INTERVAL ABSTRACT DOMAIN

- $I = \{[a, b] \mid a, b \in \mathbb{R} \cup \{-\infty, \infty\}\} \cup \{\perp\}$
    - $D = V \to I$
    - Also called Box abstract domain.
- $[a_1, b_1] \sqsubseteq [a_2, b_2] \Leftrightarrow a_2 \leq a_1 \wedge b_1 \leq b_2$, $\forall d \in I \,.\, \perp \sqsubseteq d$
    - Is $(I, \sqsubseteq)$ a lattice?
    - Is $(I, \sqsubseteq)$ a complete lattice?
    - Maximal element?
- $[a_1, b_1] \sqcup [a_2, b_2] = ???$

INTERVAL ABSTRACT DOMAIN

- $I = \{[a, b] \mid a, b \in \mathbb{R} \cup \{-\infty, \infty\}\} \cup \{\bot\}$
  - $D = V \to I$
  - Also called Box abstract domain.
- $[a_1, b_1] \sqsubseteq [a_2, b_2] \Leftrightarrow a_2 \leq a_1 \wedge b_1 \leq b_2$, $\forall d \in I \,.\, \bot \sqsubseteq d$
  - Is $(I, \sqsubseteq)$ a lattice?
  - Is $(I, \sqsubseteq)$ a complete lattice?
  - Maximal element?
- $[a_1, b_1] \sqcup [a_2, b_2] = [min(a_1, a_2), max(b_1, b_2)]$

INTERVAL ABSTRACT DOMAIN

- $I = \{[a, b] \mid a, b \in \mathbb{R} \cup \{-\infty, \infty\}\} \cup \{\perp\}$
  - $D = V \to I$
  - Also called Box abstract domain.
- $[a_1, b_1] \sqsubseteq [a_2, b_2] \Leftrightarrow a_2 \leq a_1 \wedge b_1 \leq b_2$, $\forall d \in I \,.\, \perp \sqsubseteq d$
  - Is $(I, \sqsubseteq)$ a lattice?
  - Is $(I, \sqsubseteq)$ a complete lattice?
  - Maximal element?
- $[a_1, b_1] \sqcup [a_2, b_2] = [min(a_1, a_2), max(b_1, b_2)]$
- $(D, \sqsubseteq)$: $\forall d_1, d_2 \in D \,.\, d_1 \sqsubseteq d_2 \Leftrightarrow \forall v \in V \,.\, d_1(v) \sqsubseteq d_2(v)$

INTERVAL ABSTRACT DOMAIN

ABSTRACTION AND CONCRETIZATION FUNCTION

- $\alpha : \mathbb{P}(State) \rightarrow D, \gamma : D \rightarrow \mathbb{P}(State)$
- $\alpha(c) = d$
  - $d(v) = [min\{\sigma(v) \,|\, \sigma \in c\}, max\{\sigma(v) \,|\, \sigma \in c\}]$
- $\gamma(d) = \{\sigma \mid \forall v \in V \,.\, d(v) = [a, b] \Rightarrow a \leq \sigma(v) \leq b\}$
- Is $(\mathbb{P}(State), \subseteq) \underset{\gamma}{\overset{\alpha}{\rightleftarrows}} (D, \sqsubseteq)$ a Galois Connection?
  - Is it an Onto Galois Connection?

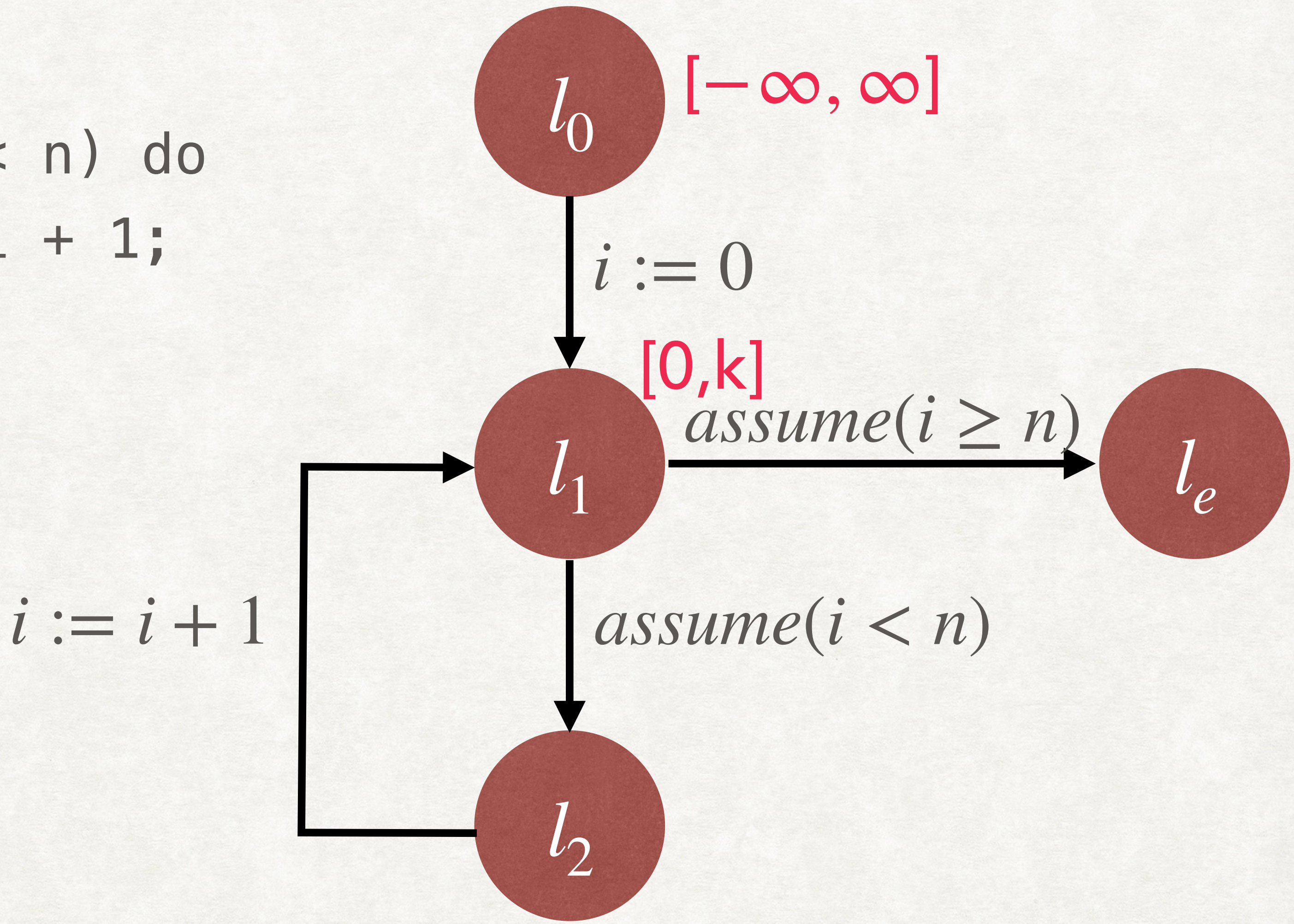
INTERVAL ABSTRACT DOMAIN

ABSTRACT TRANSFER FUNCTION

- Consider $c : x := x + y$
  - We can use interval arithmetic for $\hat{f}_c$
- Assuming $d(x) = [l_x, u_x], d(y) = [l_y, u_y]$
  - $\hat{f}_c(d) = d[x \mapsto [l_x + l_y, u_x + u_y]]$
- Is $\hat{f}_c$ monotonic?
  - Is it distributive?

USING INTERVAL DOMAIN

```
i := 0;
while(i < n) do
    i := i + 1;
```



Interval Abstract Domain does not satisfy ACC, hence Kildall's Algorithm may not terminate

WIDENING

- A widening function $\nabla : D \times D \to D$ on a poset $(D, \leq)$ satisfies the following properties:
  - $\forall x, y \in D \,.\, x \sqcup y \leq x \nabla y$
  - For an increasing chain $x_0, x_1, \ldots$, the increasing chain $y_0, y_1, \ldots$ where $y_0 = x_0$ and $y_n = y_{n-1} \nabla x_n$ eventually stabilizes.

WIDENING FOR THE INTERVAL DOMAIN

- We can define the widening operator for interval domain as follows:
  - $[a, b] \triangledown \perp = [a, b]$
  - $\perp \triangledown [a, b] = [a, b]$
  - $[a_1, b_1] \triangledown [a_2, b_2] = [(a_2 < a_1)? - \infty : a_1, (b_1 < b_2)? \infty : b_1]$
- Examples
  - $[1,2] \triangledown [0,2] = ???$

WIDENING FOR THE INTERVAL DOMAIN

- We can define the widening operator for interval domain as follows:
  - $[a, b] \triangledown \bot = [a, b]$
  - $\bot \triangledown [a, b] = [a, b]$
  - $[a_1, b_1] \triangledown [a_2, b_2] = [(a_2 < a_1)? - \infty : a_1, (b_1 < b_2)? \infty : b_1]$
- Examples
  - $[1,2] \triangledown [0,2] = [-\infty,2]$

WIDENING FOR THE INTERVAL DOMAIN

- We can define the widening operator for interval domain as follows:
  - $[a, b] \triangledown \bot = [a, b]$
  - $\bot \triangledown [a, b] = [a, b]$
  - $[a_1, b_1] \triangledown [a_2, b_2] = [(a_2 < a_1) ? -\infty : a_1, (b_1 < b_2) ? \infty : b_1]$
- Examples
  - $[1,2] \triangledown [0,2] = [-\infty,2]$
  - $[0,2] \triangledown [1,2] = ???$

WIDENING FOR THE INTERVAL DOMAIN

- We can define the widening operator for interval domain as follows:
  - $[a, b] \triangledown \bot = [a, b]$
  - $\bot \triangledown [a, b] = [a, b]$
  - $[a_1, b_1] \triangledown [a_2, b_2] = [(a_2 < a_1)? -\infty : a_1, (b_1 < b_2)? \infty : b_1]$
- Examples
  - $[1,2] \triangledown [0,2] = [-\infty,2]$
  - $[0,2] \triangledown [1,2] = [0,2]$

WIDENING FOR THE INTERVAL DOMAIN

- We can define the widening operator for interval domain as follows:
  - $[a, b] \triangledown \bot = [a, b]$
  - $\bot \triangledown [a, b] = [a, b]$
  - $[a_1, b_1] \triangledown [a_2, b_2] = [(a_2 < a_1)? -\infty : a_1, (b_1 < b_2)? \infty : b_1]$
- Examples
  - $[1,2] \triangledown [0,2] = [-\infty,2]$
  - $[0,2] \triangledown [1,2] = [0,2]$
  - $[2,3] \triangledown [4,6] = ???$

WIDENING FOR THE INTERVAL DOMAIN

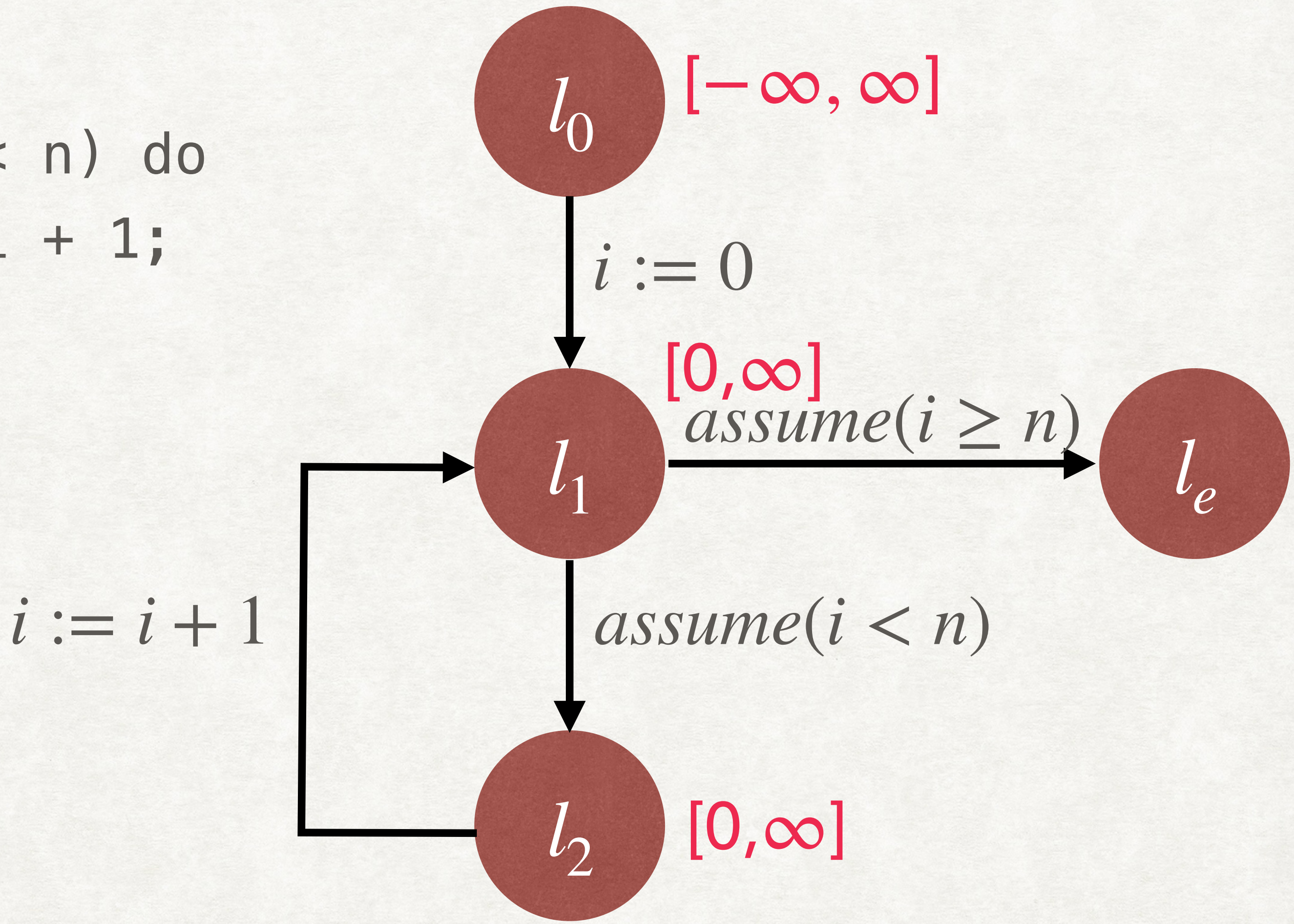
- We can define the widening operator for interval domain as follows:
  - $[a, b] \triangledown \bot = [a, b]$
  - $\bot \triangledown [a, b] = [a, b]$
  - $[a_1, b_1] \triangledown [a_2, b_2] = [(a_2 < a_1)? - \infty : a_1, (b_1 < b_2)? \infty : b_1]$
- Examples
  - $[1,2] \triangledown [0,2] = [-\infty, 2]$
  - $[0,2] \triangledown [1,2] = [0,2]$
  - $[2,3] \triangledown [4,6] = [2, \infty]$

KILDALL'S ALGORITHM WITH WIDENING

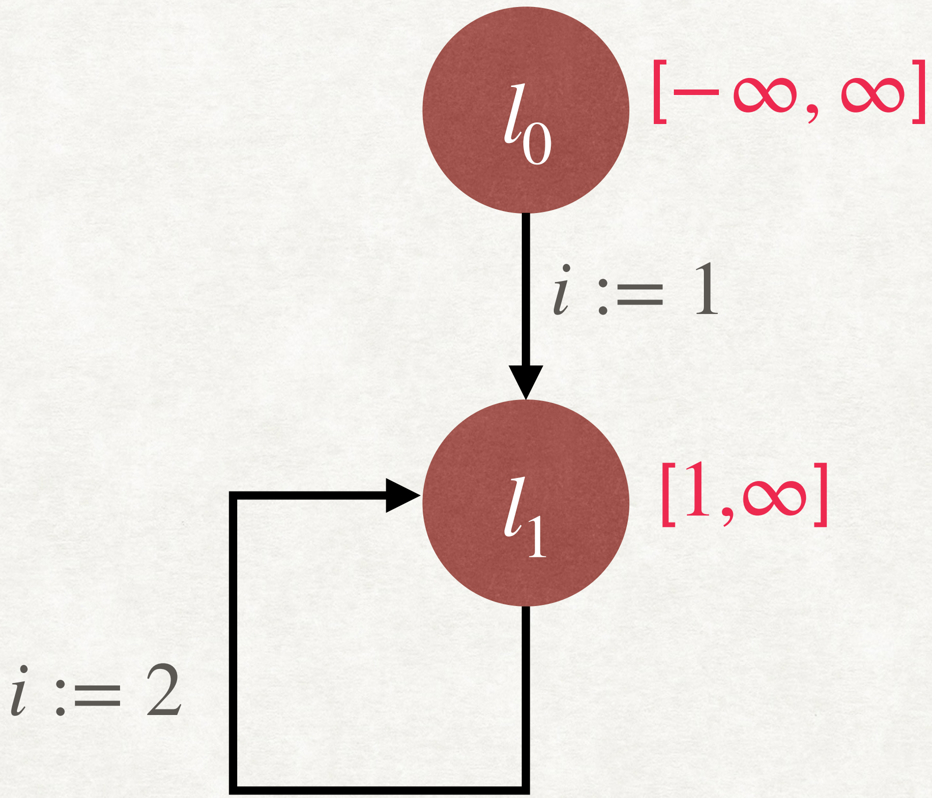
```
AbstractForwardPropagate(Γ_c,P)
  S := {l_0};
  μ̂_K(l_0) := α(P);
  μ̂_K(l) := ⊥, for l ∈ L\{l_0};
  while S ≠ ∅ do{
     l := Choose S;
     S := S \ {l};
     foreach (l,c,l′) ∈ T do{
        F := f̂_c(μ̂_K(l));
        if ¬(F ≤ μ̂_K(l′)) then{
           μ̂_K(l′) := μ̂_K(l′) ▽ F;
           S := S ∪ {l′};
        }
     }
  }
```

WIDENING EXAMPLE

```
i := 0;
while(i < n) do
    i := i + 1;
```



ANOTHER WIDENING EXAMPLE



NARROWING

- A narrowing function $\triangle : D \times D \rightarrow D$ on a poset $(D, \leq)$ satisfies the following properties:
  - $\forall x, y \in D \,.\, y \leq x \Rightarrow y \leq x \triangle y \leq x$
  - For a decreasing chain $x_0 \geq x_1 \geq \ldots$, the decreasing chain $y_0, y_1, \ldots$ where $y_0 = x_0$ and $y_n = y_{n-1} \triangle x_n$ eventually stabilizes.

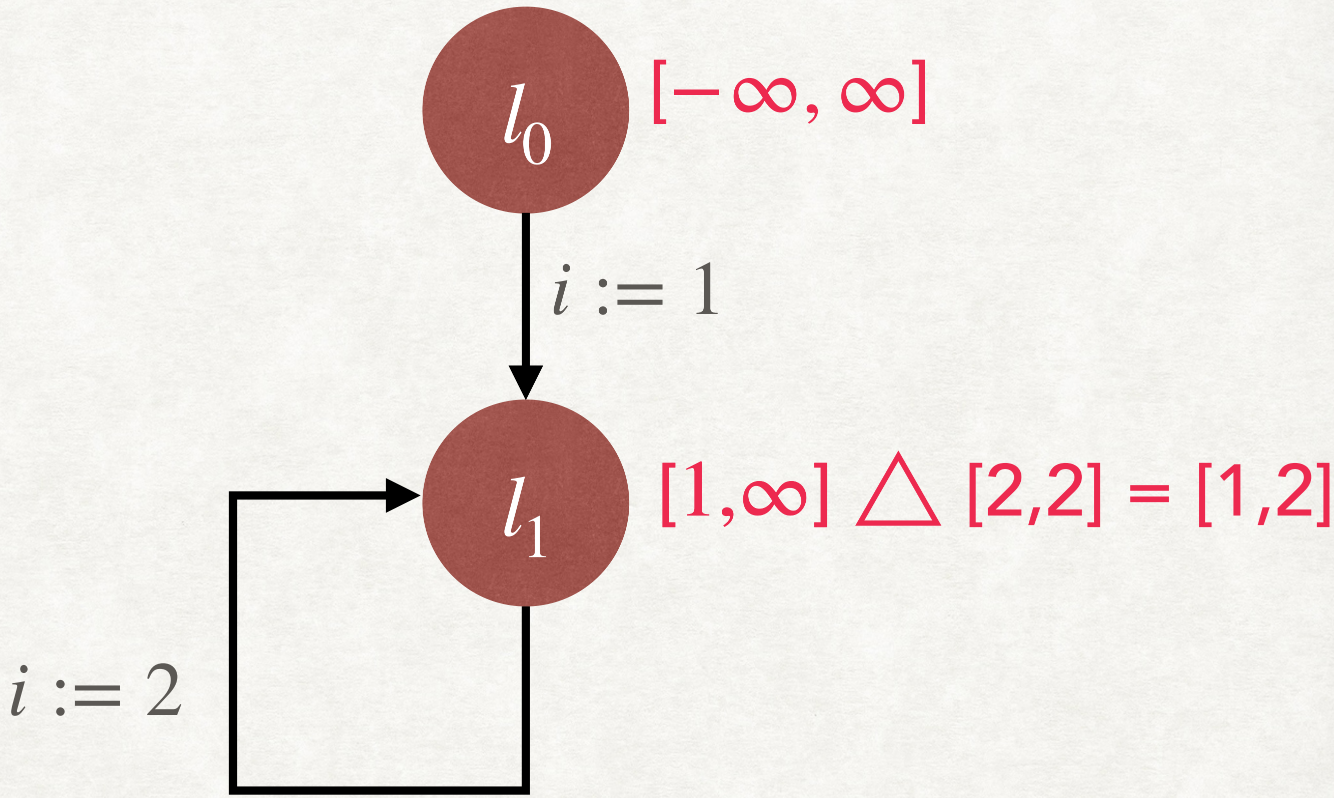
NARROWING FOR THE INTERVAL DOMAIN

- We can define the narrowing operator for interval domain as follows:
  - $[a, b] \bigtriangleup \bot = \bot$
  - $[a_1, b_1] \bigtriangleup [a_2, b_2] = [(a_1 = -\infty)?a_2 : a_1, (b_1 = \infty)?b_2 : b_1]$
- Examples
  - $[1,3] \bigtriangleup [1,2] =$
  - $[-\infty,6] \bigtriangleup [1,3] =$

NARROWING FOR THE INTERVAL DOMAIN

- We can define the narrowing operator for interval domain as follows:
  - $[a, b] \bigtriangleup \bot = \bot$
  - $[a_1, b_1] \bigtriangleup [a_2, b_2] = [(a_1 = -\infty)?a_2 : a_1, (b_1 = \infty)?b_2 : b_1]$
- Examples
  - $[1,3] \bigtriangleup [1,2] = [1,3]$
  - $[-\infty,6] \bigtriangleup [1,3] = [1,6]$

NARROWING EXAMPLE



Apply Narrowing pass after Widening

RELATIONAL DOMAINS

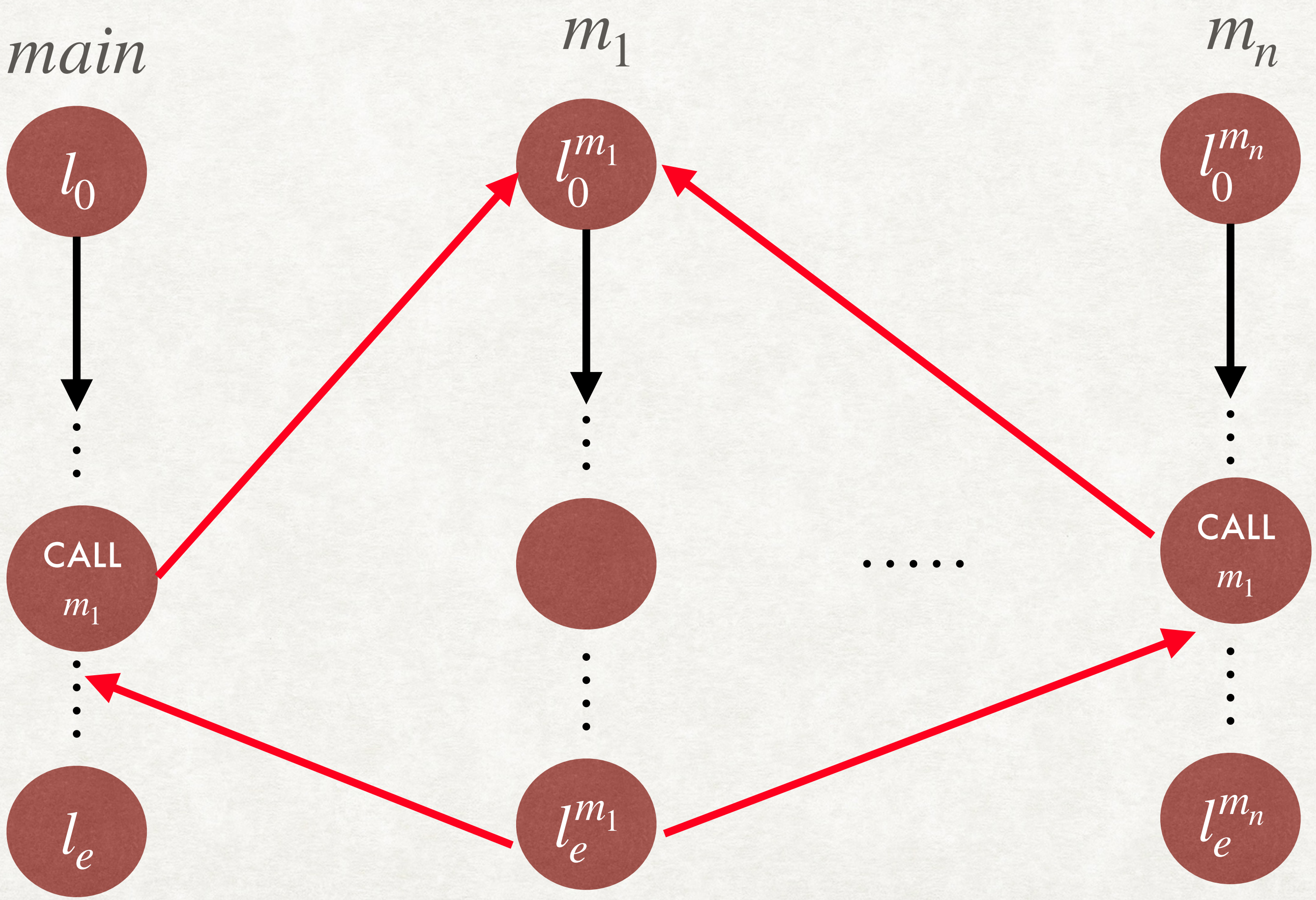
- Both the sign and the interval abstract domains are non-relational, i.e. they do not track relationships between variables.
- Relational domains track relationships between variables and are more powerful.
- Examples of relational domains
  - Karr's Domain: Tracks equalities between linear expressions (e.g. $x = 2y + z$)
  - Octagon Domain: Constraints of the form $\pm x \pm y \leq c$
  - Polyhedra Domain: Constraints of the form $c_1 x_1 + \ldots c_n x_n \leq c$

INTER-PROCEDURAL ABSTRACT INTERPRETATION

- For programs with multiple functions, we first consider the inter-procedural LTS:

INTER-PROCEDURAL ABSTRACT INTERPRETATION

- For programs with multiple functions, we first consider the inter-procedural LTS:



INTER-PROCEDURAL ABSTRACT INTERPRETATION

- Assuming that variable names are distinct across functions, function call and return statements can be replaced by assignments to parameters and return variables.
- However, the challenge is to only consider inter-procedurally valid paths.
- Naively applying AI on the inter-procedural LTS will result in highly imprecise analysis.

SHARIR AND PNUELI'S APPROACHES TO INTER-PROCEDURAL AI

- Call-Strings based approach
  - Change the abstract domain to also record the history of call-sites.
  - Since call-strings can be infinite in size, two practical approaches are also proposed: Approximate call-string method and Bounded call-string method.
- Functional approach
  - Maintain an abstract summary of every method which maps abstract value of input parameter(s) to abstract value of return variable.
  - Abstract summaries calculated on-the-fly during the analysis.

Micha Sharir and Amir Pnueli: Two approaches to interprocedural data flow analysis (1981)

LIMITATIONS OF ABSTRACT INTERPRETATION

- Precision depends upon the choice of the abstract domain.
- Hard to choose the right abstract domain: may depend on the program and the specification.
- Hard to interpret a negative result
    - If verification fails, then we don’t know whether the program is actually incorrect, or the abstract domain was not precise enough.
    - No counterexample is provided as output.