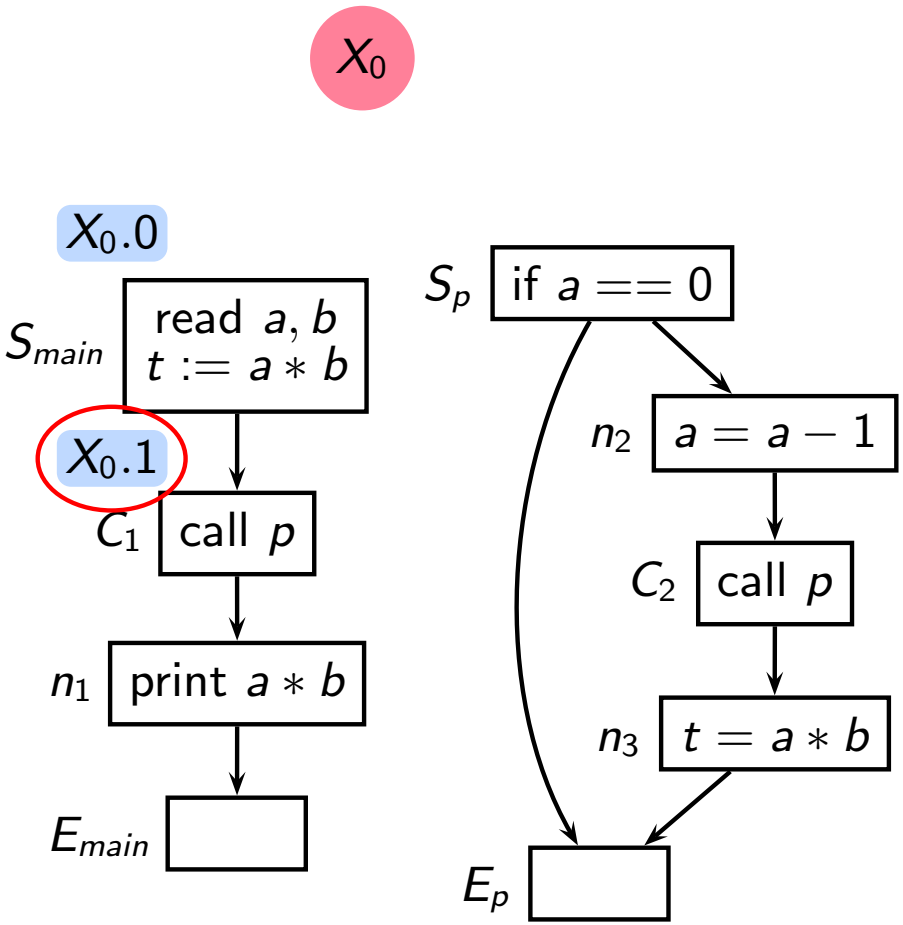


Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1

Compute the data flow values for S_m for context X_0

Available Expressions Analysis Using Value Contexts

$WL = [X_0|C_1, X_0|n_1, X_0|E_m]$



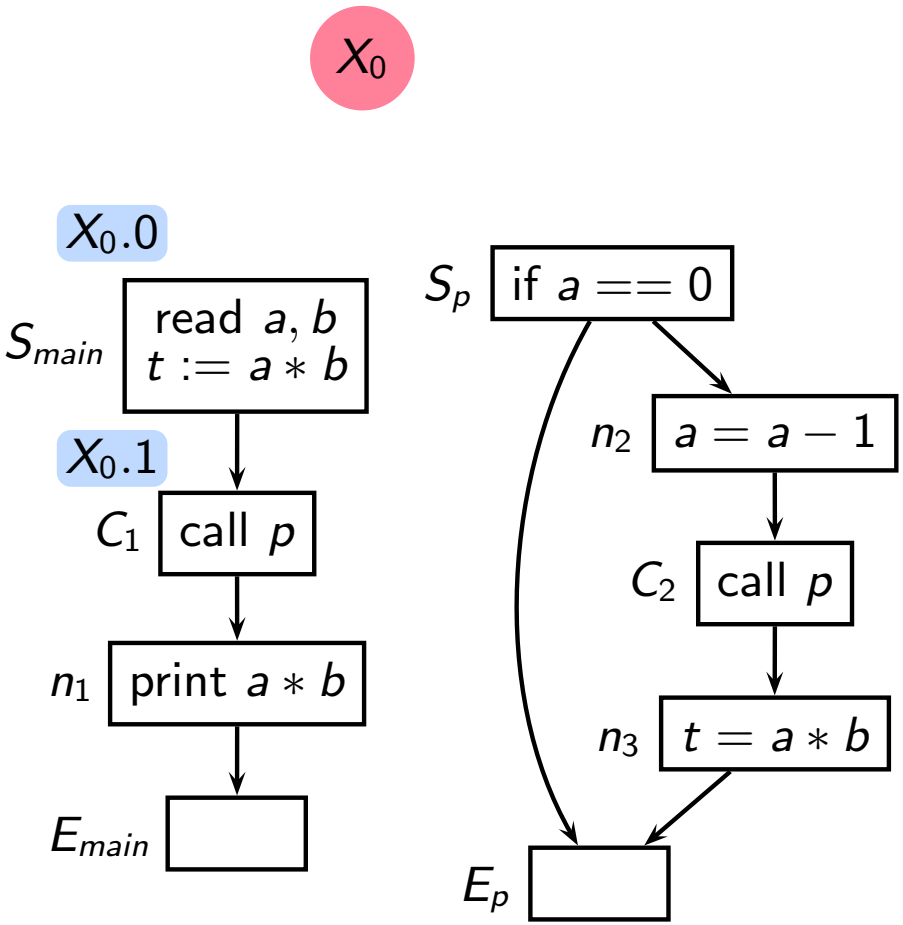
Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1

Compute the data flow values for S_m for context X_0

It does not change

Available Expressions Analysis Using Value Contexts

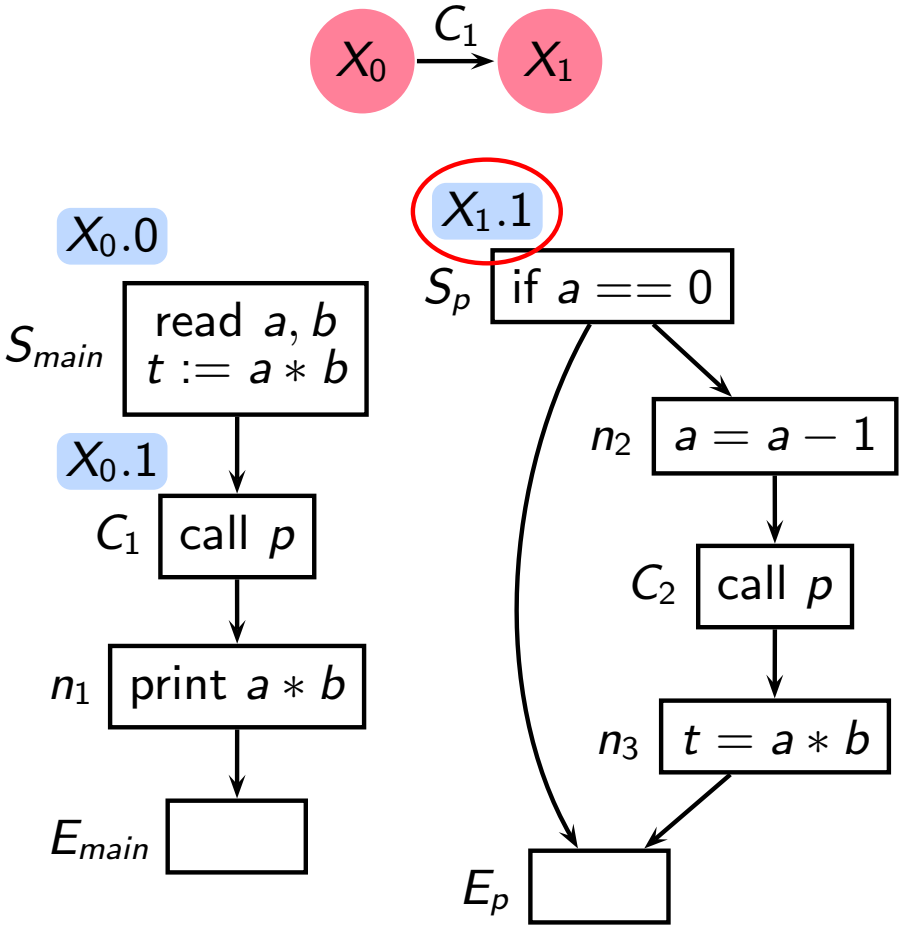
$WL = [X_0|C_1, X_0|n_1, X_0|E_m]$



Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1

Available Expressions Analysis Using Value Contexts

$WL = [X_1|S_p, X_1|n_2, X_1|C_2, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$



Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1

Create a new context X_1 with entry value 1

Record the transition to X_1

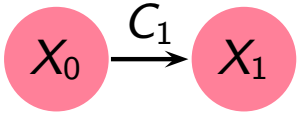
Initialize $exitValue(X_1)$ to $\top = 1$

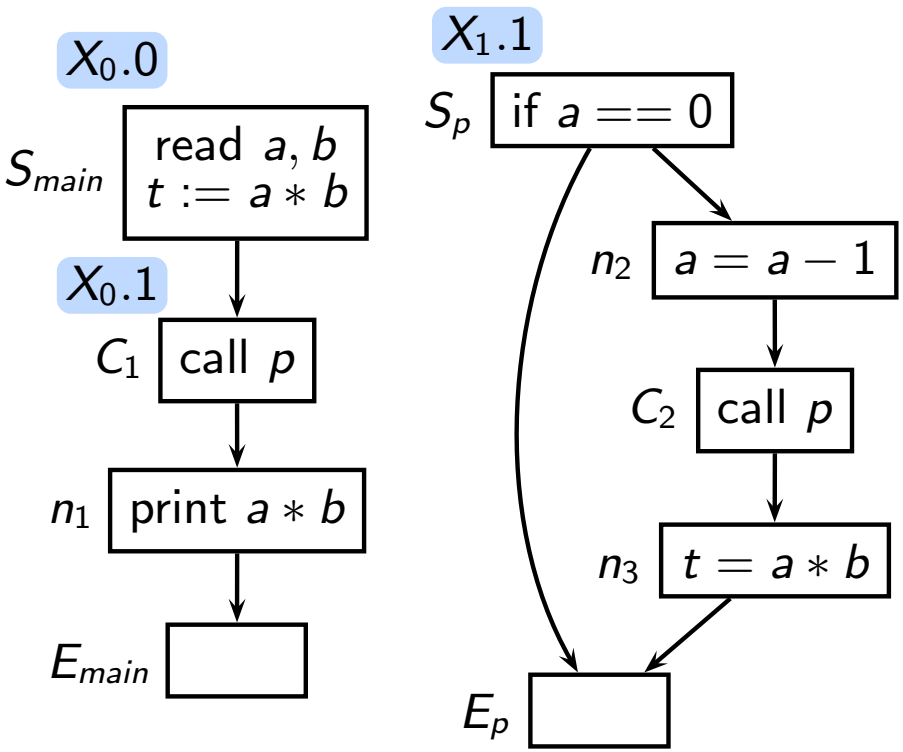
Add all nodes of procedure p to the work list for X_1

Initialize $Out_n[X_1]$ for all n in p to $\top$

Available Expressions Analysis Using Value Contexts

$WL = [X_1|S_p, X_1|n_2, X_1|C_2, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$

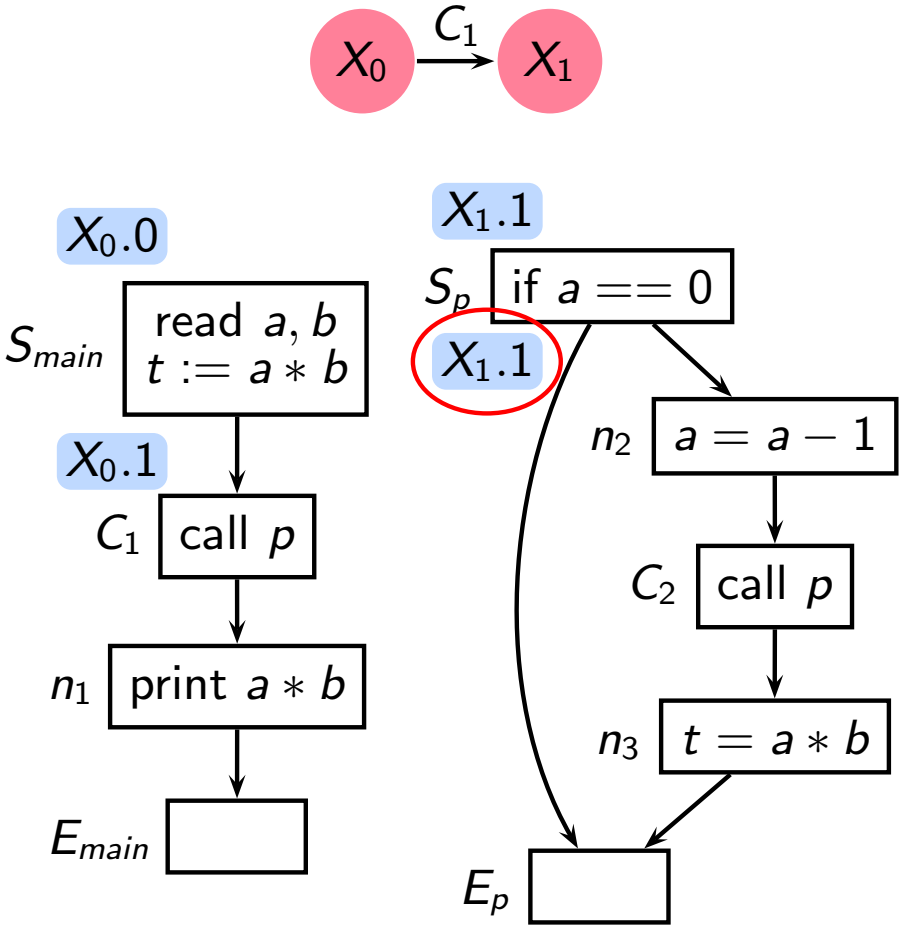




Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1

Available Expressions Analysis Using Value Contexts

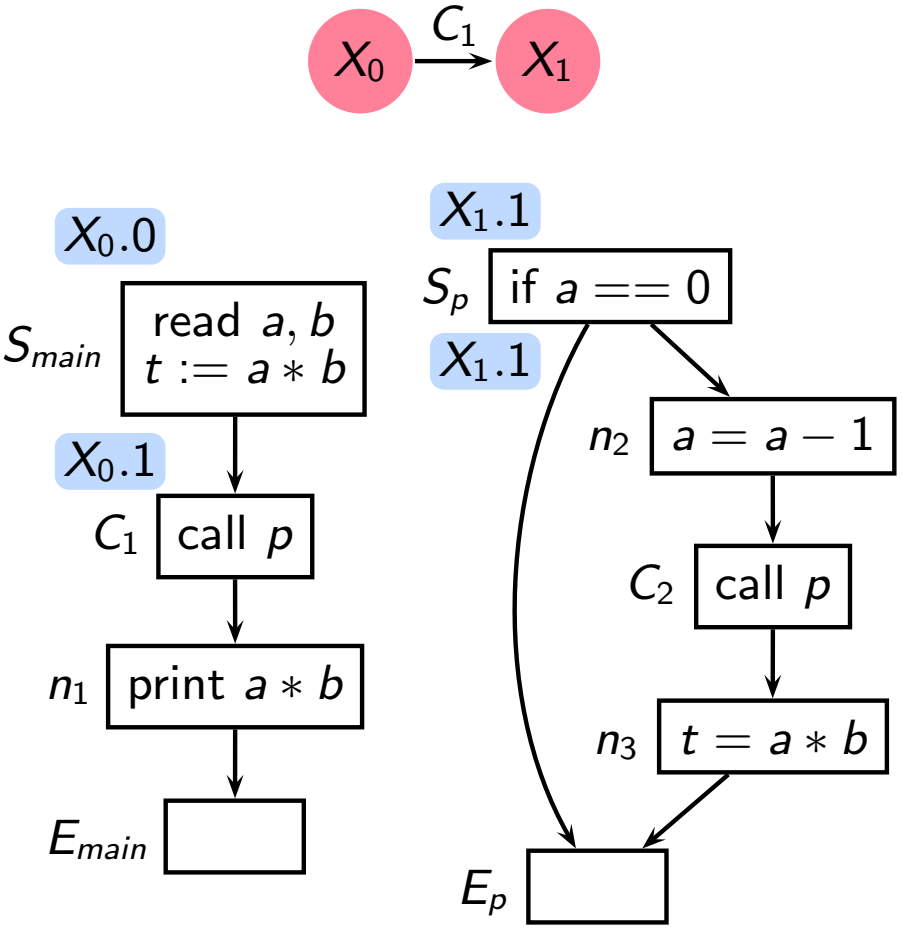
$$WL = [X_1|n_2, X_1|C_2, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$$



Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1

Available Expressions Analysis Using Value Contexts

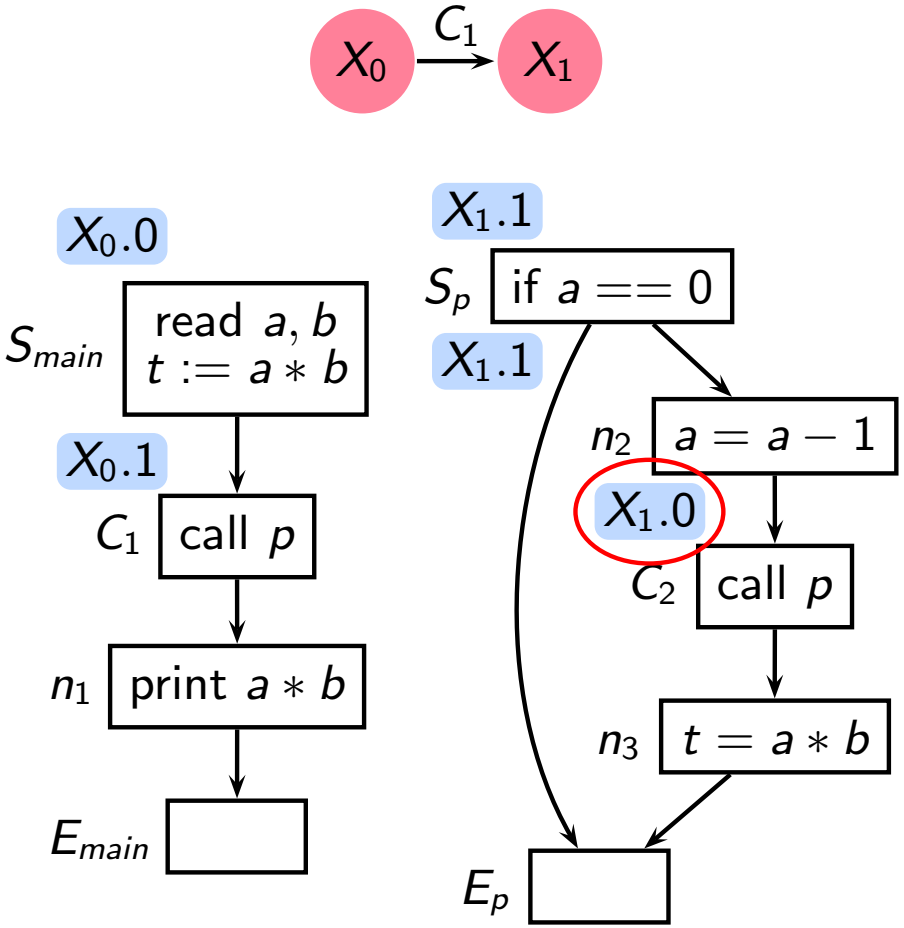
$WL = [X_1|n_2, X_1|C_2, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$



Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1

Available Expressions Analysis Using Value Contexts

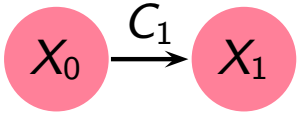
$$WL = [X_1|n_2, X_1|C_2, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$$

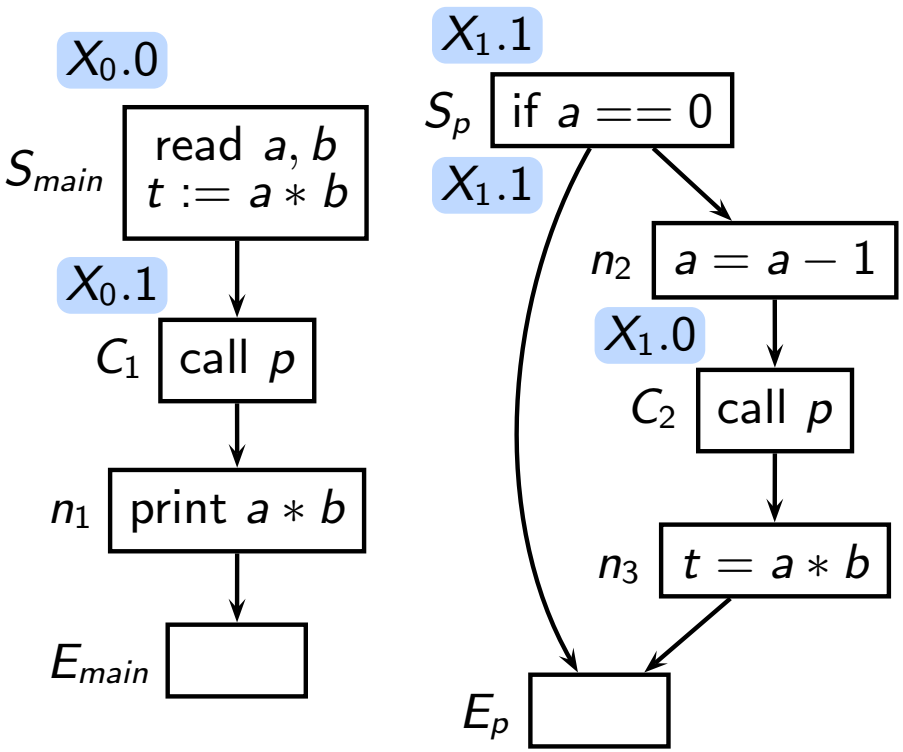


Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1

Available Expressions Analysis Using Value Contexts

$WL = [X_1|C_2, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$

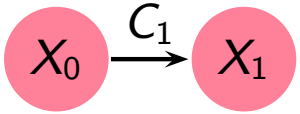


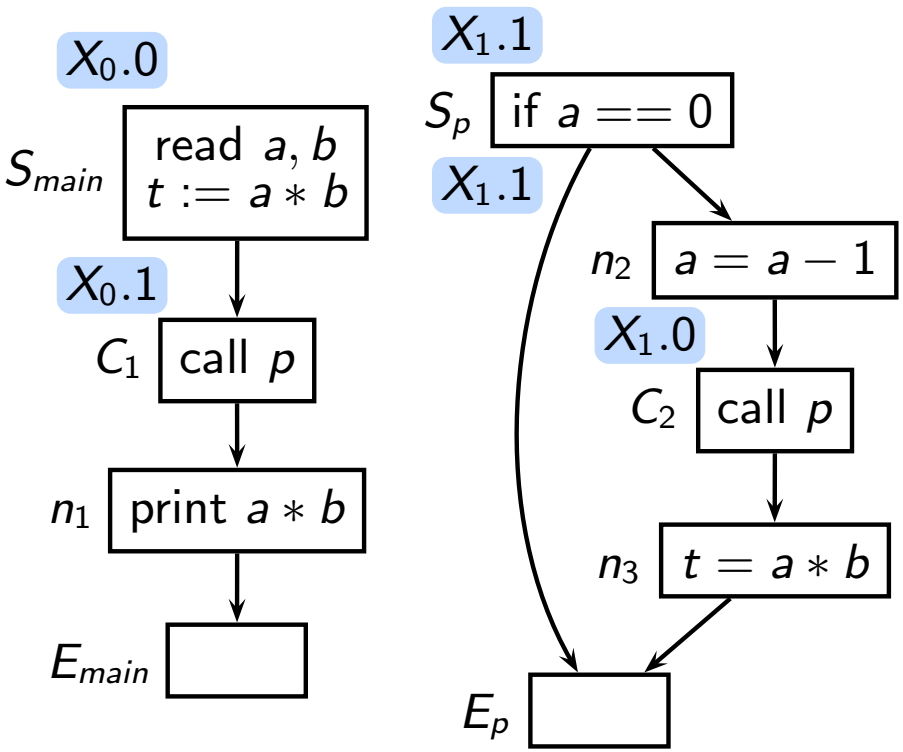


Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1

Available Expressions Analysis Using Value Contexts

$WL = [X_1|C_2, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$

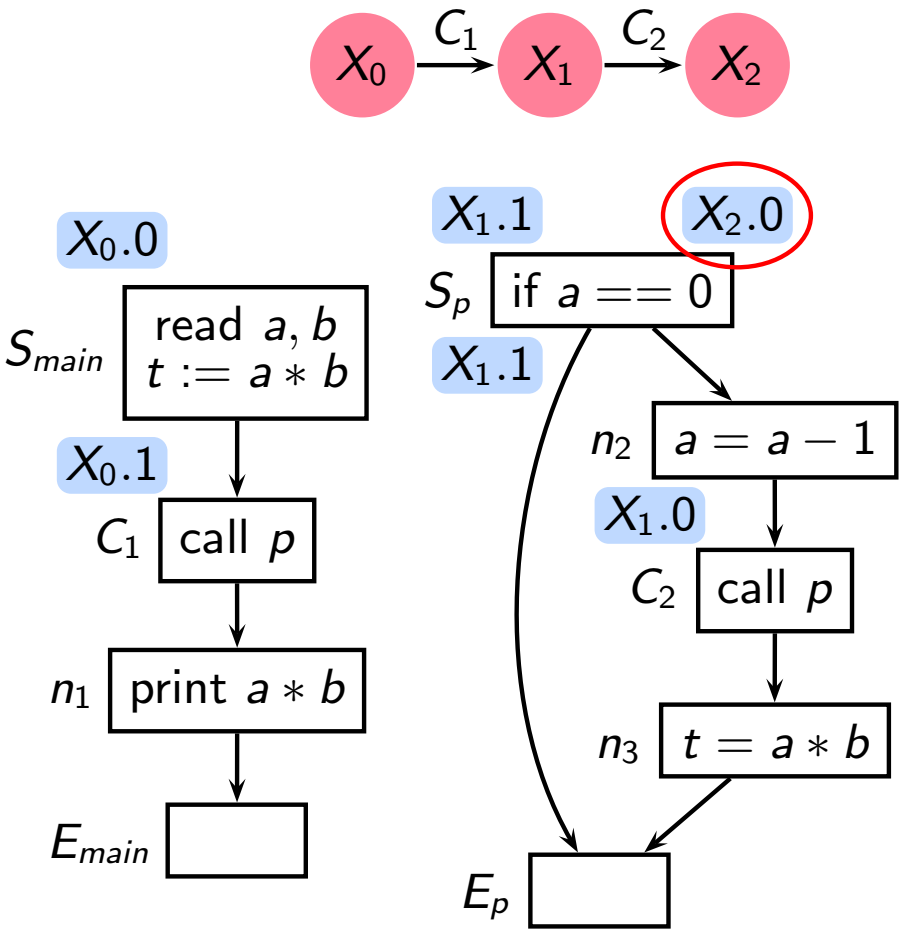




Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1

Available Expressions Analysis Using Value Contexts

$WL = [X_1|C_2, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$



Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1
$X_2 = \langle$p,0$\rangle$	1

Since there is no context for p with value 0, create context X_2

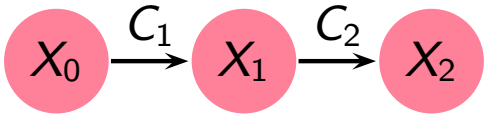
Record the transition to X_2

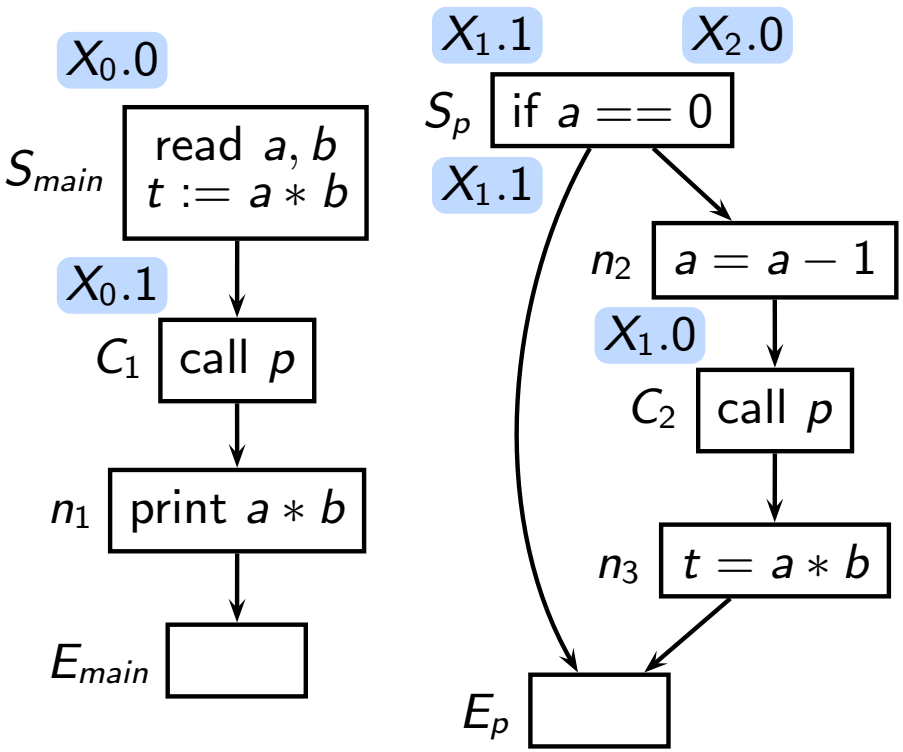
Initialize $exitValue(X_2)$ to $\top = 1$

Add all nodes of procedure p to the work list for X_2 Initialize $Out_n[X_2]$ for all n in p to $\top$

Available Expressions Analysis Using Value Contexts

$$WL = [X_2|S_p, X_2|n_2, X_2|C_2, X_2|n_3, X_2|E_p, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$$

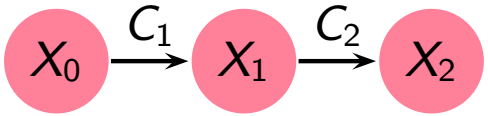


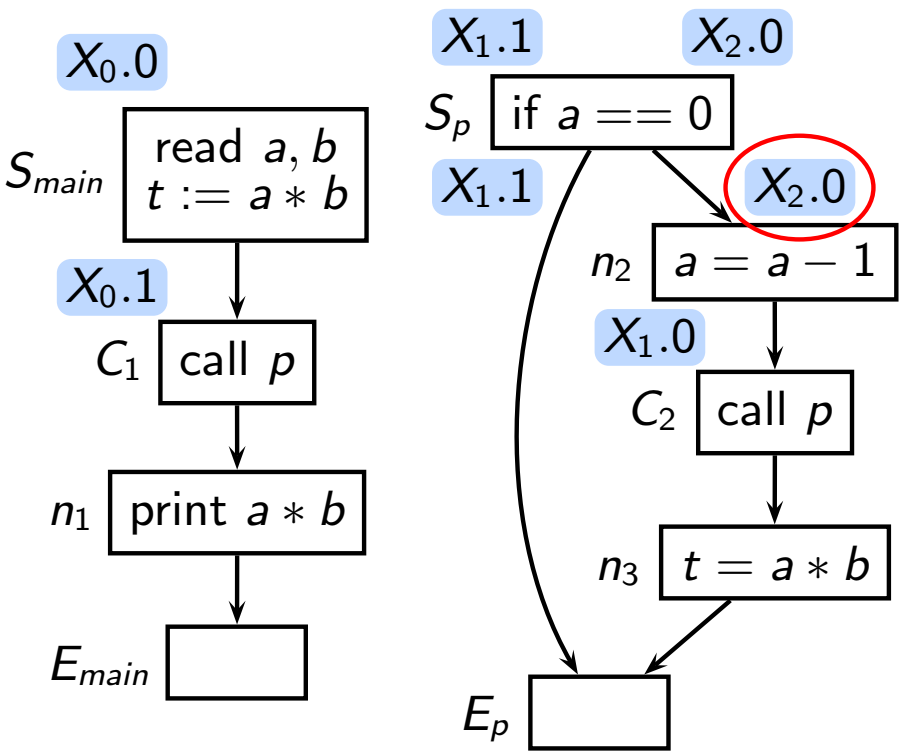


Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1
$X_2 = \langle$p,0$\rangle$	1

Available Expressions Analysis Using Value Contexts

$$WL = [X_2|S_p, X_2|n_2, X_2|C_2, X_2|n_3, X_2|E_p, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$$

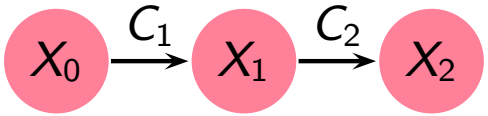


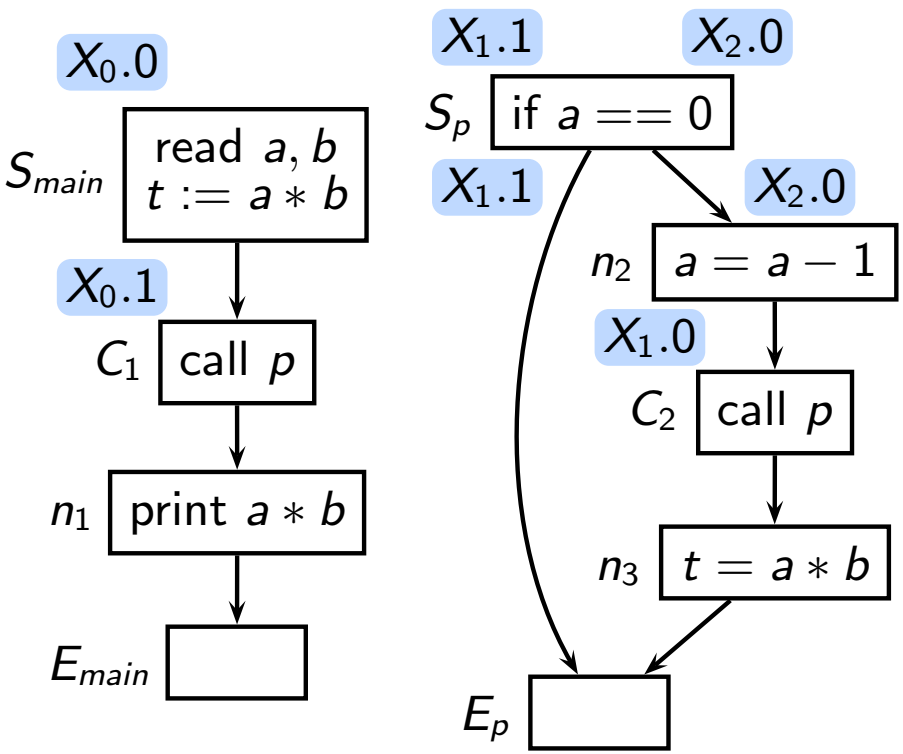


Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1
$X_2 = \langle$p,0$\rangle$	1

Available Expressions Analysis Using Value Contexts

$$WL = [X_2|n_2, X_2|C_2, X_2|n_3, X_2|E_p, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$$

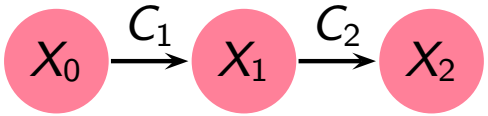


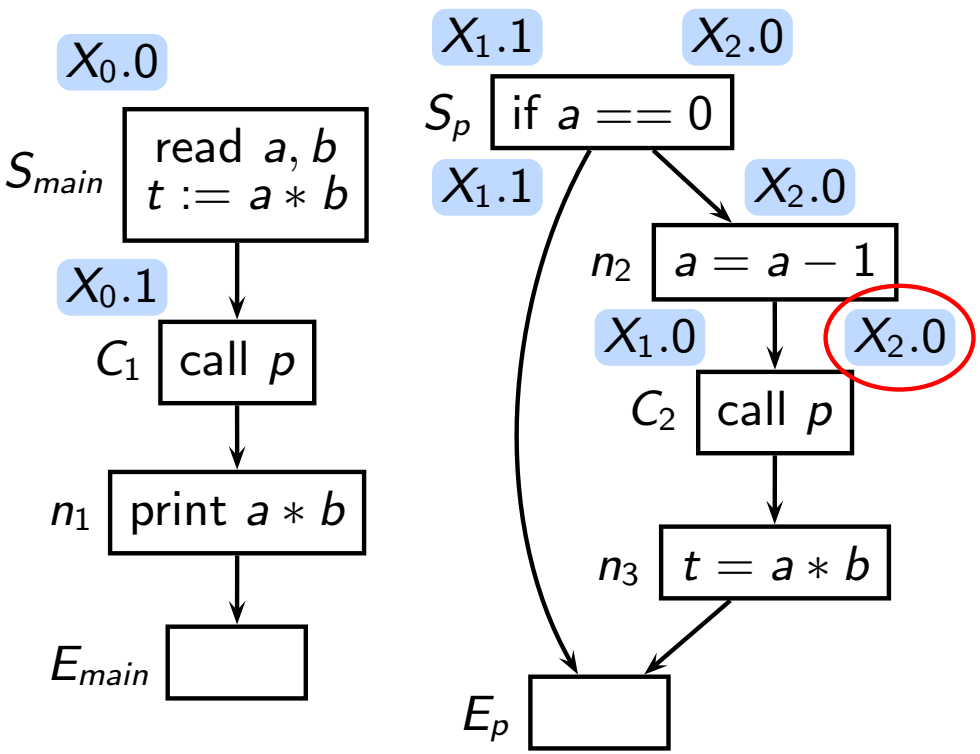


Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1
$X_2 = \langle$p,0$\rangle$	1

Available Expressions Analysis Using Value Contexts

$$WL = [X_2|n_2, X_2|C_2, X_2|n_3, X_2|E_p, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$$

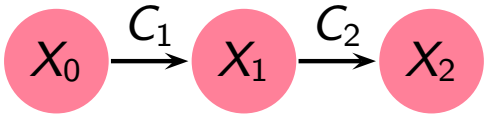


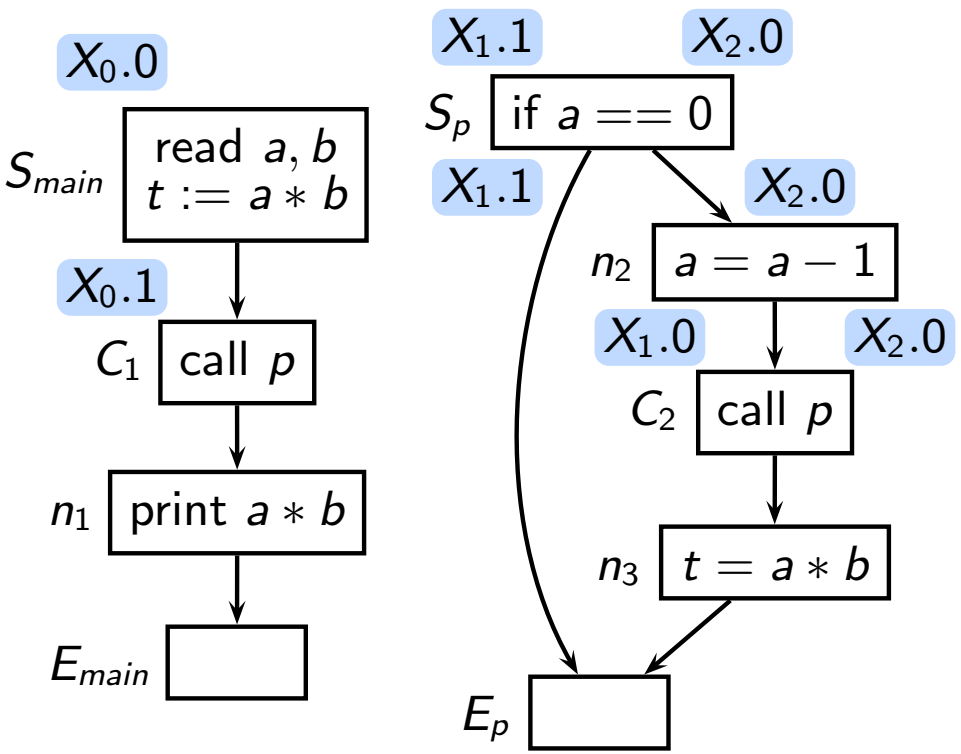


Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1
$X_2 = \langle$p,0$\rangle$	1

Available Expressions Analysis Using Value Contexts

$WL = [X_2|C_2, X_2|n_3, X_2|E_p, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$

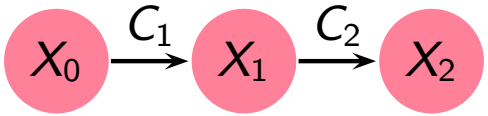


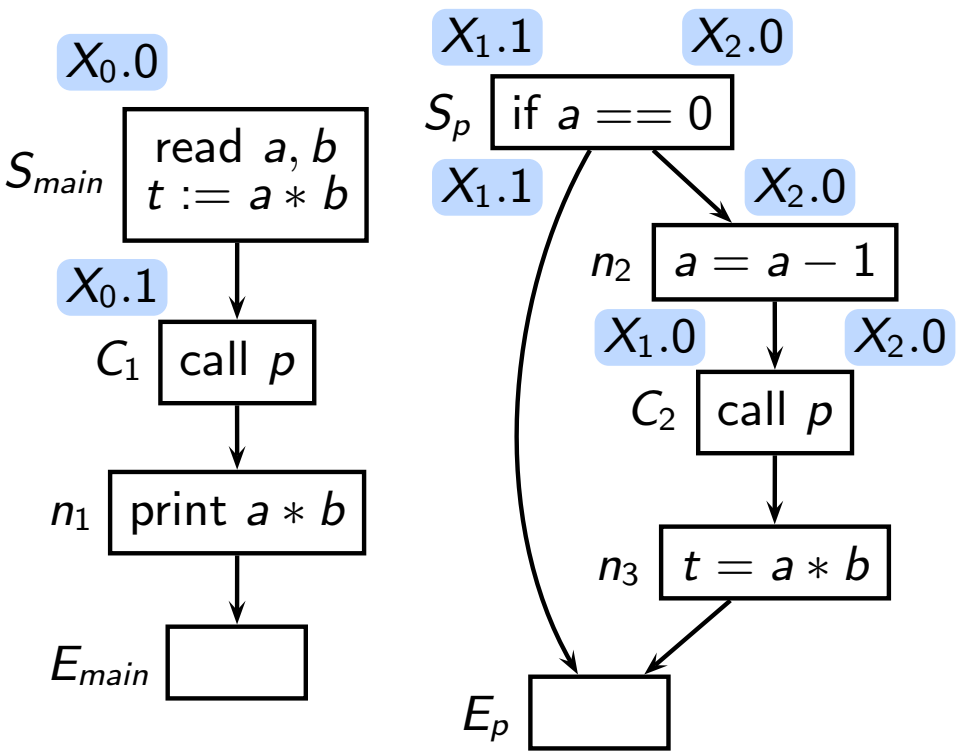


Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1
$X_2 = \langle$p,0$\rangle$	1

Available Expressions Analysis Using Value Contexts

$WL = [X_2|C_2, X_2|n_3, X_2|E_p, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$



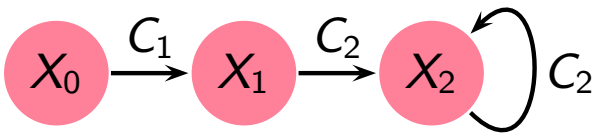


Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1
$X_2 = \langle$p,0$\rangle$	1

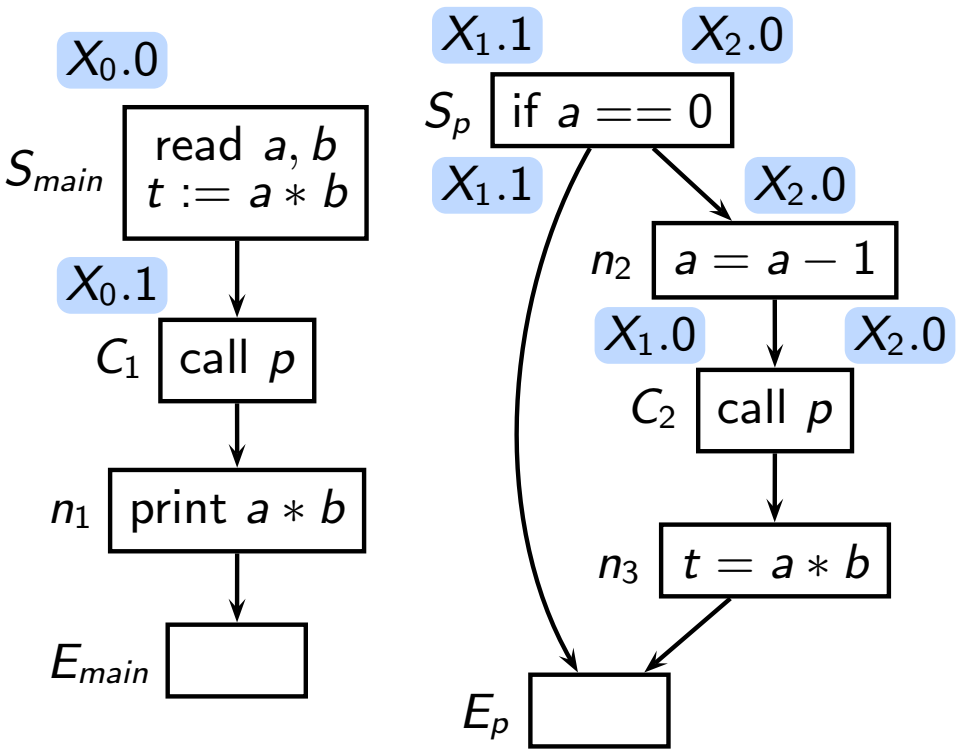
p has context X_2 with value 0 so no need to create a new context

Available Expressions Analysis Using Value Contexts

$WL = [X_2|C_2, X_2|n_3, X_2|E_p, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$



Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	1

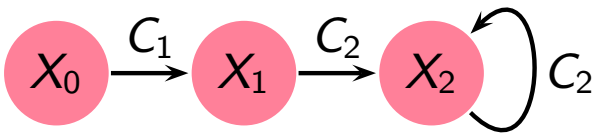


p has context X_2 with value 0 so no need to create a new context

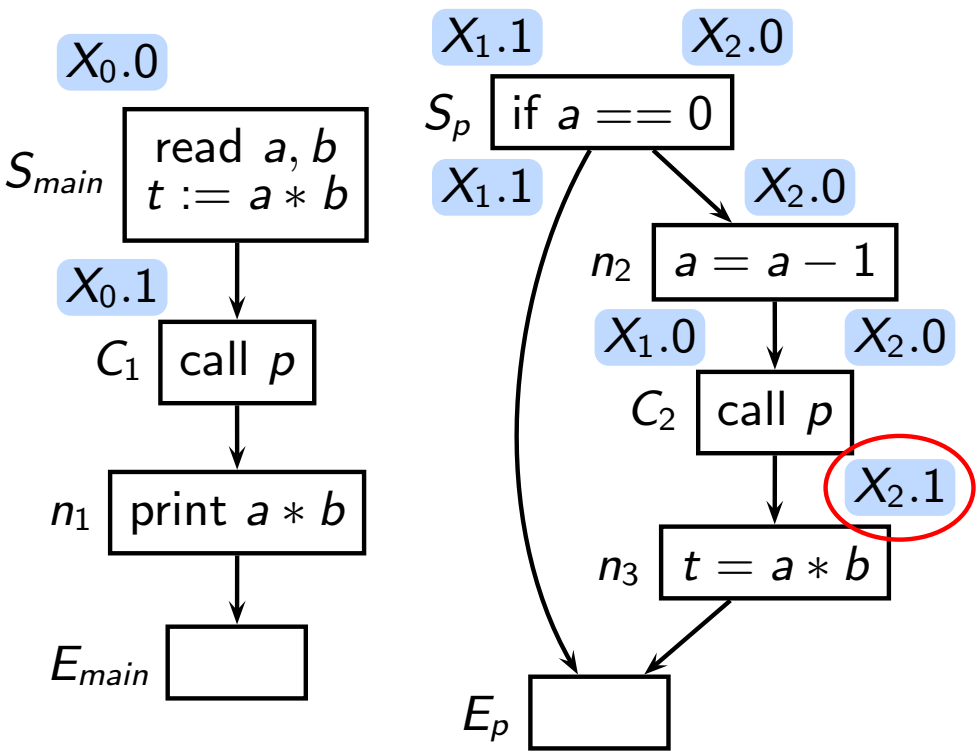
Record the transition from context X_2 to itself

Available Expressions Analysis Using Value Contexts

$$WL = [X_2|C_2, X_2|n_3, X_2|E_p, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$$



Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	1



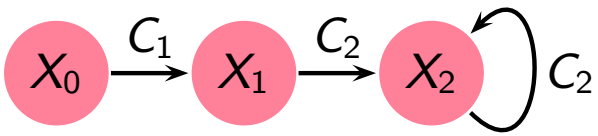
p has context X_2 with value 0 so no need to create a new context

Record the transition from context X_2 to itself

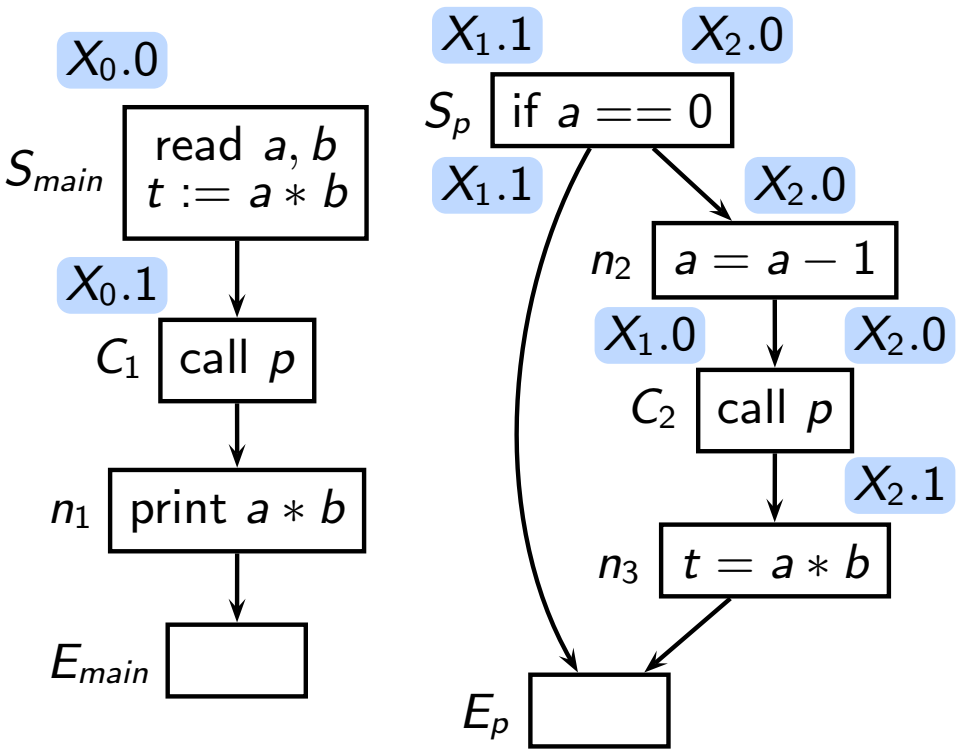
Use the $exitValue(X_2)$ to compute $Out_{C_2}[X_2]$

Available Expressions Analysis Using Value Contexts

$WL = [X_2|n_3, X_2|E_p, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$

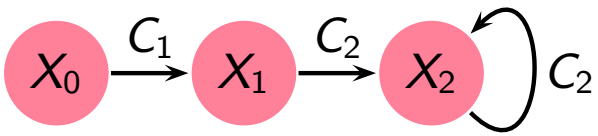


Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1
$X_2 = \langle$p,0$\rangle$	1

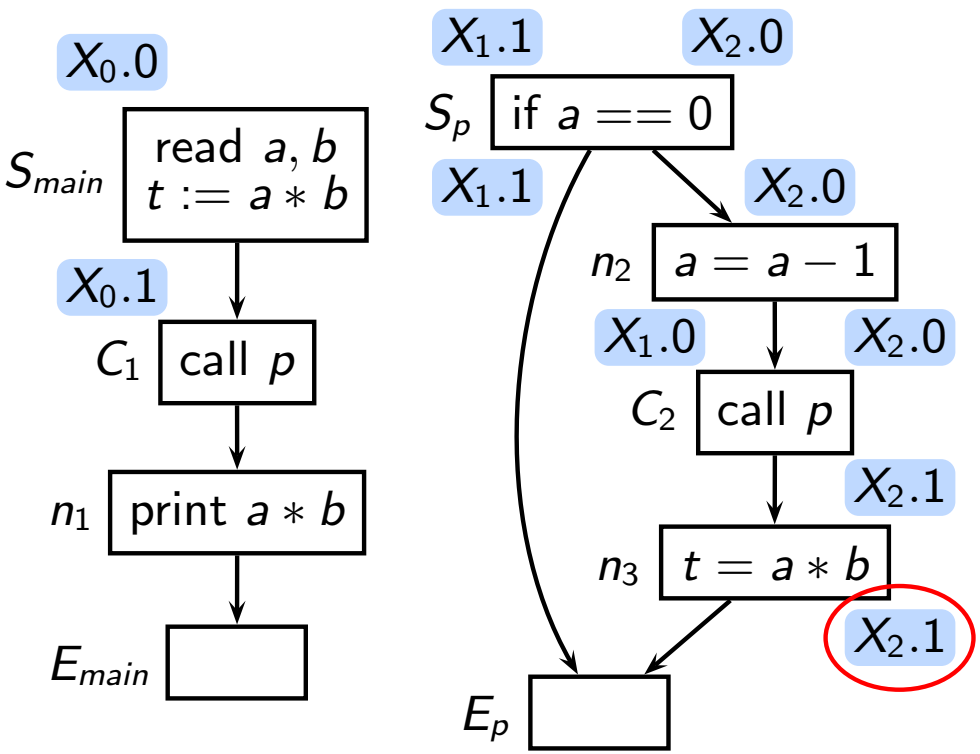


Available Expressions Analysis Using Value Contexts

$WL = [X_2|n_3, X_2|E_p, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$

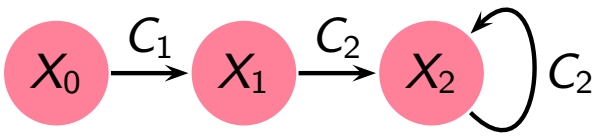


Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	1

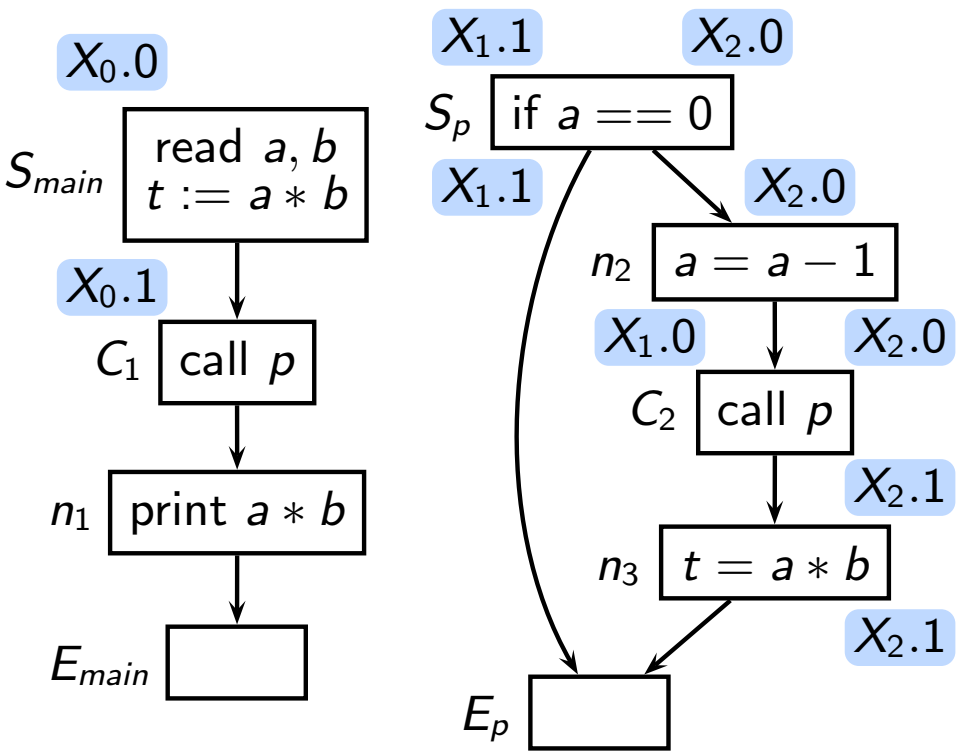


Available Expressions Analysis Using Value Contexts

$WL = [X_2|E_p, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$

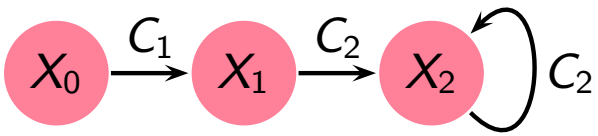


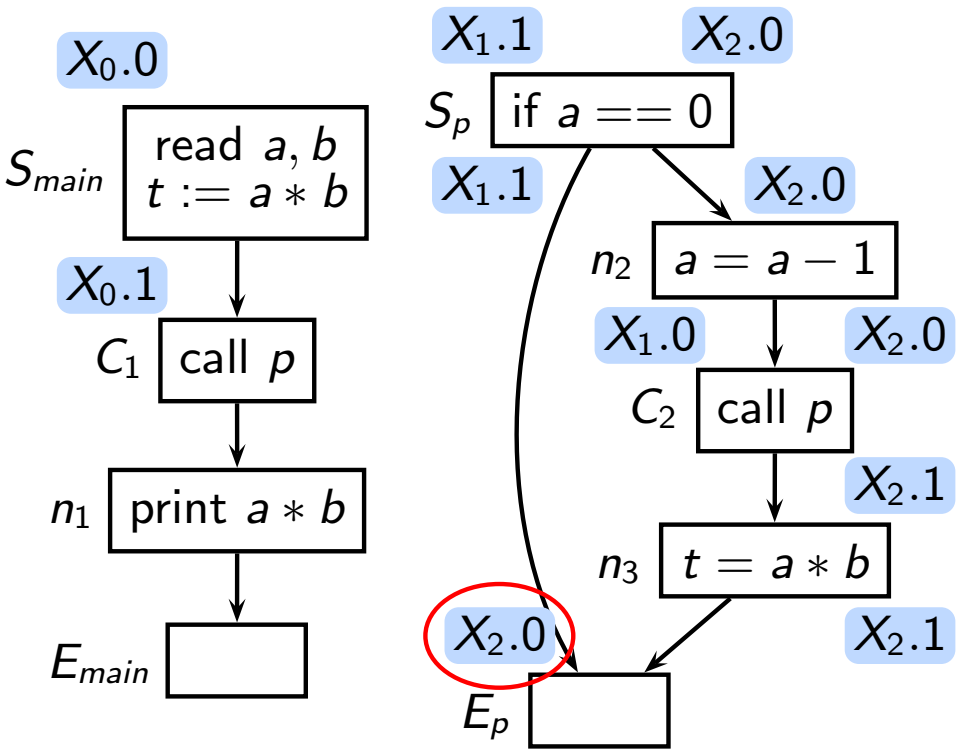
Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1
$X_2 = \langle$p,0$\rangle$	1



Available Expressions Analysis Using Value Contexts

$WL = [X_2|E_p, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$



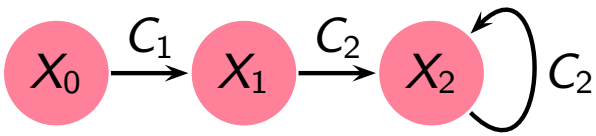


Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1
$X_2 = \langle$p,0$\rangle$	1

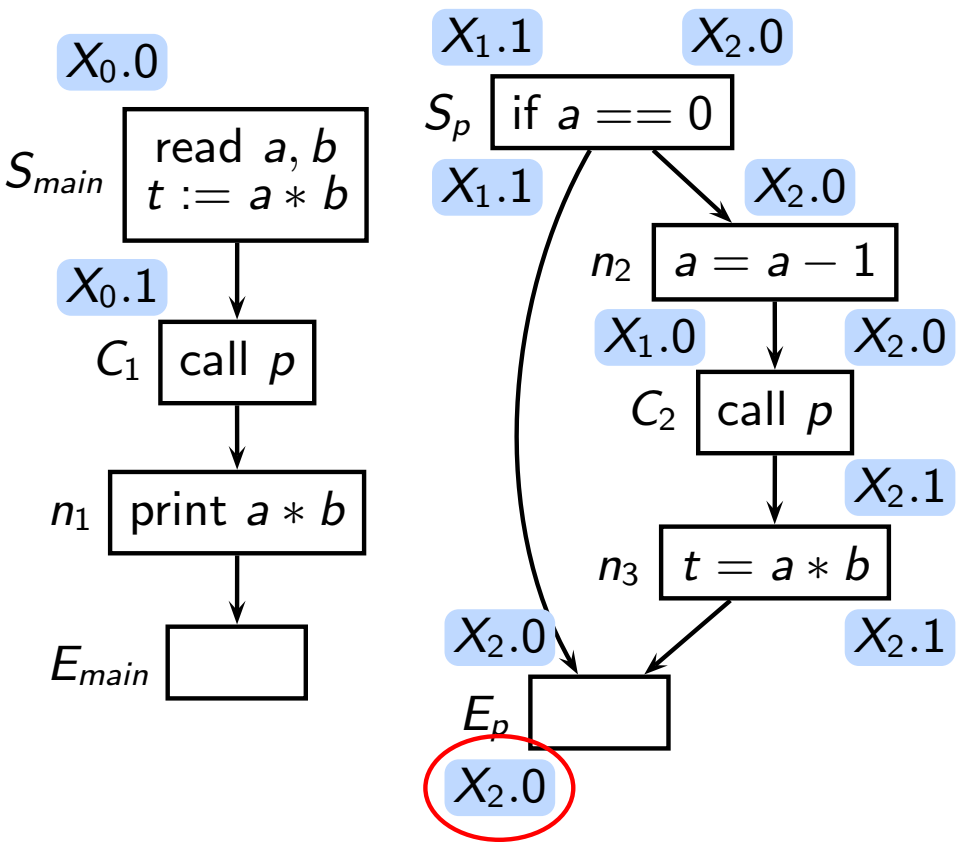
At E_p the values from S_p and n_3 are merged for context X_2

Available Expressions Analysis Using Value Contexts

$WL = [X_2|E_p, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$



Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	0

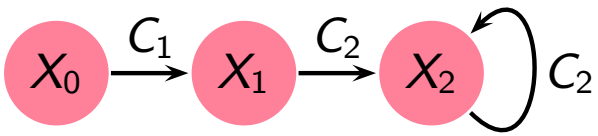


At E_p the values from S_p and n_3 are merged for context X_2

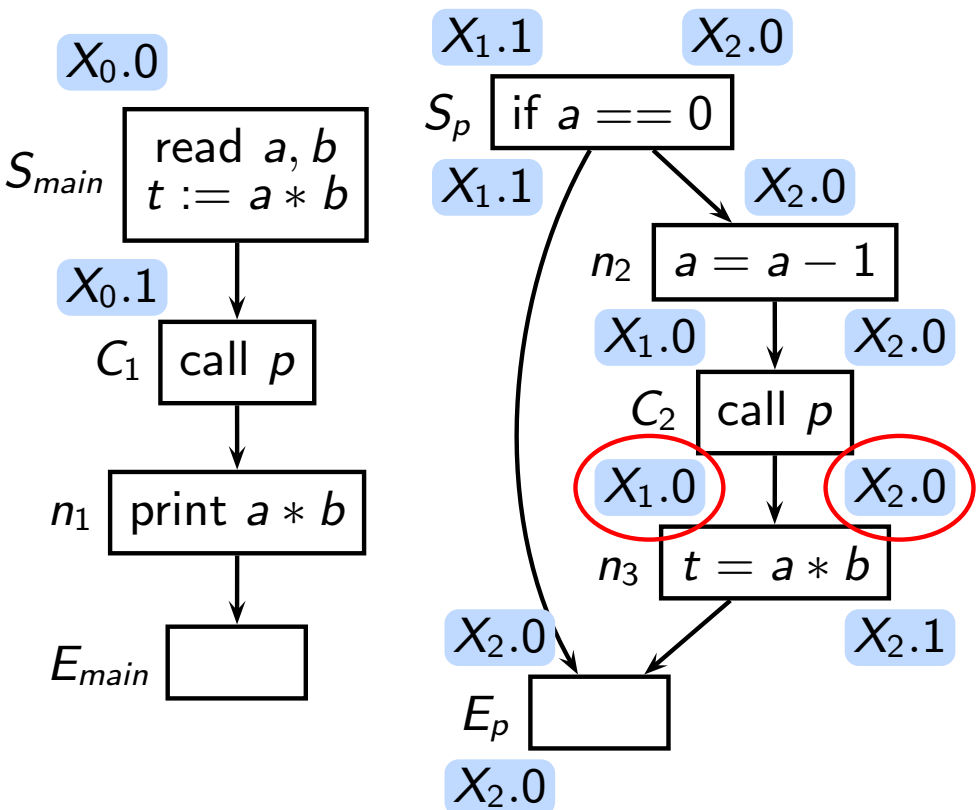
exitValue(X_2) is set to 0

Available Expressions Analysis Using Value Contexts

$WL = [X_2|E_p, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$



Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	0



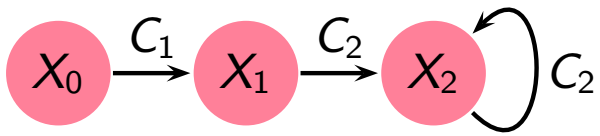
At E_p the values from S_p and n_3 are merged for context X_2

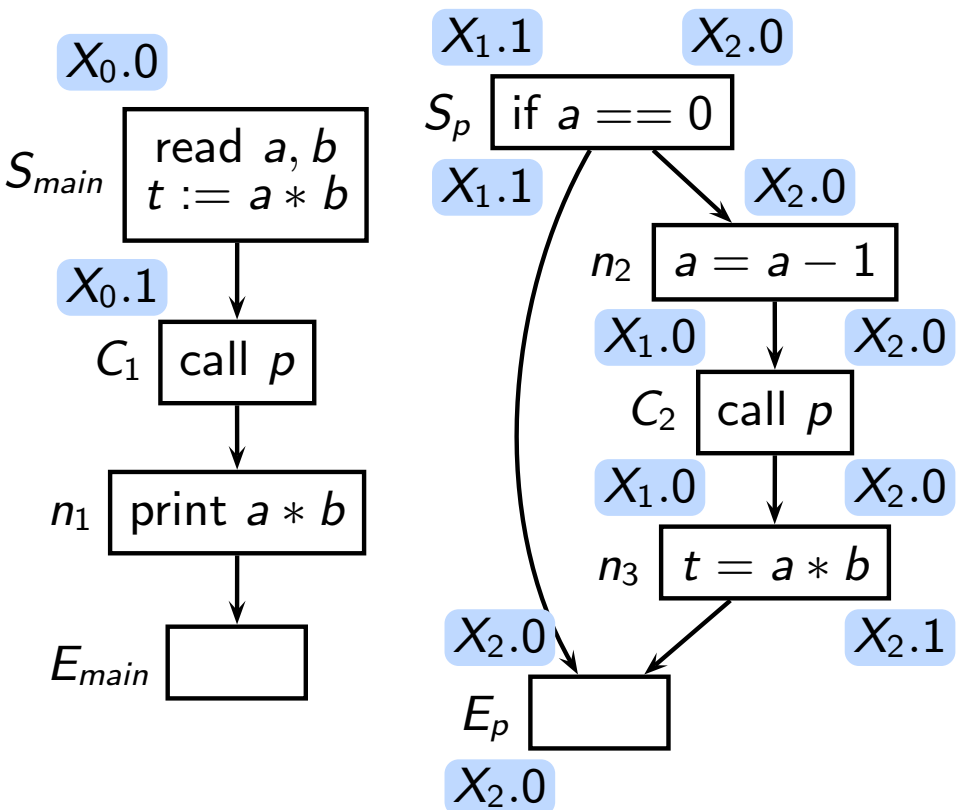
exitValue(X_2) is set to 0

Since X_2 has transitions $X_1 \xrightarrow{C_2} X_2$ and $X_2 \xrightarrow{C_2} X_2$, $Out_{C_2}[X_1]$ and $Out_{C_2}[X_2]$ become 0

Available Expressions Analysis Using Value Contexts

$WL = [X_2|n_3, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$





Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1
$X_2 = \langle$p,0$\rangle$	0

At E_p the values from S_p and n_3 are merged for context X_2

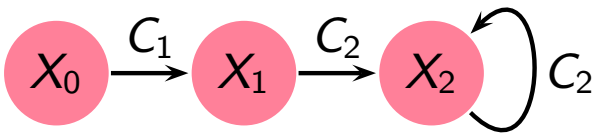
exitValue(X_2) is set to 0

Since X_2 has transitions $X_1 \xrightarrow{C_2} X_2$ and $X_2 \xrightarrow{C_2} X_2$, $Out_{C_2}[X_1]$ and $Out_{C_2}[X_2]$ become 0

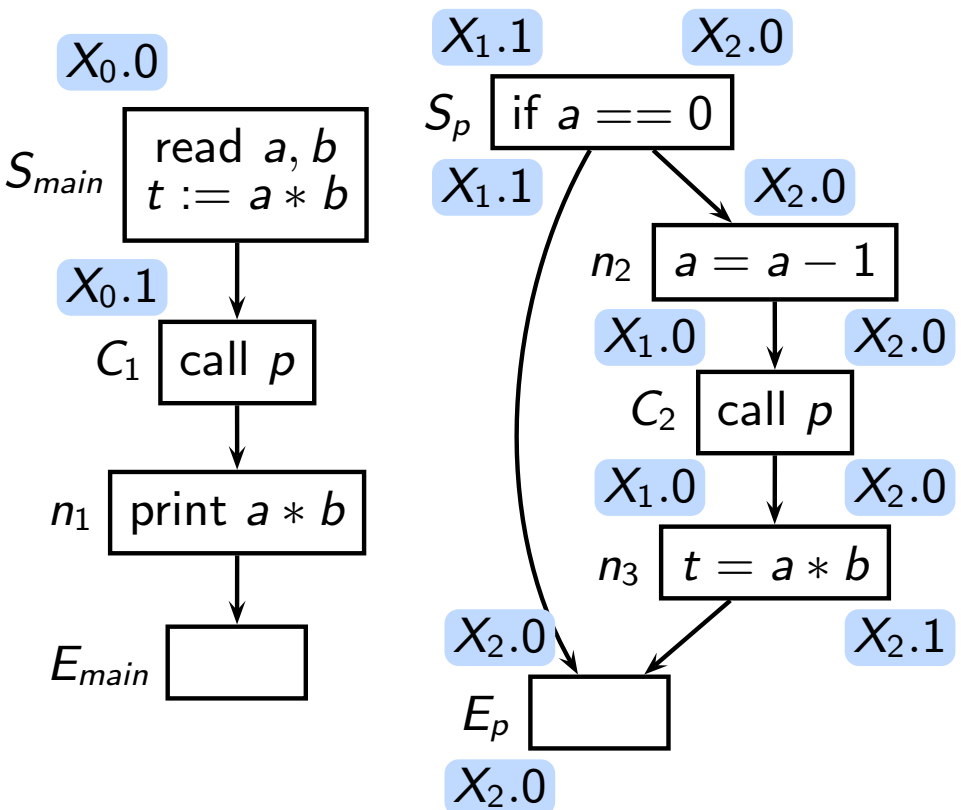
Since $Out_{C_2}[X_2]$ changes, $X_2|n_3$ is added to the work list

Available Expressions Analysis Using Value Contexts

$WL = [X_2|n_3, X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$



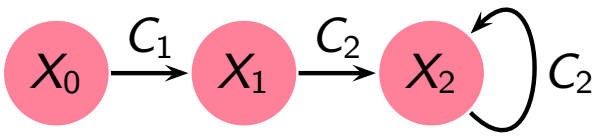
Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	0



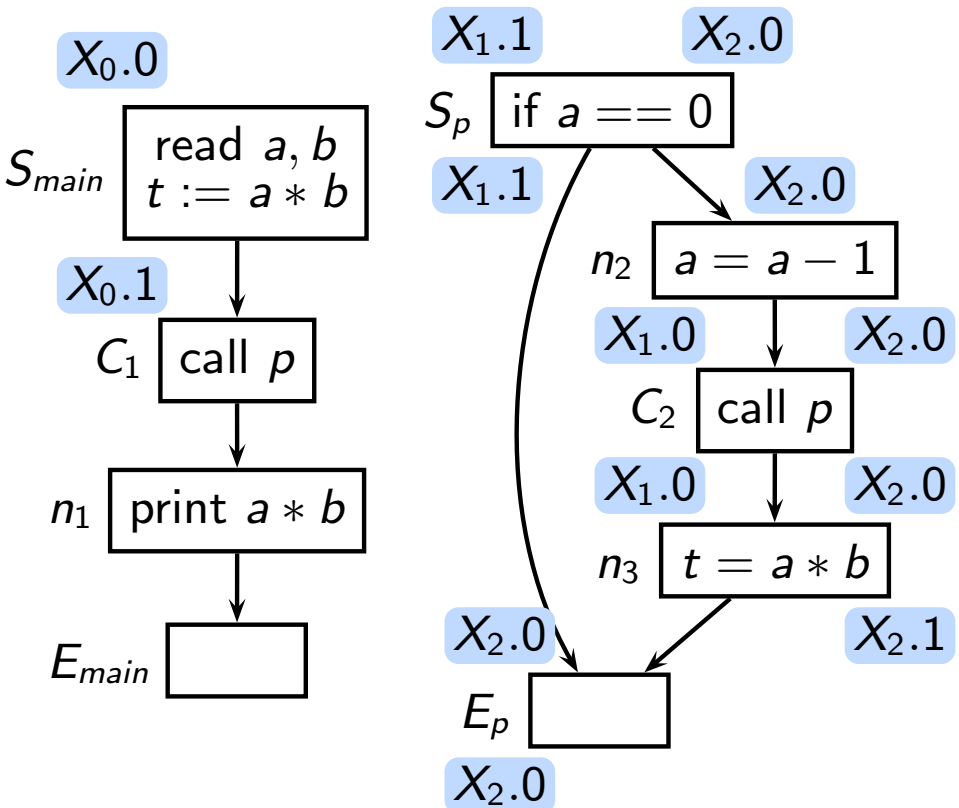
There is no change in $Out_{n_3}[X_2]$ (because it was initialized to $\top$)

Available Expressions Analysis Using Value Contexts

$WL = [X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$



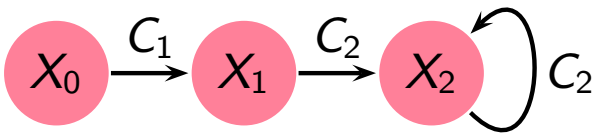
Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	0



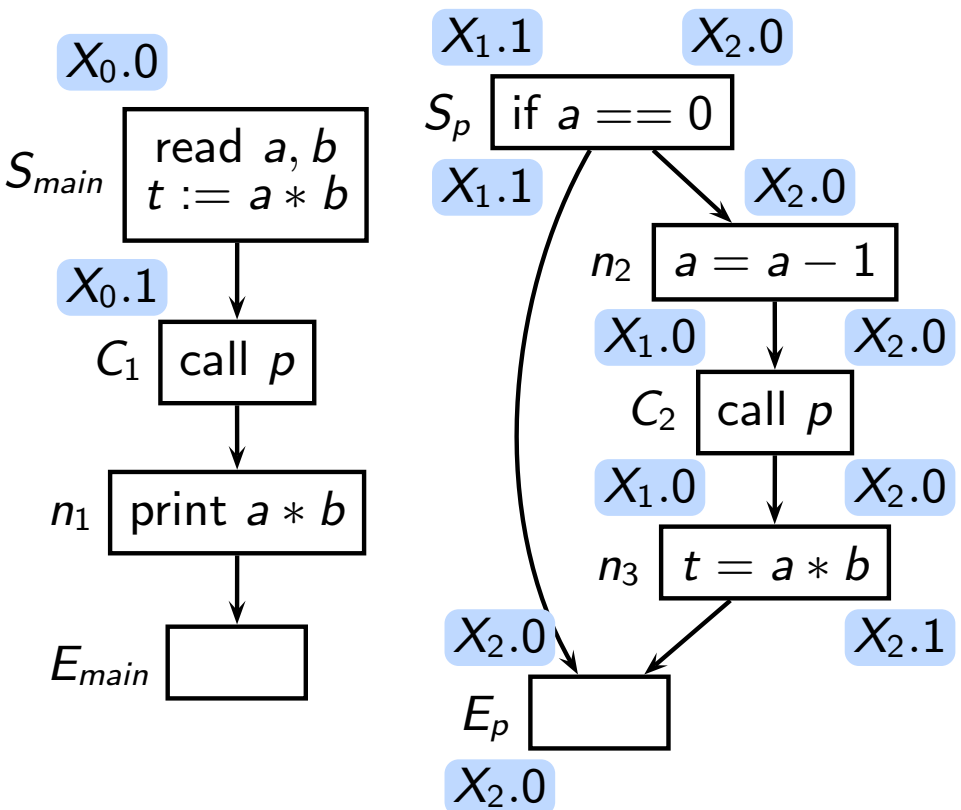
There is no change in $Out_{n_3}[X_2]$ (because it was initialized to $\top$)

Available Expressions Analysis Using Value Contexts

$WL = [X_1|n_3, X_1|E_p, X_0|n_1, X_0|E_m]$



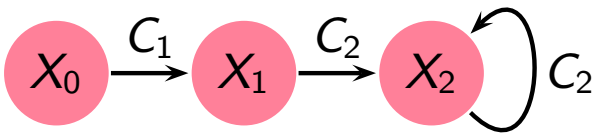
Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	0



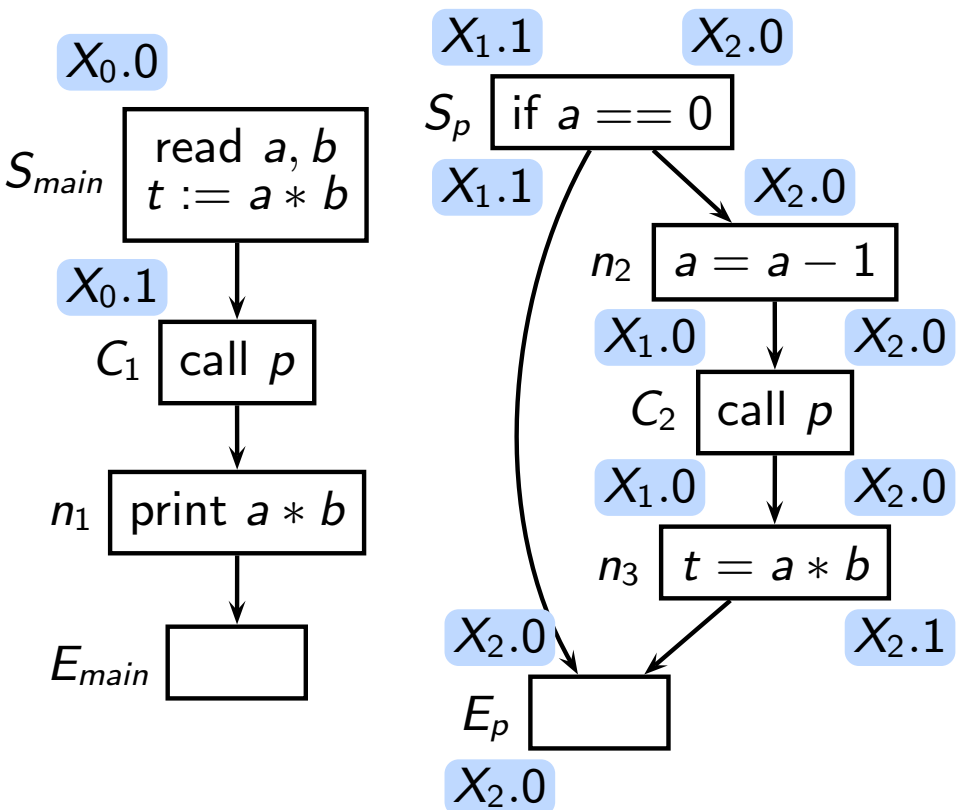
There is no change in $Out_{n_3}[X_1]$ either

Available Expressions Analysis Using Value Contexts

$WL = [X_1|E_p, X_0|n_1, X_0|E_m]$



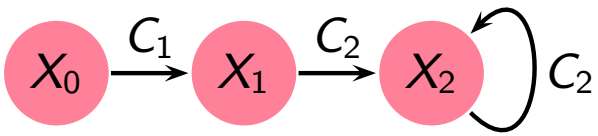
Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1
$X_2 = \langle$p,0$\rangle$	0

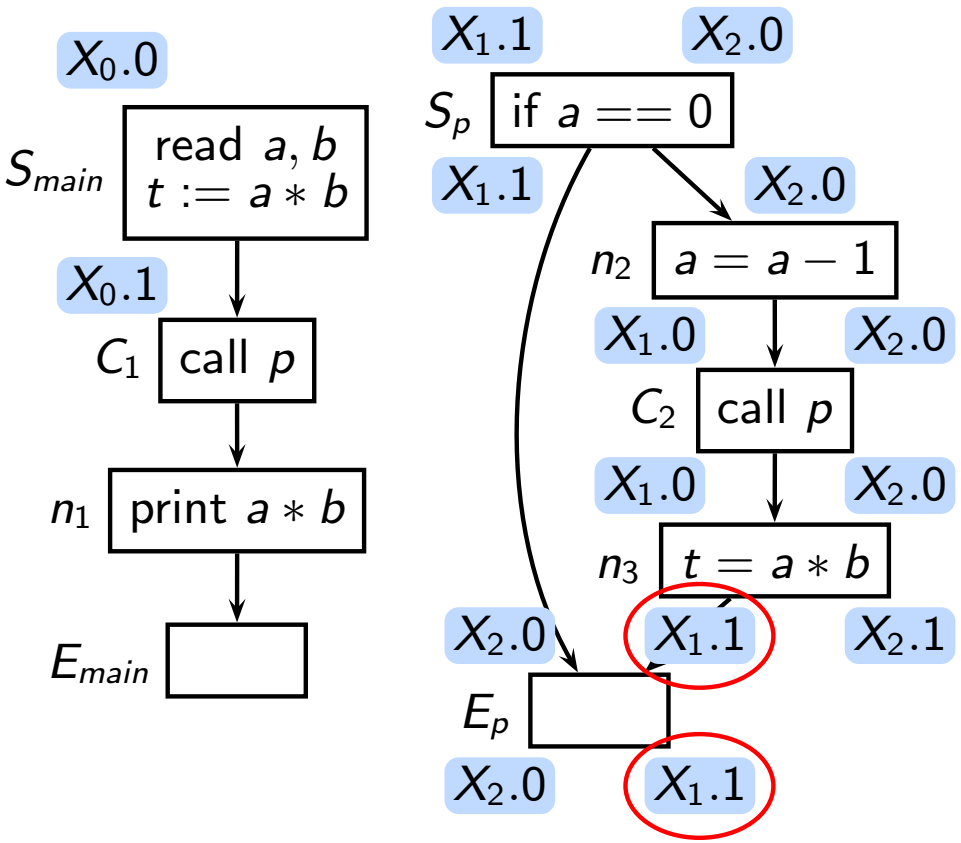


There is no change in $Out_{n_3}[X_1]$ either

Available Expressions Analysis Using Value Contexts

$WL = [X_1|E_p, X_0|n_1, X_0|E_m]$



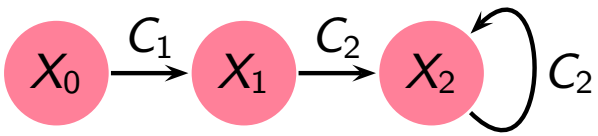


Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1
$X_2 = \langle$p,0$\rangle$	0

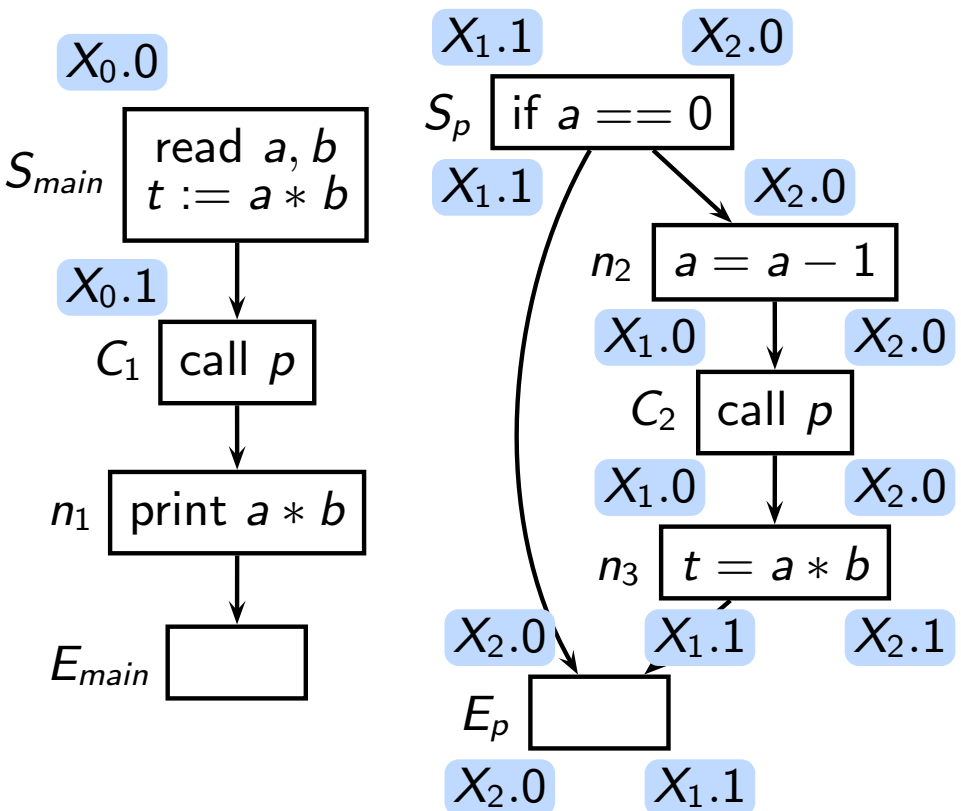
At E_p the values from S_p and n_3 are merged for context X_1

Available Expressions Analysis Using Value Contexts

$WL = [X_1|E_p, X_0|n_1, X_0|E_m]$



Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	0

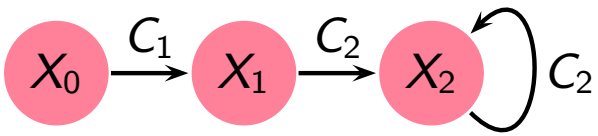


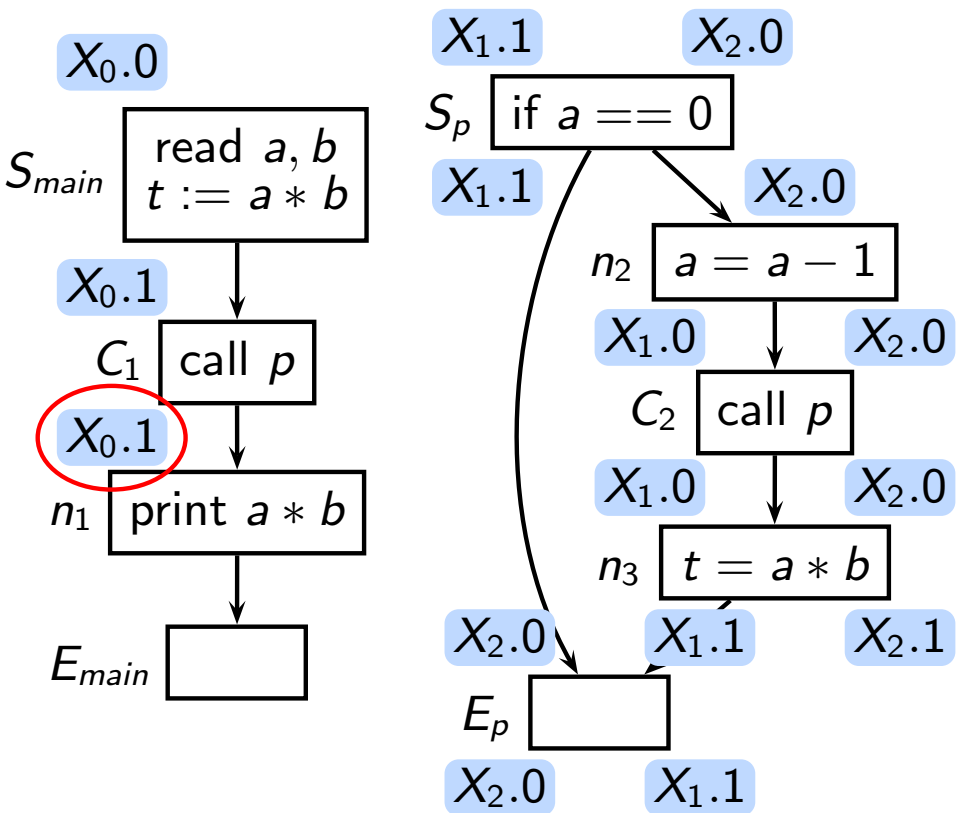
At E_p the values from S_p and n_3 are merged for context X_1

exitValue(X_1) remains 1

Available Expressions Analysis Using Value Contexts

$WL = [X_1|E_p, X_0|n_1, X_0|E_m]$





Context	*exitValue*
$X_0 = \langle$main,0$\rangle$	1
$X_1 = \langle$p,1$\rangle$	1
$X_2 = \langle$p,0$\rangle$	0

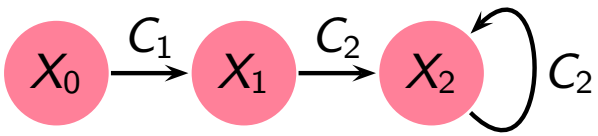
At E_p the values from S_p and n_3 are merged for context X_1

$exitValue(X_1)$ remains 1

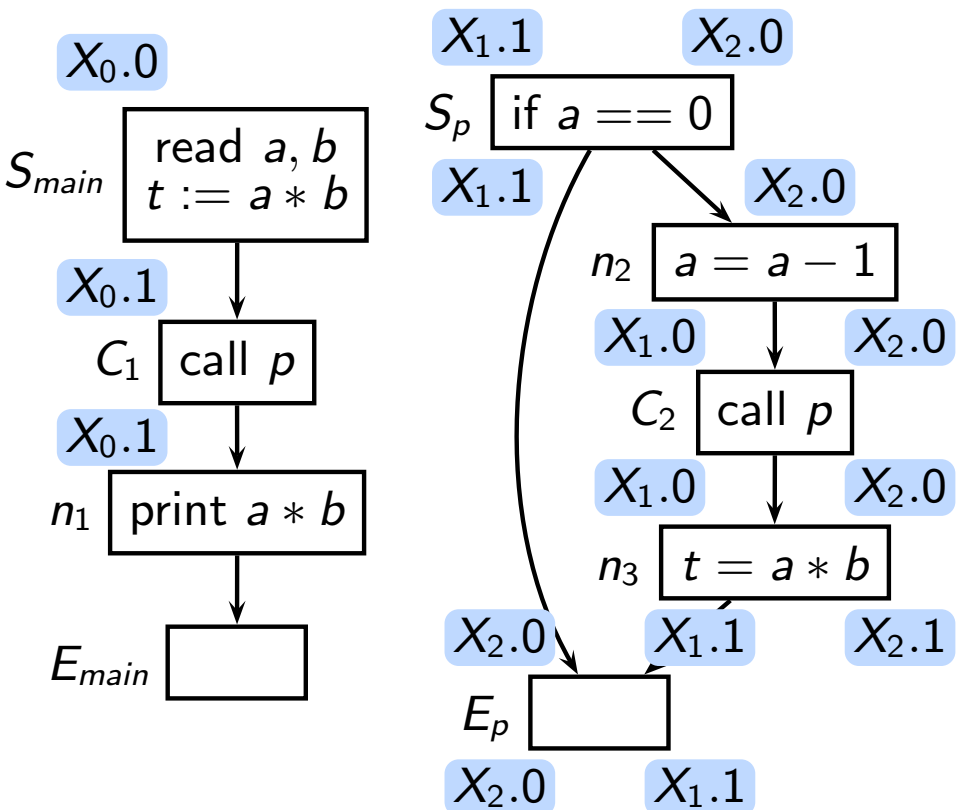
Since X_1 has transition $X_0 \xrightarrow{C_1} X_1$, $Out_{C_1}[X_0]$ becomes 1

Available Expressions Analysis Using Value Contexts

$WL = [X_0|n_1, X_0|E_m]$

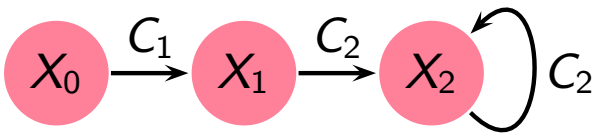


Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	0

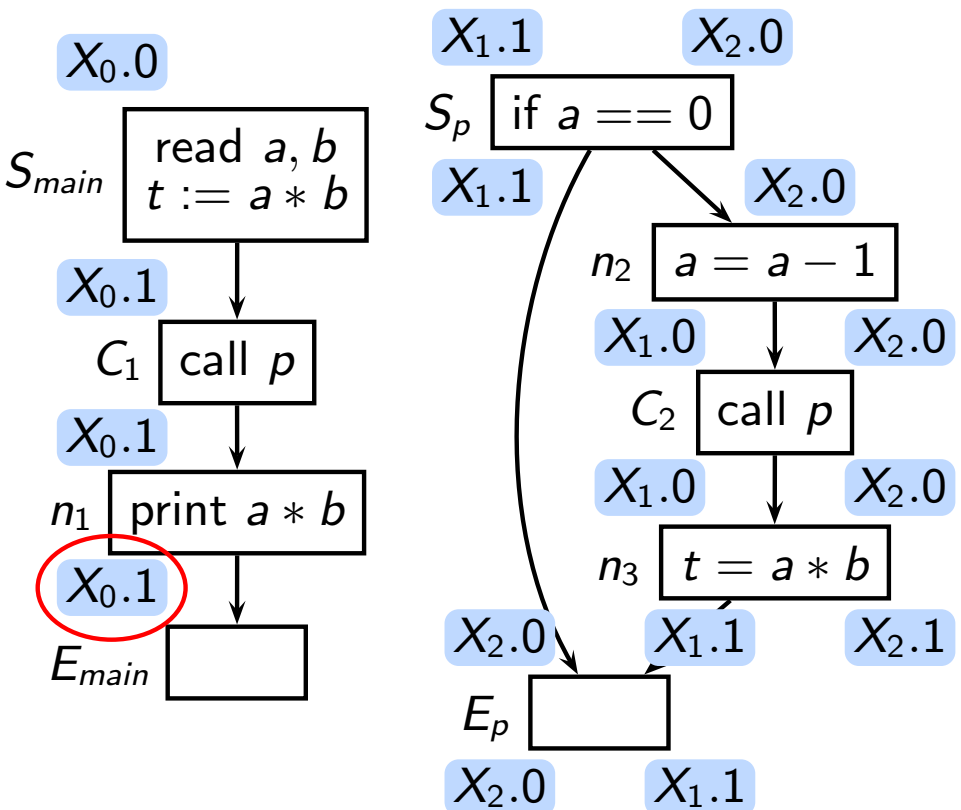


Available Expressions Analysis Using Value Contexts

$WL = [X_0|n_1, X_0|E_m]$

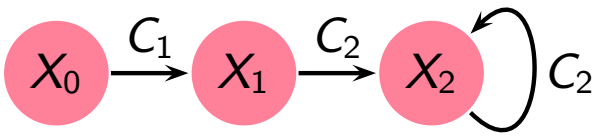


Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	0

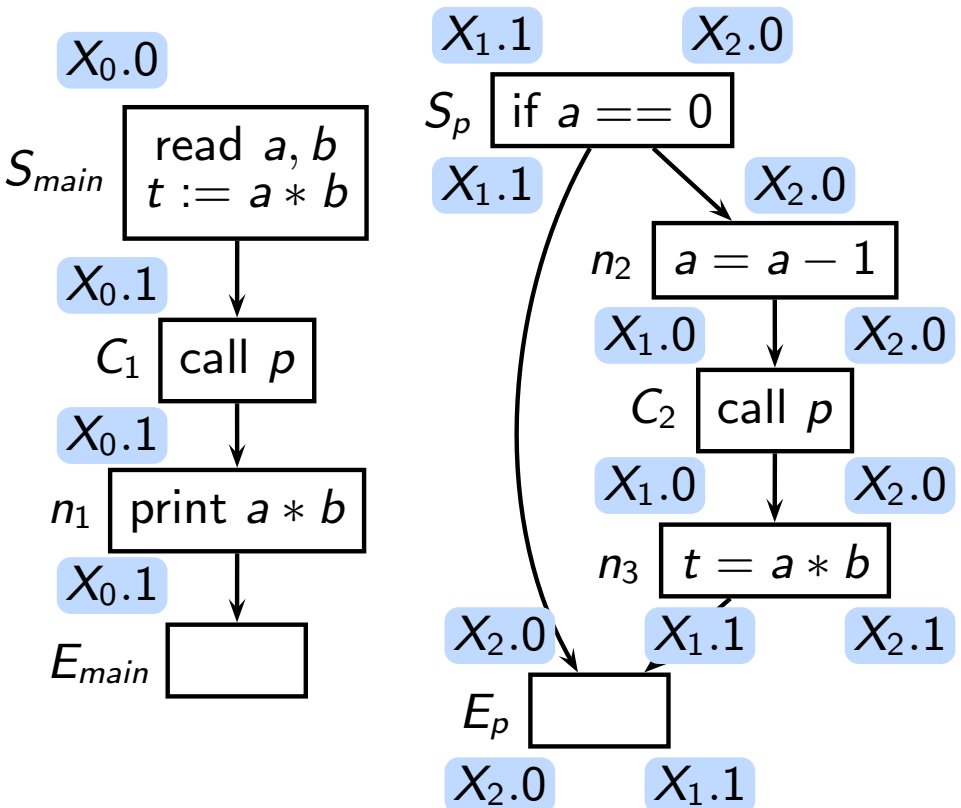


Available Expressions Analysis Using Value Contexts

$WL = [X_0|E_m]$

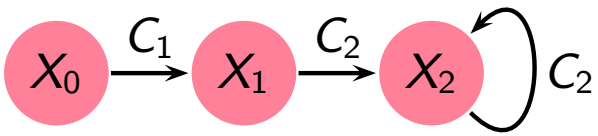


Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	0

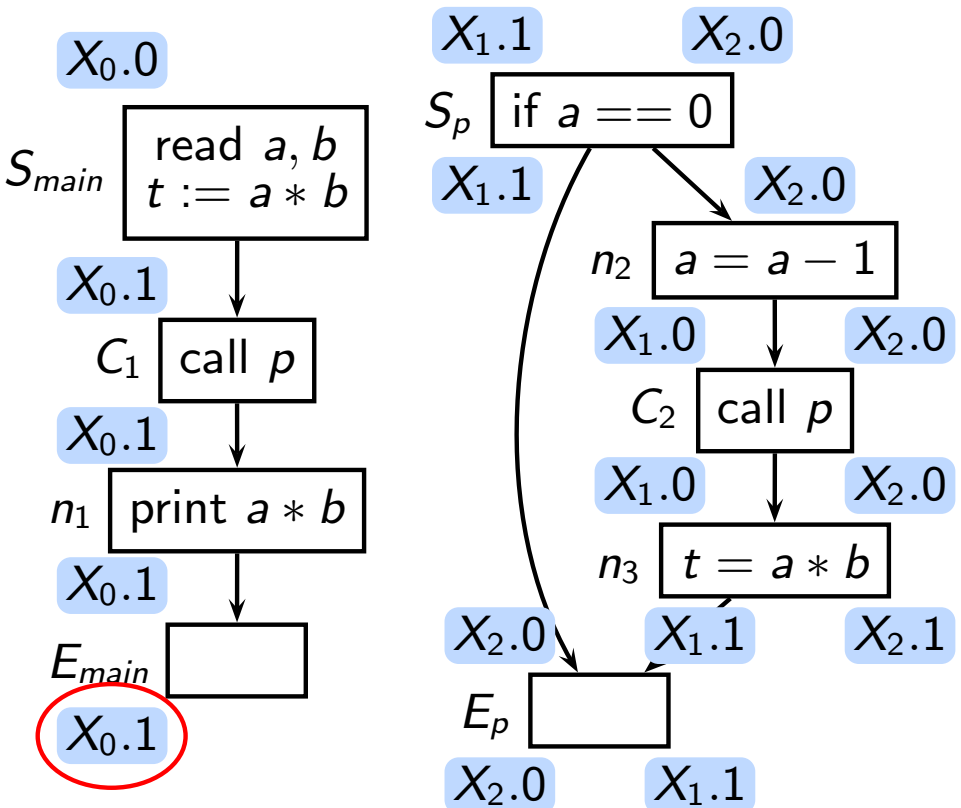


Available Expressions Analysis Using Value Contexts

$WL = [X_0 | E_m]$

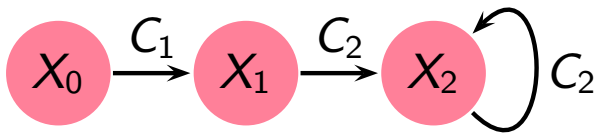


Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	0

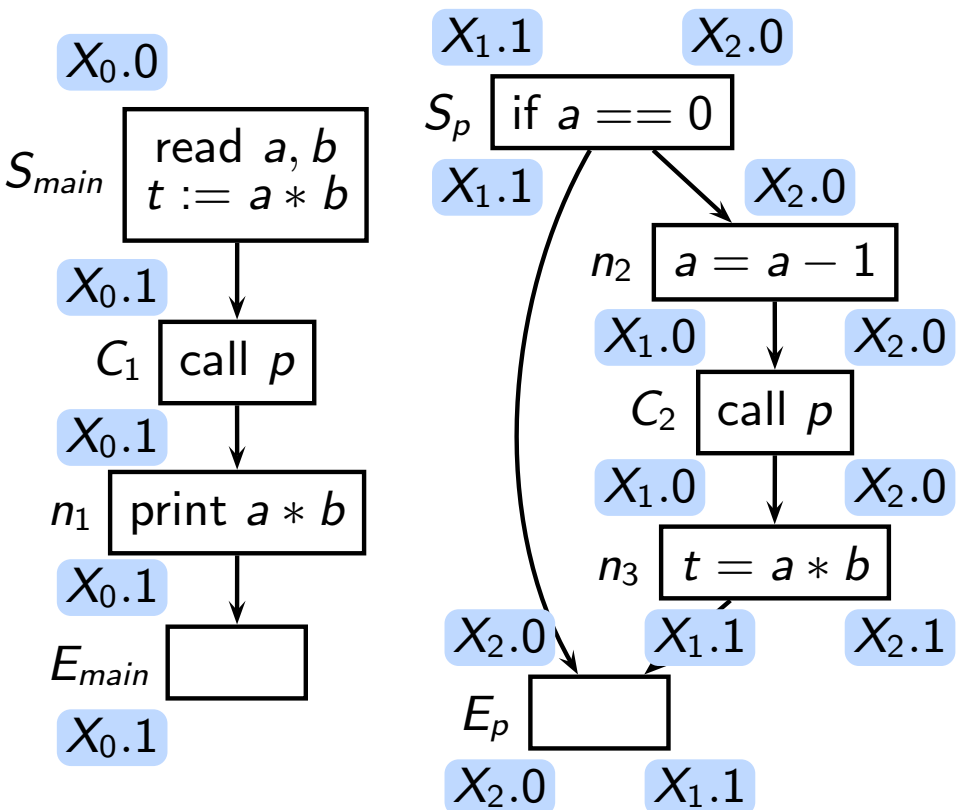


Available Expressions Analysis Using Value Contexts

$WL = [X_0|E_m]$

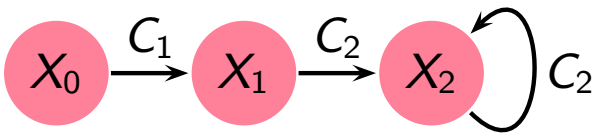


Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	0

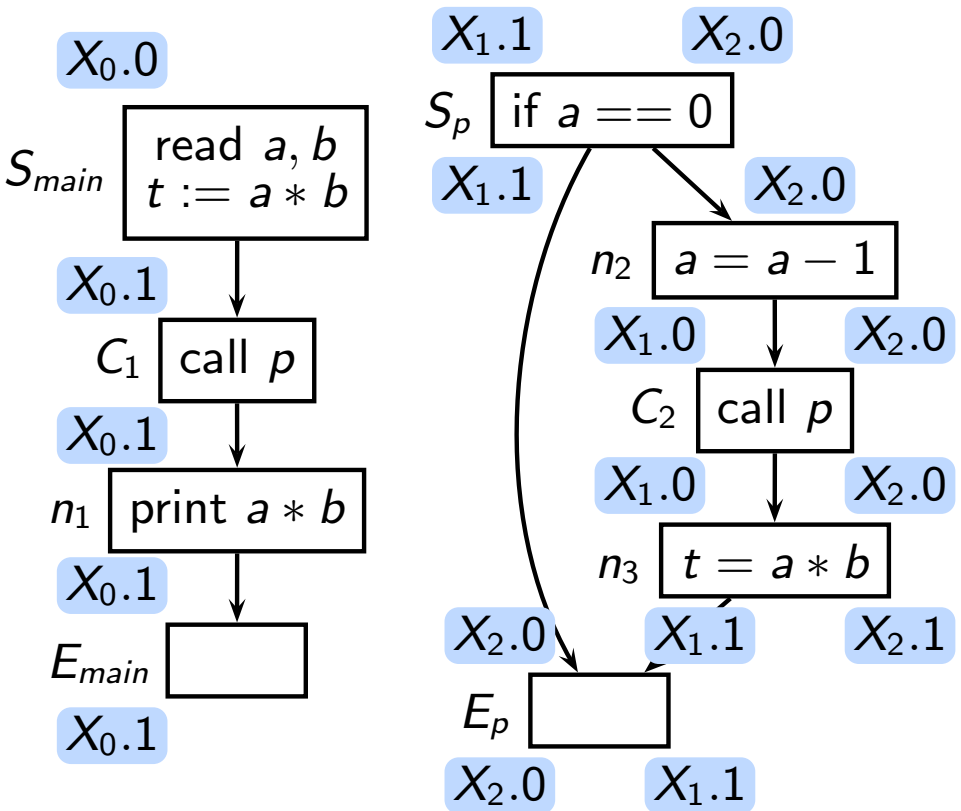


Available Expressions Analysis Using Value Contexts

$WL = [\ ]$



Context	*exitValue*
$X_0 = \langle \text{main},0 \rangle$	1
$X_1 = \langle \text{p},1 \rangle$	1
$X_2 = \langle \text{p},0 \rangle$	0



Work list is empty and the analysis is over

A Trace of Value Context Based Analysis (1)

S. No.	Work List	Sel. node	Data flow value	New context	New trans.	exit value	Addition to the work list
1				$X_0 = \langle m,0 \rangle$		$X_0.1$	$X_0 \vert S_m, X_0 \vert C_1, X_0 \vert n_1, X_0 \vert E_m$
2	$X_0 \vert S_m, X_0 \vert C_1, X_0 \vert n_1, X_0 \vert E_m$	S_m	$Out_{S_m}[X_0] = 1$				
3	$X_0 \vert C_1, X_0 \vert n_1, X_0 \vert E_m$	C_1		$X_1 = \langle p,1 \rangle$	$X_0 \xrightarrow{C_1} X_1$	$X_1.1$	$X_1 \vert S_p, X_1 \vert n_2, X_1 \vert C_2, X_1 \vert n_3, X_1 \vert E_p$
4	$X_1 \vert S_p, X_1 \vert n_2, X_1 \vert C_2, X_1 \vert n_3, X_1 \vert E_p, X_0 \vert n_1, X_0 \vert E_m$	S_p	$Out_{S_p}[X_1] = 1$				
5	$X_1 \vert n_2, X_1 \vert C_2, X_1 \vert n_3, X_1 \vert E_p, X_0 \vert n_1, X_0 \vert E_m$	n_2	$Out_{n_2}[X_1] = 0$				
6	$X_1 \vert C_2, X_1 \vert n_3, X_1 \vert E_p, X_0 \vert n_1, X_0 \vert E_m$	C_2		$X_2 = \langle p,0 \rangle$	$X_1 \xrightarrow{C_2} X_2$	$X_2.1$	$X_2 \vert S_p, X_2 \vert n_2, X_2 \vert C_2, X_2 \vert n_3, X_2 \vert E_p$
7	$X_2 \vert S_p, X_2 \vert n_2, X_2 \vert C_2, X_2 \vert n_3, X_2 \vert E_p, X_1 \vert n_3, X_1 \vert E_p, X_0 \vert n_1, X_0 \vert E_m$	S_p	$Out_{S_p}[X_2] = 0$				

A Trace of Value Context Based Analysis (2)

S. No.	Work List	Sel. node	Data flow value	New context	New trans.	exit value	Addition to the work list
8	$X_2\|n_2, X_2\|C_2, X_2\|n_3, X_2\|E_p,$ $X_1\|n_3, X_1\|E_p, X_0\|n_1, X_0\|E_m$	n_2	$Out_{n_2}[X_2]=0$				
9	$X_2\|C_2, X_2\|n_3, X_2\|E_p, X_1\|n_3,$ $X_1\|E_p, X_0\|n_1, X_0\|E_m$	C_2	$Out_{C_2}[X_2]=1$		$X_2 \xrightarrow{C_2} X_2$		
10	$X_2\|n_3, X_2\|E_p, X_1\|n_3, X_1\|E_p,$ $X_0\|n_1, X_0\|E_m$	n_3	$Out_{n_3}[X_2]=1$				
11	$X_2\|E_p, X_1\|n_3, X_1\|E_p, X_0\|n_1,$ $X_0\|E_m$	E_p	$Out_{E_p}[X_2]=0$ $Out_{C_2}[X_2]=0$ $Out_{C_2}[X_1]=0$			$X_2.0$	$X_2\|n_3$
12	$X_2\|n_3, X_1\|n_3, X_1\|E_p, X_0\|n_1,$ $X_0\|E_m$	n_3	No change				
13	$X_1\|n_3, X_1\|E_p, X_0\|n_1, X_0\|E_m$	n_3	$Out_{n_3}[X_1]=1$				
14	$X_1\|E_p, X_0\|n_1, X_0\|E_m$	E_p	$Out_{E_p}[X_1]=1$ $Out_{C_1}[X_0]=1$			$X_1.1$	
15	$X_0\|n_1, X_0\|E_m$	n_1	$Out_{n_1}[X_0]=1$				
16	$X_0\|E_m$	E_m	$Out_{E_m}[X_0]=1$				

Merging ExitValue with Previous Out Value at the Call Site

Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

- If $n = C_i$ calling procedure p
  - If some context $\langle p,v \rangle$ exists (say Y) /* p is the callee */
    - record the transition $X \xrightarrow{C_i} Y$
    - $Out_{C_i}[X] = Out_{C_i}[X] \sqcap exitValue(Y)$
    - if there is a change, add $X|m$, $\forall m \in succ(C_i)$ to WL

Merging ExitValue with Previous Out Value at the Call Site

Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

- If $n = C_i$ calling procedure p
  - If some context $\langle p,v \rangle$ exists (say Y) /* p is the callee */
    - record the transition $X \xrightarrow{C_i} Y$
    - $Out_{C_i}[X] = \boxed{Out_{C_i}[X] \sqcap} exitValue(Y)$
    - if there is a change, add $X|m$, $\forall m \in succ(C_i)$ to WL

Merging ExitValue with Previous Out Value at the Call Site

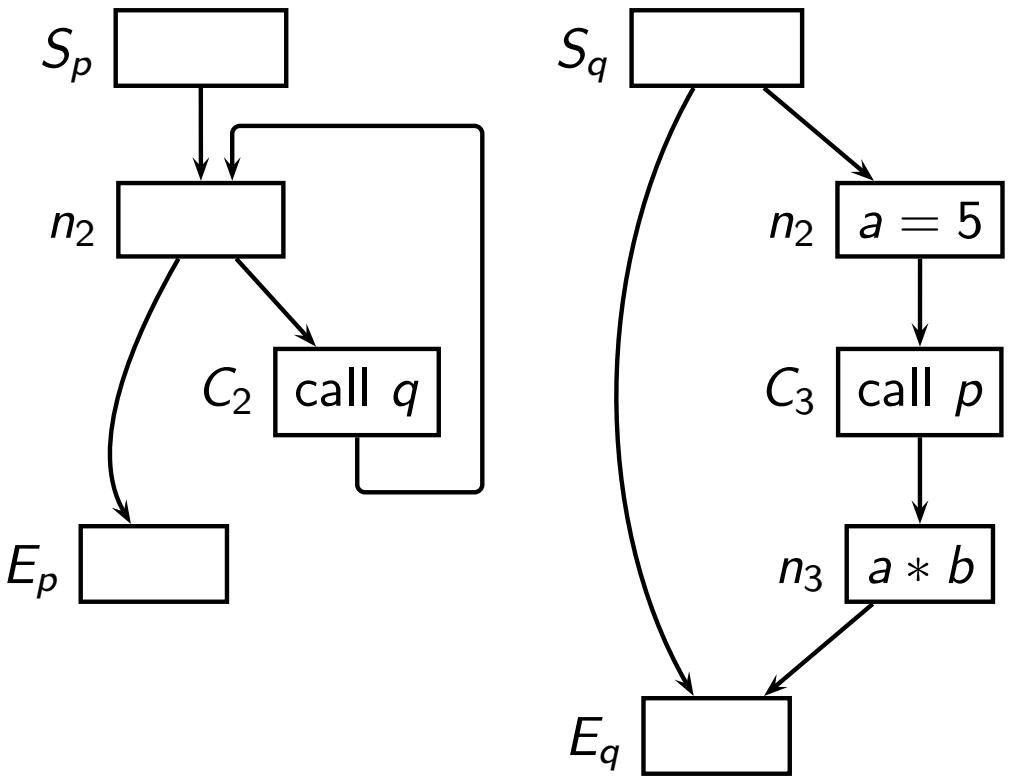
Select $X|n$ from WL. Compute In_n. Let $X.v$ be in In_n

- If $n = C_i$ calling procedure p
  - If some context $\langle p,v \rangle$ exists (say Y) /* p is the callee */
    - record the transition $X \xrightarrow{C_i} Y$
    - $Out_{C_i}[X] = \boxed{Out_{C_i}[X] \sqcap} exitValue(Y)$
    - if there is a change, add $X|m$, $\forall m \in succ(C_i)$ to WL

  Analogy:
  - At the intraprocedural level, we merge the values at the entry of a loop to compute the glb across all iterations of the loop
  - At the interprocedural level, we want to compute the glb across repeated calls at the same call site (perhaps in a loop)

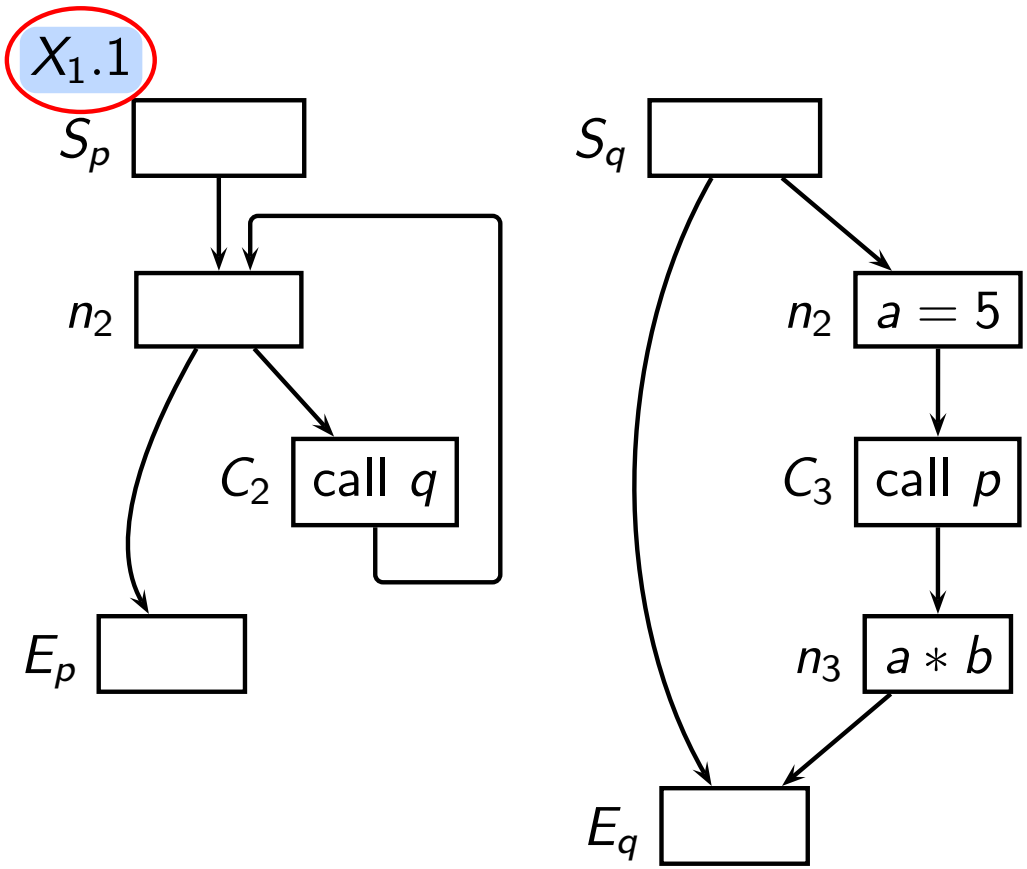
Partially Available Expressions Analysis Using Value Contexts

This example illustrates non-termination of analysis if the *exitValue* is not merged with the previous *Out* value



We assume that procedure main calls procedure p (and not q) and the expression $a * b$ is partially available on entry to p

Partially Available Expressions Analysis Using Value Contexts

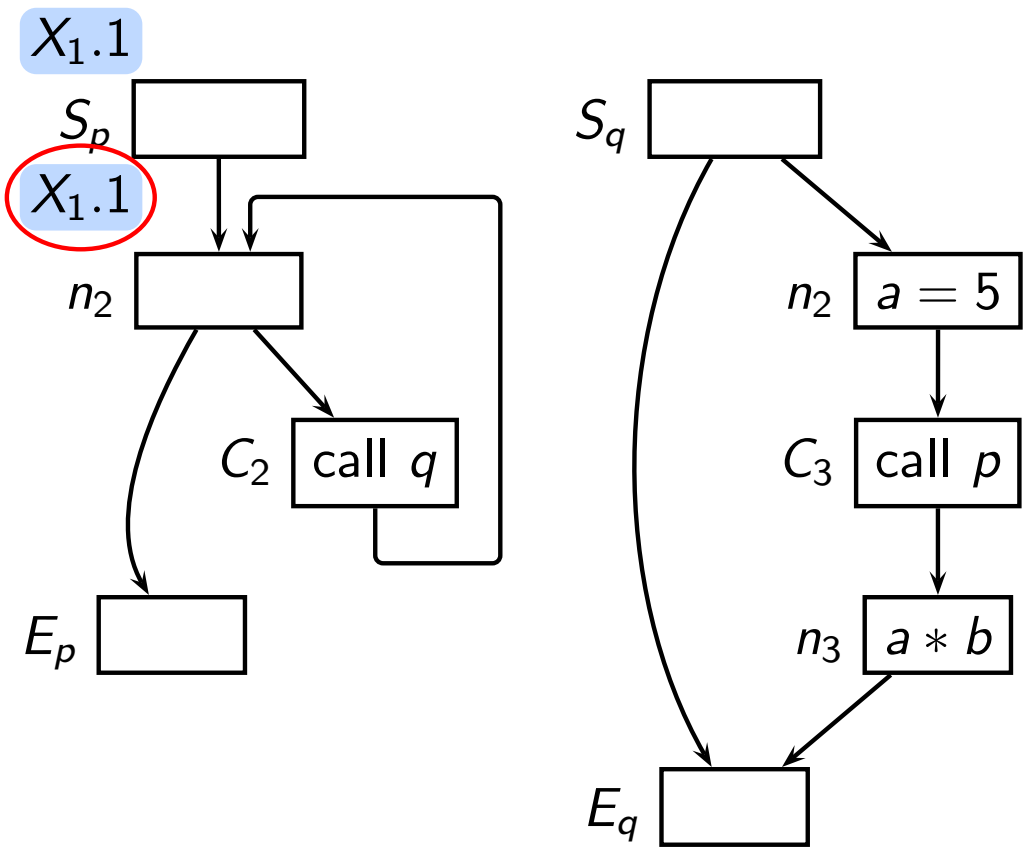


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0

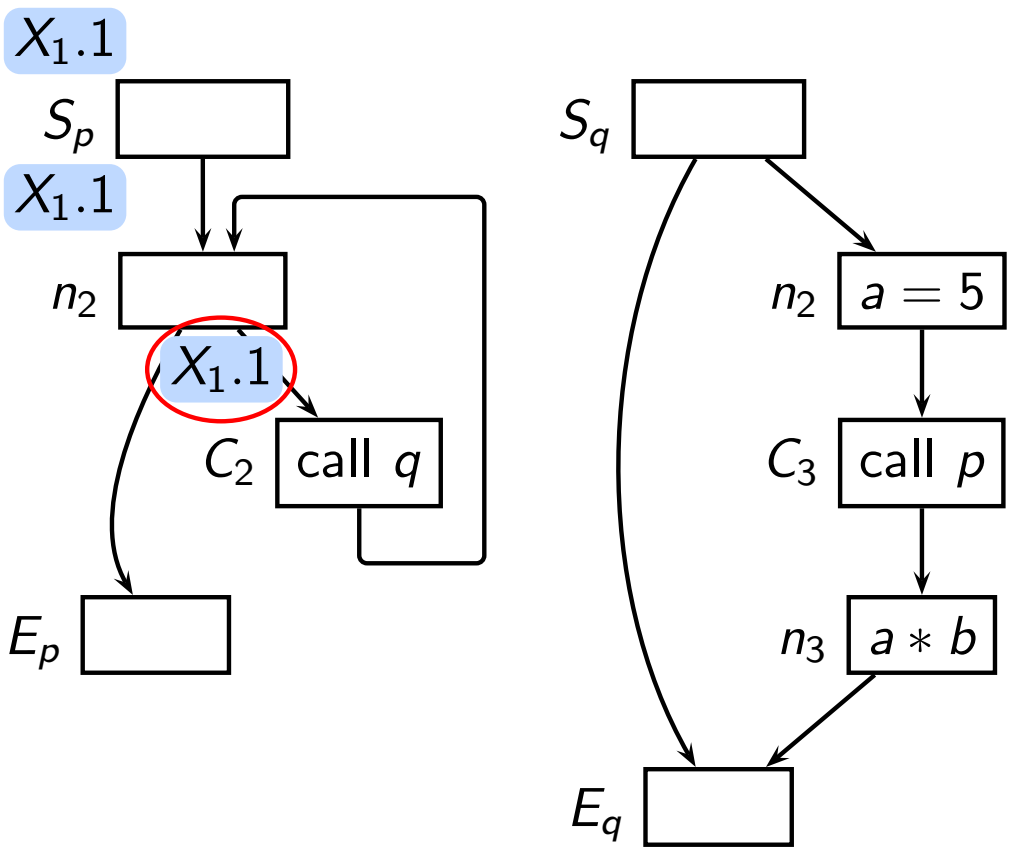
We create context X_1 for entry value 1 with *exitValue* as 0 ($\top$ for partially available expressions analysis)

Partially Available Expressions Analysis Using Value Contexts

Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0



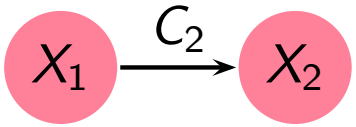
Partially Available Expressions Analysis Using Value Contexts

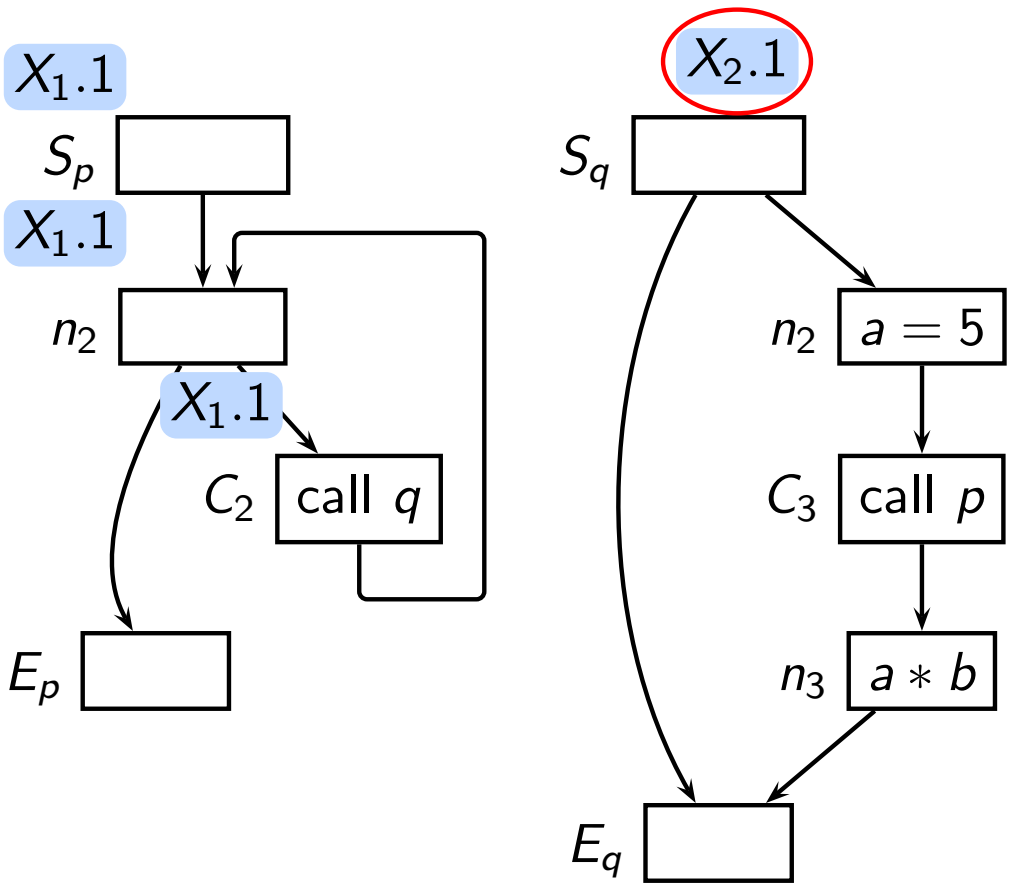


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0

Value 1 reaches q and a new context must be created for it

Partially Available Expressions Analysis Using Value Contexts

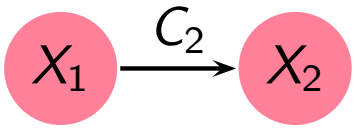


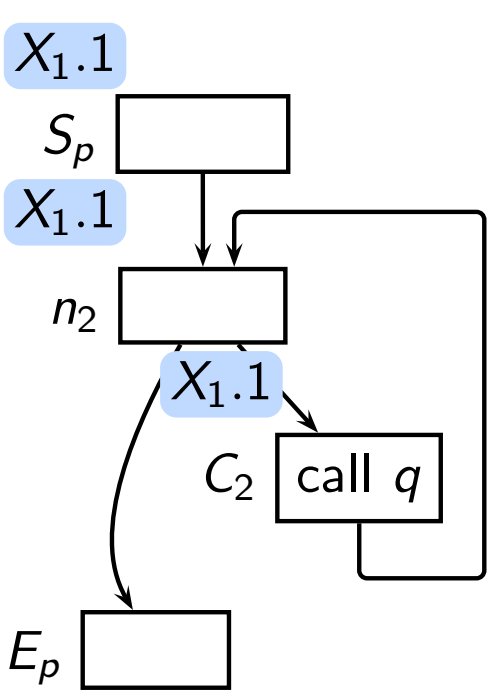


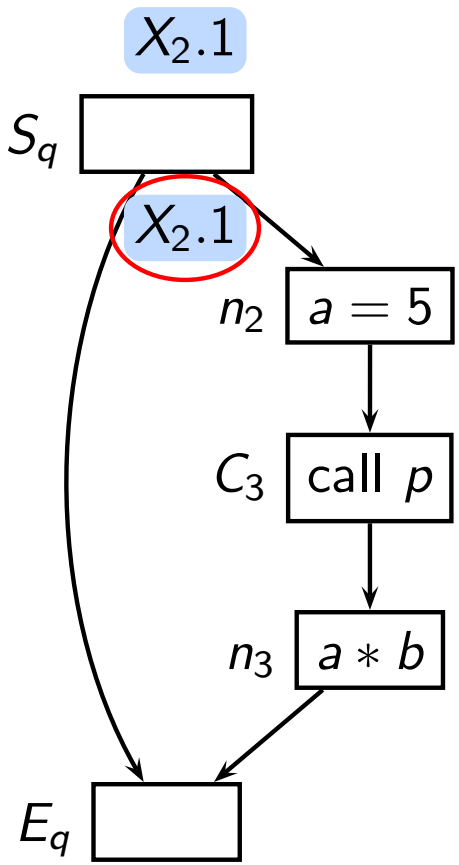
Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0

We create context X_2 for value 1 reaching q and record a transition from X_1 to X_2 on C_2

Partially Available Expressions Analysis Using Value Contexts

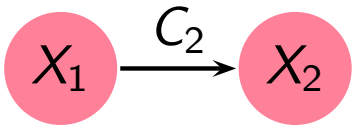


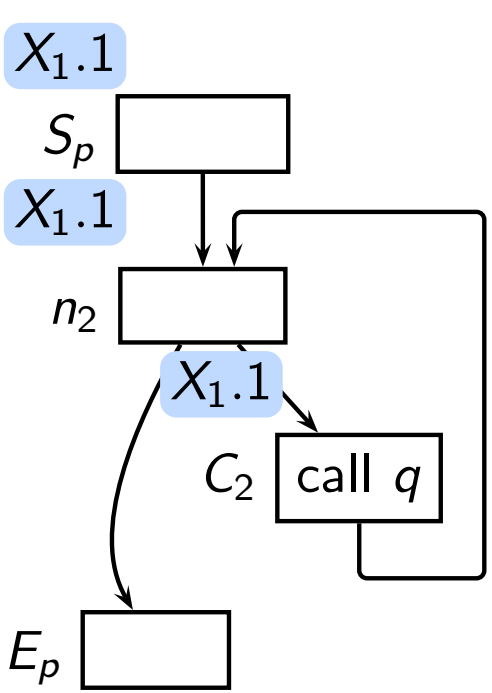


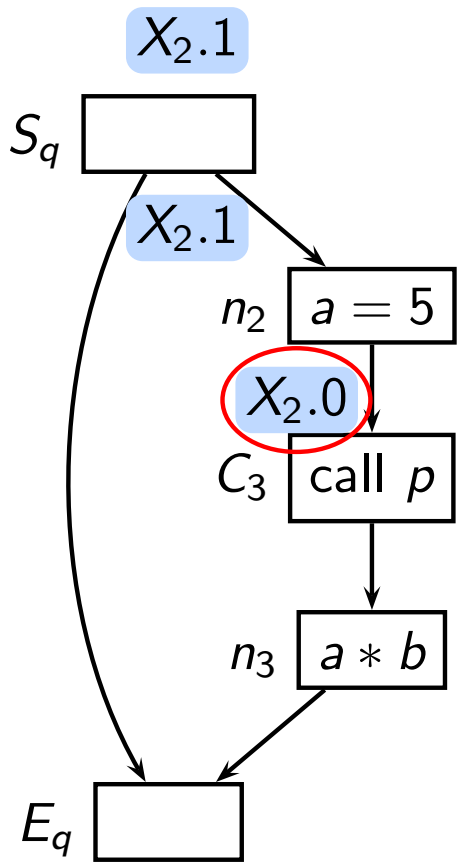


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0

Partially Available Expressions Analysis Using Value Contexts



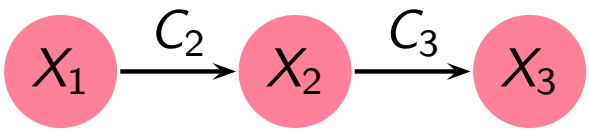


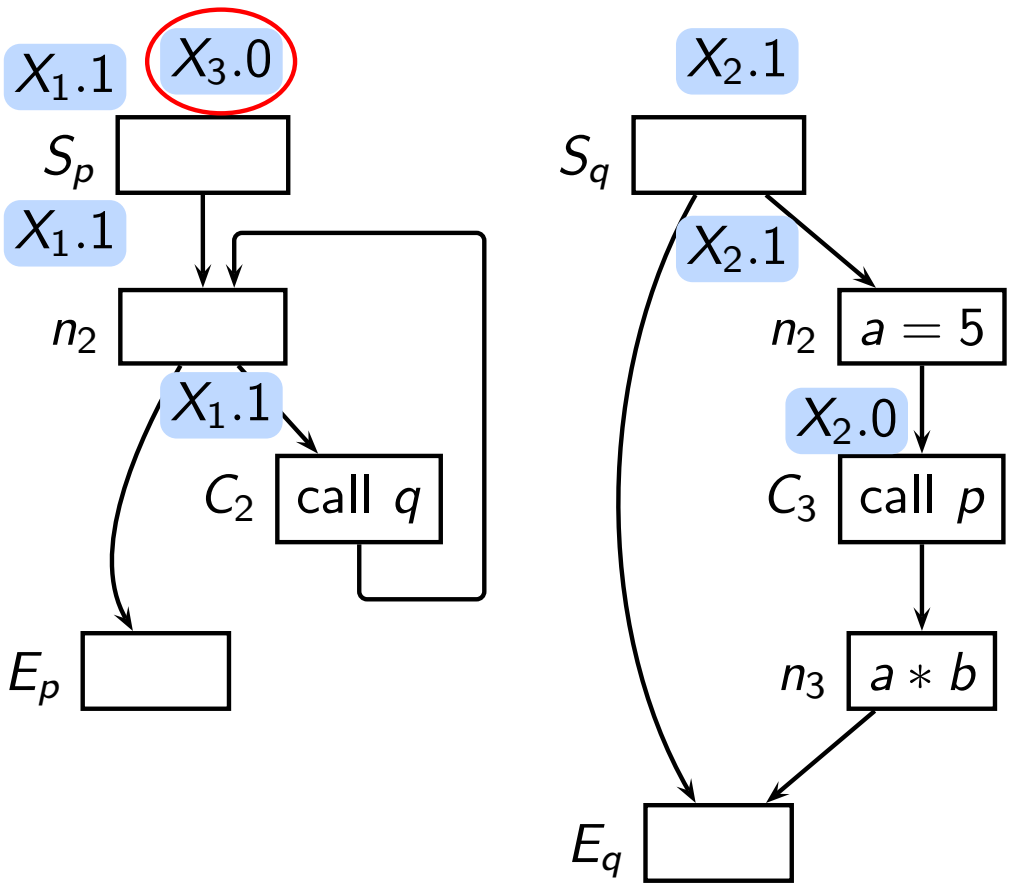


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0

The expression is killed in node n_2 and data flow value 0 reaches the call site C_3 that calls p

Partially Available Expressions Analysis Using Value Contexts

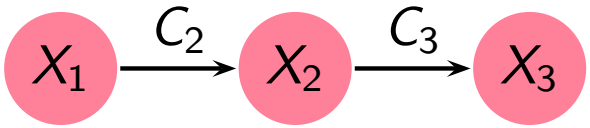


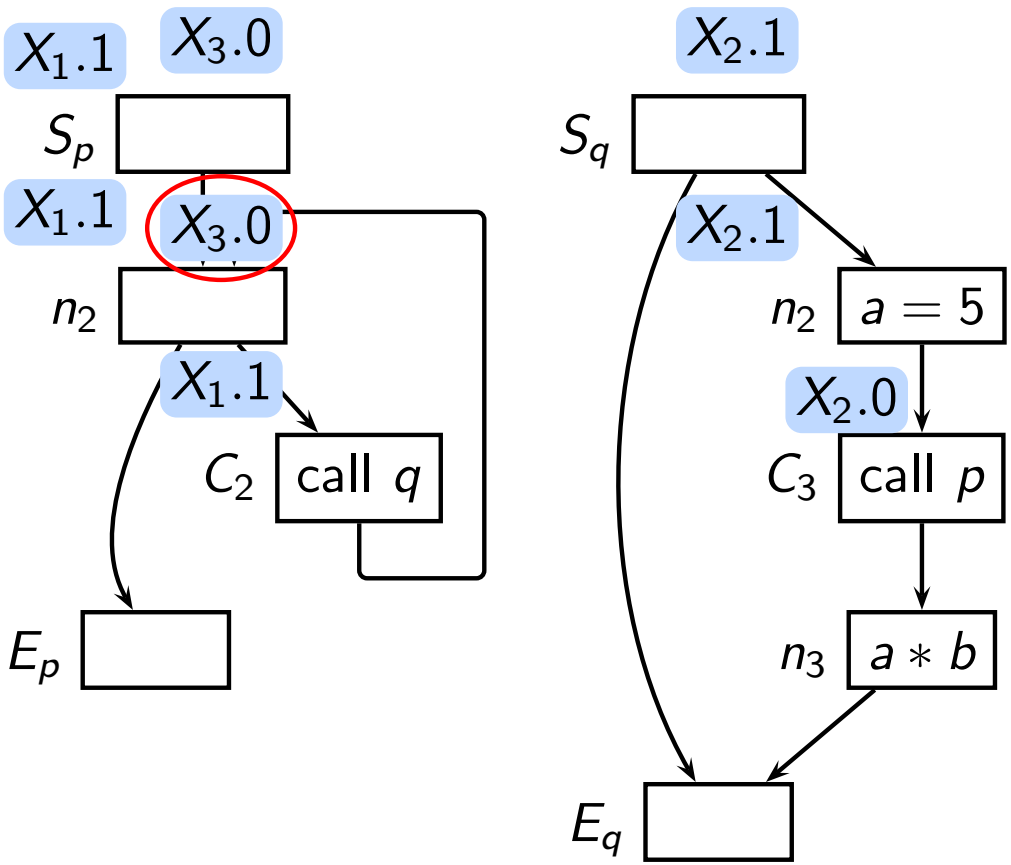


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0

We create context X_3 for the new value (0) reaching p and record transition from X_2 to X_3 on C_3

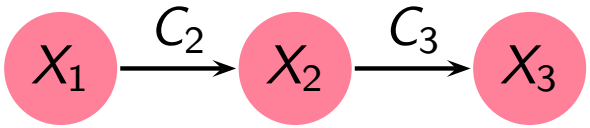
Partially Available Expressions Analysis Using Value Contexts

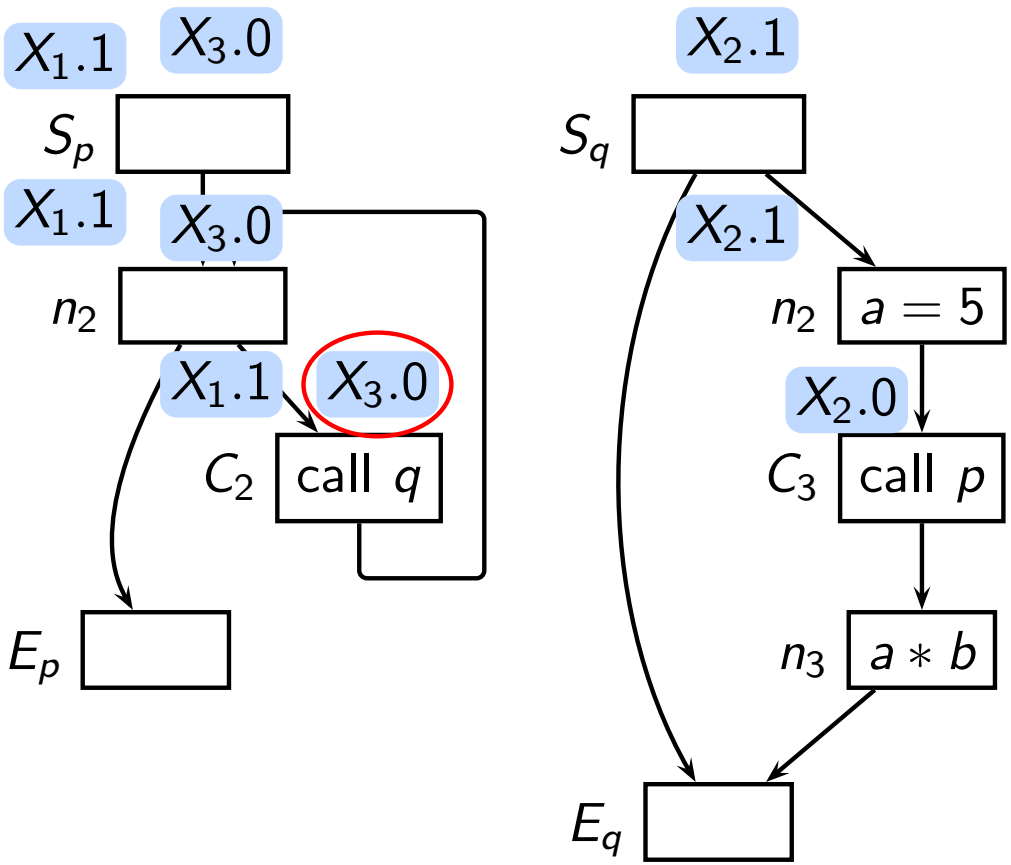




Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0

Partially Available Expressions Analysis Using Value Contexts

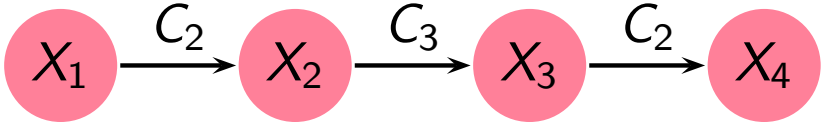


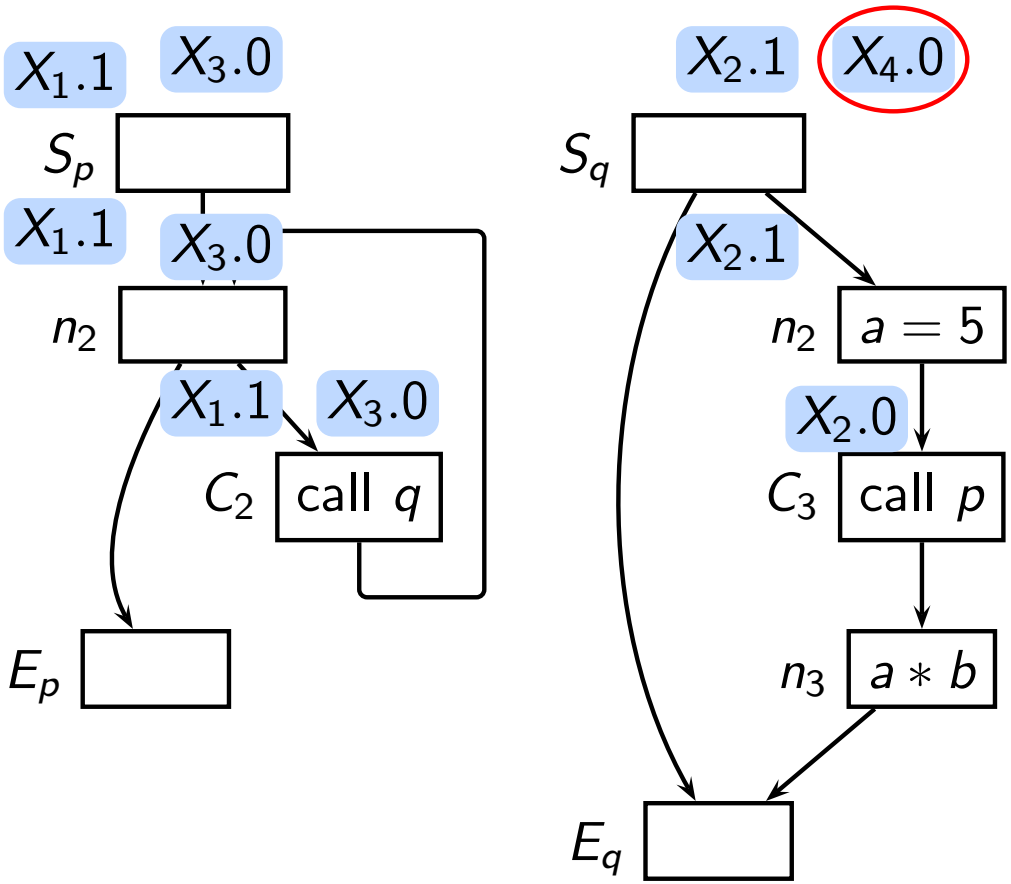


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0

And now the value 0 reaches q at call site C_2

Partially Available Expressions Analysis Using Value Contexts

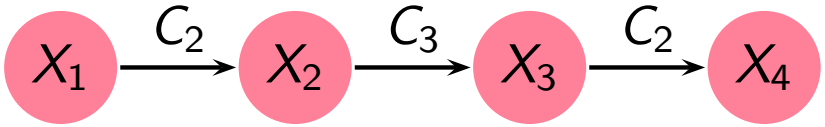


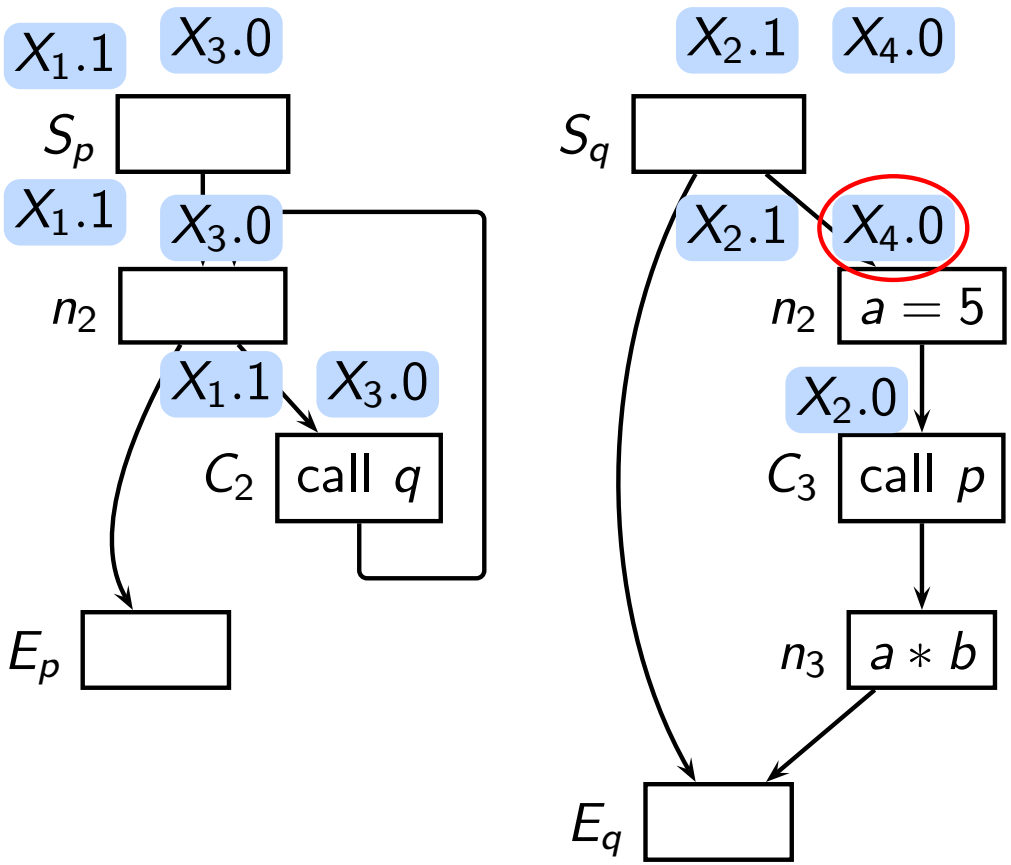


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	0

We create context X_4 for the new value (0) reaching p and record transition from X_3 to X_4 on C_2

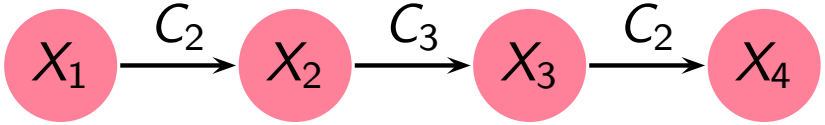
Partially Available Expressions Analysis Using Value Contexts



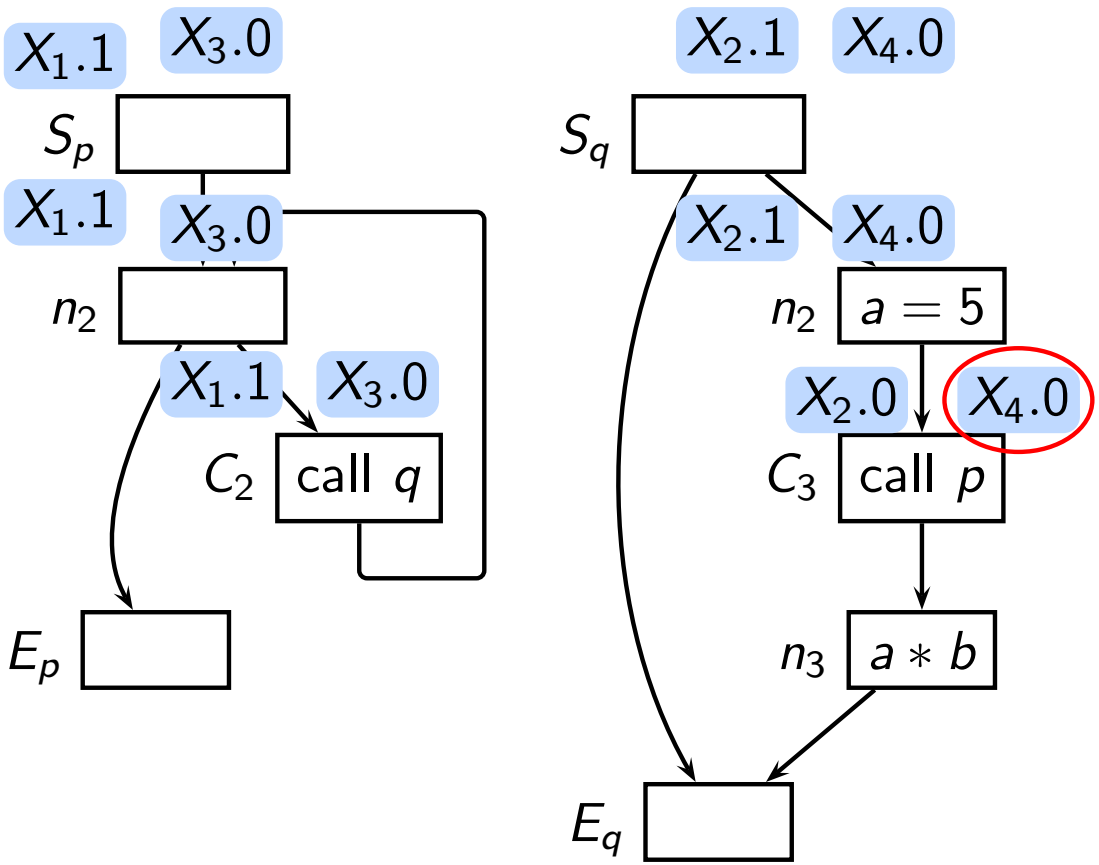


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	0

Partially Available Expressions Analysis Using Value Contexts

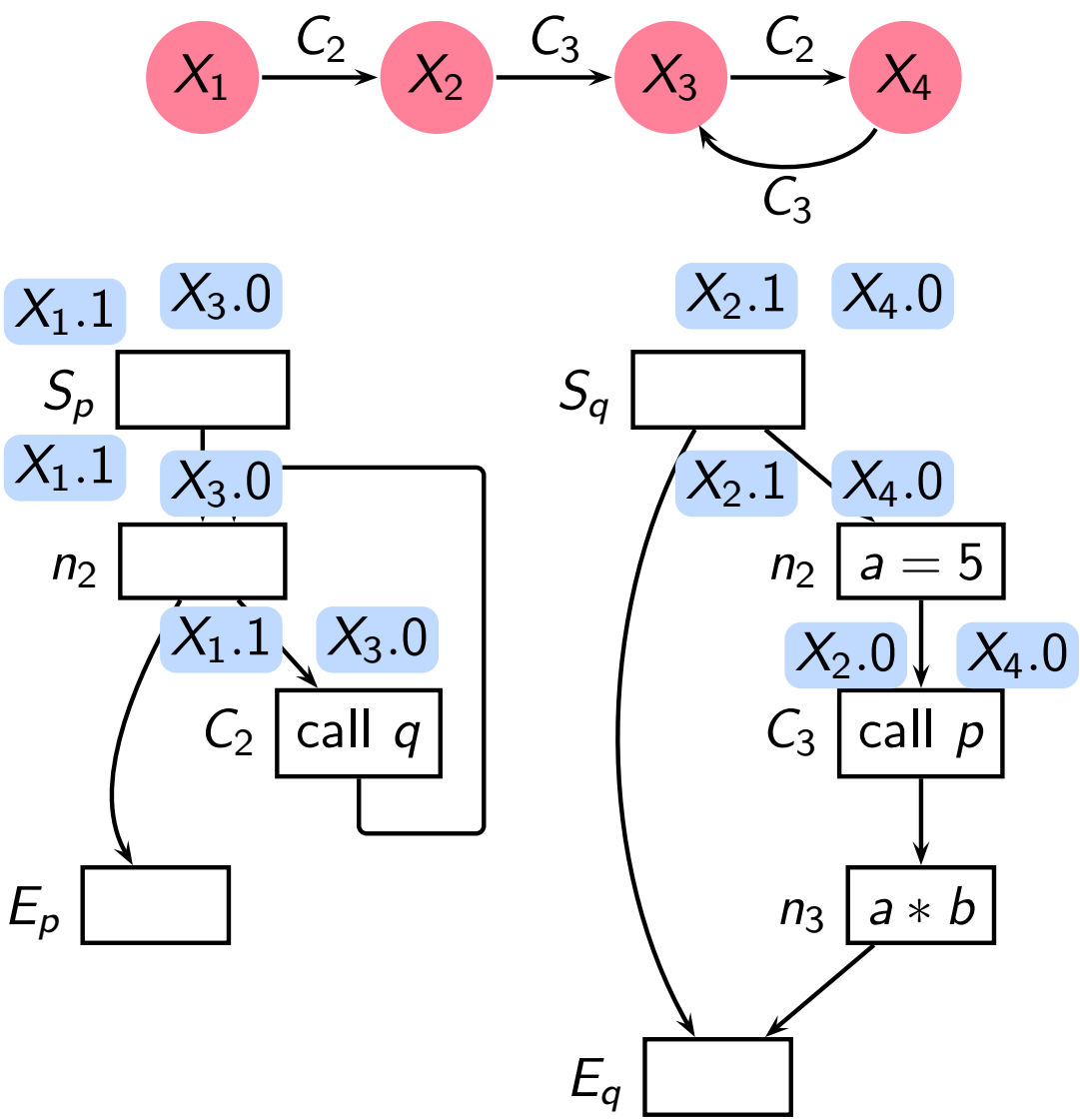


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	0



And now the value 0 reaches p at call site C_3

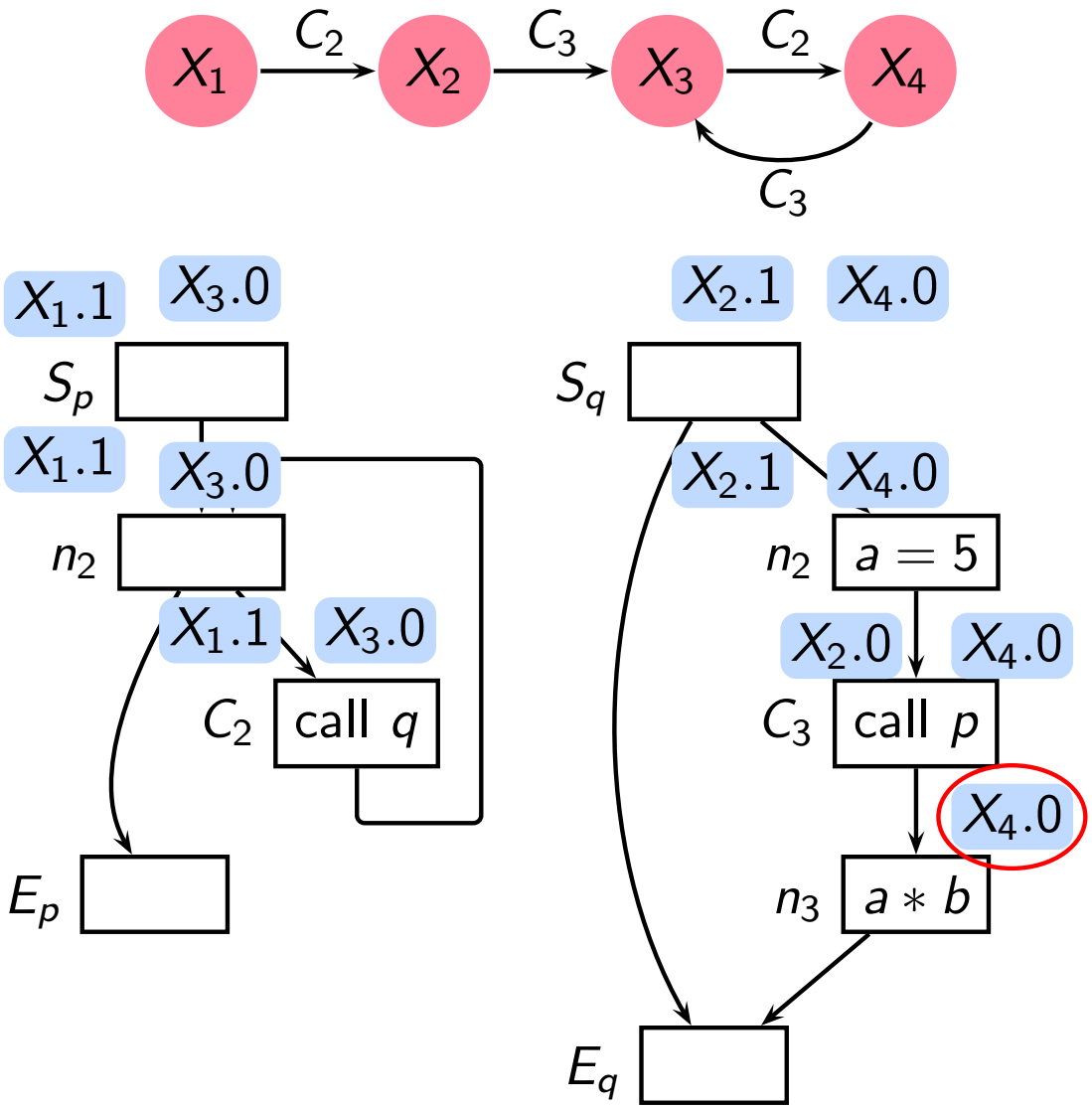
Partially Available Expressions Analysis Using Value Contexts



Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	0

We already have context X_3 with entry value 0 for p so no need to analyse p again

Partially Available Expressions Analysis Using Value Contexts

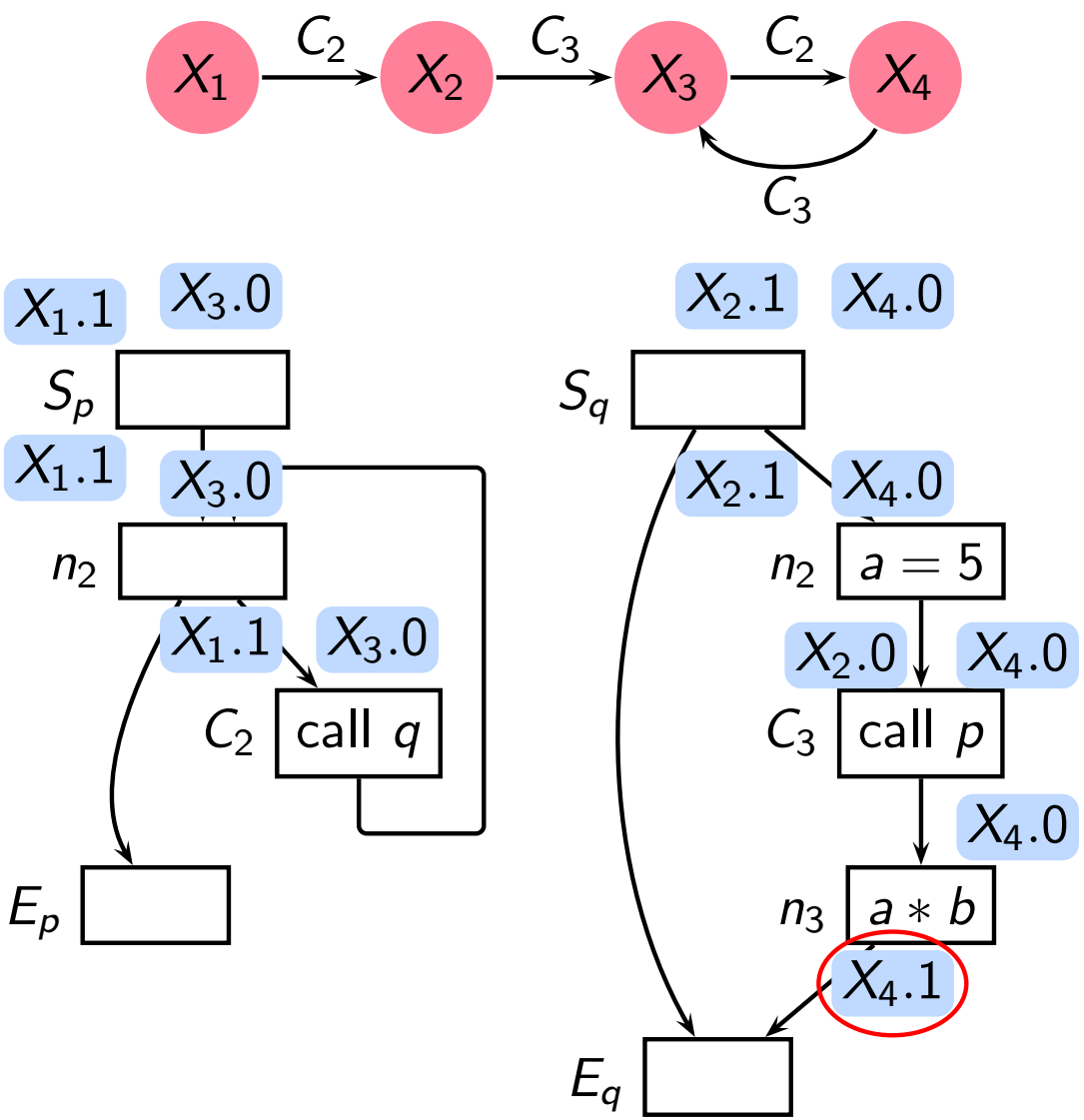


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	0

We use the *exitValue* for X_3 to compute Out_{C_3} for the context X_4 (because of the transition $X_4 \xrightarrow{C_3} X_3$)

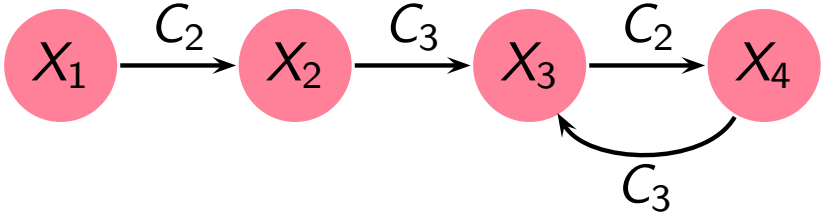
The analysis of p is not yet over for any context, and so we get the $\top$ value

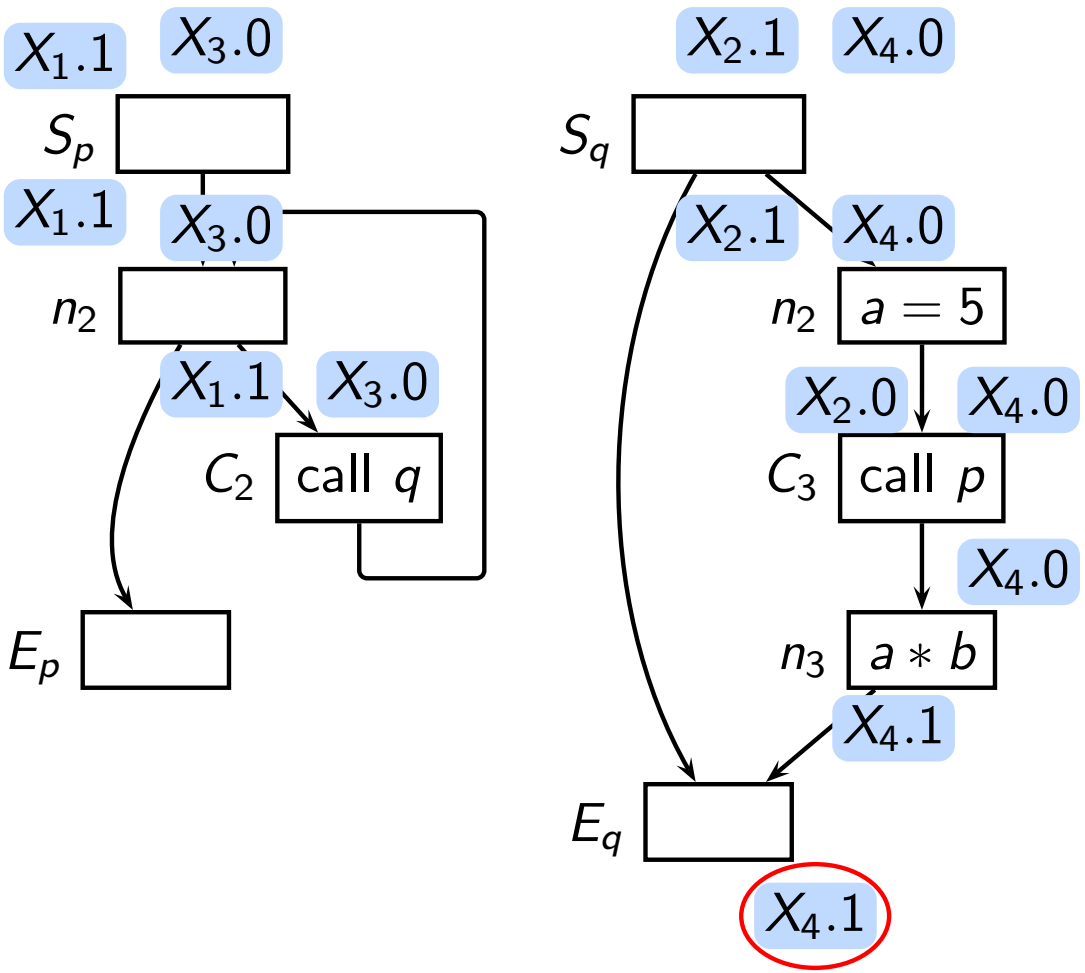
Partially Available Expressions Analysis Using Value Contexts



Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	0

Partially Available Expressions Analysis Using Value Contexts



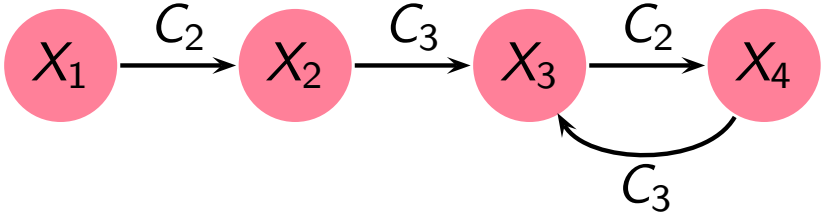


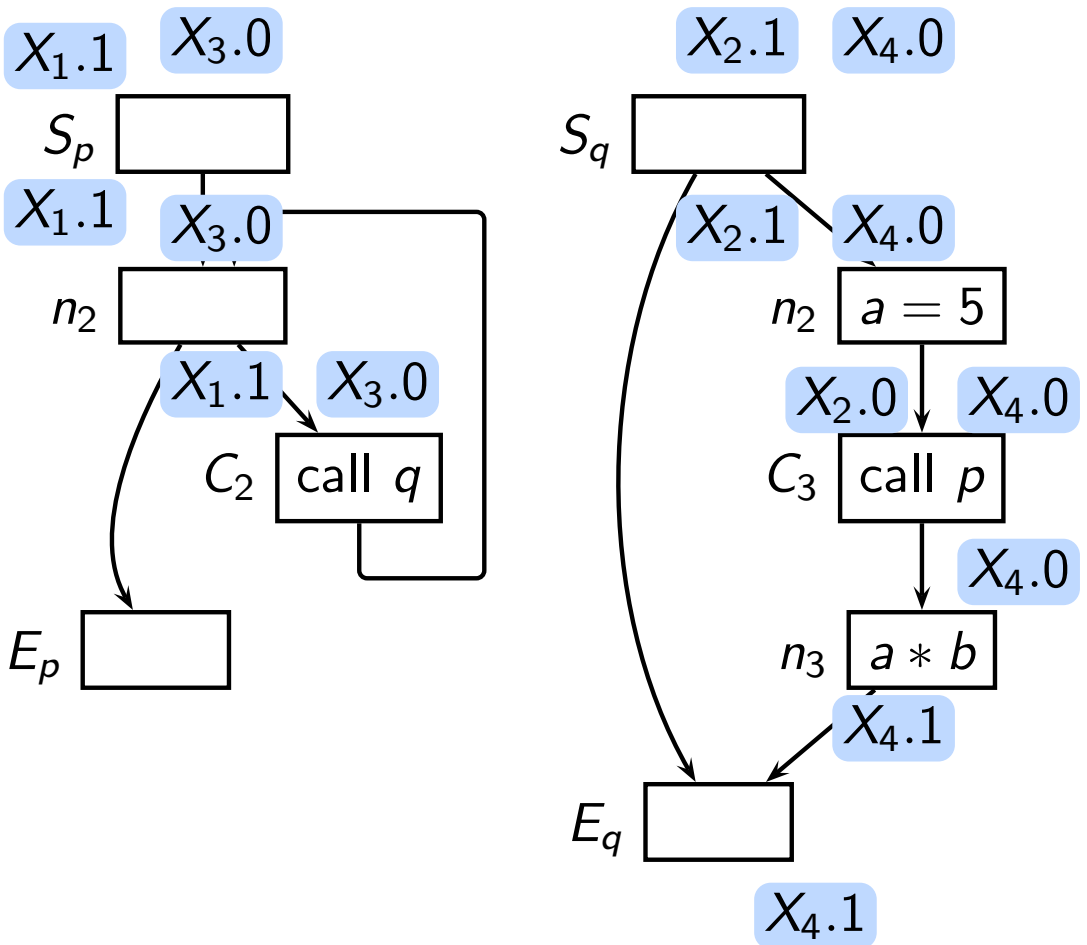
Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	0

The analysis of q for X_4 is now and the *exitValue* of X_4 becomes 1

This change in X_4 must be propagated to X_3 in the caller p (identified from the transition $X_3 \xrightarrow{C_2} X_4$)

Partially Available Expressions Analysis Using Value Contexts



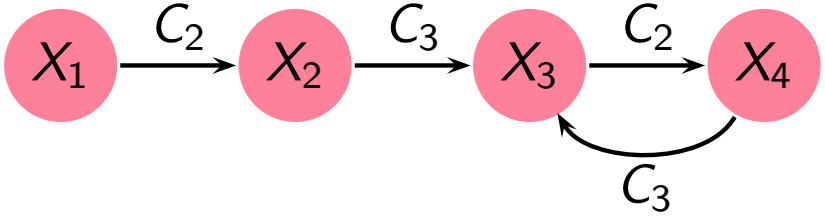


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	1

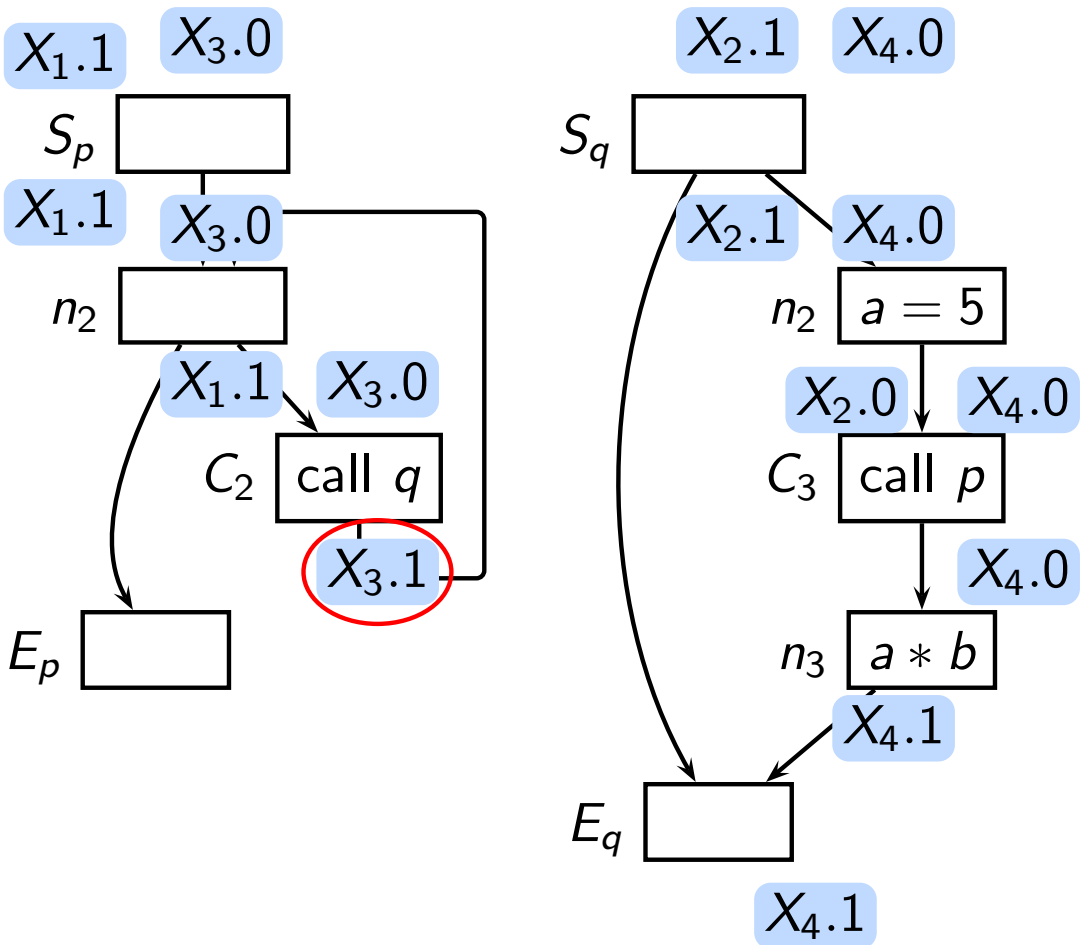
The analysis of q for X_4 is now and the *exitValue* of X_4 becomes 1

This change in X_4 must be propagated to X_3 in the caller p (identified from the transition $X_3 \xrightarrow{C_2} X_4$)

Partially Available Expressions Analysis Using Value Contexts

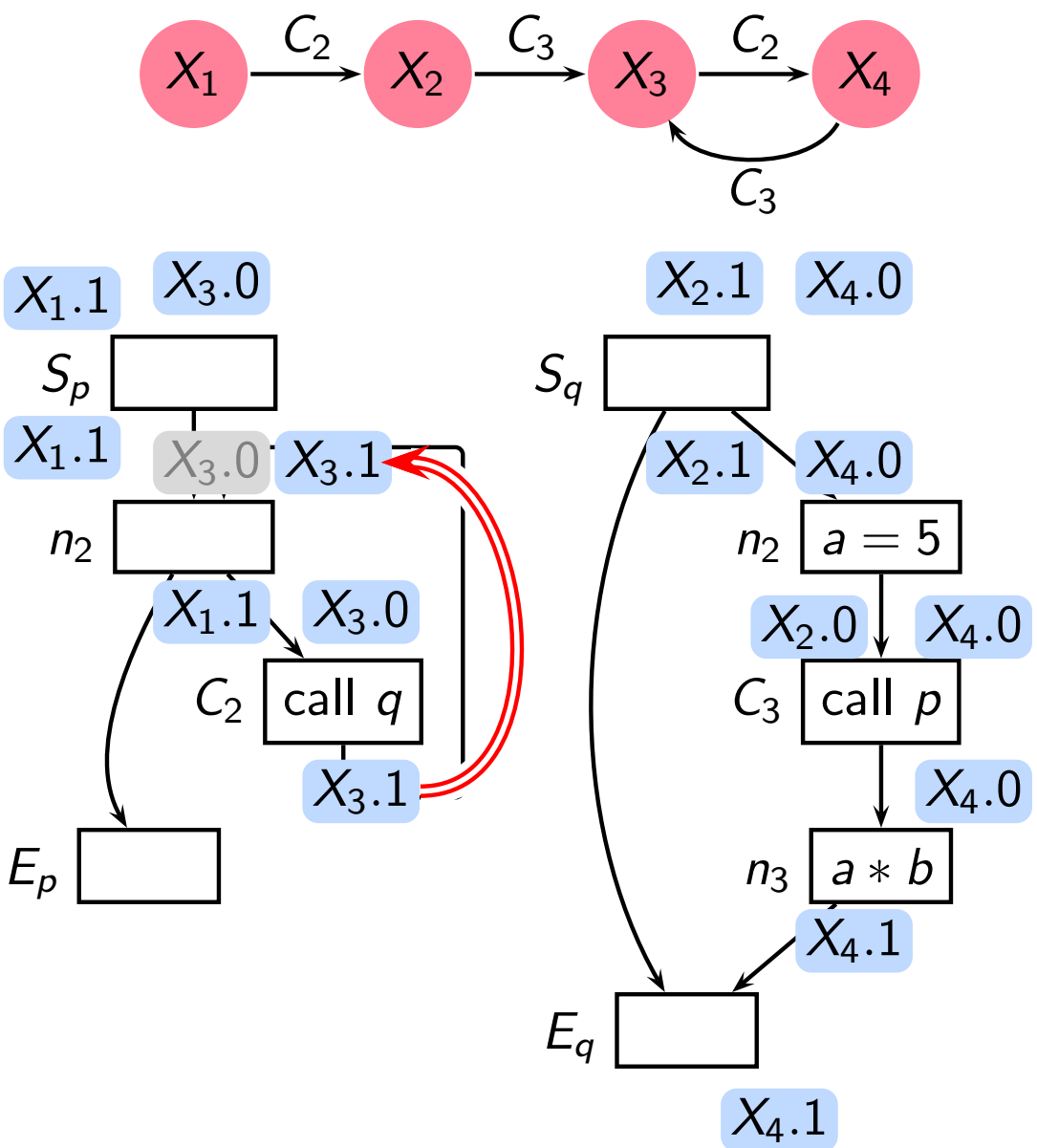


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	1



Out_{C_2} becomes 1 for X_3 which changes the value at In_{n_2} for X_3 to 1

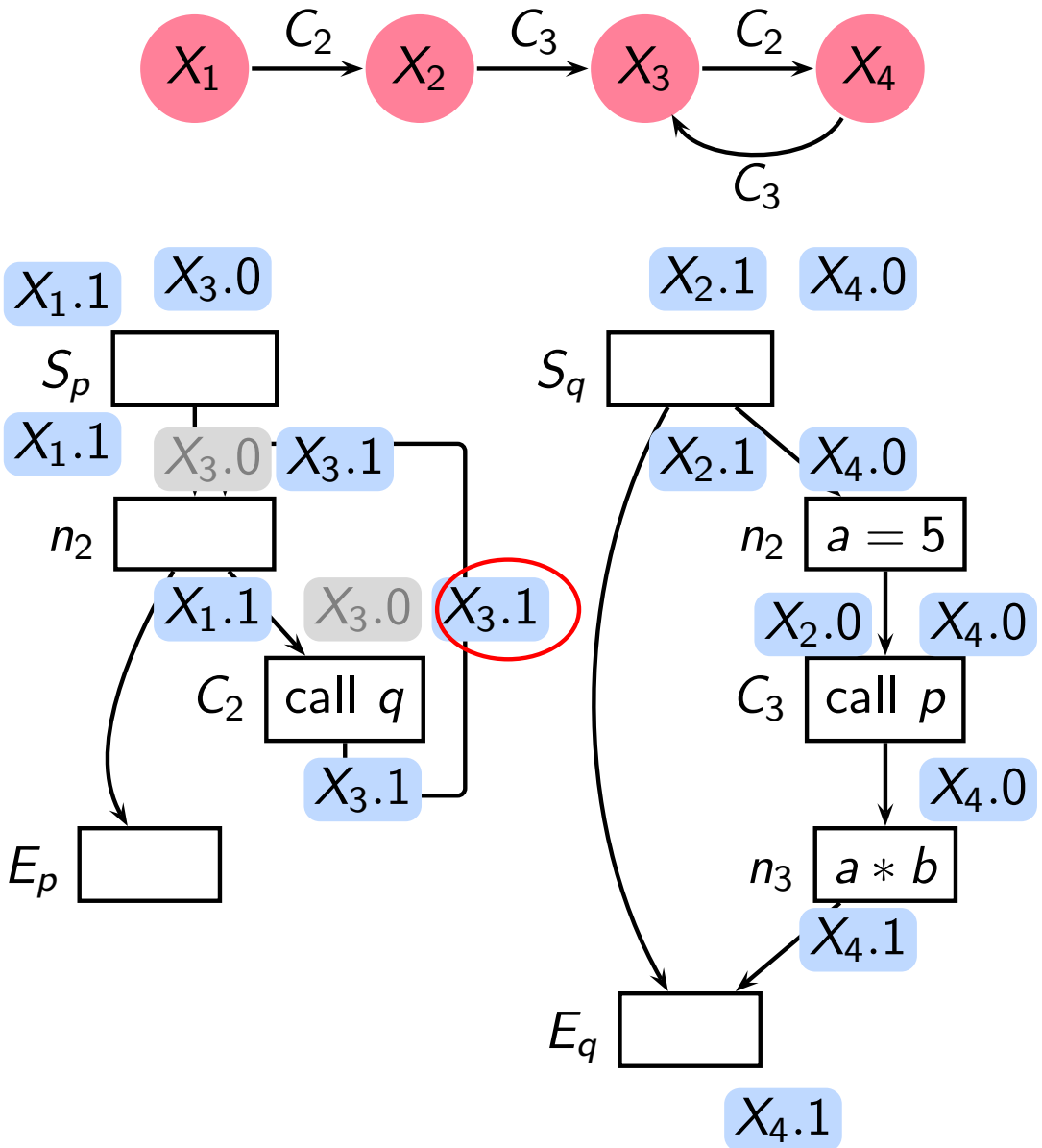
Partially Available Expressions Analysis Using Value Contexts



Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	1

Out_{C_2} becomes 1 for X_3 which changes the value at In_{n_2} for X_3 to 1

Partially Available Expressions Analysis Using Value Contexts

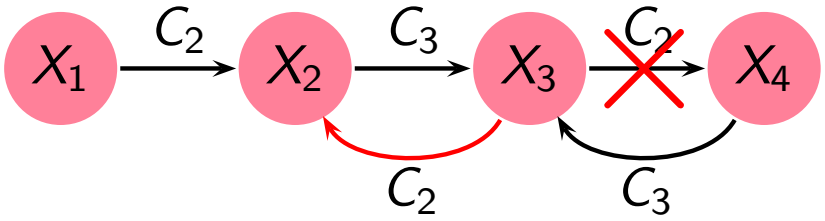


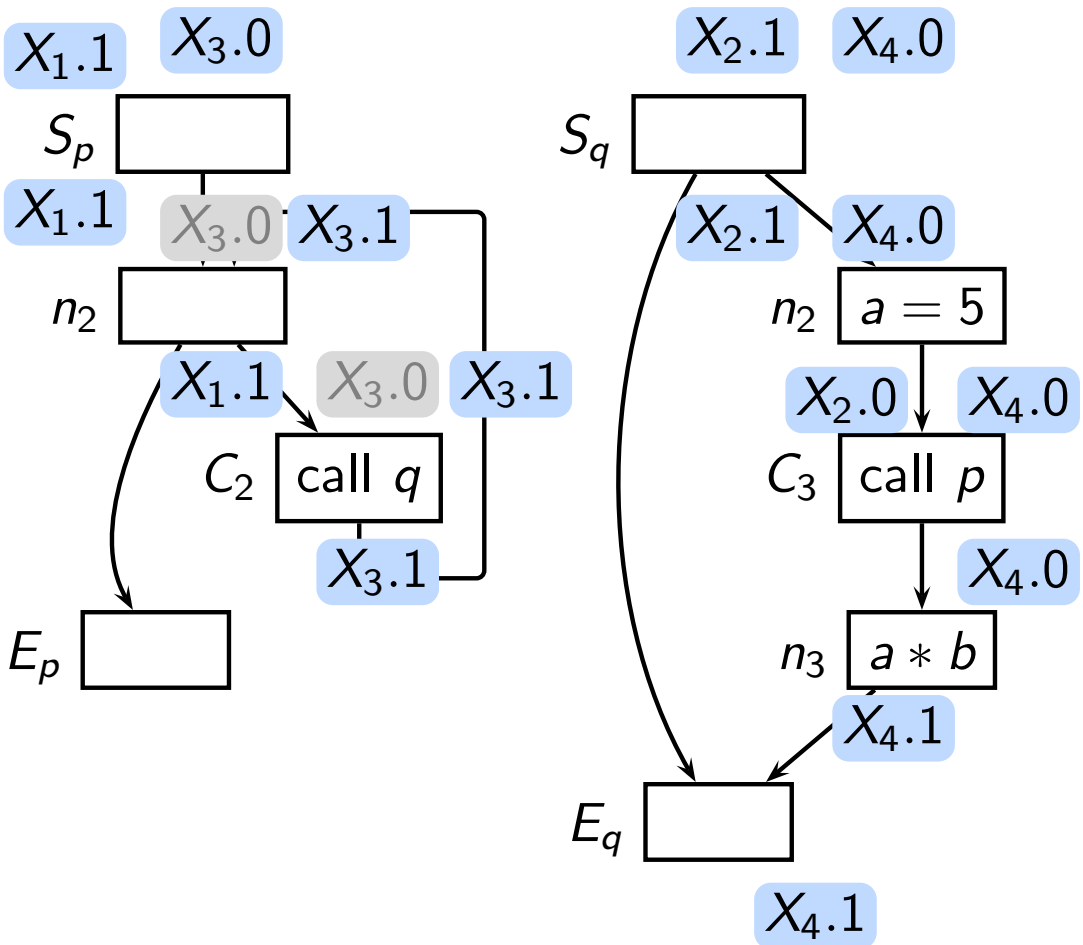
Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	1

In_{C_2} becomes 1 for X_3

Since we have a context for q with entry value 1 (X_2), we remove the transition $X_3 \xrightarrow{C_2} X_4$ and add the transition $X_3 \xrightarrow{C_2} X_2$

Partially Available Expressions Analysis Using Value Contexts



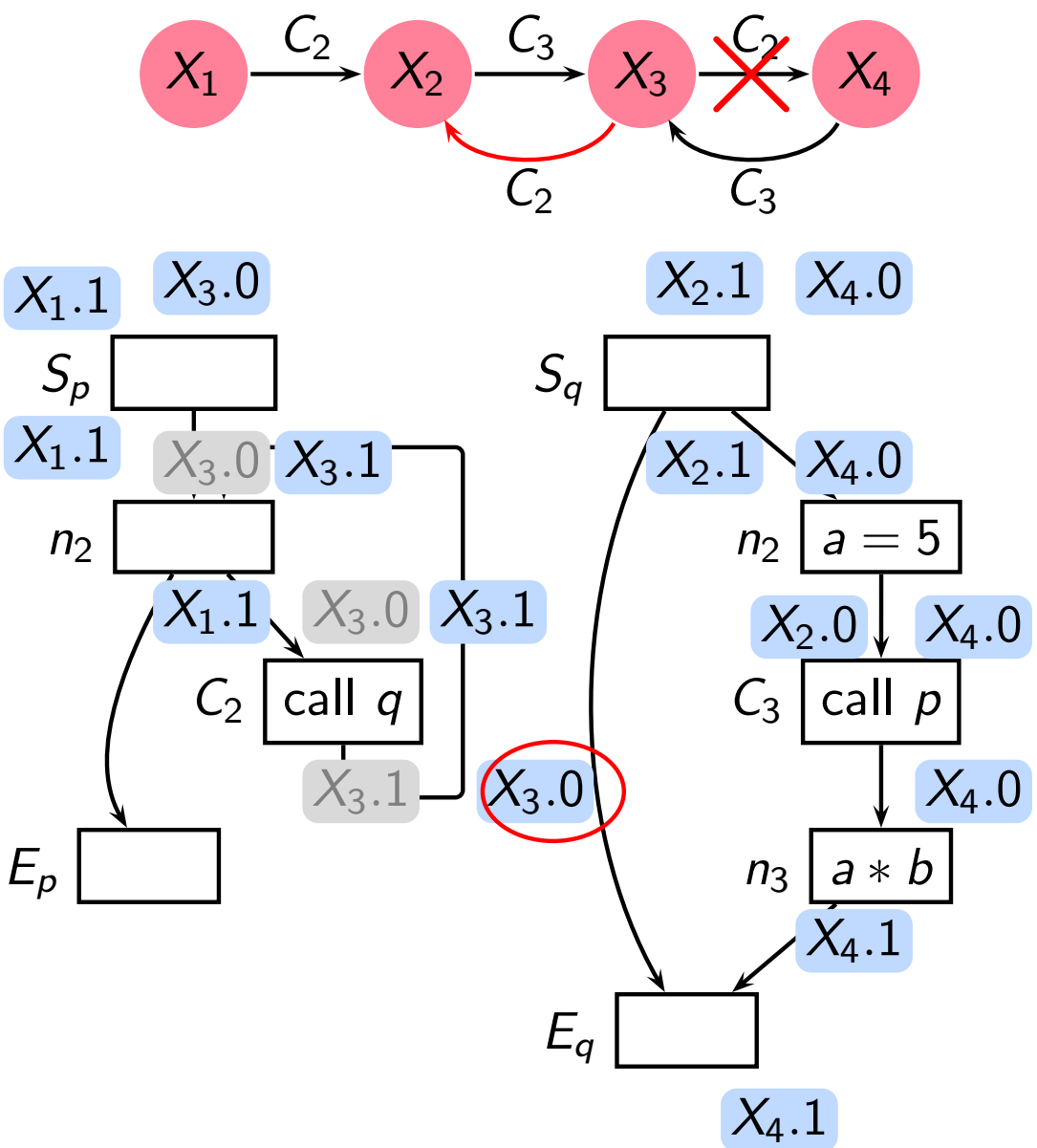


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	1

In_{C_2} becomes 1 for X_3

Since we have a context for q with entry value 1 (X_2), we remove the transition $X_3 \xrightarrow{C_2} X_4$ and add the transition $X_3 \xrightarrow{C_2} X_2$

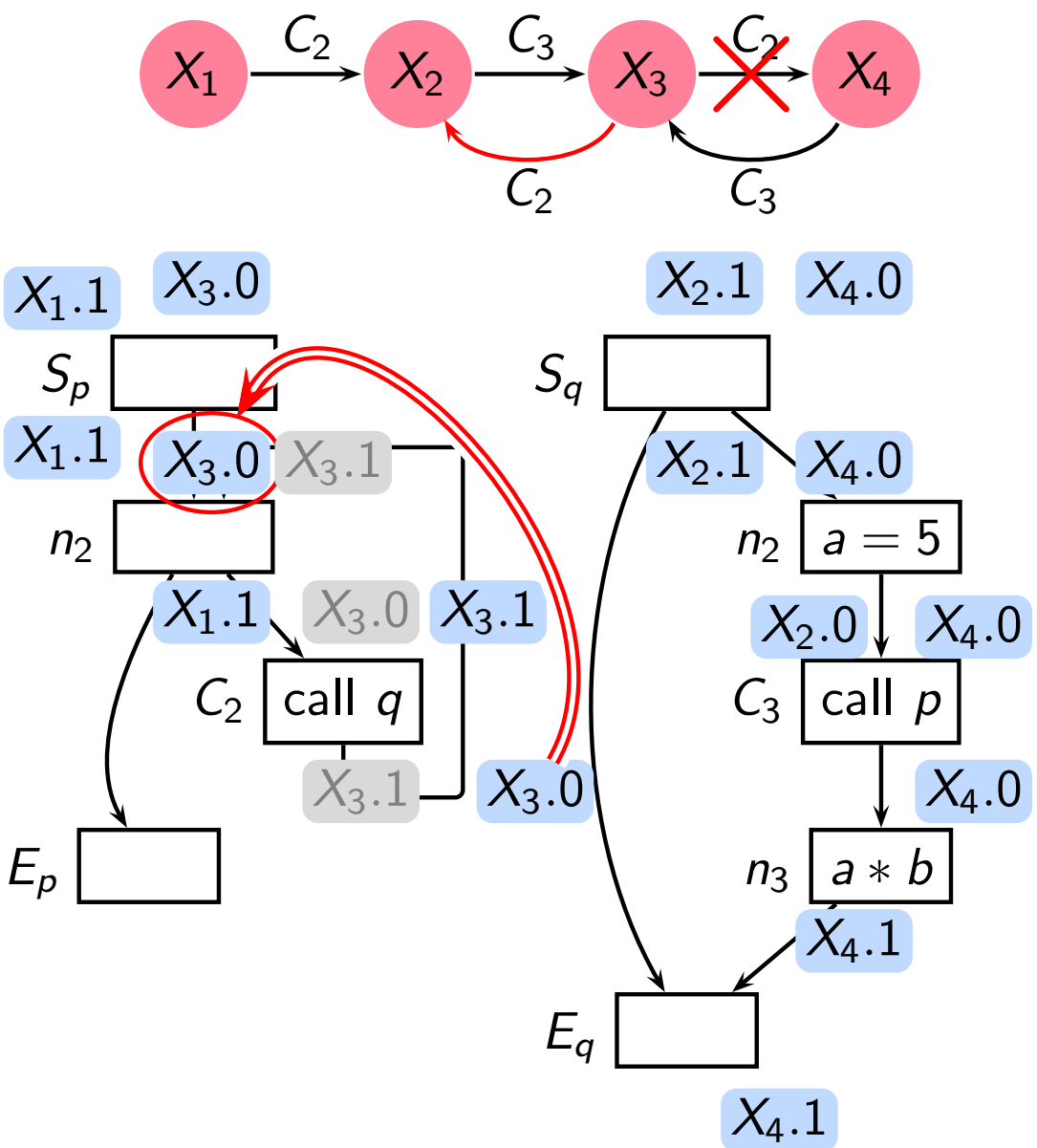
Partially Available Expressions Analysis Using Value Contexts



Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	1

We use the *exitValue* of X_2 to compute the value of X_3 in Out_{C_2}

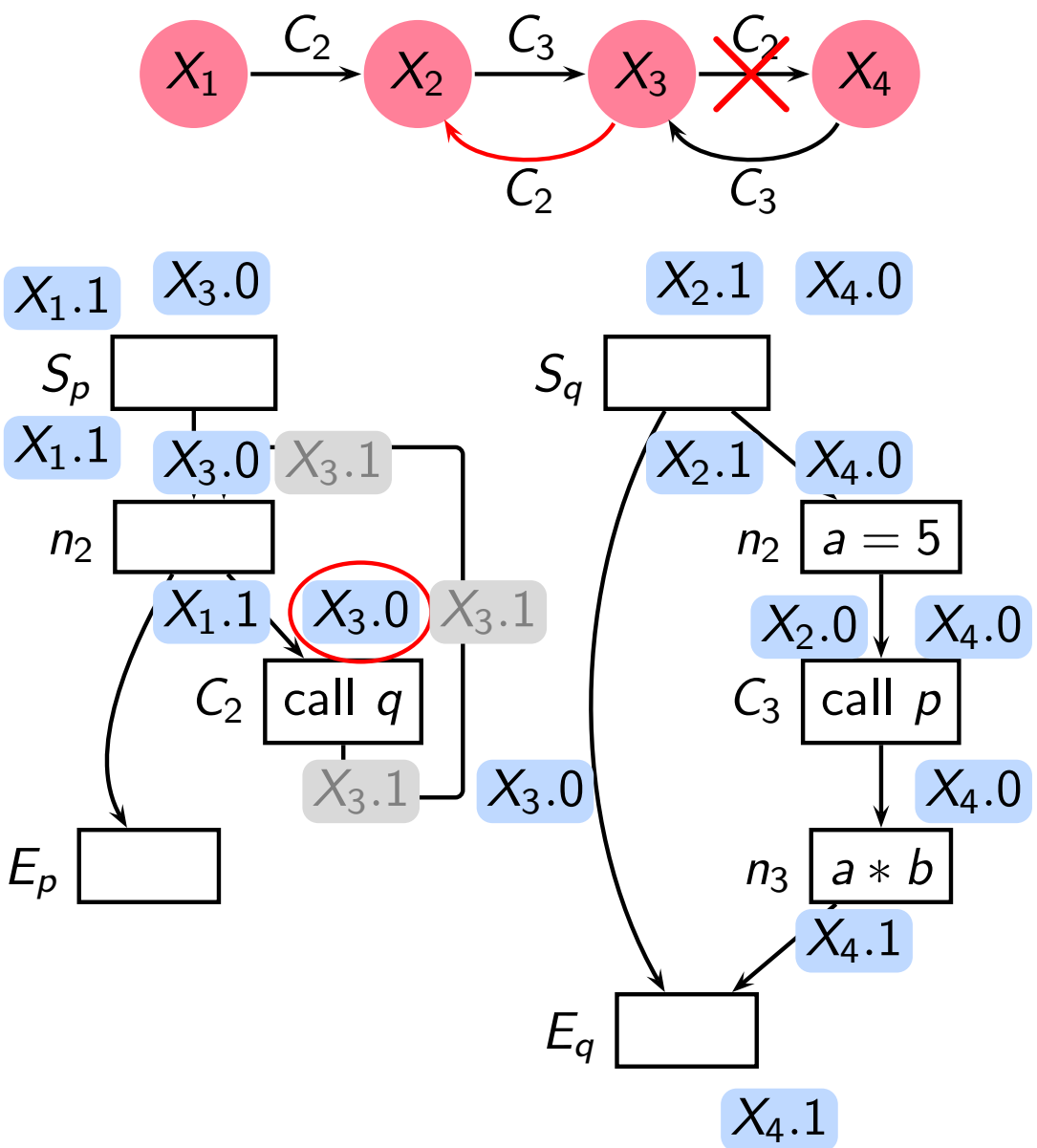
Partially Available Expressions Analysis Using Value Contexts



Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	1

The value of X_3 in In_{n_2} once again becomes 0

Partially Available Expressions Analysis Using Value Contexts

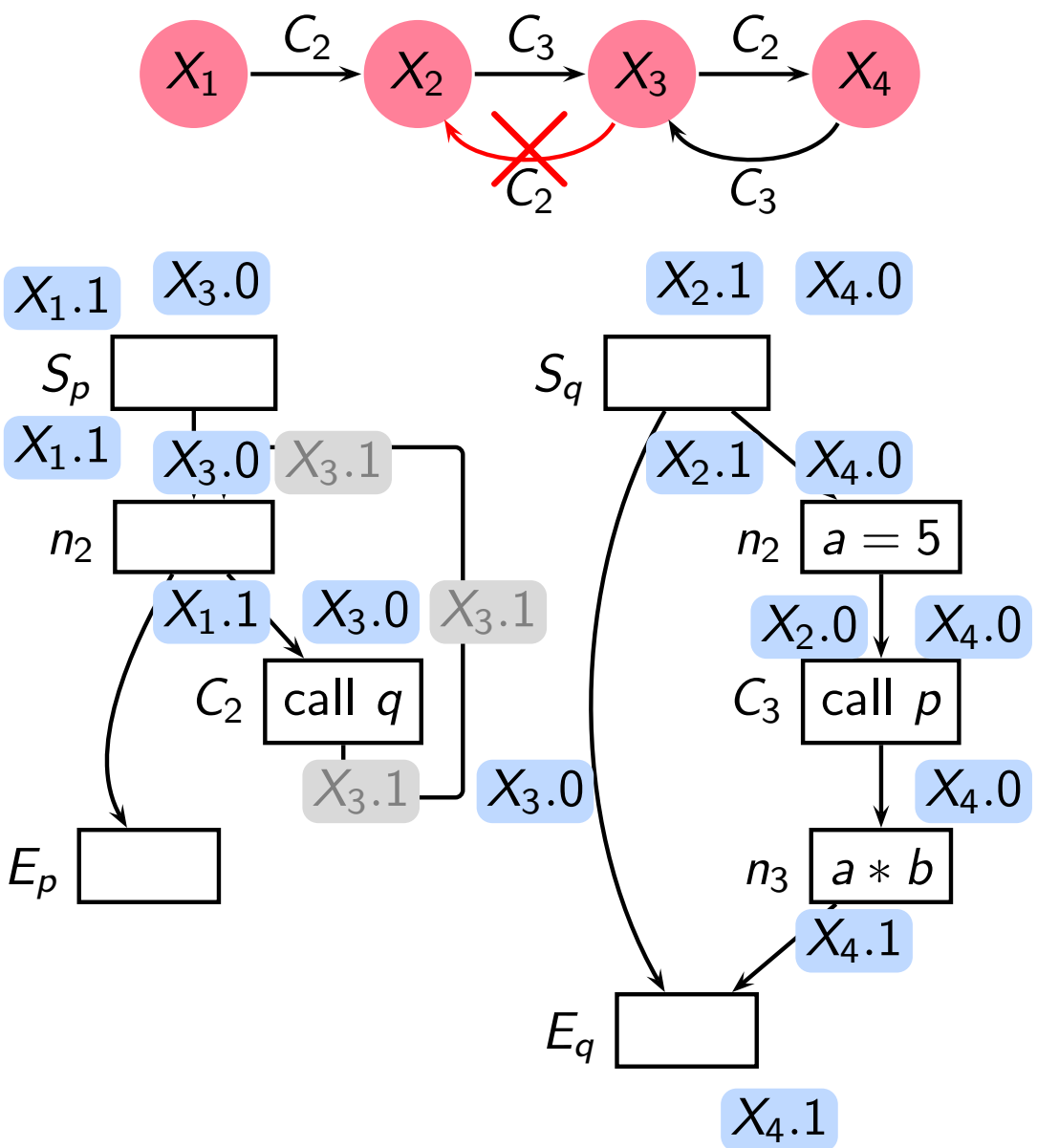


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	1

The value of X_3 in In_{C_2} once again becomes 0

The transition from X_3 needs to be restored to $X_3 \xrightarrow{C_2} X_4$ removing the transition $X_3 \xrightarrow{C_2} X_2$

Partially Available Expressions Analysis Using Value Contexts

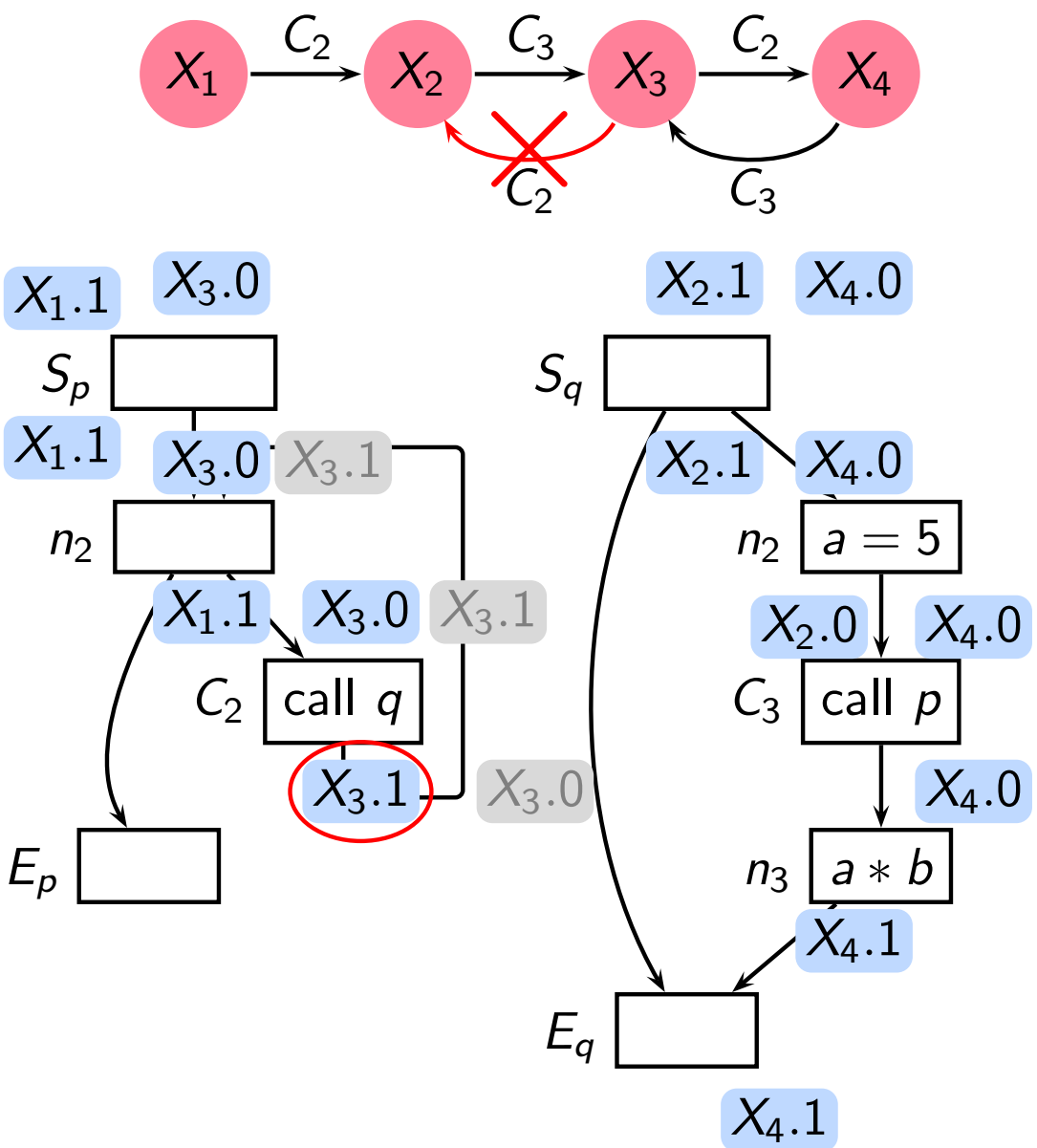


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	1

The value of X_3 in In_{C_2} once again becomes 0

The transition from X_3 needs to be restored to $X_3 \xrightarrow{C_2} X_4$ removing the transition $X_3 \xrightarrow{C_2} X_2$

Partially Available Expressions Analysis Using Value Contexts

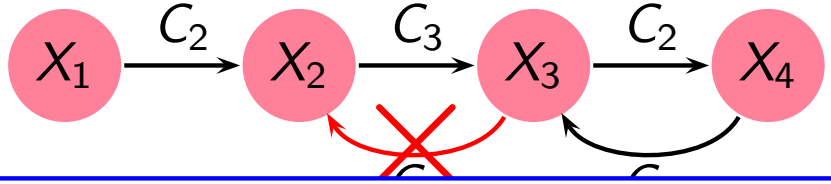


Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0
$X_2 = \langle q,1 \rangle$	0
$X_3 = \langle p,0 \rangle$	0
$X_4 = \langle q,0 \rangle$	1

We use the *exitValue* of X_4 to compute Out_{C_2} for X_3 which once again becomes 0

Thus we are back to the same situation

Partially Available Expressions Analysis Using Value Contexts



Context	*exitValue*
$X_1 = \langle p,1 \rangle$	0

- The process would not terminate so long as the processing of the nodes in the loop continues
- If the work list organization allows processing of E_p, then the *exitValue* of X_3 will also change to 1 which will lead to termination
- Our underlying flow functions are monotonic and a fixed point exists; non-termination is caused by the algorithm because its progress depends on the order of the nodes in the work list
- We avoid this problem by taking a meet at the exit of call nodes when the exit values of existing contexts are used at the call sites in the callers

$X_4.1$

Defining Value Context Method Using Data Flow Equations

- The overall data flow values Γ are sets of $X.v$ where X is a context and $v \in L$ is the underlying data flow value.
- We merge underlying data flow values only if the contexts are same

$$\Gamma_1 \uplus \Gamma_2 = \{X.w \mid \begin{array}{lcll} X.u \in \Gamma_1 \wedge X.v \in \Gamma_2 & \Rightarrow & w = u \sqcap v, \\ X.u \in \Gamma_1 \wedge X.v \notin \Gamma_2 & \Rightarrow & w = u, \\ X.u \notin \Gamma_1 \wedge X.v \in \Gamma_2 & \Rightarrow & w = v \end{array} \}$$

  Effectively, if a context does not exist in Γ, its value is $\top$ in Γ
- Data flow variables for node n in procedure p are $In(p, n)$ and $Out(p, n)$
- The flow function for node n in procedure p is $f(p, n)$

Defining Value Context Method Using Data Flow Equations

We assume the following auxiliary functions

- Function *context* maintains the context information

  $context(p, v)$ returns the context of procedure p for entry value v

  If no such context exists, the function creates a new context and returns it
- Function $exitValue(X)$ returns the exit valus of context X

  If context X does not exist, the function returns $\top \in L$
- Function *gpred* extends the predcessor relation pred (which is local to a procedure) to a global level across procedures

$$gpred(p, n) = \begin{cases} \{(q, m) \mid \text{ call site } m \text{ in } q \text{ calls } p\} & n \text{ is } S_p \\ \{(p, m) \mid m \in pred(n)\} & \text{otherwise} \end{cases}$$

Defining Value Context Method Using Data Flow Equations

We define data flow equations for a forward data flow analysis

$$In(p,n) = \begin{cases} \{X.v \mid X = context(p,v), Y.v \in In(q,m), \\ \qquad (q,m) \in gpred(p,n)\} & n \text{ is } S_p \\ \biguplus_{(p,m) \in gpred(p,n)} Out(p,m) & \text{otherwise} \end{cases}$$

$$Out(p,n) = \begin{cases} Out(p,n) \uplus \{X.v \mid X.v' \in In(p,m), & n \text{ calls } q \\ \qquad Y = context(q,v'), \\ \qquad v = exitValue(Y)\ \} \\ \{X.v \mid X.v' \in In(p,m), v = f(p,n)(v')\} & \text{otherwise} \end{cases}$$

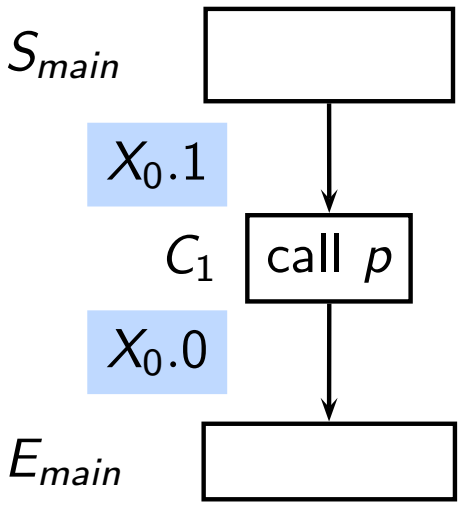
Value Contexts and Interprocedurally Valid Paths

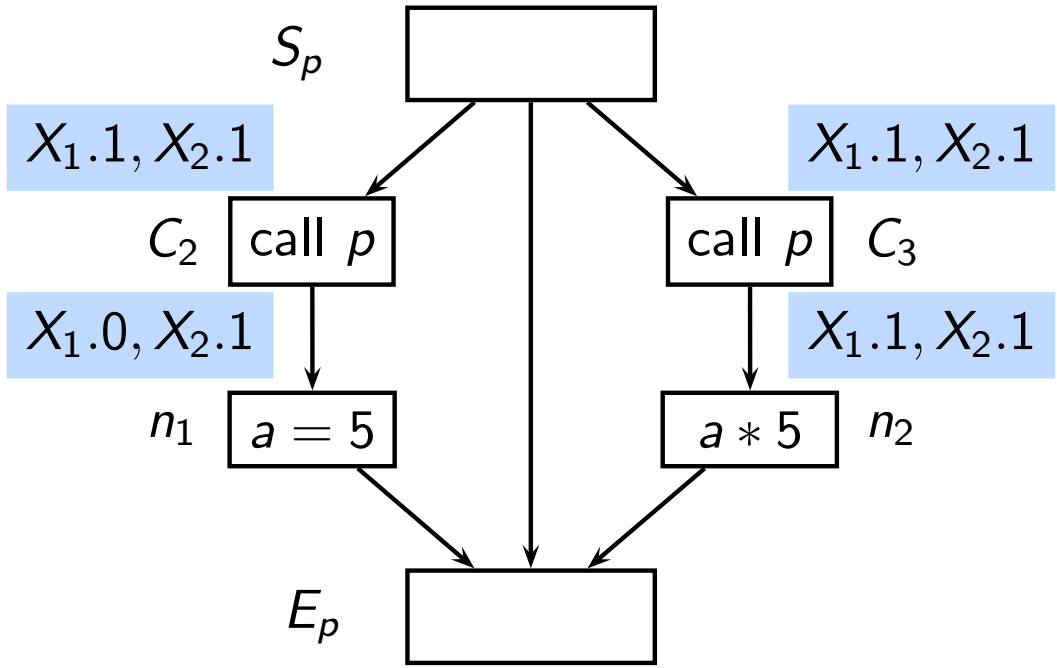
The role of value contexts in context sensitivity

- Value contexts preserve interprocedurally valid paths
- Value contexts consider only interprocedurally valid paths

We explain this with the help of an example by illustrating paths using a staircase diagram

Value Contexts and Interprocedurally Valid Paths: Example

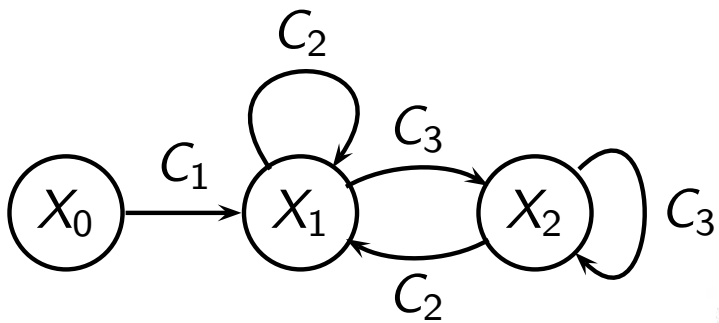




Context Transition Table

Context	*exitValue*
X_0 : $\langle \text{main}, 0 \rangle$	1
X_1 : $\langle p, 0 \rangle$	1
X_2 : $\langle p, 1 \rangle$	1

Context Transition Graph



Value Contexts and Interprocedurally Valid Paths: Example

We explain the data flow value at the entry of C_2 by dividing the paths into the following two categories:

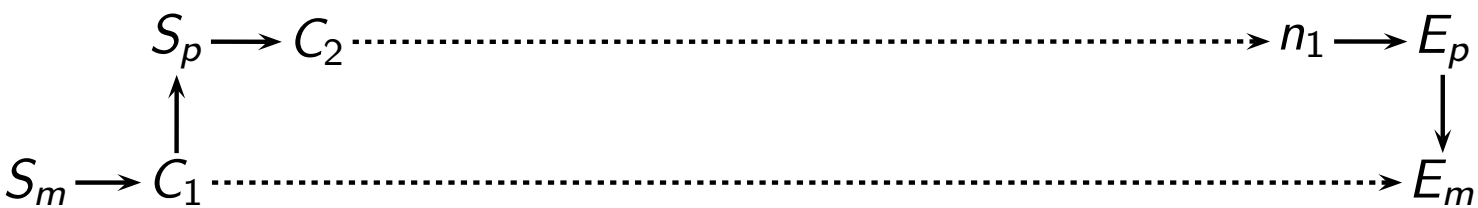
A. Paths in which the innermost recursion is along the call at C_2.

B. Paths in which the innermost recursion is along the call at C_3.

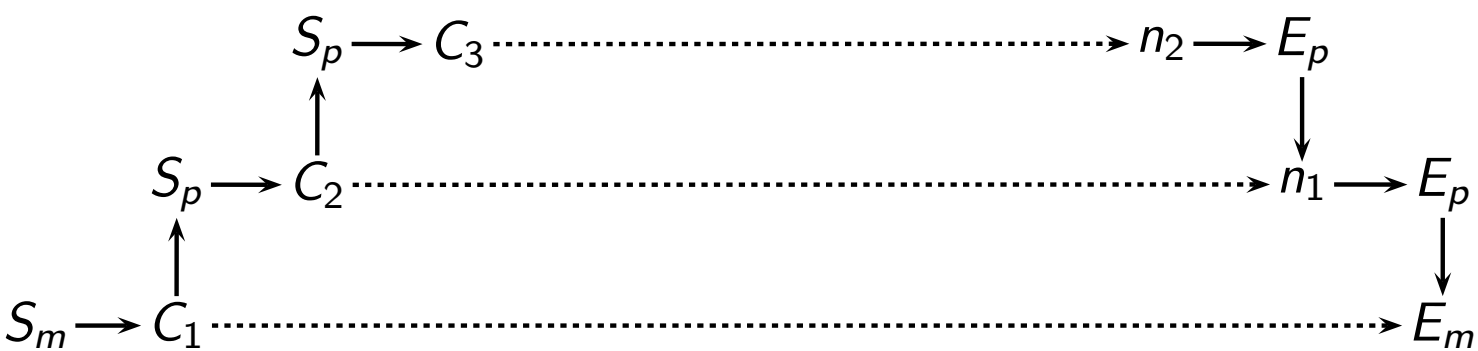
We draw the staircase diagrams of the example paths in the two categories

Innermost Recursion Along the Call at C_2

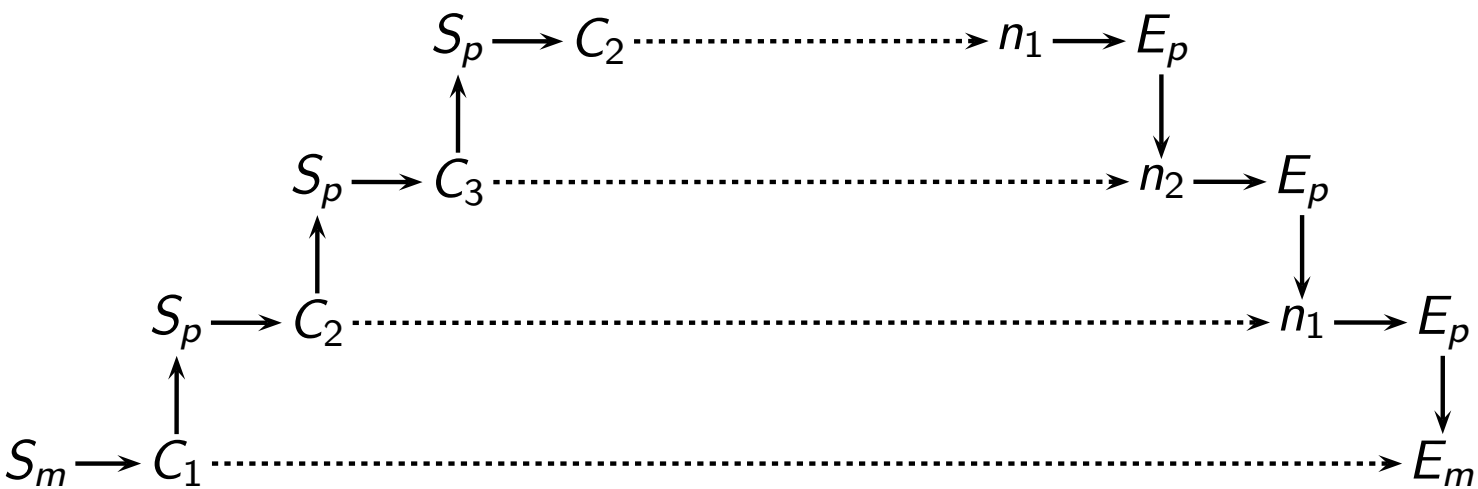
Innermost Recursion Along the Call at C_2



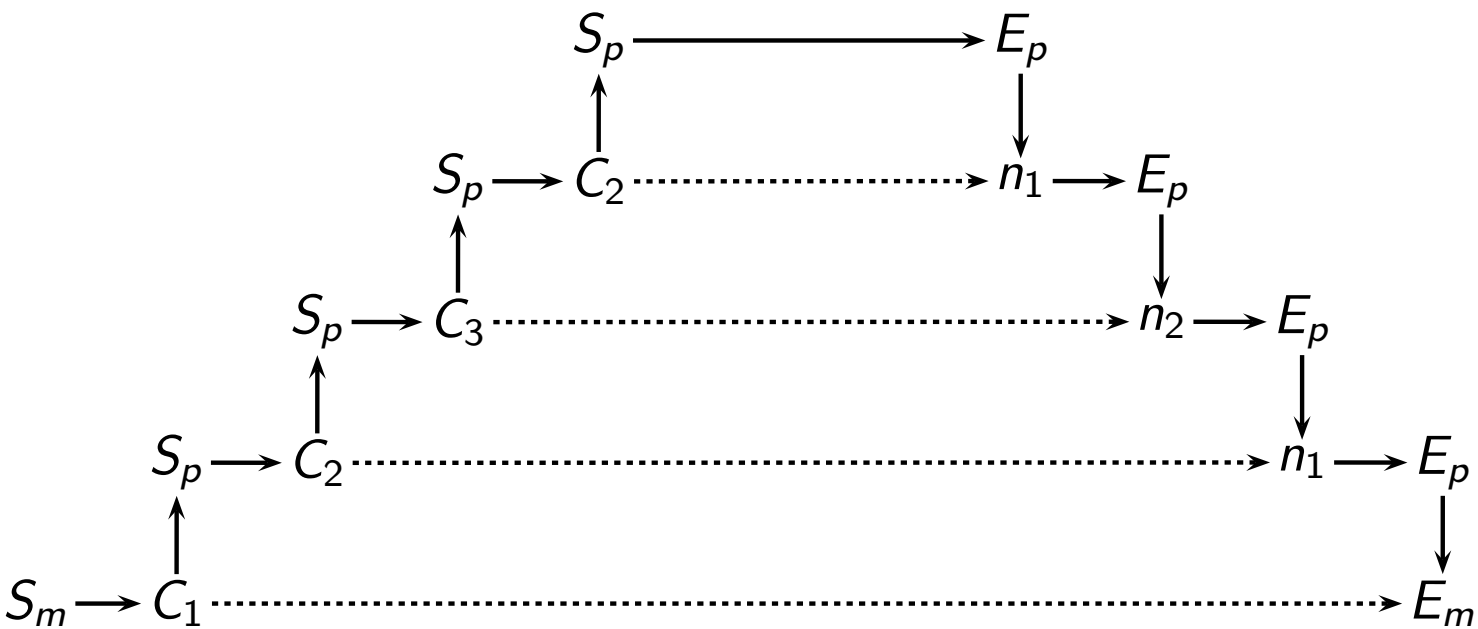
Innermost Recursion Along the Call at C_2



Innermost Recursion Along the Call at C_2

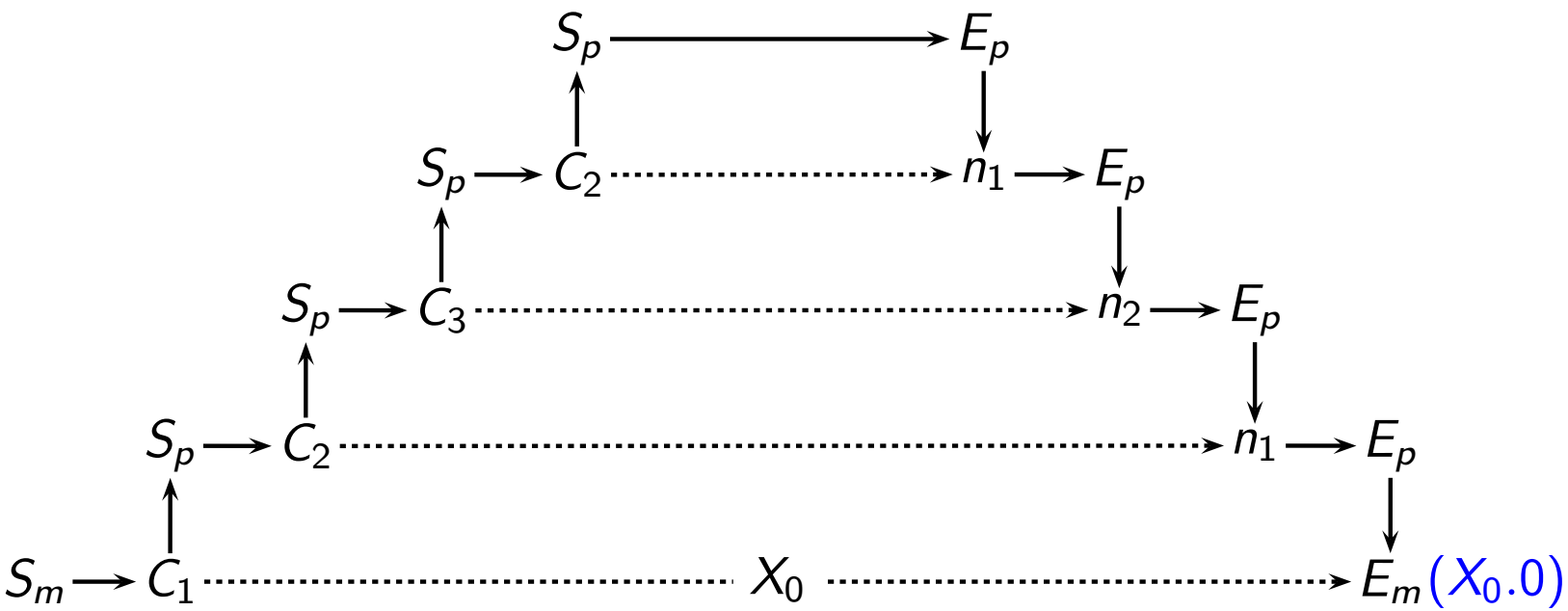


Innermost Recursion Along the Call at C_2



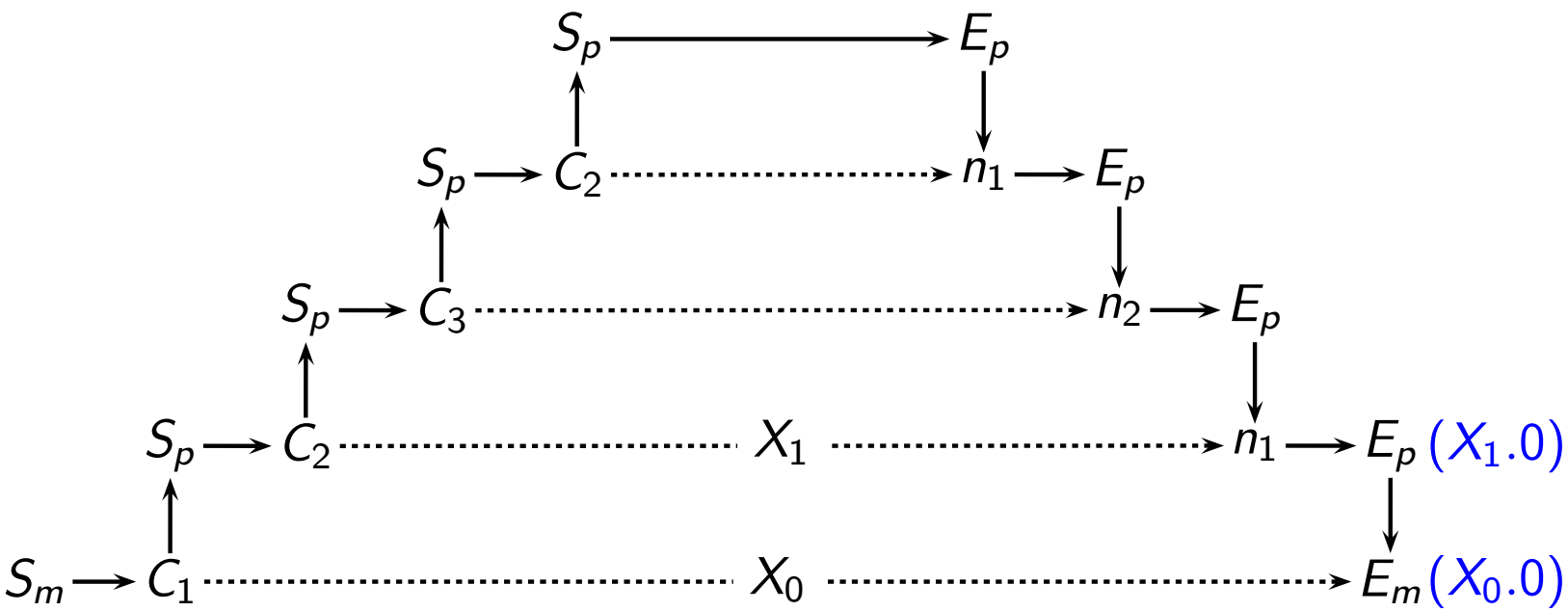
Innermost Recursion Along the Call at C_2

B_l is 0



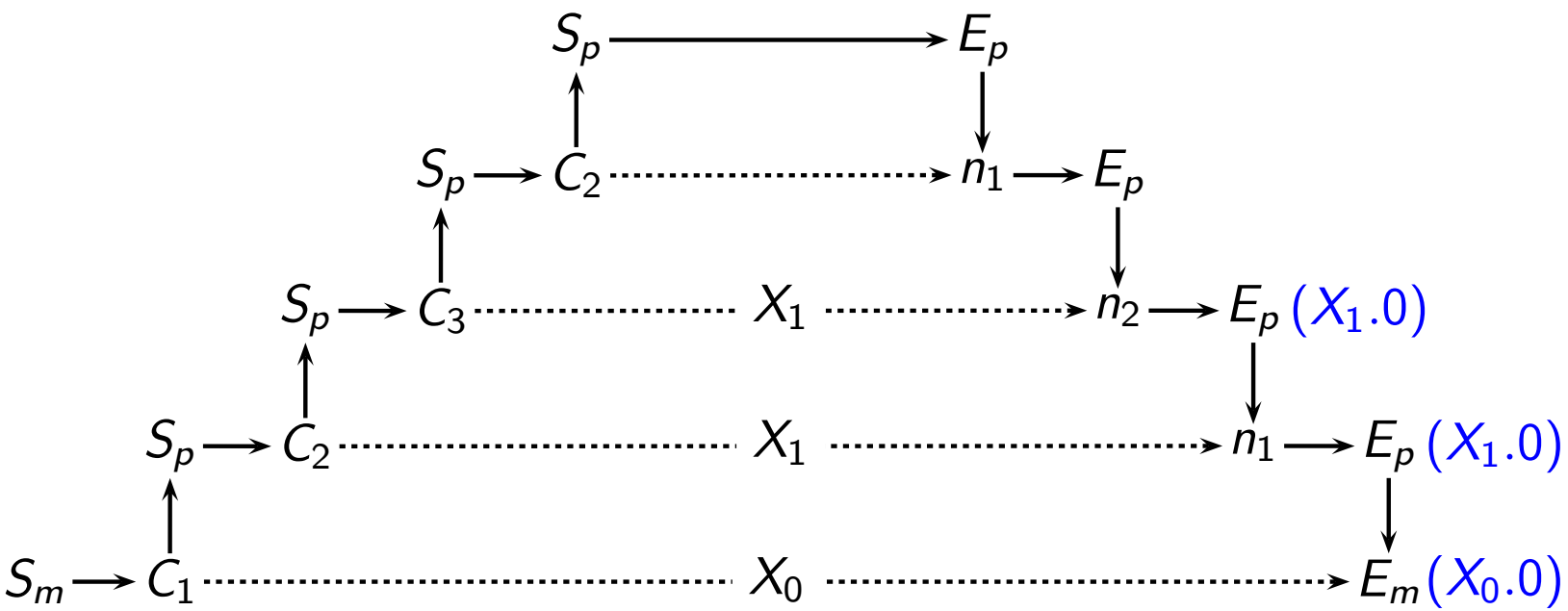
Innermost Recursion Along the Call at C_2

$Bl$is 0



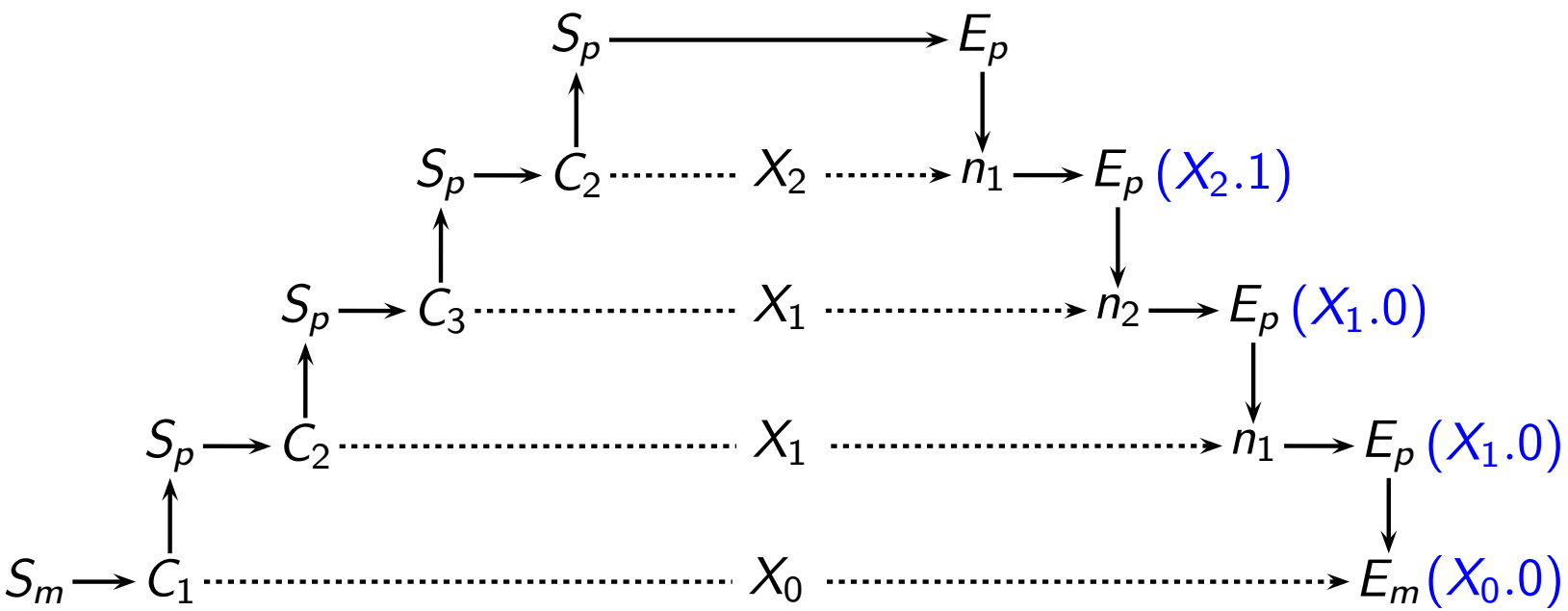
Innermost Recursion Along the Call at C_2

n_1 kills the liveness of a
New context is not required



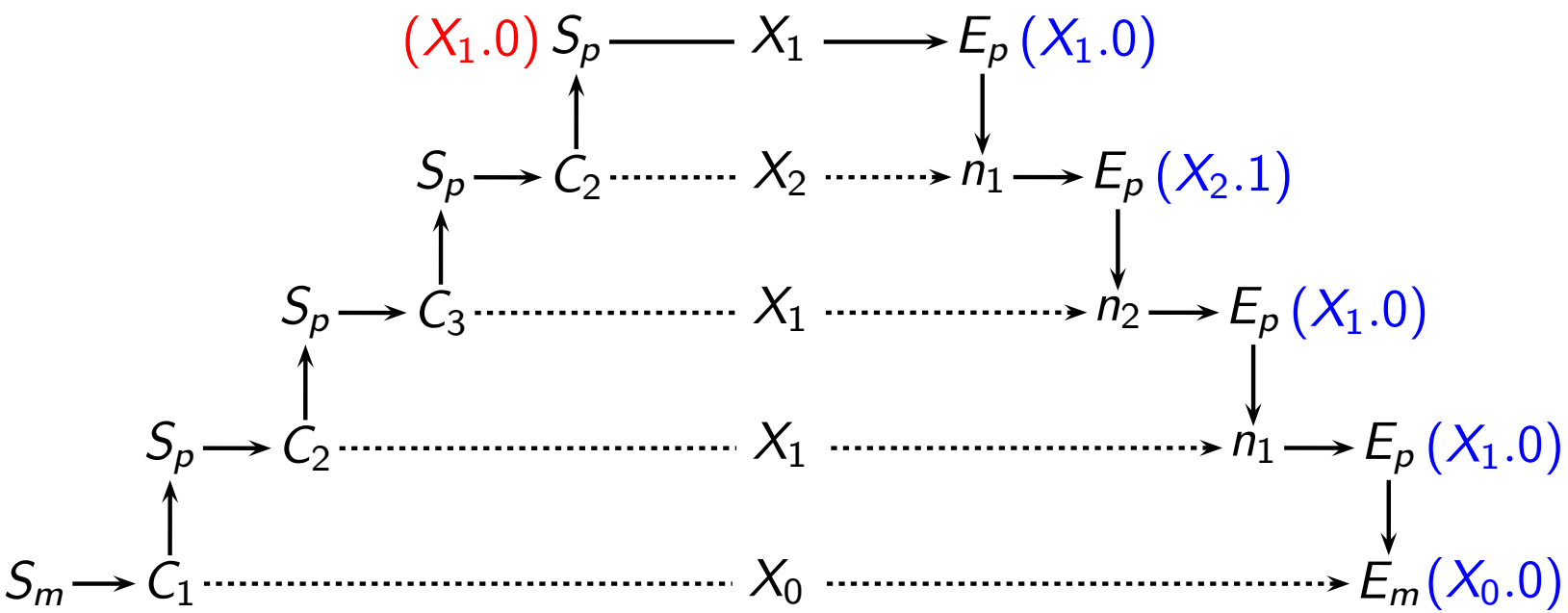
Innermost Recursion Along the Call at C_2

n_2 generates the liveness of a
New context is required



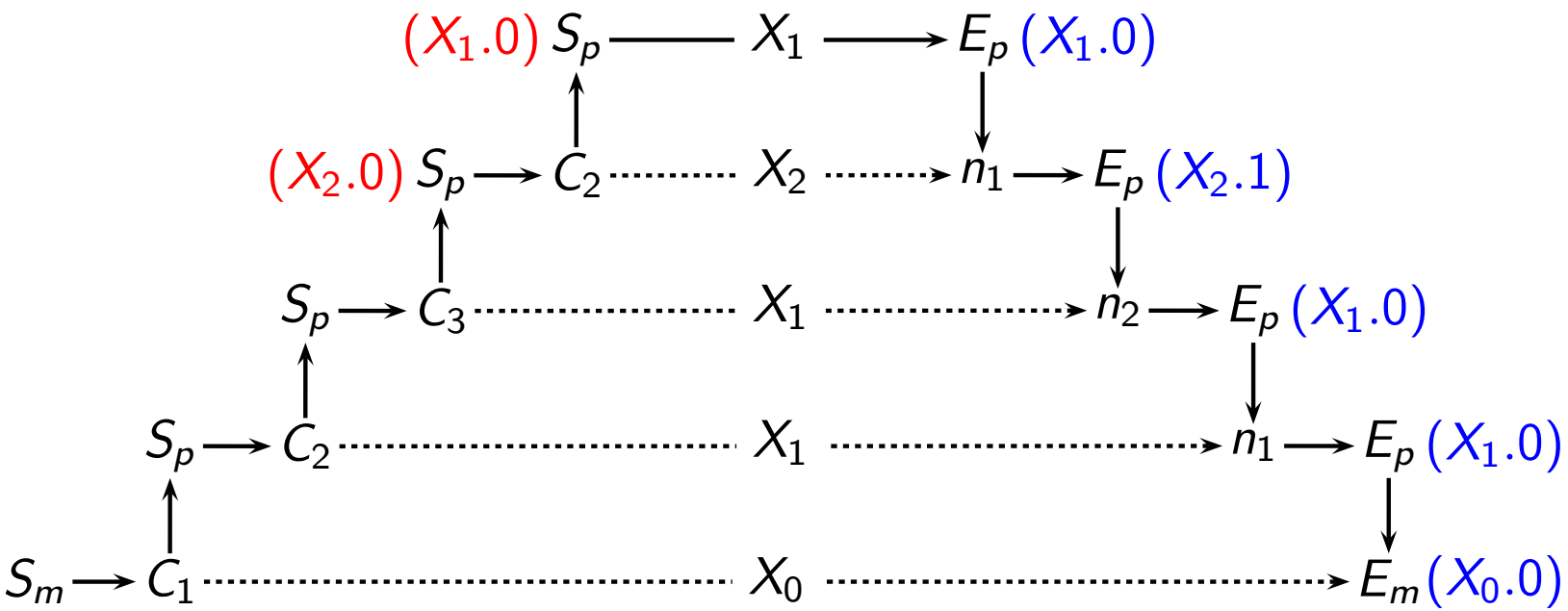
Innermost Recursion Along the Call at C_2

exitValue of X_1 is 0



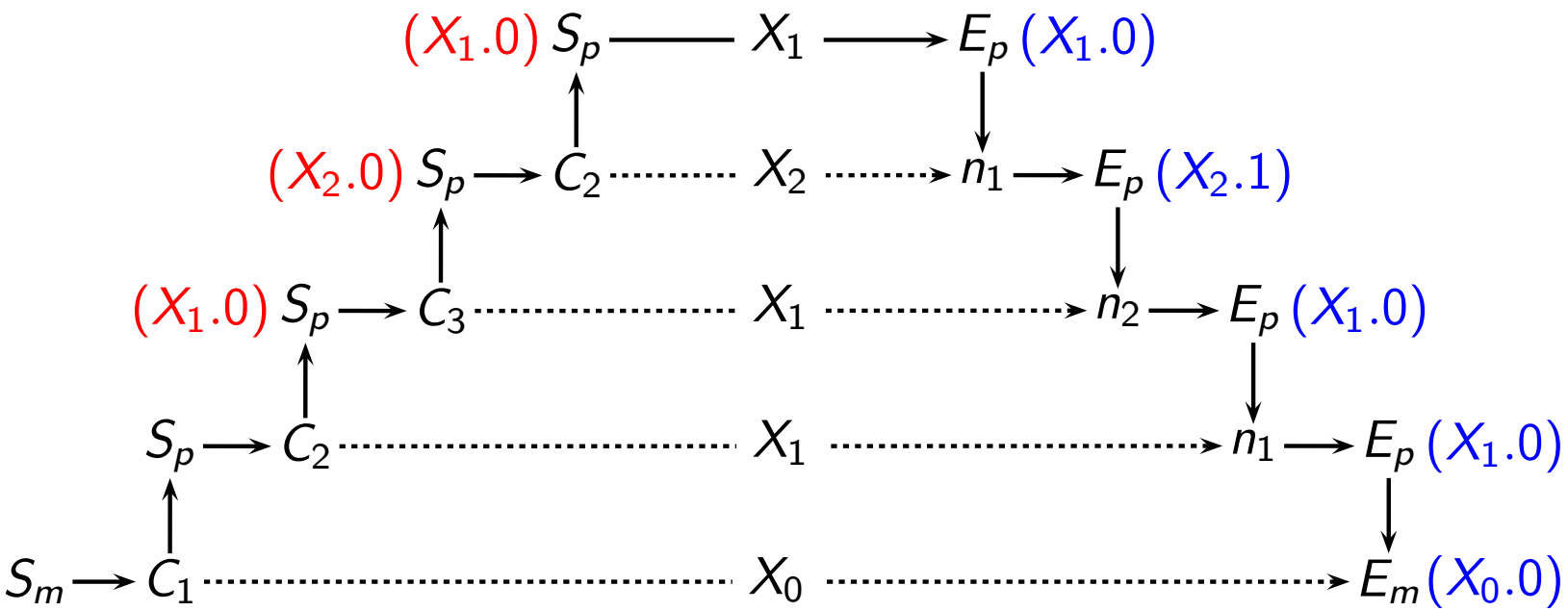
Innermost Recursion Along the Call at C_2

exitValue of X_2 is 0



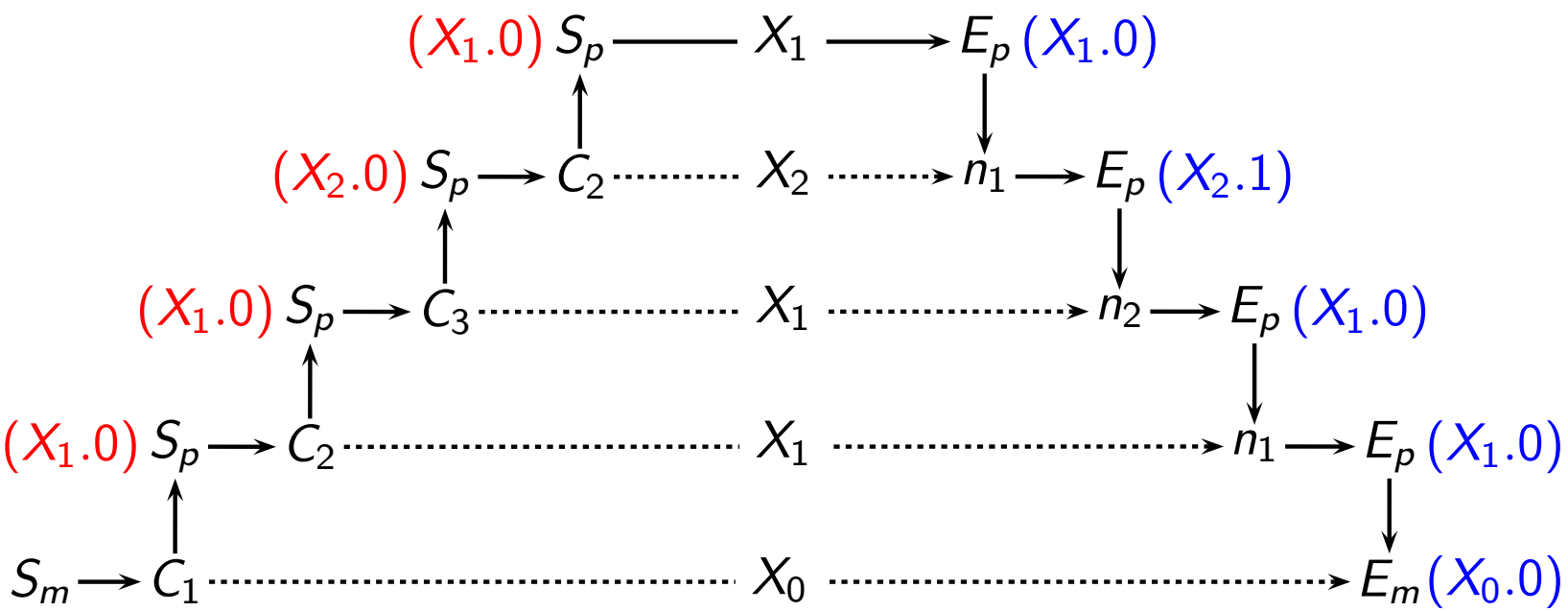
Innermost Recursion Along the Call at C_2

exitValue of X_1 remains 0



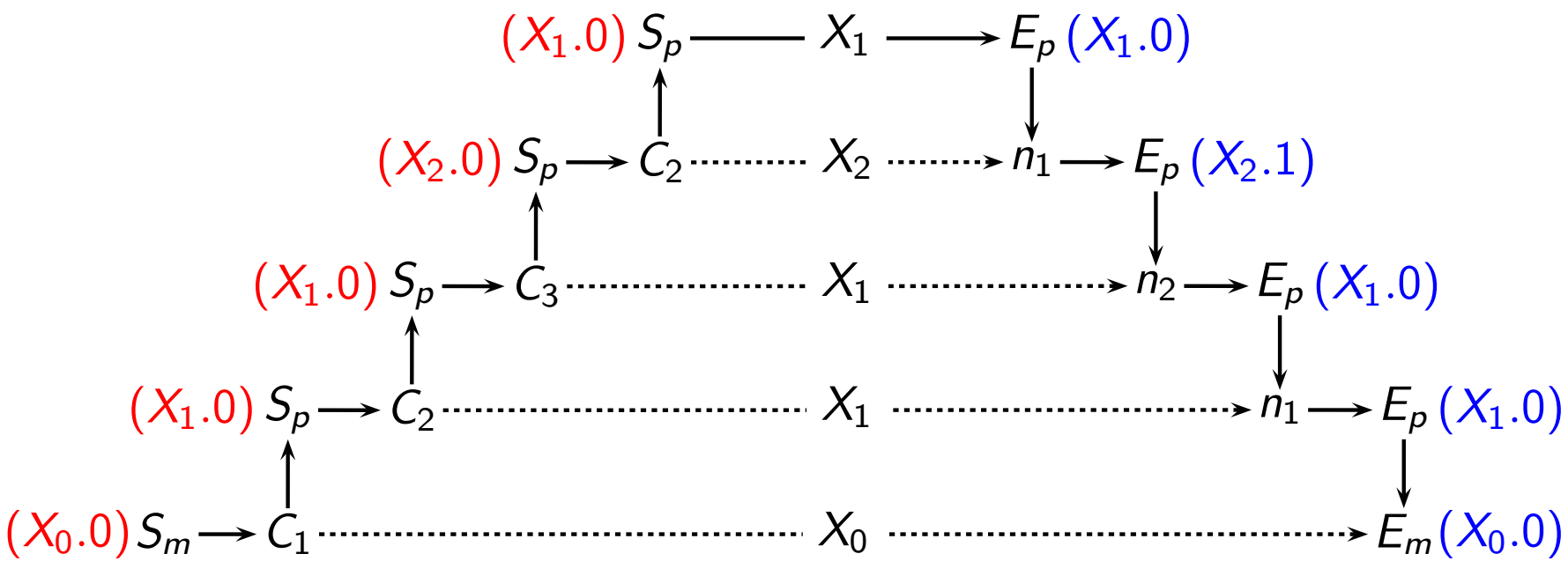
Innermost Recursion Along the Call at C_2

exitValue of X_1 remains 0



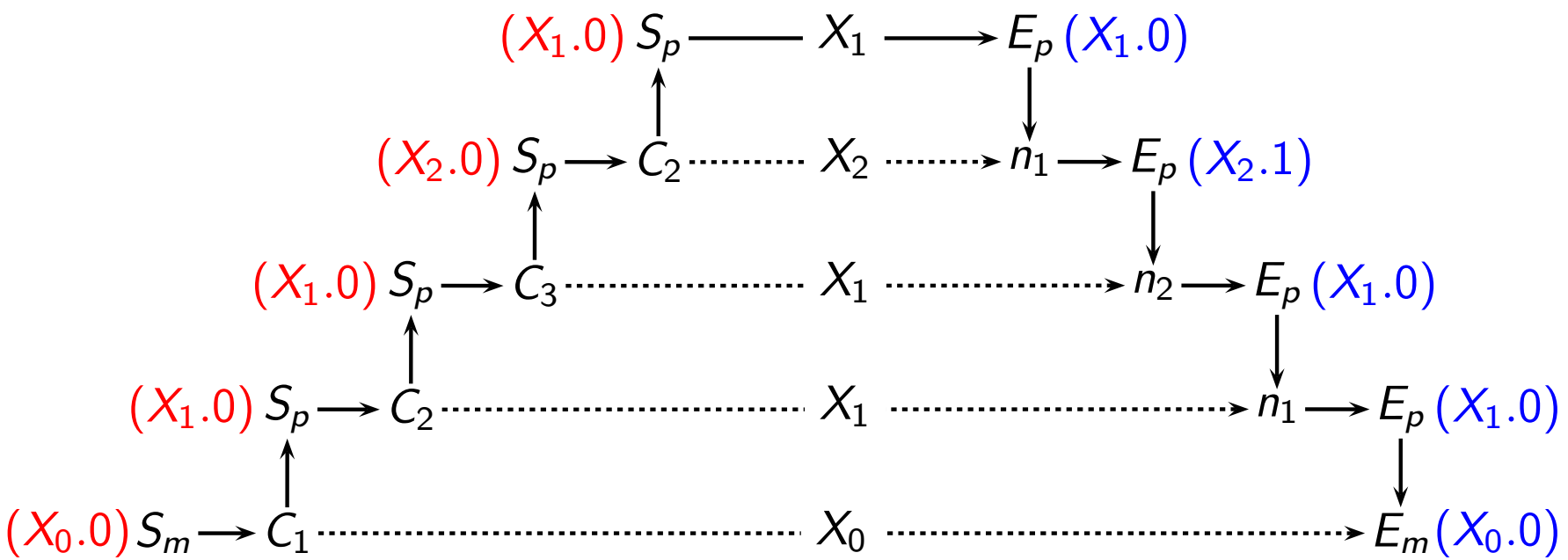
Innermost Recursion Along the Call at C_2

exitValue of X_0 is 0



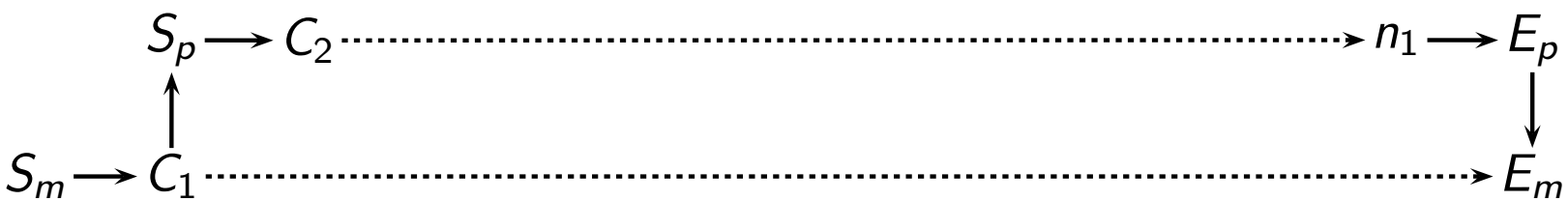
Innermost Recursion Along the Call at C_2

For this example, the innermost call determines the *exitValue* of contexts

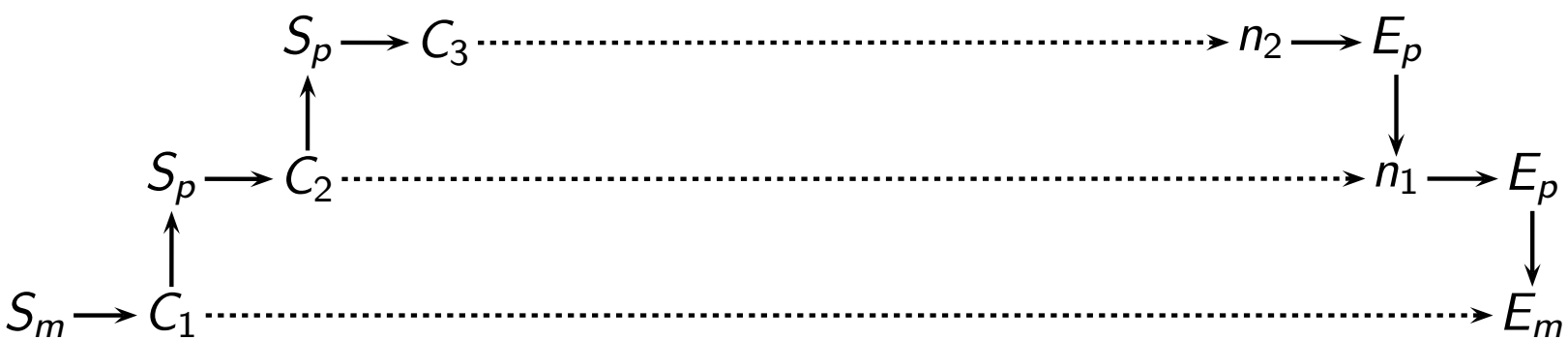


Innermost Recursion Along the Call at C_3

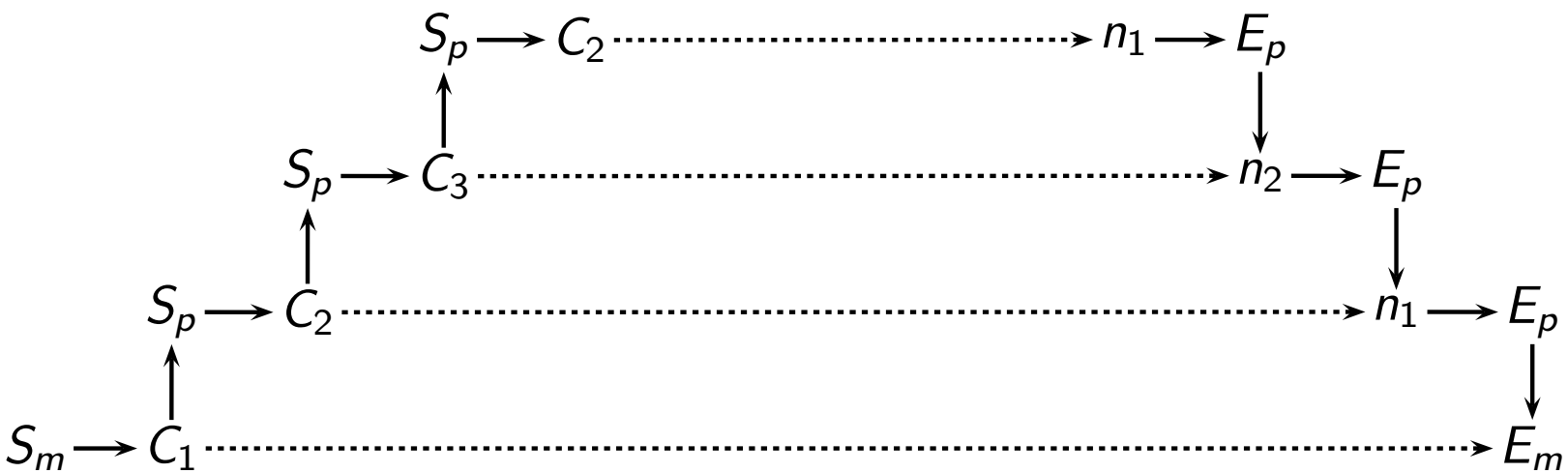
Innermost Recursion Along the Call at C_3



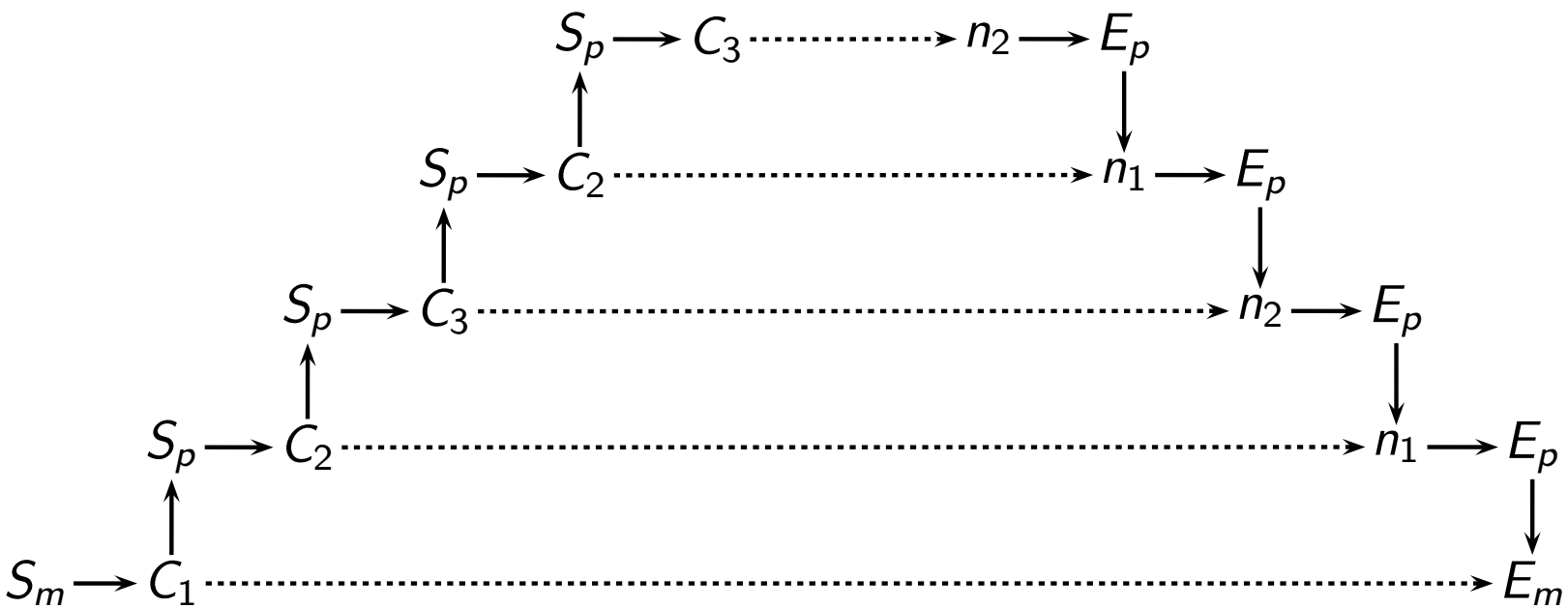
Innermost Recursion Along the Call at C_3



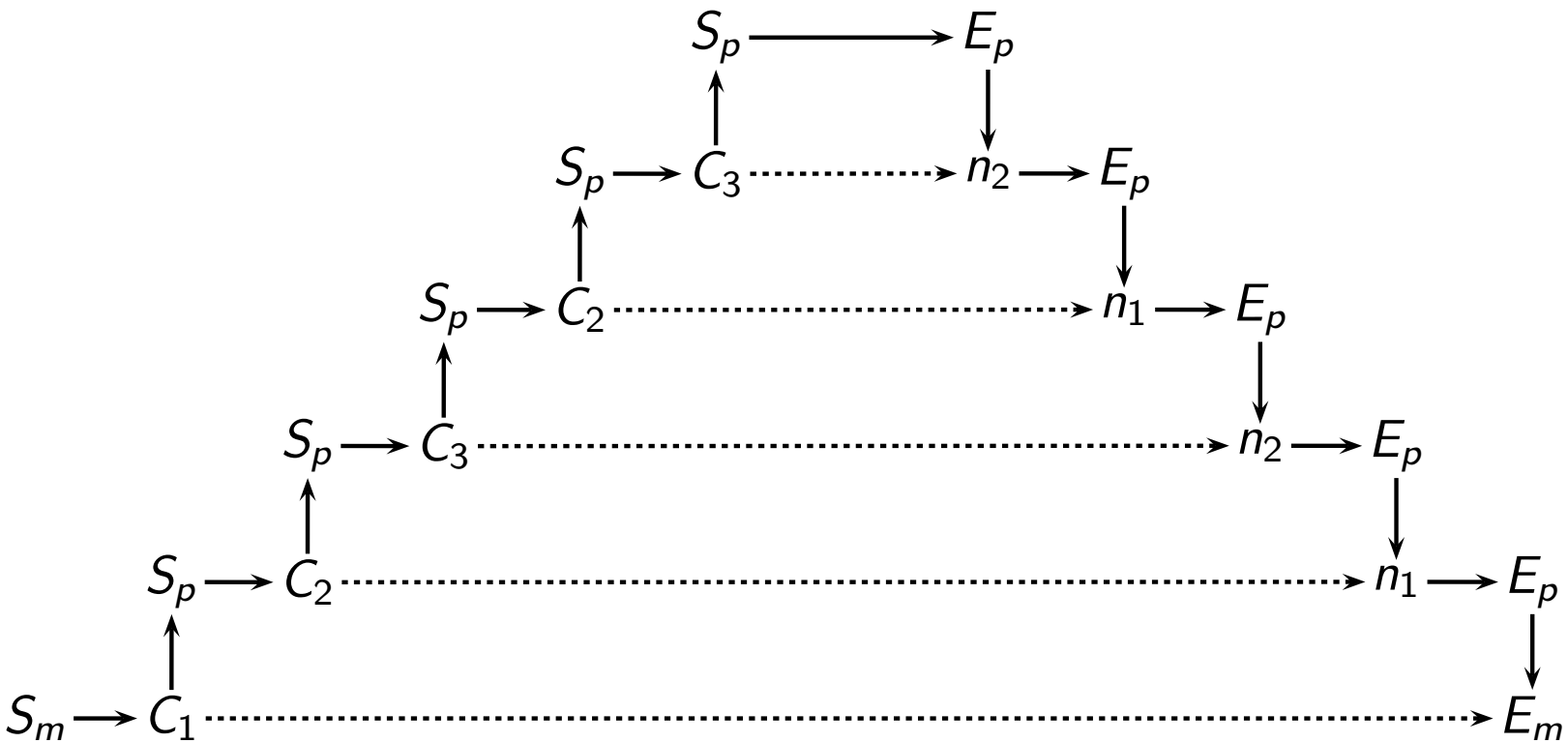
Innermost Recursion Along the Call at C_3



Innermost Recursion Along the Call at C_3

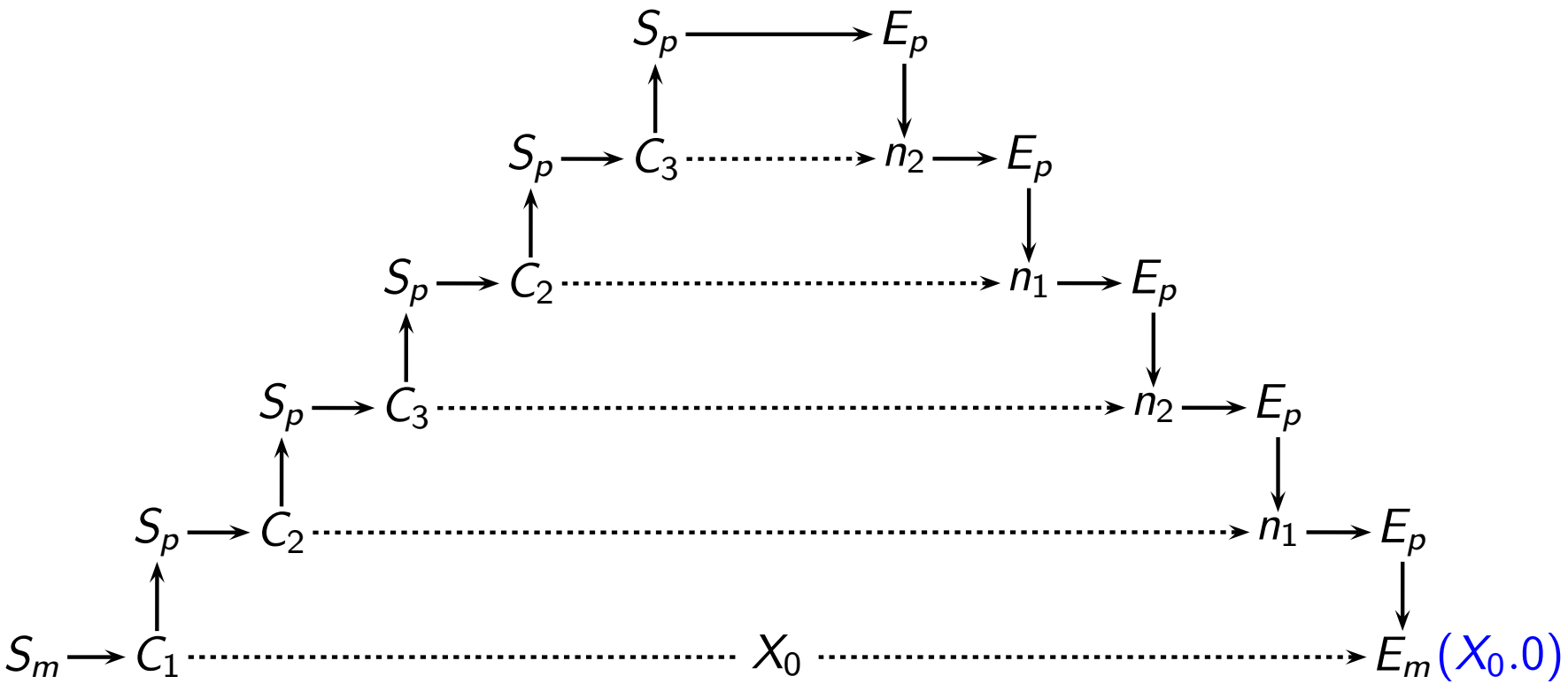


Innermost Recursion Along the Call at C_3



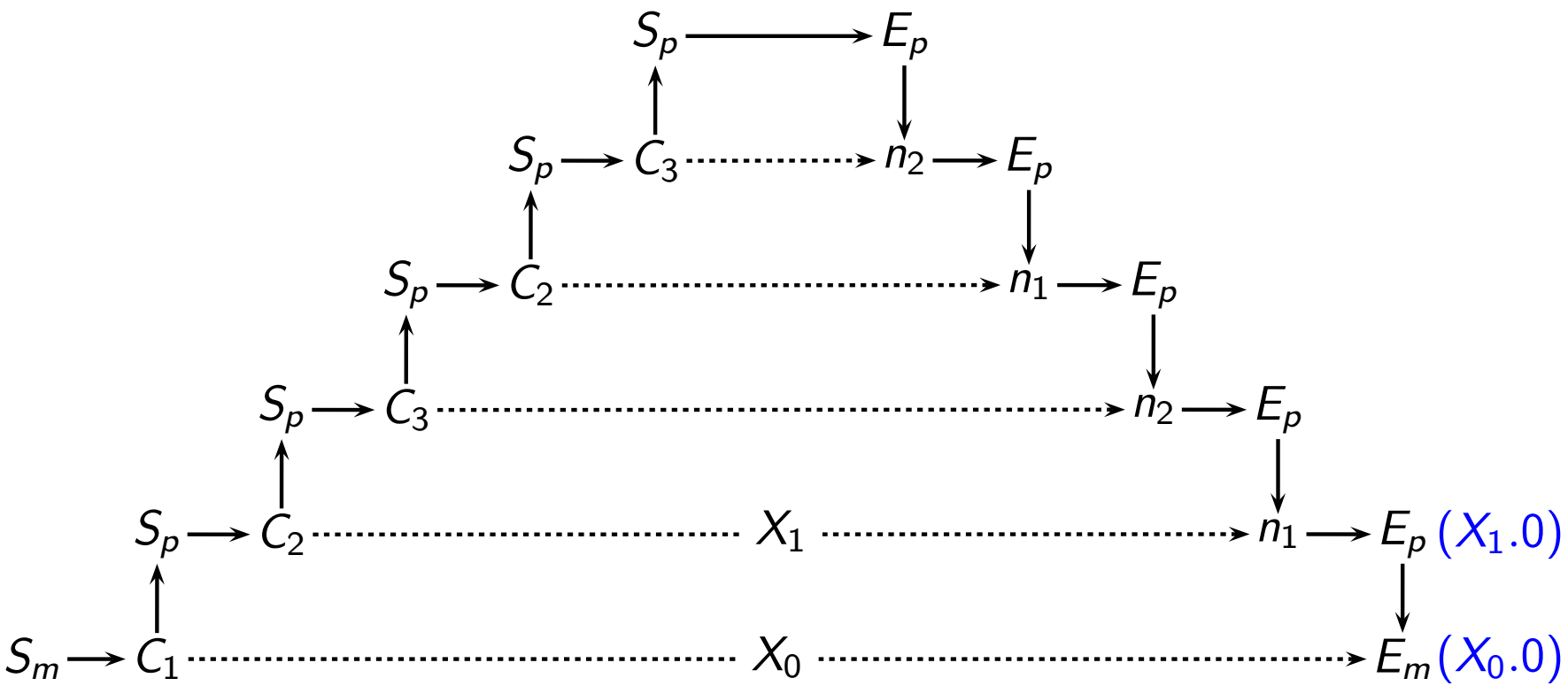
Innermost Recursion Along the Call at C_3

$Bl$is 0



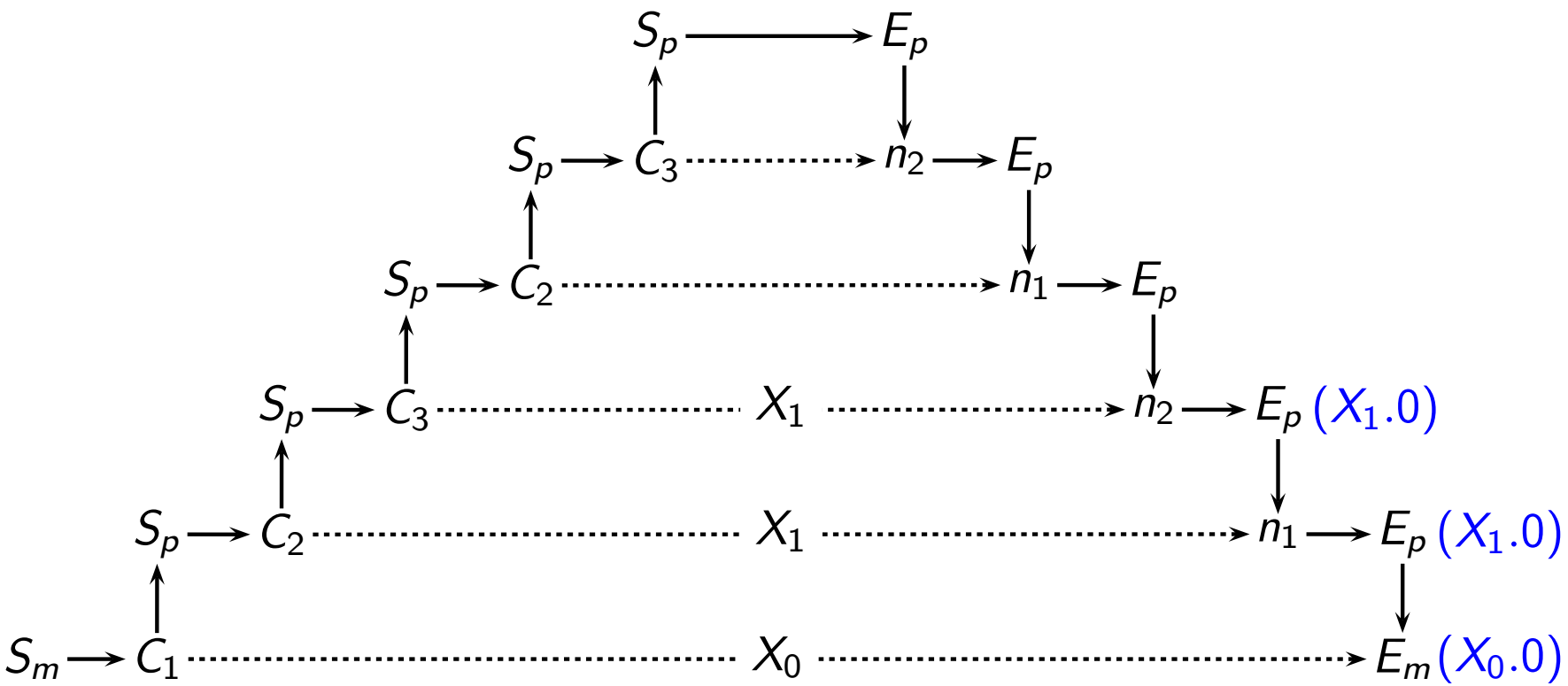
Innermost Recursion Along the Call at C_3

n_1 kills the liveness of a
New context is not required



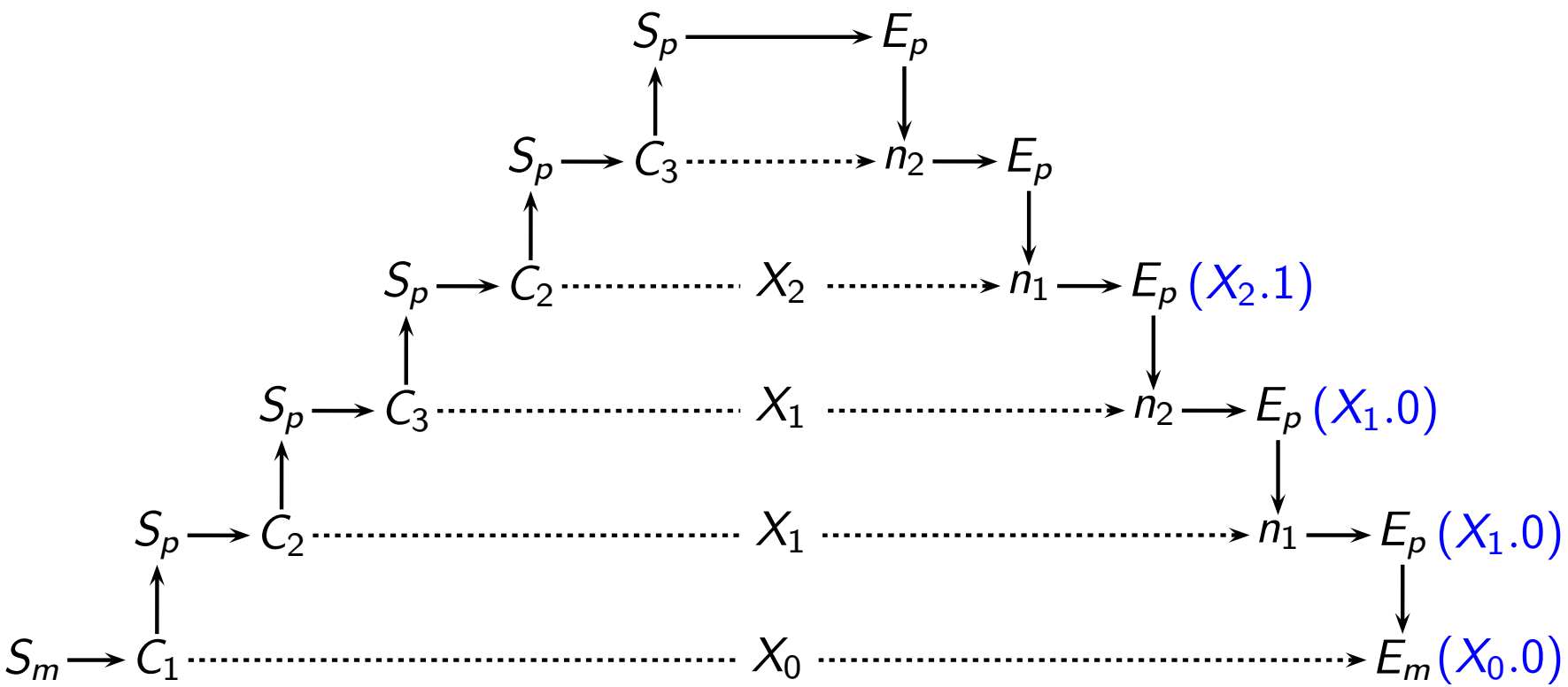
Innermost Recursion Along the Call at C_3

n_1 kills the liveness of a
New context is not required



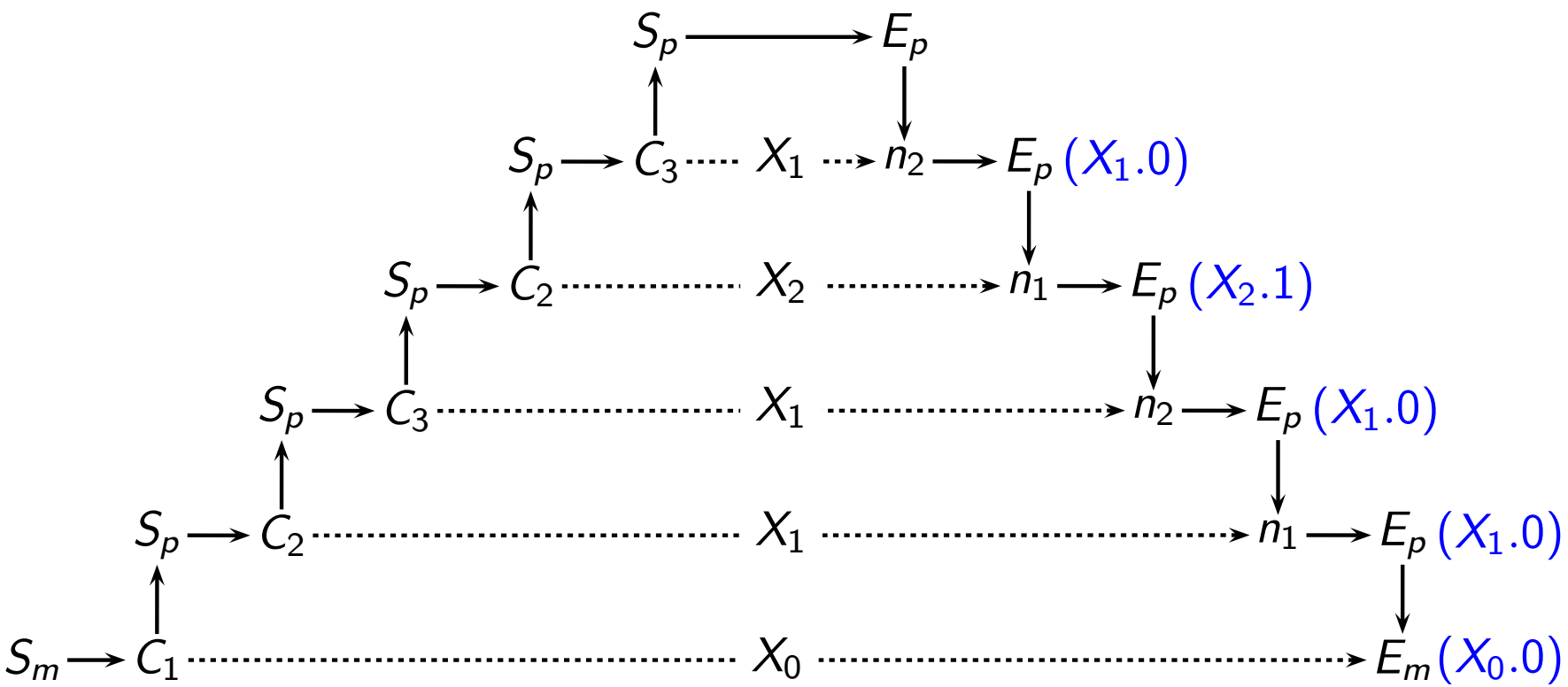
Innermost Recursion Along the Call at C_3

n_2 generates the liveness of a
New context is required



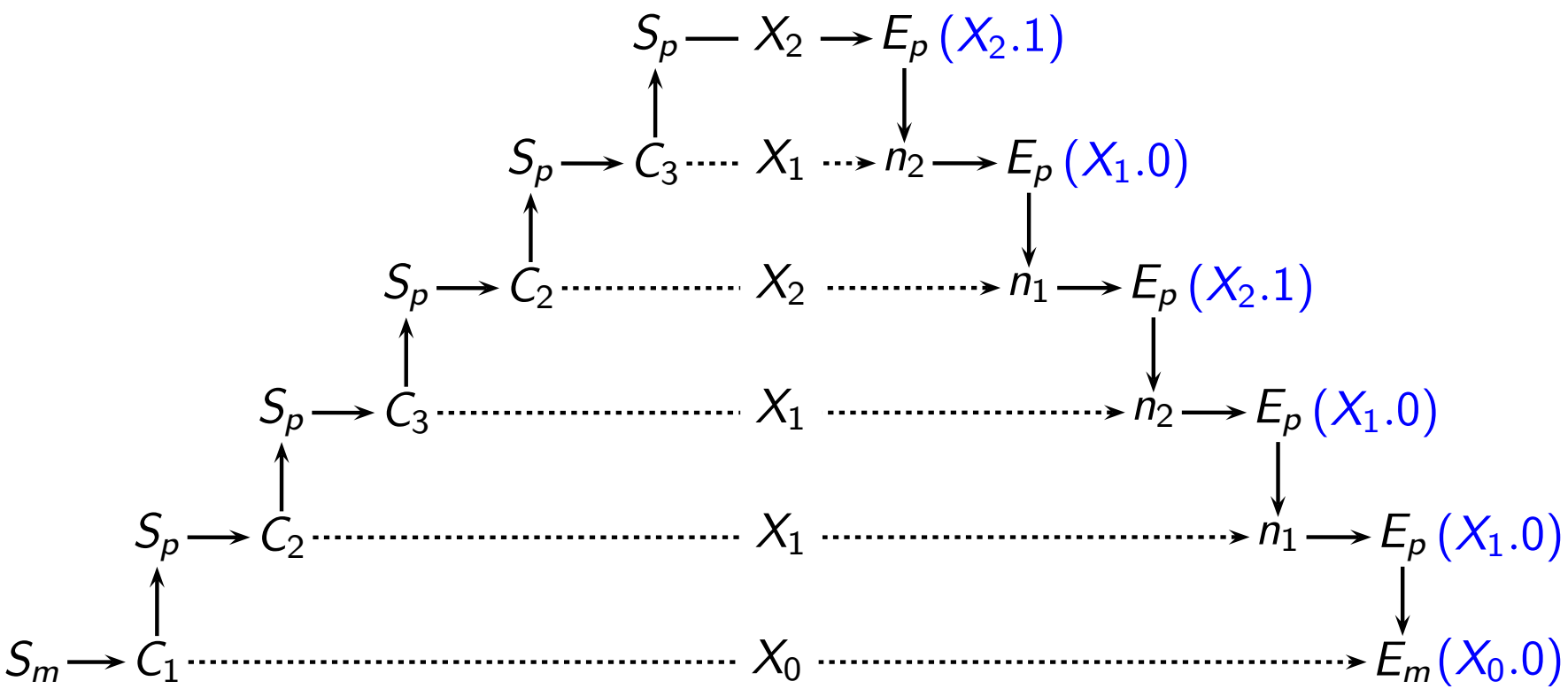
Innermost Recursion Along the Call at C_3

n_1 kills the liveness of a
New context is not required



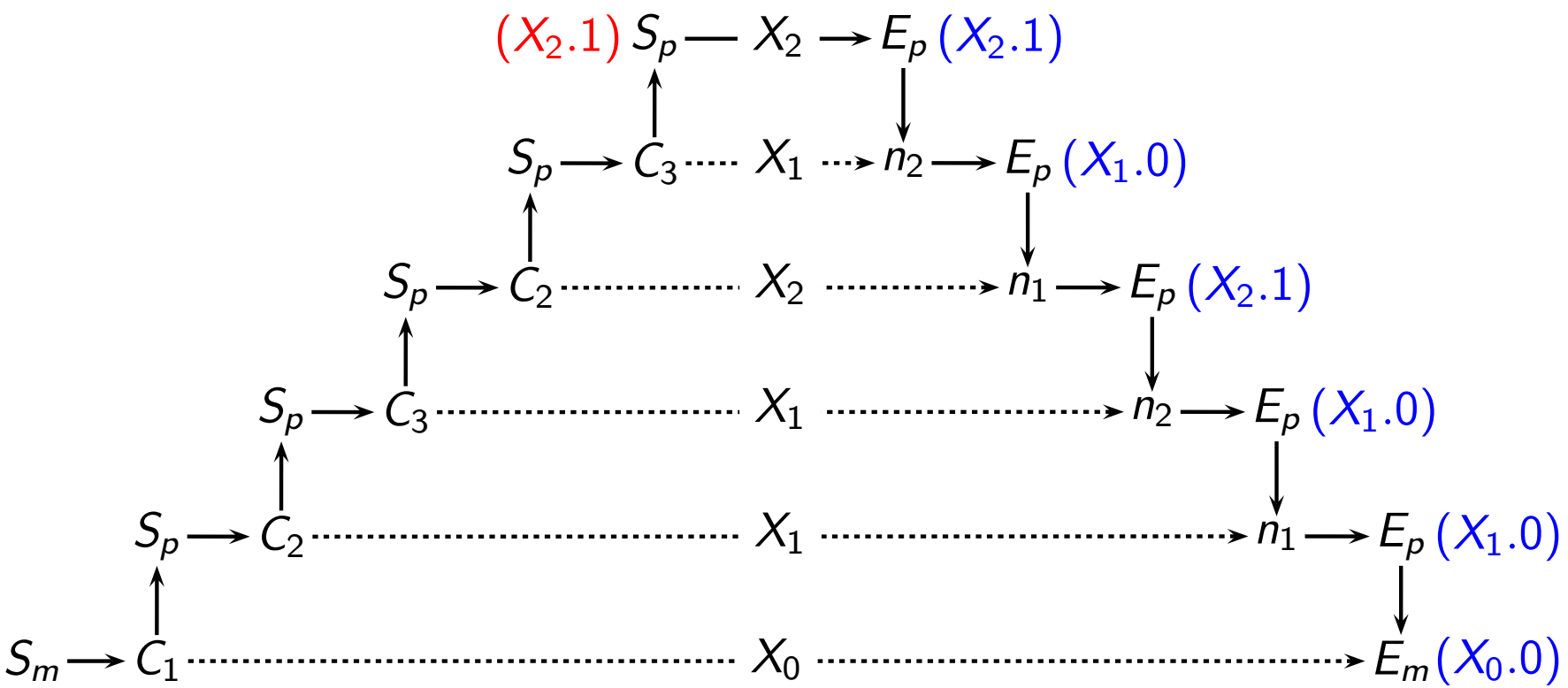
Innermost Recursion Along the Call at C_3

n_2 generates the liveness of a
New context is not required



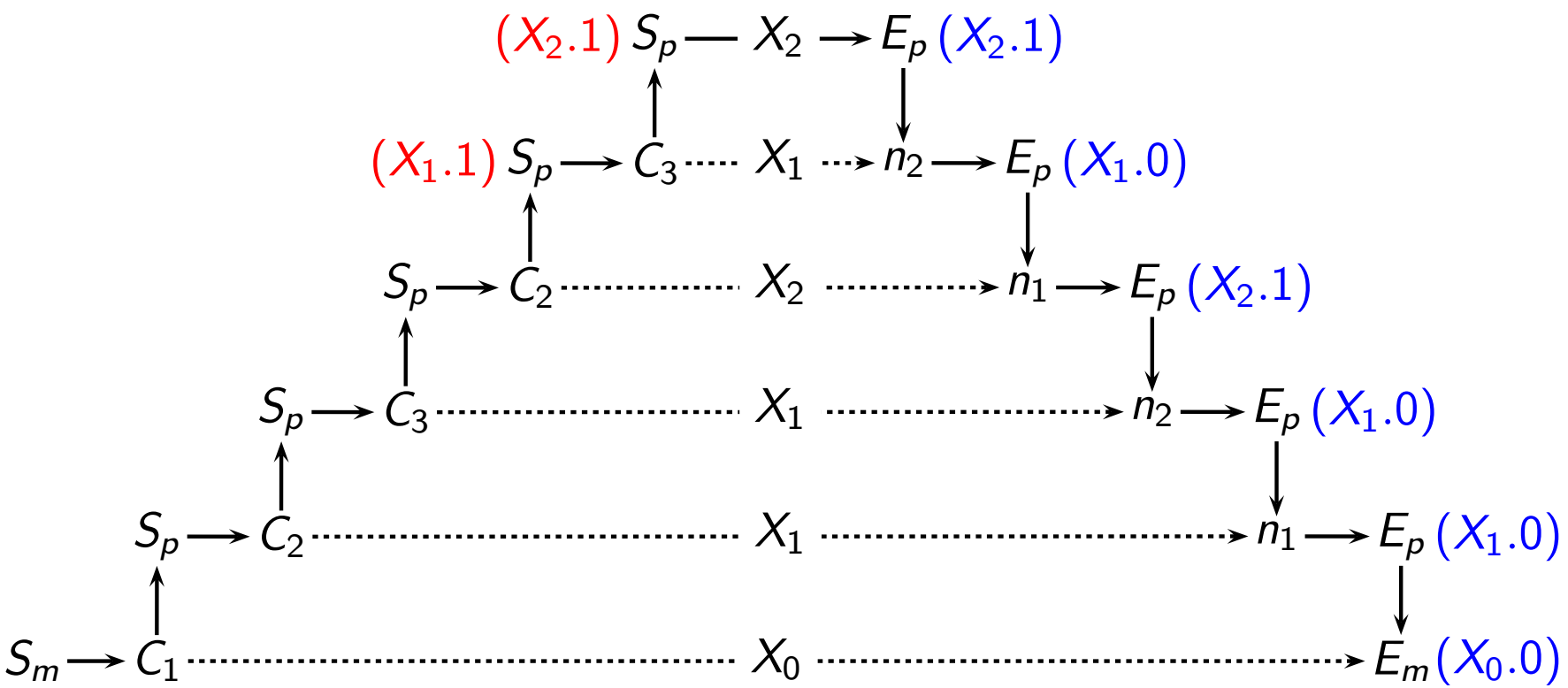
Innermost Recursion Along the Call at C_3

exitValue of X_2 is 1 (after merging with previous value 0)



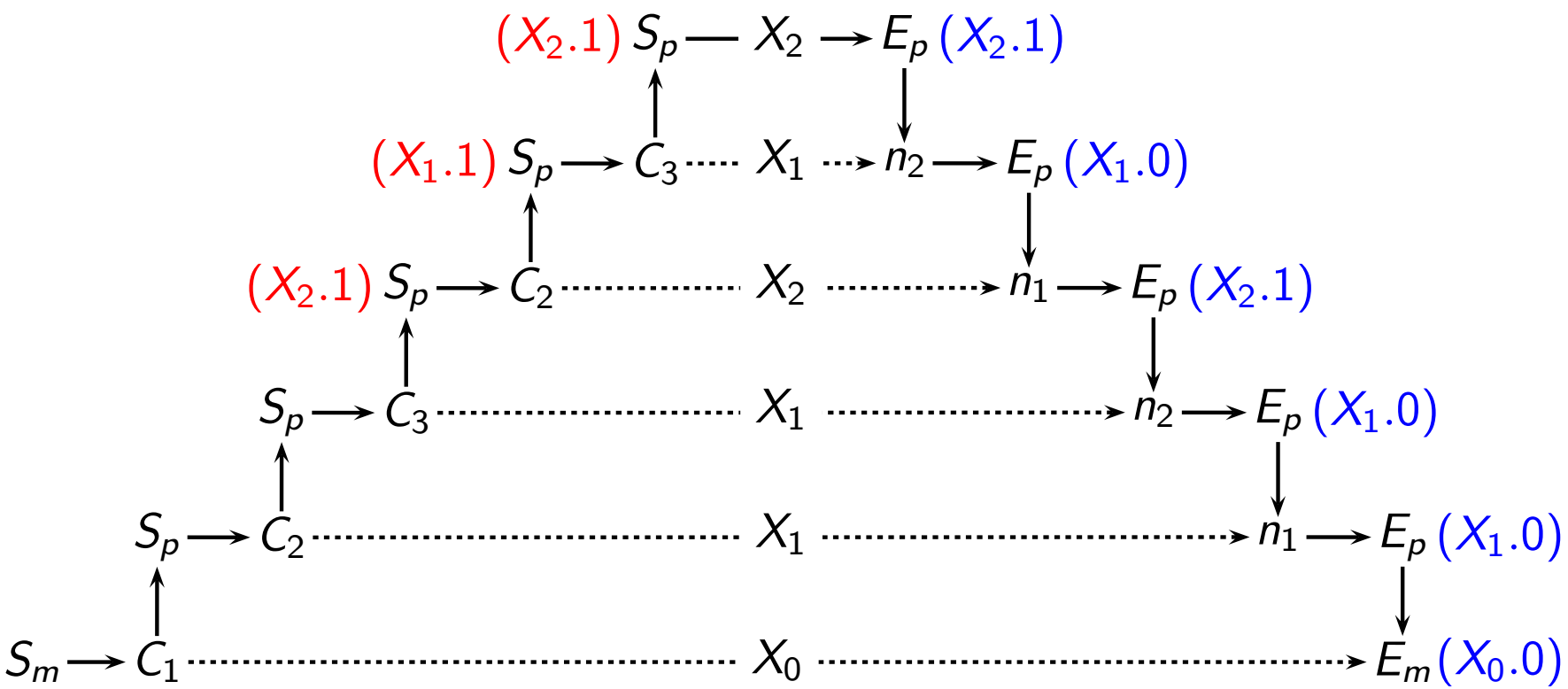
Innermost Recursion Along the Call at C_3

exitValue of X_1 is 1 (after merging with previous value 0)



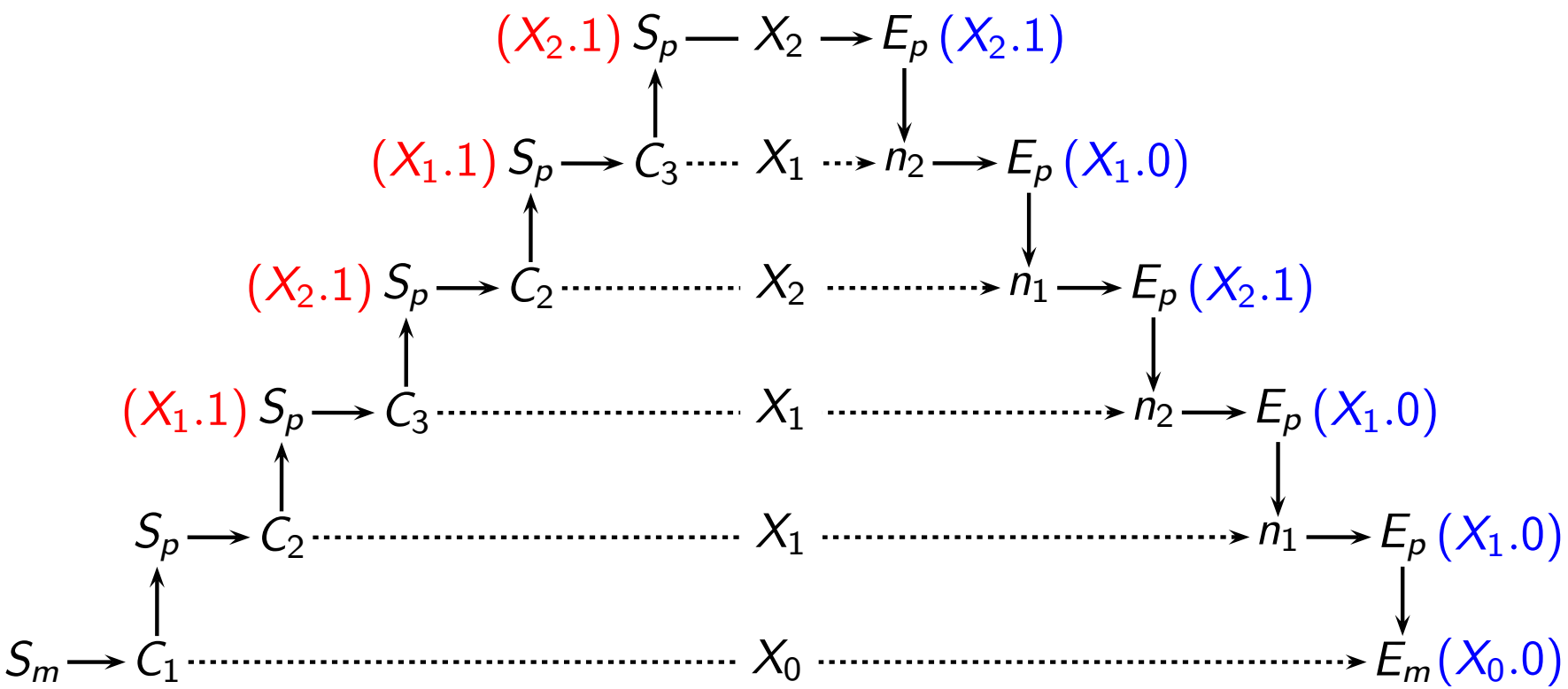
Innermost Recursion Along the Call at C_3

exitValue of X_2 remains 1



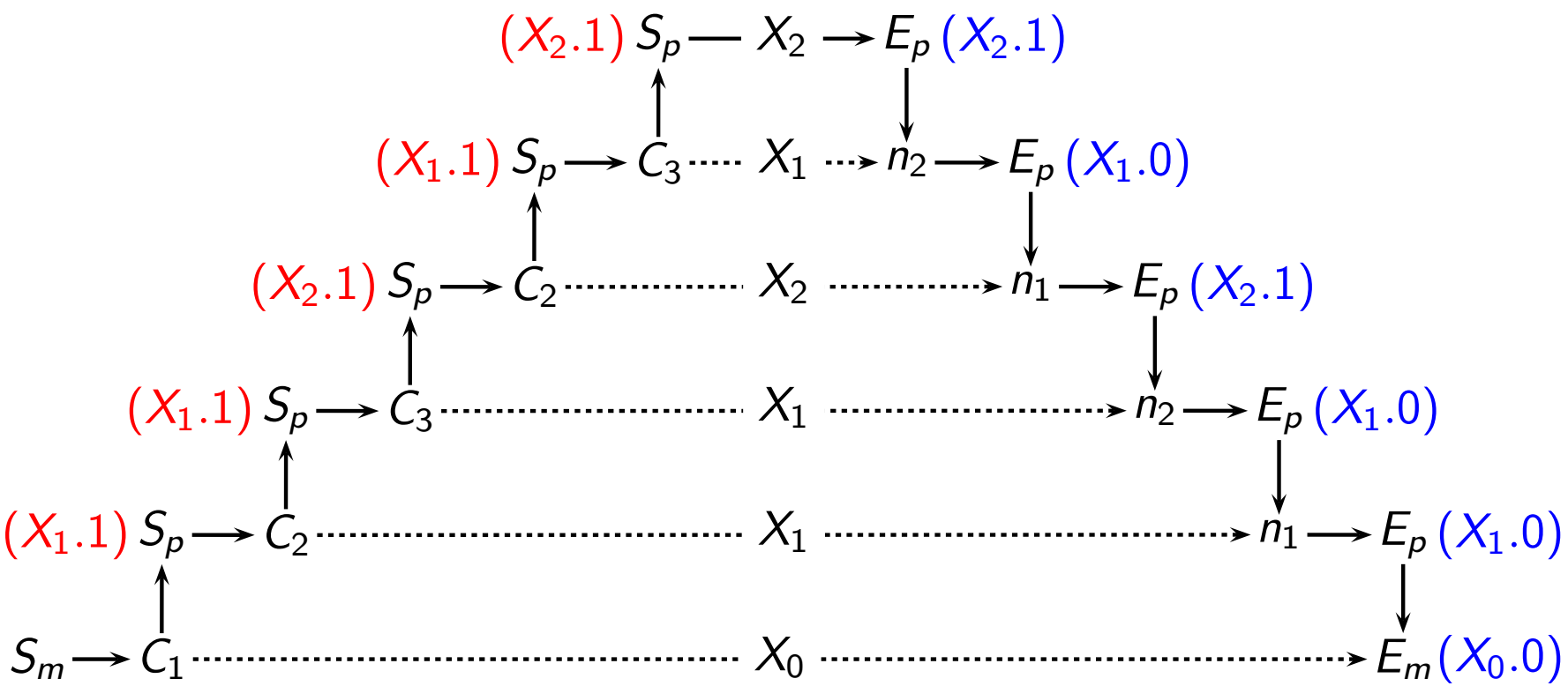
Innermost Recursion Along the Call at C_3

exitValue of X_1 remains 1



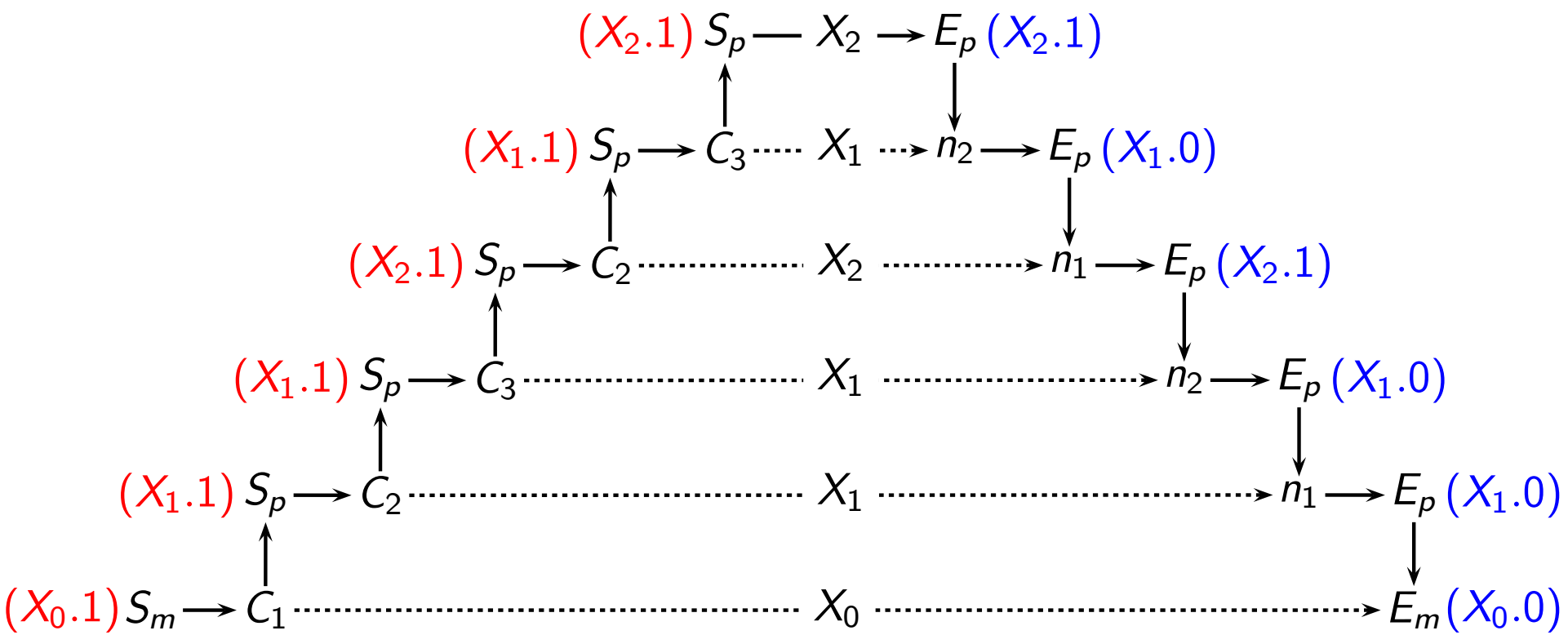
Innermost Recursion Along the Call at C_3

exitValue of X_1 remains 1



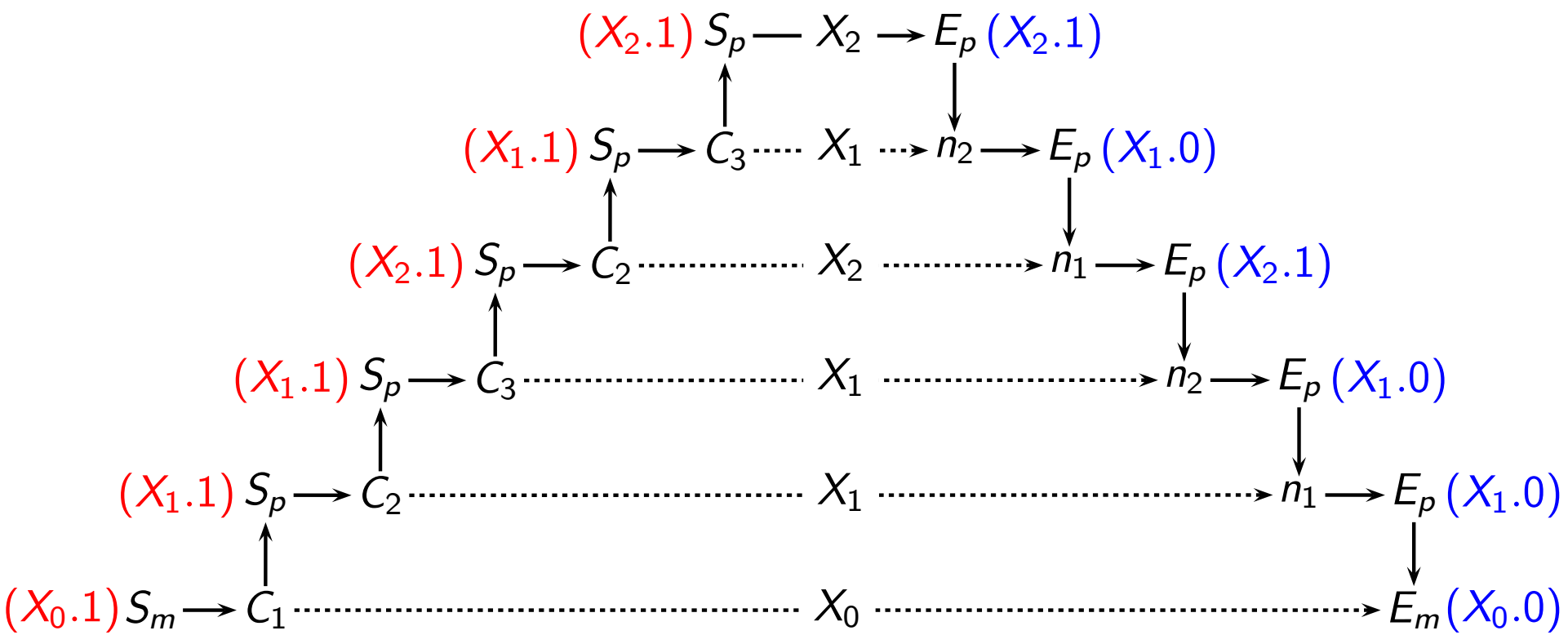
Innermost Recursion Along the Call at C_3

exitValue of X_0 is 1 (after merging with previous value 0)



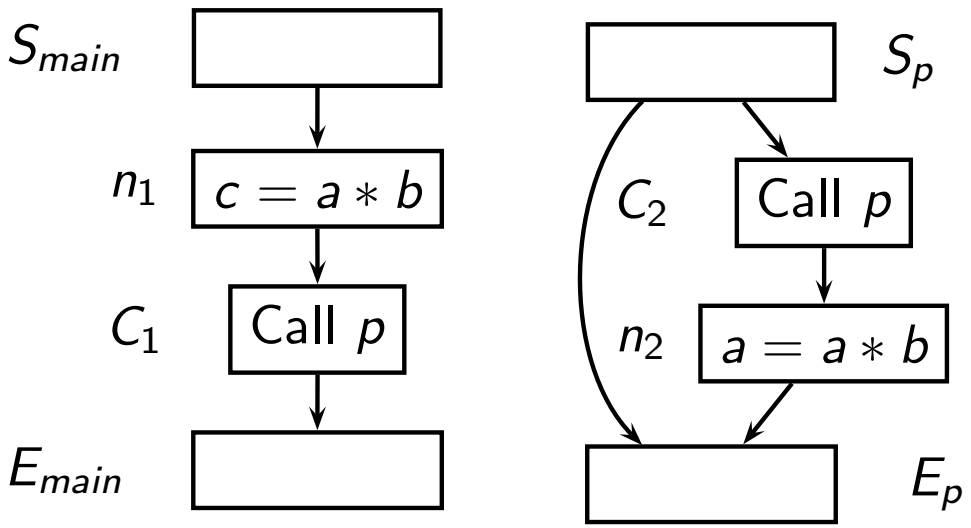
Innermost Recursion Along the Call at C_3

Again, the innermost call determines the *exitValue* of contexts
The final values at the entry of C_3 are 1 (union of 1 and 0)



Tutorial Problem #1 for Value Contexts

```
 1. int a,b,c;
 2. void main()
 3. {   c = a*b;
 4.     p();
 5. }
 6. void p()
 7. {  if (...)
 8.    {  p();
 9.       Is a*b available?
10.       a = a*b;
11.    }
12. }
```



Tutorial Problem #2 for Value Contexts

Perform interprocedural live variables analysis using value contexts

```
main()
{
   p();
}
```

```
p()
{  while (...)
   {  printf ("%d\n",a);
      p();
   }
}
```

Observe the change in edges in the transition diagram

Tutorial Problem #3 for Value Contexts

Perform interprocedural available expressions analysis using value contexts

```
main()
{
    c = a*b;
    p();
}
```

```
p()
{
    while (a > b)
    {
        p();
        a = a*b;
    }
}
```

Observe the change in edges in the transition diagram

Tutorial Problem #4 for Value Contexts

Perform interprocedural available expressions analysis using value contexts

```
1.  main()
2.  {
3.      c = a*b;
4.      p();
5.      a = a*b;
6.  }
```

```
 7.  p()
 8.  {   if (...)
 9.      {   a = a*b;
10.          p();
11.      }
12.      else if (...)
13.      {   c = a * b;
14.          p();
15.          c = a;
16.      }
17.      else
18.          ; /* ignore */
19.  }
```


Tutorial Problem #5 for Value Contexts

Perform interprocedural live variables analysis using value contexts

```
main()
{
  a = 5; b = 3;
  c = 7; d = 2;
  p();
  a = a + 2;
  e = c+d;
  d = a*b;
  q();
  print a+c+e;
}
```

```
p()
{
  b = 2;
  if (b<d)
     c = a+b;
  else
     q();
  print c+d;
}
```

```
q()
{
  a = 1;
  p();
  a = a*b;
}
```

Context sensitivity: e is live on entry to p but not before its call in main

Result of Tutorial #5

```
main()
{
  a = 5; b = 3;
  c = 7; d = 2;
  /*{a,d}*/
  p();
  /*{a,b,c,d}*/
  a = a + 2;
  e = c+d;
  /*{a,b,e}*/
  d = a*b;
  /*{d,e}*/
  q();
  /*{a,c,e}*/
  print a+c+e;
}
```

```
p()
{  /*{a,d,e}*/
  b = 2;
  if (b<d)
    /*{a,b,d,e}*/
     c = a+b;
  else
    /*{d,e}*/
     q();
  /*{a,b,c,d,e}*/
  print c+d;
}
```

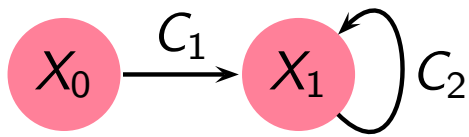
```
q()
{
  /*{d,e}*/
  a = 1;
  /*{a,d,e}*/
  p();
  /*{a,b,c,d,e}*/
  a = a*b;
}
```

Tutorial Problem #6: Interprocedural Points-to Analysis

```
main()
{   x = &y;
    z = &x;
    y = &z;
    p(); /* C1 */
}

p()
{   if (...)
    {   p(); /* C2 */
        x = *x;
    }
}
```

Value contexts method requires three contexts as shown below in the transition diagram



Reaching Definitions Analysis in GCC 4.0

Program	LoC	$\#F$	$\#C$	$3K$ length bound				Proposed Approach		
				K	#CS	Max	Time	#CS	Max	Time
hanoi	33	2	4	4	100000+	99922	3973 $\times 10^3$	8	7	2.37
bit_gray	53	5	11	7	100000+	31374	2705 $\times 10^3$	17	6	3.83
analyzer	288	14	20	2	21	4	20.33	21	4	1.39
distray	331	9	21	6	96	28	322.41	22	4	1.11
mason	350	9	13	8	100000+	22143	432 $\times 10^3$	14	4	0.43
fourinarow	676	17	45	5	510	158	397.76	46	7	1.86
sim	1146	13	45	8	100000+	33546	1427 $\times 10^3$	211	105	234.16
181_mcf	1299	17	24	6	32789	32767	484 $\times 10^3$	41	11	5.15
256_bzip2	3320	63	198	7	492	63	258.33	406	34	200.19

- LoC is the number of lines of code,
- $\#F$ is the number of procedures,
- $\#C$ is the number of call sites,
- $\#CS$ is the number of call strings
- Max denotes the maximum number of call strings reaching any node.
- Analysis time is in milliseconds.

(Implementation was carried out by Seema Ravandale.)

Some Observations

- Compromising on precision may not be necessary for efficiency.
- Separating the necessary information from redundant information is much more significant.
- Data flow propagation in real programs seems to involve only a small subset of all possible values.

  Much fewer changes than the theoretically possible worst case number of changes.
- A precise modelling of the process of analysis is often an eye opener.

Some Observations

- Compromising on precision may not be necessary for efficiency.
- Separating the necessary information from redundant information is much more significant.
- Data flow propagation in real programs seems to involve only a small subset of all possible values.

  Much fewer changes than the theoretically possible worst case number of changes.
- A precise modelling of the process of analysis is often an eye opener.

  # distinct tagged values =

  Min (# actual contexts, # actual data flow values)