COURSE STRUCTURE

CONSTRAINT SOLVERS

- Propositional Logic, SAT solving, DPLL
- First-Order Logic, SMT
- First-Order Theories

DEDUCTIVE VERIFICATION

- Operational Semantics
- Strongest Post-condition, Weakest Pre-condition
- Hoare Logic

MODEL CHECKING AND OTHER VERIFICATION TECHNIQUES

- Abstract Interpretation
- Predicate Abstraction, CEGAR
- Property-directed Reachability

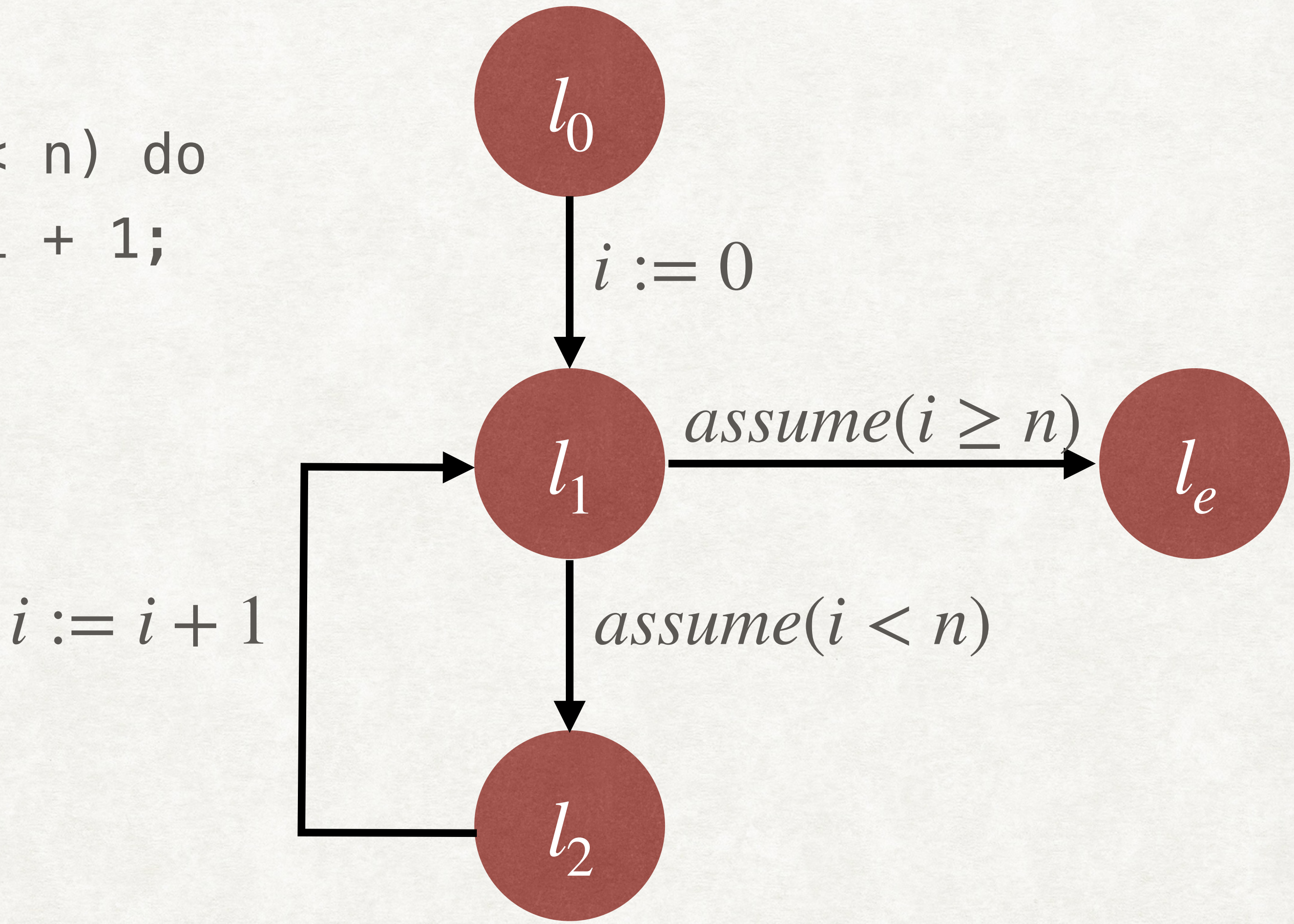
ABSTRACT INTERPRETATION

LABELLED TRANSITION SYSTEM

- We express the program c as a labelled transition system $\Gamma_c \equiv (V, L, l_0, l_e, T)$
  - V is the set of program variables
  - L is the set of program locations
  - l_0 is the start location
  - l_e is the end location
  - $T \subseteq L \times c \times L$ is the set of labelled transitions between locations.

EXAMPLE

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



PROGRAMS AS LTS

- There are various ways to construct the LTS of a program
  - We can use control flow graph
  - We can use basic paths as defined by the book (BM Chapter 5). A basic path is a sequence of instructions that begins at the start of the program or a loop head, and ends at a loop head or the end of the program.
- Program State (σ, l) consists of the values of the variables $(\sigma : V \to \mathbb{R})$ and the location.
- An execution is a sequence of program states, $(\sigma_0, l_0), (\sigma_1, l_1), \ldots, (\sigma_n, l_n)$, such that for all i, $0 \leq i \leq n-1$, $(l_i, c, l_{i+1}) \in T$ and $(\sigma_i, c) \hookrightarrow^* (\sigma_{i+1}, skip)$.
- A program satisfies its specification $\{P\}c\{Q\}$ if $\forall \sigma \in P$, for all executions $(\sigma, l_0), (\sigma_1, l_1), \ldots, (\sigma', l_e)$ of Γ_c, $\sigma' \in Q$.

INDUCTIVE ASSERTION MAP

- With each location, we associate a set of states which are reachable at that location in any execution.
  - $\mu : L \to \Sigma(V)$
- To express that such a map is an inductive assertion map, we will use Strongest Post-condition.
  - $\forall (l, c, l') \in T \,.\, sp(\mu(l), c) \to \mu(l')$
- Then, if μ is an inductive assertion map on Γ_c, the Hoare triple $\{P\}c\{Q\}$ is valid if $P \to \mu(l_0)$ and $\mu(l_e) \to Q$.

GENERATING THE INDUCTIVE ASSERTION MAP

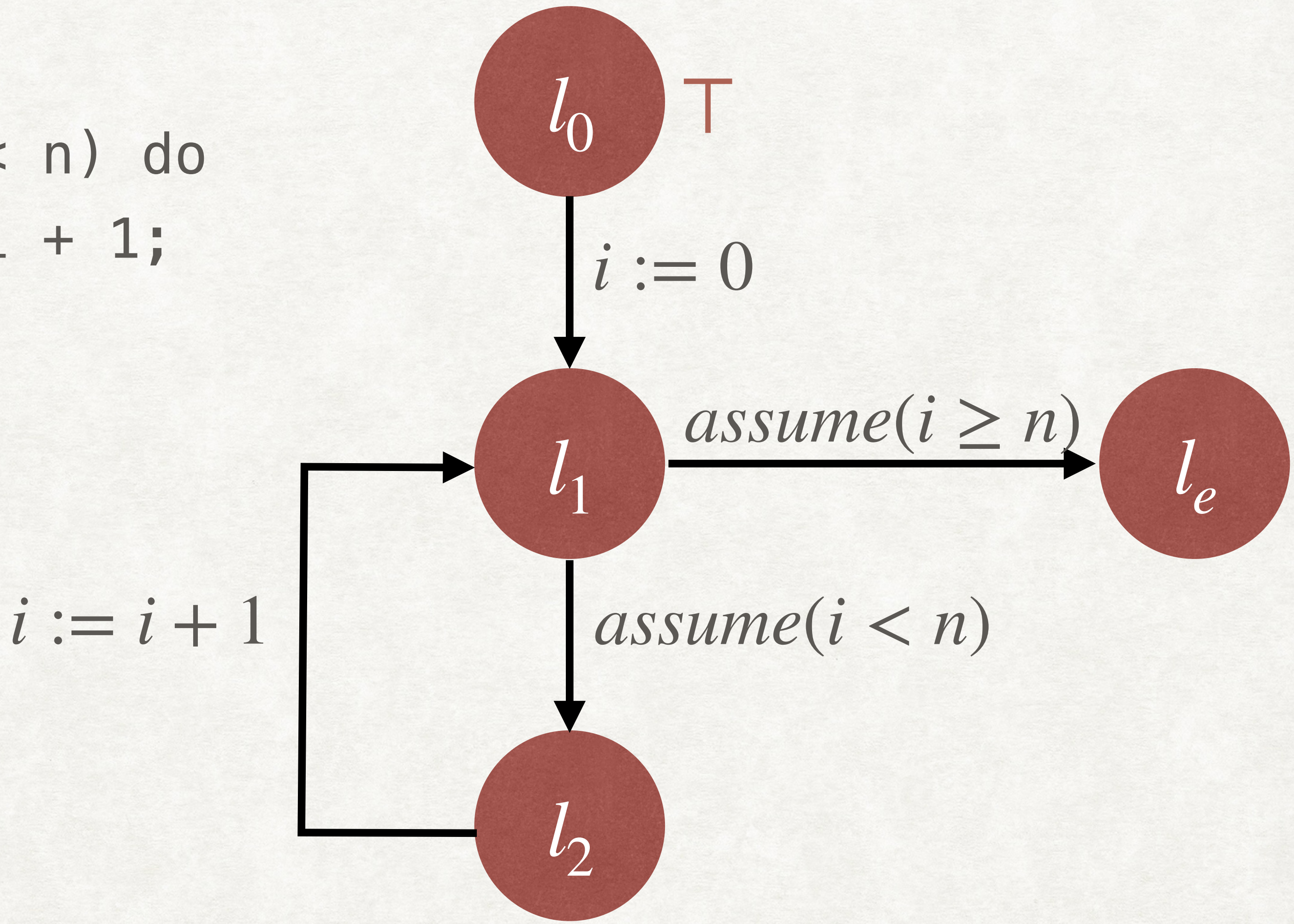
- We can express the inductive assertion map as a solution of a system of equations:
  - $X_{l_0} = P$
  - For all other locations $l \in L \backslash \{l_0\},\ X_l = \bigvee_{(l',c,l) \in T} sp(X_{l'}, c)$

GENERATING THE INDUCTIVE ASSERTION MAP

```
ForwardPropagate(Γ_c,P)
  S := {l_0};
  μ(l_0) := P;
  μ(l) := ⊥, for l ∈ L\{l_0};
  while S ≠ ∅ do{
    l := Choose S;
    S := S \ {l};
    foreach (l,c,l′) ∈ T do{
      F := sp(μ(l),c);
      if ¬(F → μ(l′)) then{
        μ(l′) := μ(l′) ∨ F;
        S := S ∪ {l′};
      }
    }
  }
```

EXAMPLE

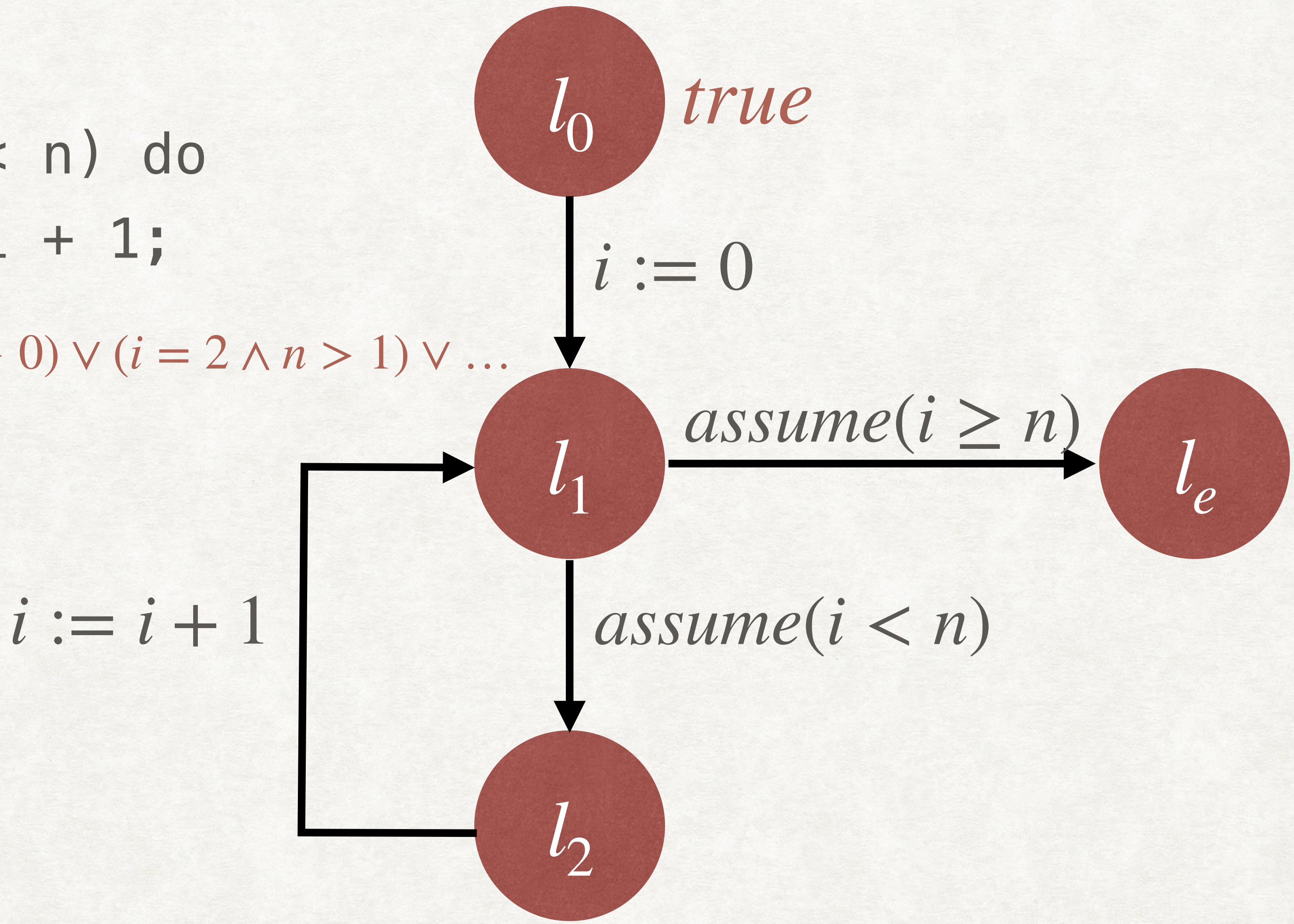
```
i := 0;
while(i < n) do
    i := i + 1;
```



EXAMPLE

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

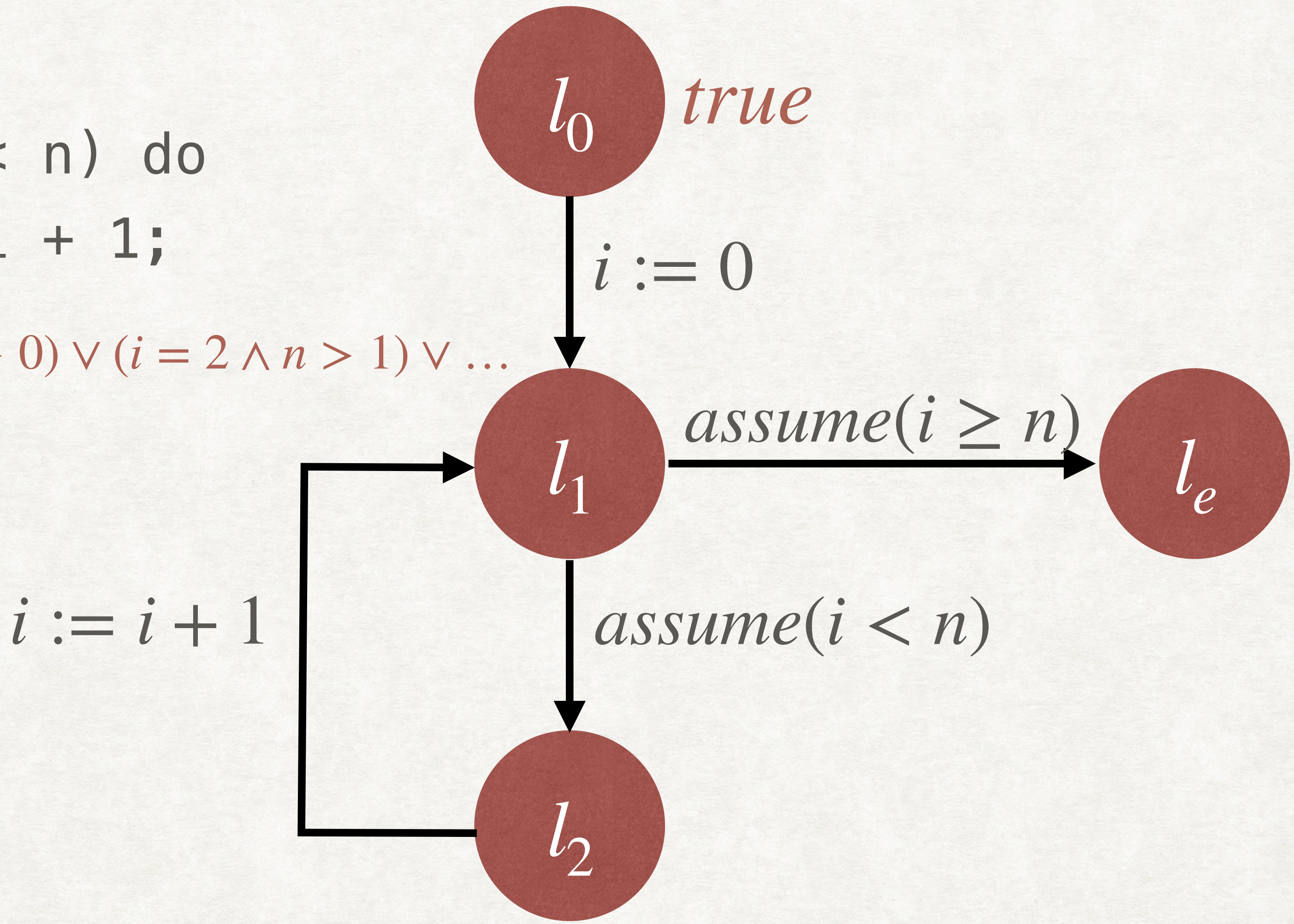
$(i = 0) \vee (i = 1 \wedge n > 0) \vee (i = 2 \wedge n > 1) \vee \ldots$



EXAMPLE

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

$(i = 0) \vee (i = 1 \wedge n > 0) \vee (i = 2 \wedge n > 1) \vee \ldots$



FORWARDPROPAGATE WILL NOT TERMINATE

ABSTRACT INTERPRETATION: OVERVIEW

- Instead of maintaining an arbitrary set of states at each location, maintain an artificially constrained set of states, coming from an abstract domain D.
  - $\hat{\mu} : L \to D$
- Let $States \triangleq V \to \mathbb{R}$ be the set of all possible concrete states.
  - Abstraction function, $\alpha : \mathbb{P}(States) \to D$
  - Concretization function, $\gamma : D \to \mathbb{P}(States)$
- $\hat{\mu}$ over approximates the set of states at every location.
  - For all locations l, $\gamma(\hat{\mu}(l)) \supseteq \mu(l)$
- Use abstract strongest post-condition operator $\hat{sp} : D \times c \to D$
  - $\gamma(\hat{sp}(d, c)) \supseteq sp(\gamma(d), c)$

GENERATING THE INDUCTIVE ASSERTION MAP

```
ForwardPropagate(Γ_c,P)
  S := {l_0};
  μ(l_0) := P;
  μ(l) := ⊥, for l ∈ L\{l_0};
  while S ≠ ∅ do{
    l := Choose S;
    S := S \ {l};
    foreach (l,c,l′) ∈ T do{
      F := sp(μ(l),c);
      if ¬(F → μ(l′)) then{
        μ(l′) := μ(l′) ∨ F;
        S := S ∪ {l′};
      }
    }
  }
```

ABSTRACT FORWARD PROPAGATE

```
AbstractForwardPropagate(Γ_c,P)
  S := {l_0};
  μ̂(l_0) := α(P);
  μ̂(l) := ⊥, for l ∈ L\{l_0};
  while S ≠ ∅ do{
      l := Choose S;
      S := S \ {l};
      foreach (l,c,l') ∈ T do{
          F := ŝp(μ̂(l),c);
          if ¬(F ≤ μ̂(l')) then{
              μ̂(l') := μ̂(l') ⊔ F;
              S := S ∪ {l'};
          }
      }
  }
```

ABSTRACT FORWARD PROPAGATE

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

Abstract Domain D

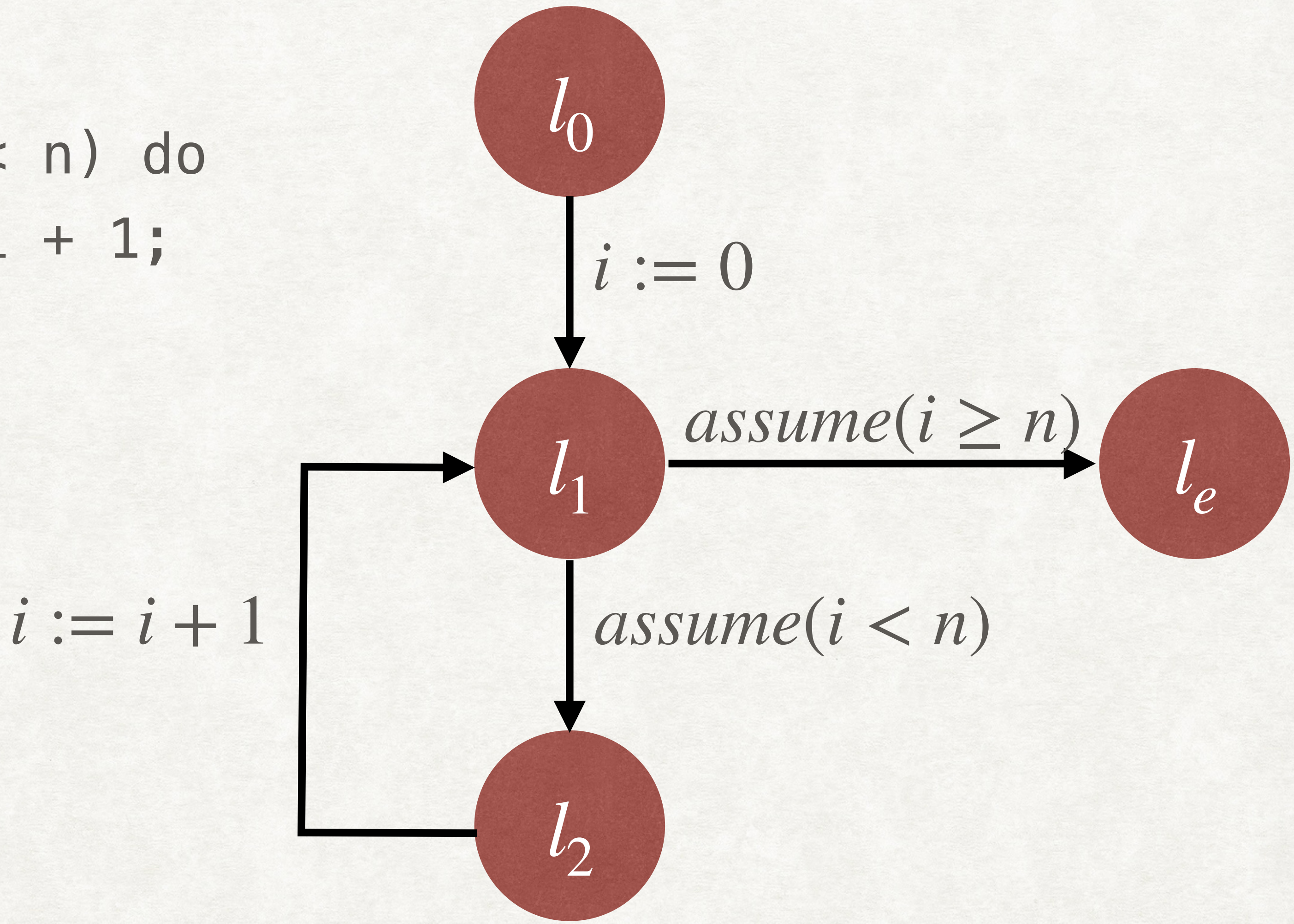
is a lattice $(D, \leq, \sqcup)$

ABSTRACT INTERPRETATION: OVERVIEW

- At the end, we will check whether $\hat{\mu}(l_e) \leq \alpha(Q)$.
  - Equivalently, $\gamma(\hat{\mu}(l_e)) \subseteq Q$

EXAMPLE

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



Suppose we want to prove the post-condition : $i \geq 0$

EXAMPLE

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

Sign Abstract Domain:

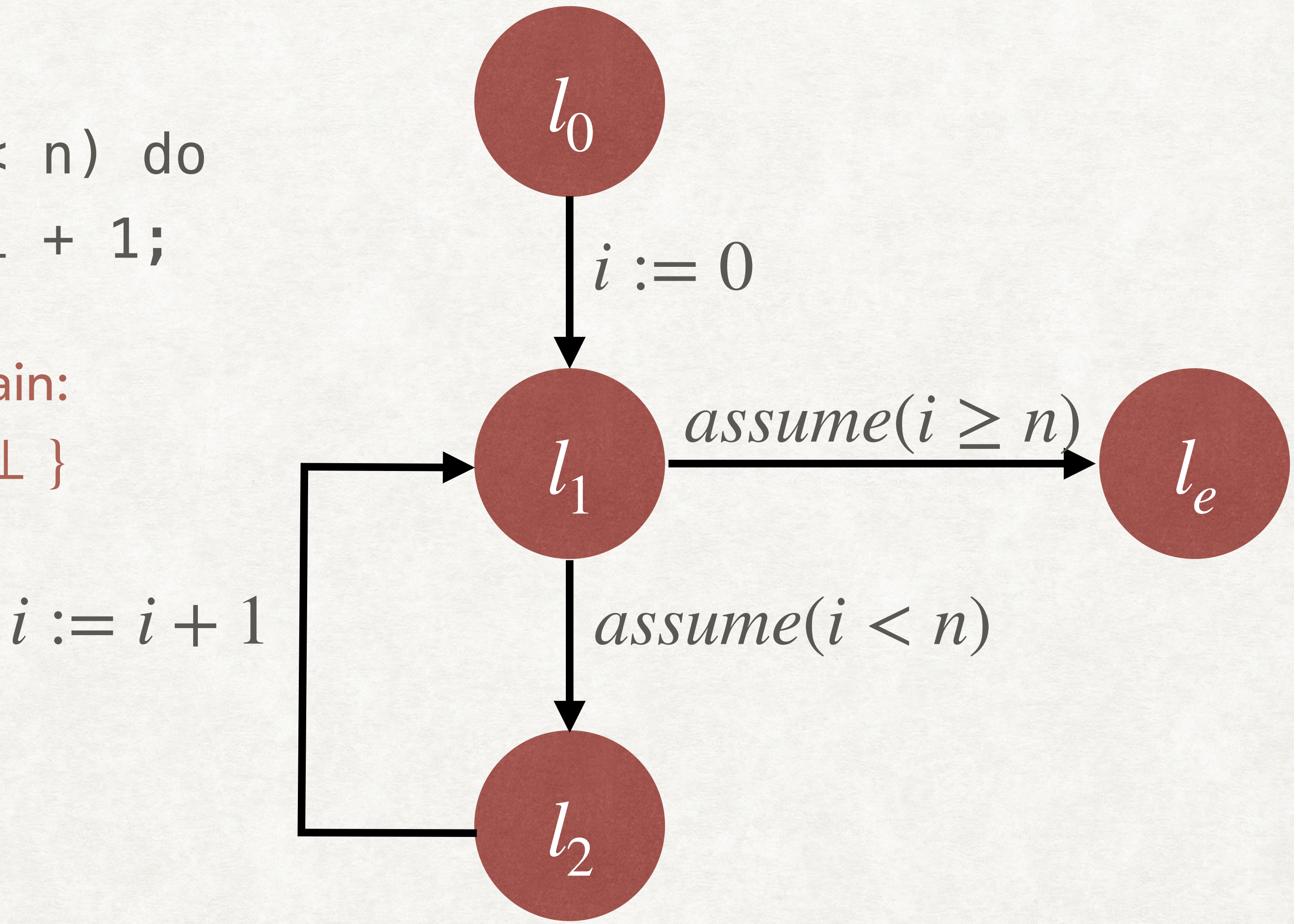
$D = \{ +-, +, -, \bot \}$

$\gamma(+-) = \top$

$\gamma(+) = i \geq 0$

$\gamma(-) = i < 0$

$\gamma(\bot) = \bot$



EXAMPLE

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

Sign Abstract Domain:

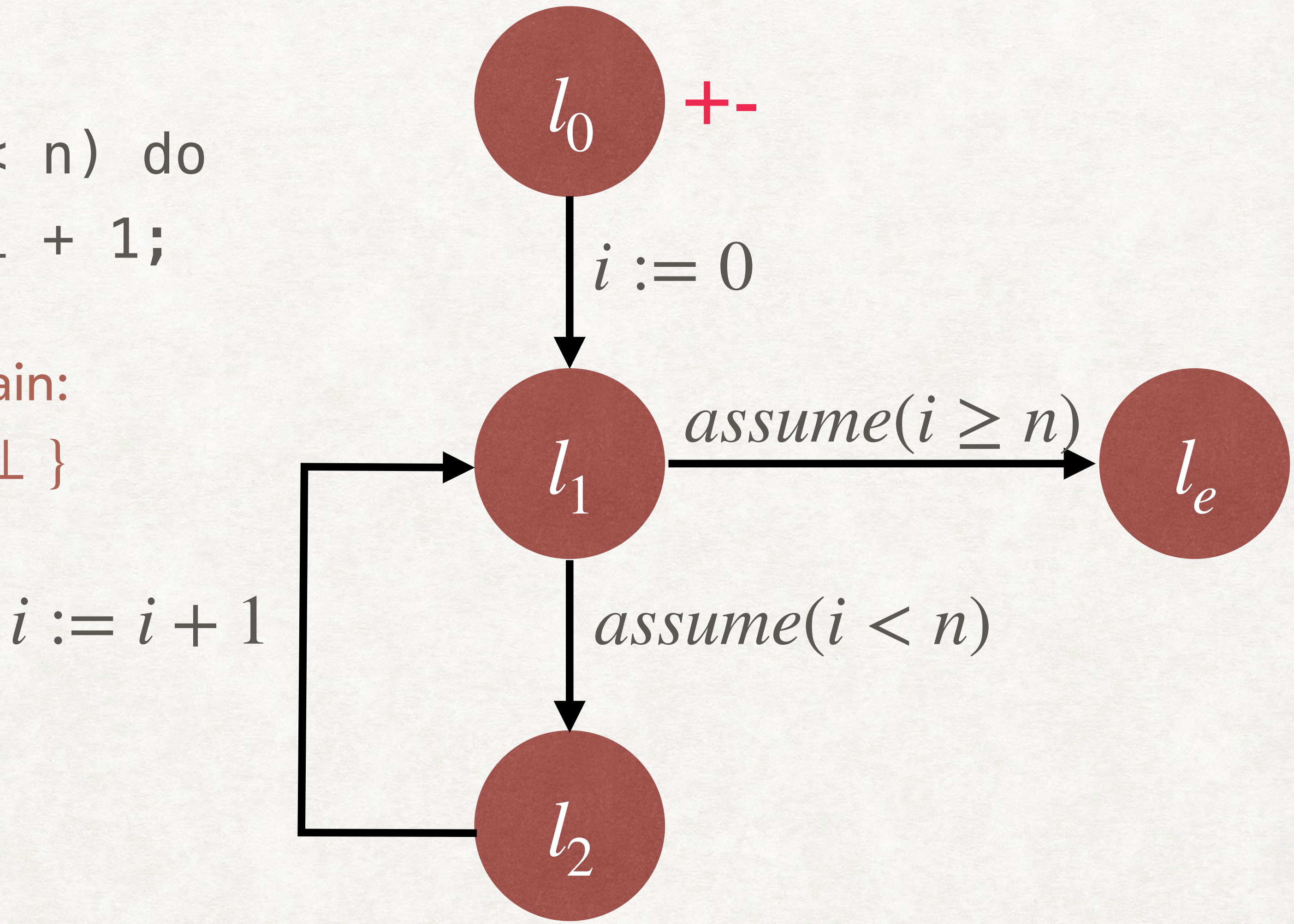
$D = \{+-, +, -, \perp\}$

$\gamma(+-) = \top$

$\gamma(+) = i \geq 0$

$\gamma(-) = i < 0$

$\gamma(\perp) = \perp$



EXAMPLE

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

Sign Abstract Domain:

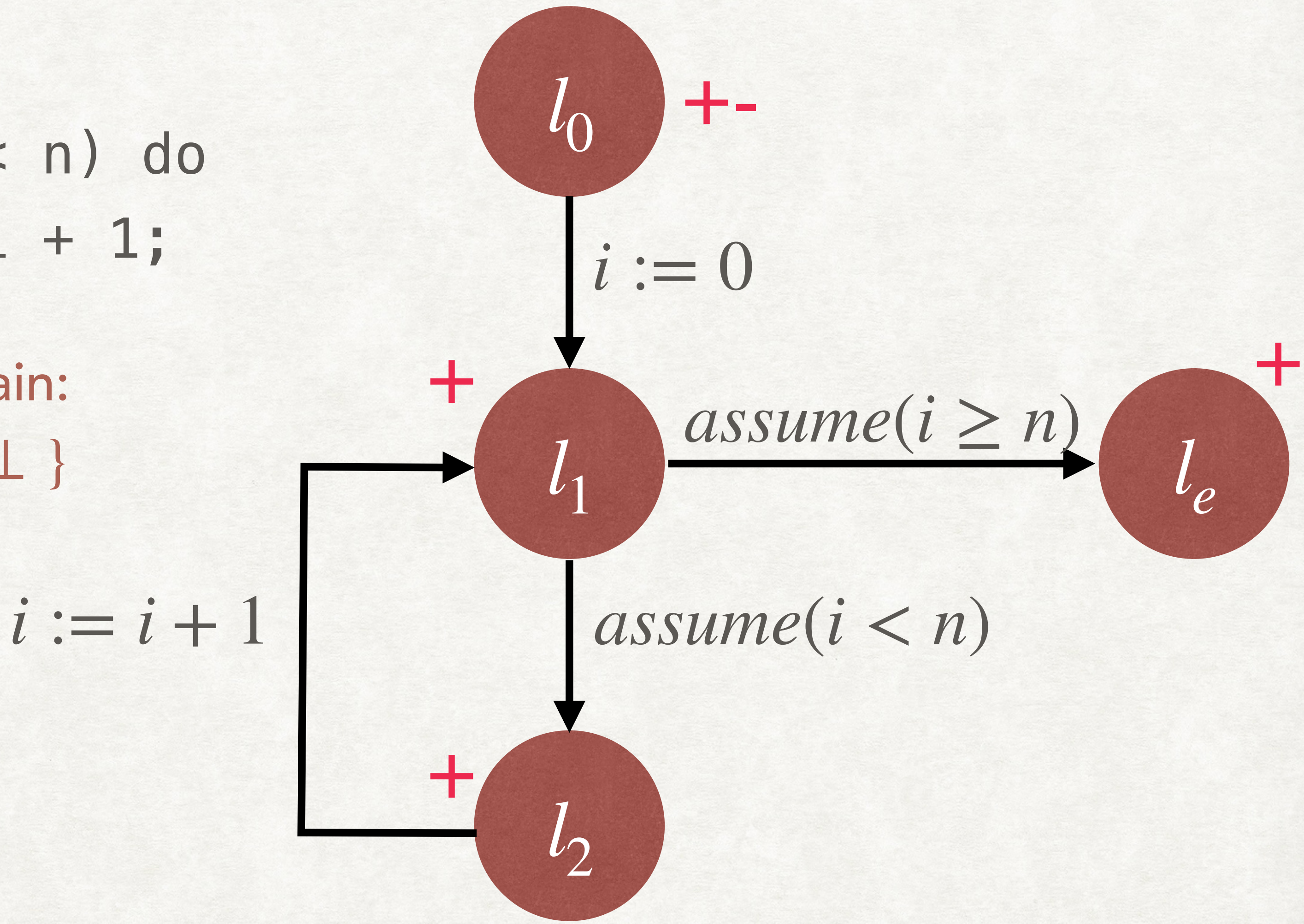
$D = \{ +-, +, -, \bot \}$

$\gamma(+-) = \top$

$\gamma(+) = i \geq 0$

$\gamma(-) = i < 0$

$\gamma(\bot) = \bot$



ABSTRACT INTERPRETATION: OVERVIEW

- Desirable properties of Abstract Interpretation
  - Soundness: $\hat{\mu}$ over approximates the set of states at every location.
  - Guaranteed termination of AbstractForwardPropagate
- We will use concepts from lattice theory to characterise the conditions required for these properties.

SNEAK PEEK

SOUNDNESS OF ABSTRACT INTERPRETATION

- An abstract interpretation $(D, \leq, \alpha, \gamma)$ is sound if:
  - $(D, \leq)$ is complete lattice.
  - $(\mathbb{P}(State), \subseteq) \underset{\gamma}{\overset{\alpha}{\rightleftarrows}} (D, \leq)$ is a Galois Connection.
  - $\hat{sp}$ is a consistent abstraction of sp.

SNEAK PEEK

GUARANTEED TERMINATION OF ABSTRACT FORWARD PROPAGATE

- AbstractForwardPropagate on abstract domain $(D, \leq)$ is guaranteed to terminate if:
  - $(D, \leq)$ is a complete lattice.
  - $\hat{sp}$ is monotonic.
  - $(D, \leq)$ satisfies the ascending chain condition.

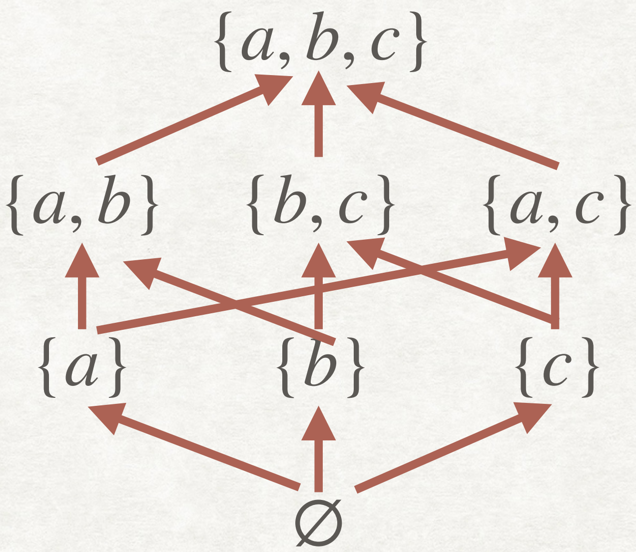
PARTIAL ORDER

- Given a set D, a binary relation $\leq \subseteq D \times D$ is a partial order on D if
  - $\leq$ is reflexive: $\forall d \in D \,.\, d \leq d$
  - $\leq$ is anti-symmetric: $\forall d, d' \in D \,.\, d \leq d' \wedge d' \leq d \rightarrow d = d'$
  - $\leq$ is transitive: $\forall d_1, d_2, d_3 \in D, d_1 \leq d_2 \wedge d_2 \leq d_3 \rightarrow d_1 \leq d_3$
- Examples
  - $\leq$ on $\mathbb{N}$ is a partial order.
  - Given a set S, $\subseteq$ on $\mathbb{P}(S)$ is a partial order.

PARTIAL ORDER - EXAMPLES

$$S = \{a, b, c\}$$

$$\mathbb{P}(S) = \{\varnothing, \{a\}, \{b\}, \{c\}, \{a, b\}, \{b, c\}, \{a, c\}, \{a, b, c\}\}$$

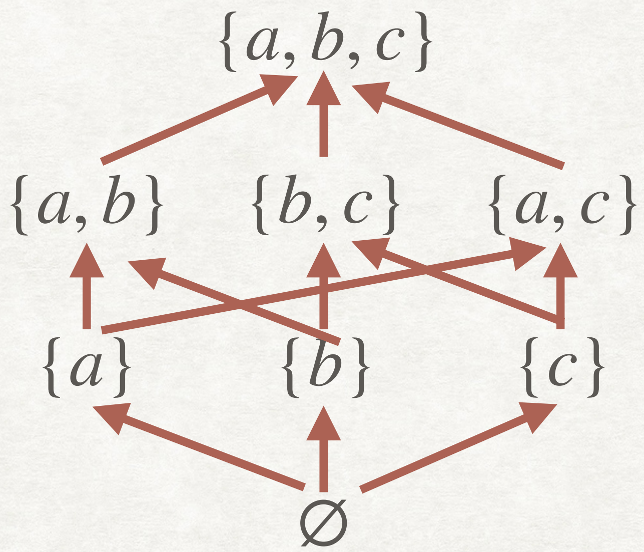


Partially Ordered Set: $(\mathbb{P}(S), \subseteq\ )$

PARTIAL ORDER - EXAMPLES

$$S = \{a, b, c\}$$

$$\mathbb{P}(S) = \{\varnothing, \{a\}, \{b\}, \{c\}, \{a, b\}, \{b, c\}, \{a, c\}, \{a, b, c\}\}$$



Hasse diagram:

- Doesn't show reflexive edges (self-loops)
- Doesn't show transitive edges

Partially Ordered Set: $(\mathbb{P}(S), \subseteq)$

PARTIAL ORDER - MORE EXAMPLES

- Which of the following are partially ordered sets (posets)?
  - $(\mathbb{N} \times \mathbb{N}, \{(a, b), (c, d) \mid a \leq c\})$
  - $(\mathbb{N} \times \mathbb{N}, \{(a, b), (c, d) \mid a \leq c \wedge b \leq d\})$
  - $(\mathbb{N} \times \mathbb{N}, \{(a, b), (c, d) \mid a \leq c \vee b \leq d\})$

LEAST UPPER BOUND

- Given a poset $(D, \leq)$ and $X \subseteq D$, $u \in D$ is called an upper bound on X if $\forall x \in X . x \leq u$.
  - $u \in D$ is called the least upper bound (lub) of X, if u is an upper bound of X, and for every other upper bound u' of X, $u \leq u'$.
  - We use the notation $\sqcup X$ to denote the least upper bound of X. Also called the join of X.
  - Exercise: Prove that the least upper bound, if it exists, is unique.

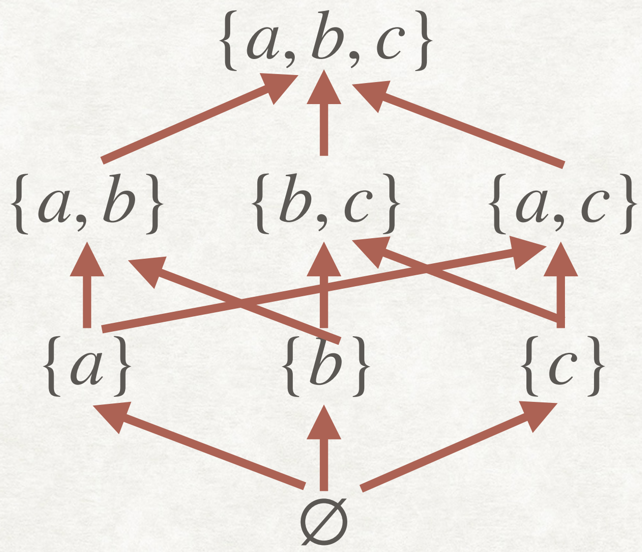
GREATEST LOWER BOUND

- Given a poset $(D, \leq)$ and $X \subseteq D$, $l \in D$ is called a lower bound on X if $\forall x \in X \,.\, l \leq x$.
  - $l \in D$ is called the greatest lower bound (glb) of X, if l is a lower bound of X, and for every other lower bound l', $l' \leq l$.
  - We use the notation $\sqcap X$ to denote the greatest lower bound of X. Also called the meet of X.
  - Homework: Prove that the greatest lower bound, if it exists, is unique.

LUB - EXAMPLE

$$S = \{a, b, c\}$$

$$\mathbb{P}(S) = \{\varnothing, \{a\}, \{b\}, \{c\}, \{a, b\}, \{b, c\}, \{a, c\}, \{a, b, c\}\}$$



- Consider $X = \{\{a\}, \{b\}\}$
- $\{a, b\}, \{a, b, c\}$ are both upper bounds of X
- $\{a, b\}$ is the least upper bound.

LATTICE

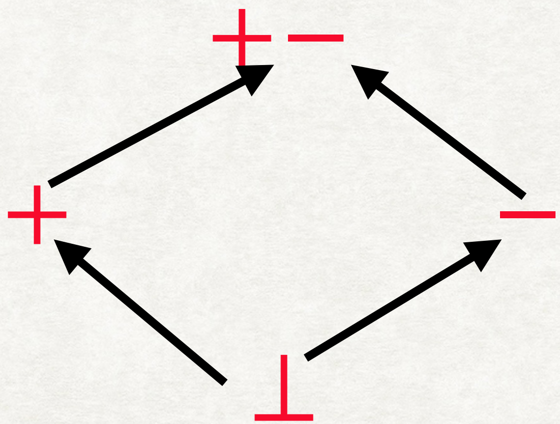
- A lattice is a poset $(D, \leq)$ such that $\forall x, y \in D$, $x \sqcup y$ and $x \sqcap y$ exist.
- A complete lattice is a lattice such that $\forall X \subseteq D$, $\sqcup X$ and $\sqcap X$ exists.
- Example: $(\mathbb{P}(S), \subseteq)$ is a complete lattice.

LATTICE - MORE EXAMPLES

- What is the simplest example of a poset that is not a lattice?
  - $(\{a, b\}, \{(a, a), (b, b)\})$
- What is an example of a lattice which is not a complete lattice?
  - $(\mathbb{N}, \leq)$

LATTICE - MORE EXAMPLES

- What is the simplest example of a poset that is not a lattice?
  - $(\{a, b\}, \{(a, a), (b, b)\})$
- What is an example of a lattice which is not a complete lattice?
  - $(\mathbb{N}, \leq)$
- Sign Lattice:



SOME PROPERTIES OF LATTICES

- $(D, \leq)$ is a lattice, $x, y, z \in D$
  - If $x \leq y$, then $x \sqcup y = y$ and $x \sqcap y = x$.
  - $x \sqcup x = x$ and $x \sqcap x = x$
  - $(x \sqcup y) \sqcup z = x \sqcup (y \sqcup z) = \sqcup \{x, y, z\}$
  - If D is finite, then D is also a complete lattice.

MINIMUM AND MAXIMUM

- Given a poset $(D, \leq)$, $x \in D$ is called the minimum element if $\forall y \in D \,.\, x \leq y$.
  - Also called the bottom element. Denoted by $\bot$.
- Given a poset $(D, \leq)$, $x \in D$ is called the maximum element if $\forall y \in D \,.\, y \leq x$.
  - Also called the top element. Denoted by $\top$.
- Complete lattices are guaranteed to have top and bottom elements.
  - $\sqcup D = \top \,, \sqcap D = \bot$
  - $\sqcup \varnothing = \bot \,, \sqcap \varnothing = \top$

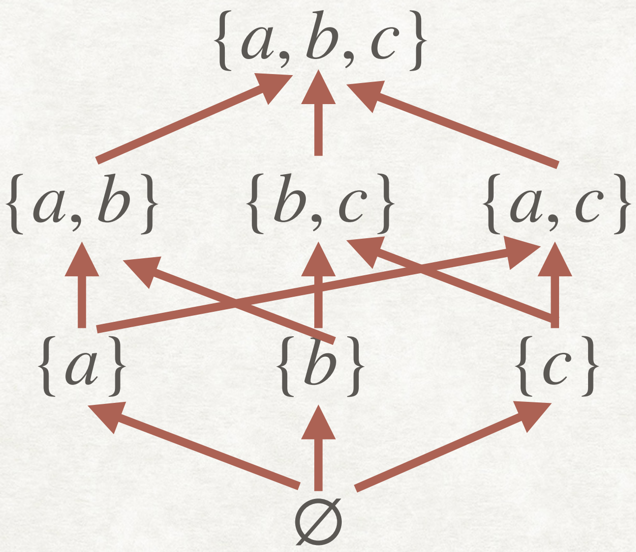
MONOTONIC FUNCTIONS

- Given two posets $(D_1, \leq_1)$ and $(D_2, \leq_2)$, function $f : D_1 \to D_2$ is called monotonic (or order-preserving) if
  - $\forall x, y \in D_1 . x \leq_1 y \to f(x) \leq_2 f(y)$
- In the special case when $D_1 = D_2 = D$, $f : D \to D$ is monotonic if
  - $\forall x, y \in D . x \leq y \to f(x) \leq f(y)$

MONOTONIC FUNCTIONS - EXAMPLE

$$S = \{a, b, c\}$$

$$\mathbb{P}(S) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{b, c\}, \{a, c\}, \{a, b, c\}\}$$



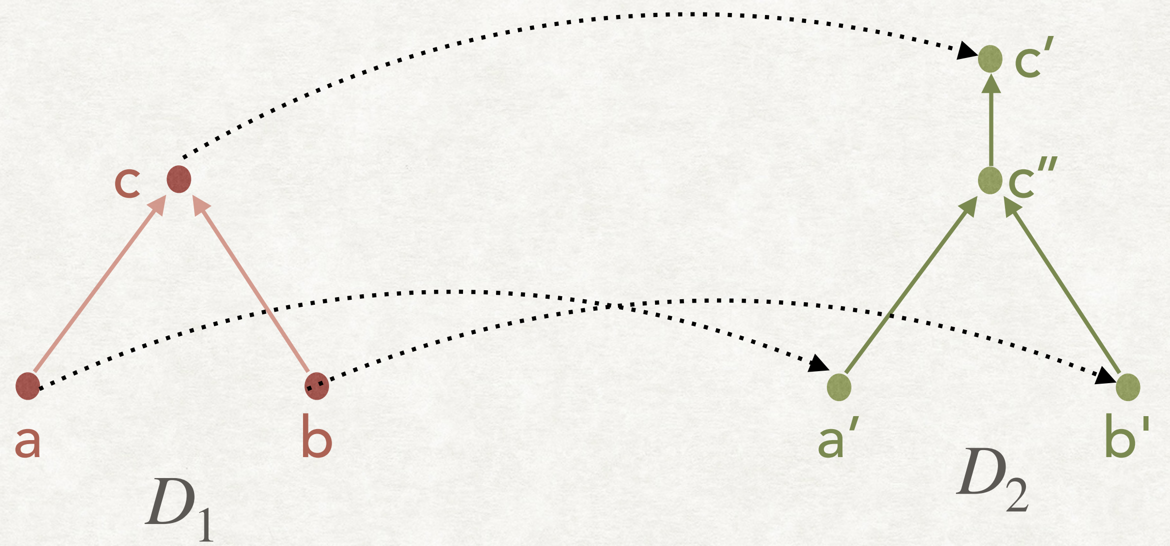
- Consider $f : \mathbb{P}(S) \to \mathbb{P}(S)$, $f(X) = X \cup \{a\}$.
  - f is monotonic.
- What about $f(X) = X \cap \{a\}$?
- Example of a non-monotonic function on $\mathbb{P}(S)$?

JOIN PRESERVING

- Given posets $(D_1, \leq_1)$ and $(D_2, \leq_2)$, a monotonic function $f : D_1 \to D_2$, and $S \subseteq D_1$, if $\sqcup_1 S$ and $\sqcup_2 f(S)$ exist, then $\sqcup_2 f(S) \leq_2 f(\sqcup_1 S)$.

JOIN PRESERVING

- Given posets $(D_1, \leq_1)$ and $(D_2, \leq_2)$, a monotonic function $f : D_1 \to D_2$, and $S \subseteq D_1$, if $\sqcup_1 S$ and $\sqcup_2 f(S)$ exist, then $\sqcup_2 f(S) \leq_2 f(\sqcup_1 S)$.



JOIN PRESERVING

- Given posets $(D_1, \leq_1)$ and $(D_2, \leq_2)$, a monotonic function $f : D_1 \to D_2$, and $S \subseteq D_1$, if $\sqcup_1 S$ and $\sqcup_2 f(S)$ exist, then $\sqcup_2 f(S) \leq_2 f(\sqcup_1 S)$.

Proof:

JOIN PRESERVING

- Given posets $(D_1, \leq_1)$ and $(D_2, \leq_2)$, a monotonic function $f : D_1 \rightarrow D_2$, and $S \subseteq D_1$, if $\sqcup_1 S$ and $\sqcup_2 f(S)$ exist, then $\sqcup_2 f(S) \leq_2 f(\sqcup_1 S)$.

Proof: Let $u = \sqcup_1 S$.

Then $\forall x \in S \,.\, x \leq_1 u$. This implies that $\forall x \in S \,.\, f(x) \leq_2 f(u)$.

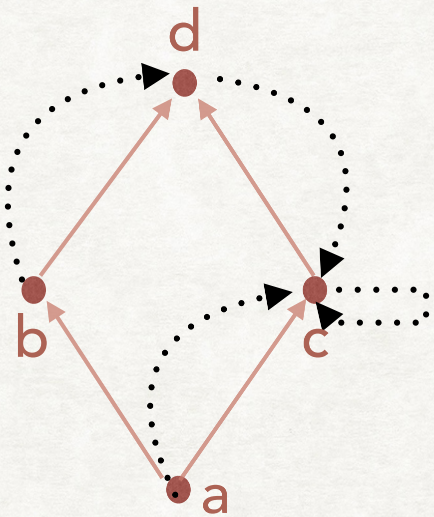
Thus $f(u)$ is an upper bound of $f(S)$.

Hence, $\sqcup_2 f(S) \leq_2 f(u)$.

FIXPOINTS

- A fixpoint of a function $f : D \to D$ is an element $x \in D$ such that $f(x) = x$.
- A pre-fixpoint of a function $f : D \to D$ is an element $x \in D$ such that $x \leq f(x)$.
- A post-fixpoint of a function $f : D \to D$ is an element $x \in D$ such that $f(x) \leq x$.

FIXPOINTS - EXAMPLE



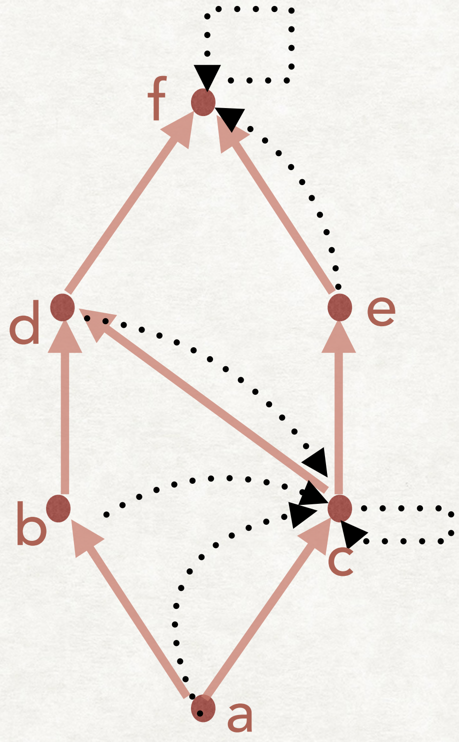
- Fixpoint : c
- Pre-fixpoints : a,b,c
- Post-fixpoint : c,d

KNASTER-TARSKI FIXPOINT THEOREM

- Let $(D, \leq)$ be a complete lattice, and $f : D \to D$ be a monotonic function on $(D, \leq)$. Then:
  - f has at least one fixpoint.
  - f has a least fixpoint (lfp), which is the same as the glb of the set of post-fixpoints of f, and a greatest fixpoint (gfp) which is the same as the lub of the set of pre-fixpoints of f.
  - The set of fixpoints of f itself forms a complete lattice under $\leq$.

KNASTER-TARSKI FIXPOINT THEOREM

ILLUSTRATION



- Complete Lattice
- Monotonic Function
- Pre-fixpoints: a,c,e,f
- Post-fixpoints: c,d,f
- Fixpoints: c,f

PROOF OF KNASTER-TARSKI THEOREM

- $Pre = \{x \mid x \leq f(x)\}$
  - We will show that $\sqcup Pre$ is a fixpoint.
  - Notice that Pre cannot be empty. Why?

Proof:

PROOF OF KNASTER-TARSKI THEOREM

- $Pre = \{x \mid x \leq f(x)\}$
  - We will show that $\sqcup Pre$ is a fixpoint.
  - Notice that Pre cannot be empty. Why?

Proof: Let $u = \sqcup Pre$.

Consider $x \in Pre$. Then, $x \leq u$. Hence, $f(x) \leq f(u)$. Since $x \leq f(x)$, we have $x \leq f(u)$. Thus, $f(u)$ is an upper bound of Pre. Since u is the least upper bound of Pre, we have $u \leq f(u)$.

$u \leq f(u) \Rightarrow f(u) \leq f(f(u))$. Hence, $f(u)$ is a pre-fixpoint. Therefore, $f(u) \leq u$.

This proves that $u = f(u)$.

PROOF OF KNASTER-TARSKI THEOREM

- $Pre = \{x \,|\, x \leq f(x)\}$
  - $\sqcup Pre$ is the greatest fixpoint.

Proof: Consider another fixpoint g.

Then, g is also a pre-fixpoint. Hence, $g \leq \sqcup Pre$.

PROOF OF KNASTER-TARSKI THEOREM

- $Post = \{x \mid f(x) \leq x)\}$
  - $\sqcap Post$ is a fixpoint of f.
  - $\sqcap Post$ is the least fixpoint.

HOMEWORK

PROOF OF KNASTER-TARSKI THEOREM

- $P = \{x \mid f(x) = x\}$
  - We will show that $(P, \leq)$ is a complete lattice.

Proof Sketch: $(P, \leq)$ is a partial order.

Let $X \subseteq P$. Let u be the $\sqcup X$ in D. Consider $U = \{a \in D \mid u \leq a\}$

Then $(U, \leq)$ is a complete lattice. [Prove this.]

Further, $f(U) \subseteq U$. [Prove this.]

Hence, f is a monotonic function on complete lattice $(U, \leq)$. By previous part of Knaster-Tarski Theorem, the least fixpoint of f in U exists.

Let v be the least fixpoint of f in U. Then v is the least upper bound of X in P. [Prove this.]

Similarly, we can show that $\sqcap X$ also exists in P. [Prove this.]

CHAINS

- Given a poset $(D, \leq)$, $C \subseteq D$ is called a chain if $\forall x, y \in C \,.\, x \leq y \vee y \leq x$.
- A poset $(D, \leq)$ satisfies the ascending chain condition, if for all sequences $x_1 \leq x_2 \leq \ldots$, $\exists k \,.\, \forall n \geq k \,.\, x_n = x_k$.
    - We say that the sequence stabilizes to x_k.
- A poset $(D, \leq)$ satisfies the descending chain condition, if for all sequences $x_1 \geq x_2 \geq \ldots$, $\exists k \,.\, \forall n \geq k \,.\, x_n = x_k$.
    - A poset that satisfies the descending chain condition is also called well-ordered.
    - Example: Is $(\mathbb{N}, \leq)$ well-ordered?
- Poset $(D, \leq)$ is said to have finite height if it satisfies both the ascending and descending chain conditions.
    - Example: Does $(\mathbb{N}, \leq)$ have finite height?

COMPUTING LFP

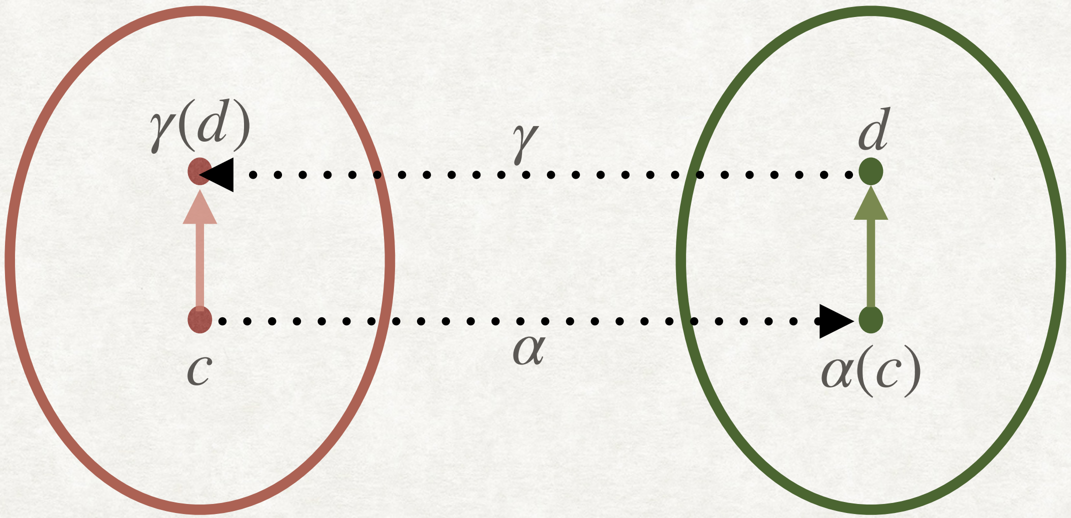
- Consider a complete lattice $(D, \leq)$ and a monotonic function $f : D \rightarrow D$.
- Consider the sequence $\perp, f(\perp), f^2(\perp), f^3(\perp), \ldots$
  - If it stabilizes, it will converge to a fixpoint of f.
  - Further, this fixpoint will be the least fixpoint of f.
- Hence, if $(D, \leq)$ satisfies the ascending chain condition, we can compute $lfp(f)$ by finding the stable value of $\perp, f(\perp), f^2(\perp), f^3(\perp), \ldots$
- Homework: If $a \in Pre$, and the sequence $a, f(a), f^2(a), \ldots$ stabilizes, it will converge to the least fixpoint greater than a (denoted by $lfp_a(f)$).

GALOIS CONNECTION

- Given posets $(C, \leq_1)$ and $(D, \leq_2)$, a pair of functions (α, γ), $\alpha : C \rightarrow D$ and $\gamma : D \rightarrow C$ is called a Galois connection if
  - $\forall c \in C \,.\, \forall d \in D \,.\, \alpha(c) \leq_2 d \Leftrightarrow c \leq_1 \gamma(d)$
- Also written as: $(C, \leq_1) \underset{\gamma}{\overset{\alpha}{\rightleftarrows}} (D, \leq_2)$

GALOIS CONNECTION

- Given posets $(C, \leq_1)$ and $(D, \leq_2)$, a pair of functions (α, γ), $\alpha : C \to D$ and $\gamma : D \to C$ is called a Galois connection if
  - $\forall c \in C \,.\, \forall d \in D \,.\, \alpha(c) \leq_2 d \Leftrightarrow c \leq_1 \gamma(d)$
- Also written as: $(C, \leq_1) \underset{\gamma}{\overset{\alpha}{\rightleftarrows}} (D, \leq_2)$



PROPERTIES OF GALOIS CONNECTION

- $c \leq_1 \gamma(\alpha(c))$
  - Proof: Consider $d = \alpha(c)$. Then, $\alpha(c) \leq d$. By definition of Galois connection, $c \leq \gamma(d)$. Hence, $c \leq \gamma(\alpha(c))$.
- $\alpha(\gamma(d)) \leq_2 d$
  - Proof: Homework.

PROPERTIES OF GALOIS CONNECTION

- α is monotonic.
  - Proof: Consider $c_1, c_2 \in C$ such that $c_1 \leq_1 c_2$.
  - We know that $c_2 \leq \gamma(\alpha(c_2))$. By transitivity, $c_1 \leq \gamma(\alpha(c_2))$. Hence, by definition of Galois connection, $\alpha(c_1) \leq_2 \alpha(c_2)$.
- γ is monotonic.
  - Proof: Homework.

GALOIS CONNECTION AND PROGRAM STATES

- Recall: $States \triangleq V \to \mathbb{R}$. The concrete domain C will be $(\mathbb{P}(States), \subseteq)$.
- The abstract domain D will be a collection of artificially constrained set of states. We can represent this as $D \subseteq C$.
- The abstraction function α will map $c \in C$ to the smallest set $d \in D$ such that $c \subseteq d$.
- The concretization function γ will simply be $\gamma(d) = d$.
- Is this a Galois Connection? We have to show that $\alpha(c) \subseteq d \Leftrightarrow c \subseteq \gamma(d)$.
  - Suppose $\alpha(c) \subseteq d$. Now, $c \subseteq \alpha(c)$ and $\gamma(d) = d$. Hence, $c \subseteq \gamma(d)$.
  - Suppose $c \subseteq \gamma(d)$. Hence, $c \subseteq d$. Now, $\alpha(c)$ is the smallest set in D containing c. Hence, $\alpha(c) \subseteq d$.

GALOIS CONNECTION AND PROGRAM STATES

EXAMPLE

- Assume that $V = \{v\}$.
  - Hence, $State = \mathbb{R}$, The concrete domain C is$(\mathbb{P}(\mathbb{R}), \subseteq)$
- Sign Abstract Domain: $D = \{ +-, +, -, \perp \}$.
  - $+- \triangleq \mathbb{R}$
  - $+ \triangleq \{n \in \mathbb{R} \mid n \geq 0\}$
  - $- \triangleq \{n \in \mathbb{R} \mid n < 0\}$
  - $\perp \triangleq \emptyset$
- Clearly $D \subseteq C$.

GALOIS CONNECTION AND PROGRAM STATES

EXAMPLE

- Define the Galois Connection: $(\mathbb{P}(\mathbb{R}), \subseteq) \underset{\gamma}{\overset{\alpha}{\rightleftarrows}} (D, \subseteq)$
    - $\alpha(c) = +$ if $min(c) \geq 0$
    - $\alpha(c) = -$ if $max(c) < 0$
    - $\alpha(\emptyset) = \bot$
    - Otherwise, $\alpha(c) = + -$.
    - $\gamma(d) = d$.
- Example: $\alpha(\{3,5\}) = +$, $\alpha(\{3,6,-1,0\}) = + -$

ONTO GALOIS CONNECTION

- The abstraction function α will map $c \in C$ to the smallest set $d \in D$ such that $c \subseteq d$.
- The concretization function γ will simply be $\gamma(d) = d$.
- Notice that $\alpha(\gamma(d)) = d$.
  - Also called Onto Galois Connection.
  - From now onwards, we will assume that Galois Connections are Onto.

JOIN OVER PATHS

- Recall: Given a program as a LTS $\Gamma_c \equiv (V, L, l_0, l_e, T)$, the assertion map $\mu : L \rightarrow \mathbb{P}(States)$ associates a set of states with every location.
  - $\mu(l)$ is the set of states reachable at l during any execution.
  - μ is also called the Concrete Join Over Paths (JOP) or the collecting semantics.
- Instead of operating over concrete states, we can also consider JOP over abstract states.

ABSTRACT TRANSFER FUNCTION

- Given a Galois Connection $(\mathbb{P}(State), \subseteq) \underset{\gamma}{\overset{\alpha}{\rightleftarrows}} (D, \leq)$, for every program command p, we can define the abstract transfer function $\hat{f}_p$ (previously called the abstract strongest post-condition operator, $\hat{sp}$)
  - $\hat{f}_p : D \to D$.
- We can define the concrete transfer function as follows: $f_p(\sigma) = \{\sigma' \mid (\sigma, p) \hookrightarrow (\sigma', skip)\}$.
  - $f_p(c) = \bigcup_{\sigma \in c} f_p(\sigma)$
- Then, the abstract transfer function must be a consistent abstraction of the concrete transfer function:
  - $\forall d \in D \,.\, f_p(\gamma(d)) \subseteq \gamma(\hat{f}_p(d))$
  - Equivalently, $\forall c \in \mathbb{P}(State) \,.\, \hat{f}_p(\alpha(c)) \leq \alpha(f(c))$

ABSTRACT TRANSFER FUNCTION

EXAMPLE

- Consider the sign abstract domain, and the program command p : x := x+1.
    - $\hat{f}_p(+) = ???$

ABSTRACT TRANSFER FUNCTION

EXAMPLE

- Consider the sign abstract domain, and the program command p : x := x+1.
  - $\hat{f}_p(+) = +$

ABSTRACT TRANSFER FUNCTION

EXAMPLE

- Consider the sign abstract domain, and the program command p : x := x+1.
  - $\hat{f}_p(+) = +$
  - $\hat{f}_p(-) = ???$

ABSTRACT TRANSFER FUNCTION

EXAMPLE

- Consider the sign abstract domain, and the program command p : x := x+1.
  - $\hat{f}_p(+) = +$
  - $\hat{f}_p(-) = + -$

ABSTRACT TRANSFER FUNCTION

EXAMPLE

- Consider the sign abstract domain, and the program command p : x := x+1.
  - $\hat{f}_p(+) = +$
  - $\hat{f}_p(-) = +-$
  - $\hat{f}_p(+-) = +-$
  - $\hat{f}_p(\perp) = \perp$

ABSTRACT TRANSFER FUNCTION

EXAMPLE

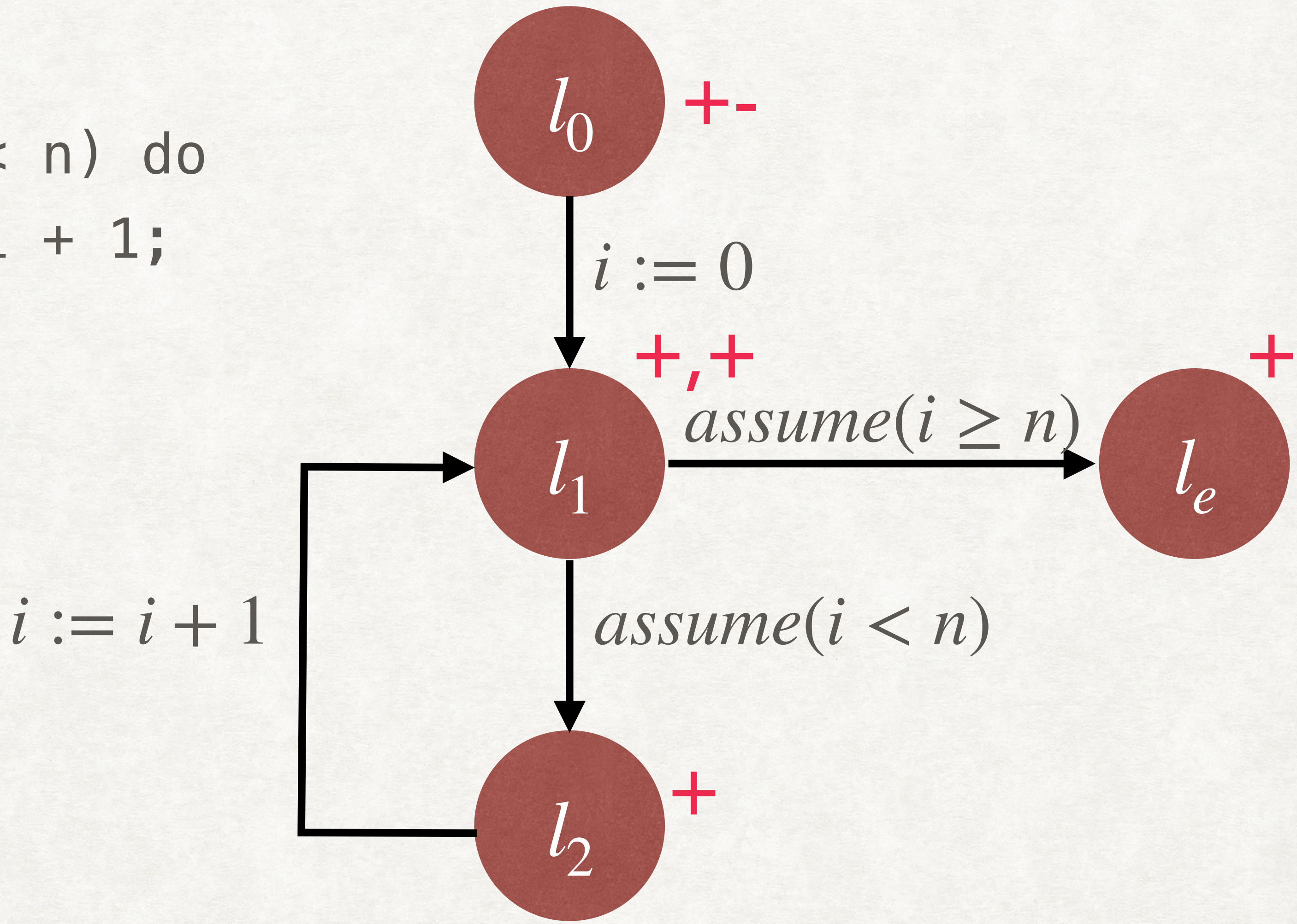
- Consider the sign abstract domain, and the program command p : x := x+1.
  - $\hat{f}_p(+) = +$
  - $\hat{f}_p(-) = + -$
  - $\hat{f}_p(+ -) = + -$
  - $\hat{f}_p(\perp) = \perp$
- A straightforward way to define consistent abstractions is to use γ, α and the concrete transfer function f_p:
  - $\hat{f}_p(d) = \alpha(f_p(\gamma(d))$

ABSTRACT JOP

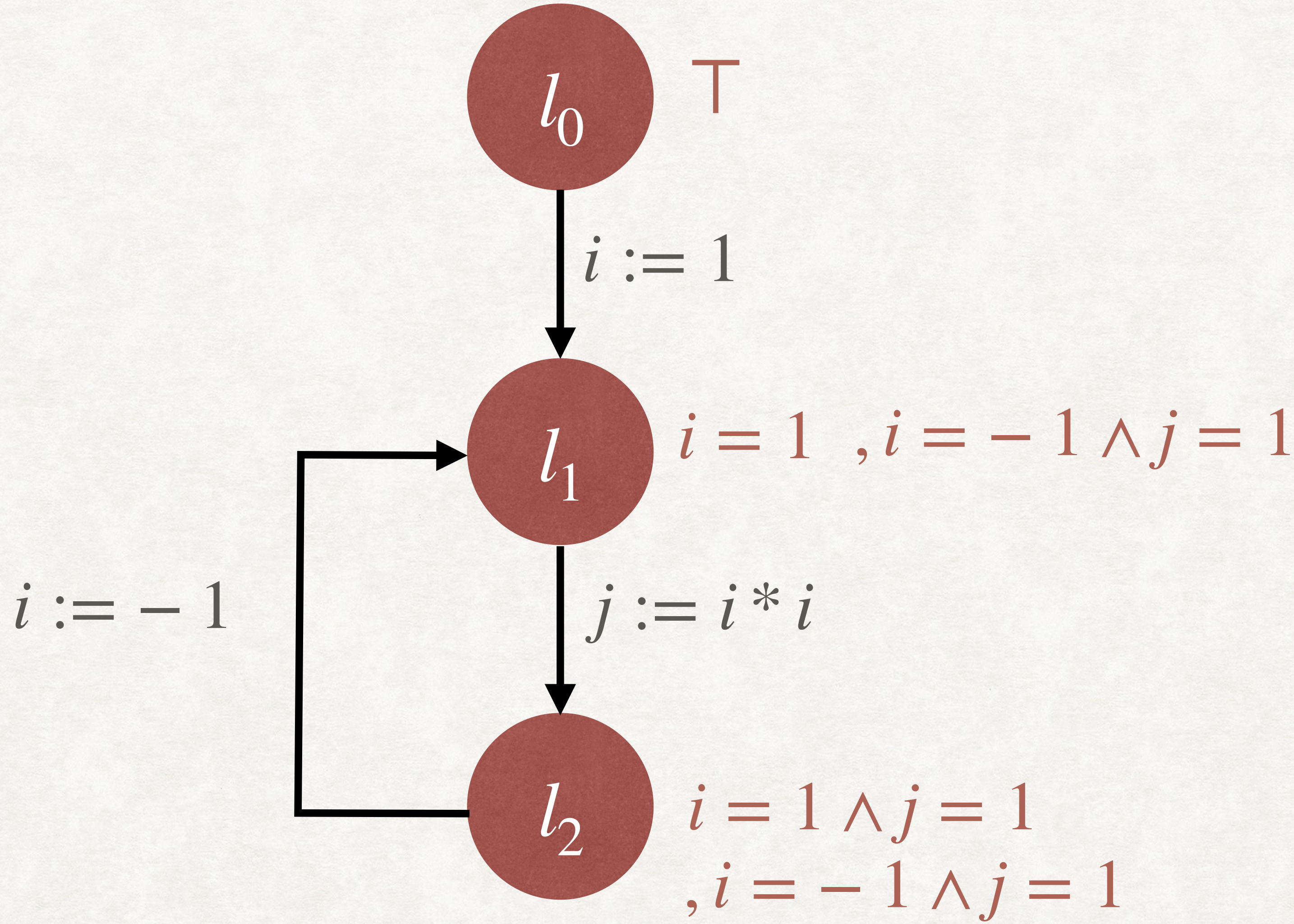
- Instead of executing the program with concrete states, we execute the program with abstract state, and the abstract transfer function for each program command.
- Collect all the abstract states at each location, for every possible execution
    - Their join is the abstract JOP map, $\hat{\mu} : L \to D$.

EXAMPLE

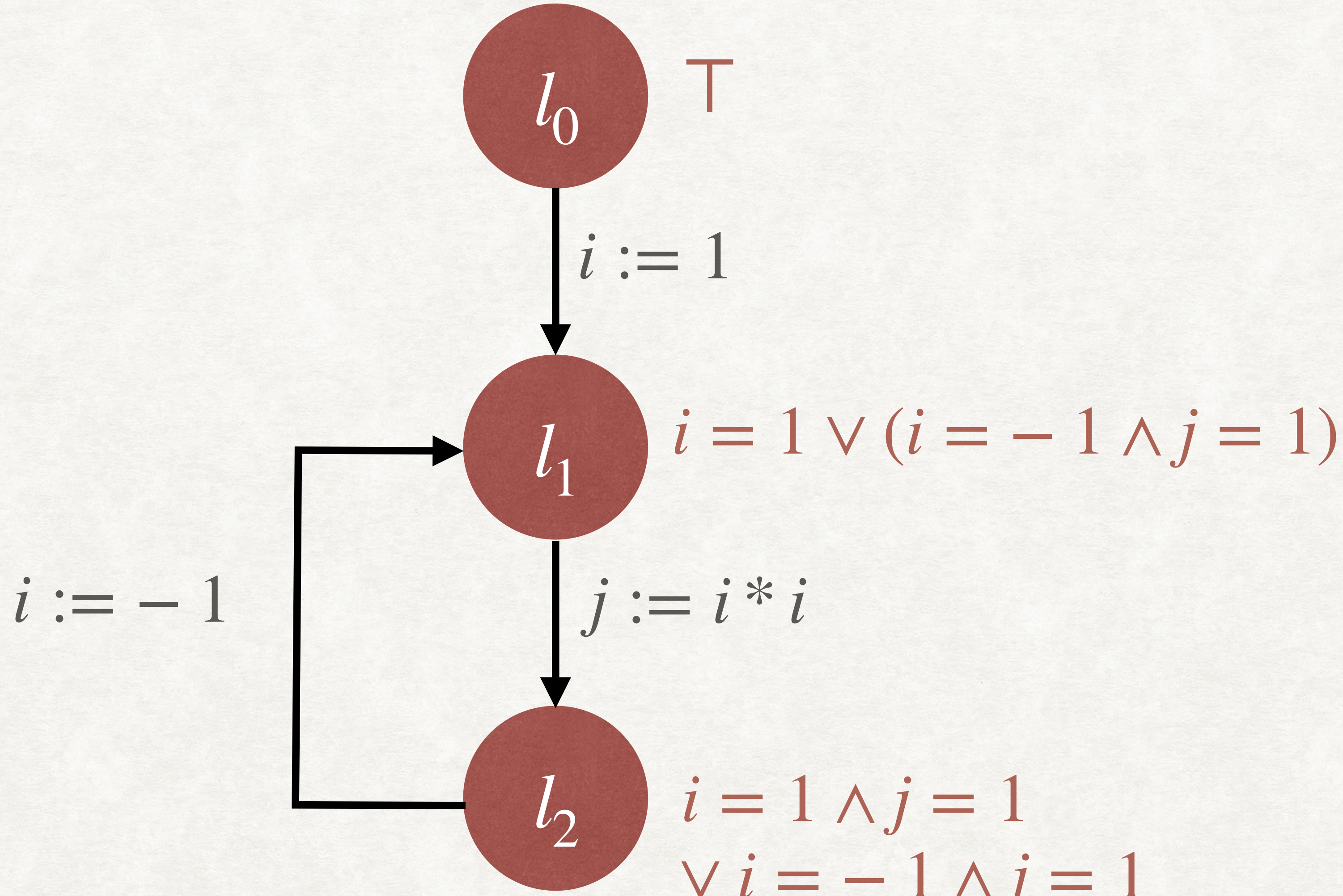
```
i := 0;
while(i < n) do
    i := i + 1;
```



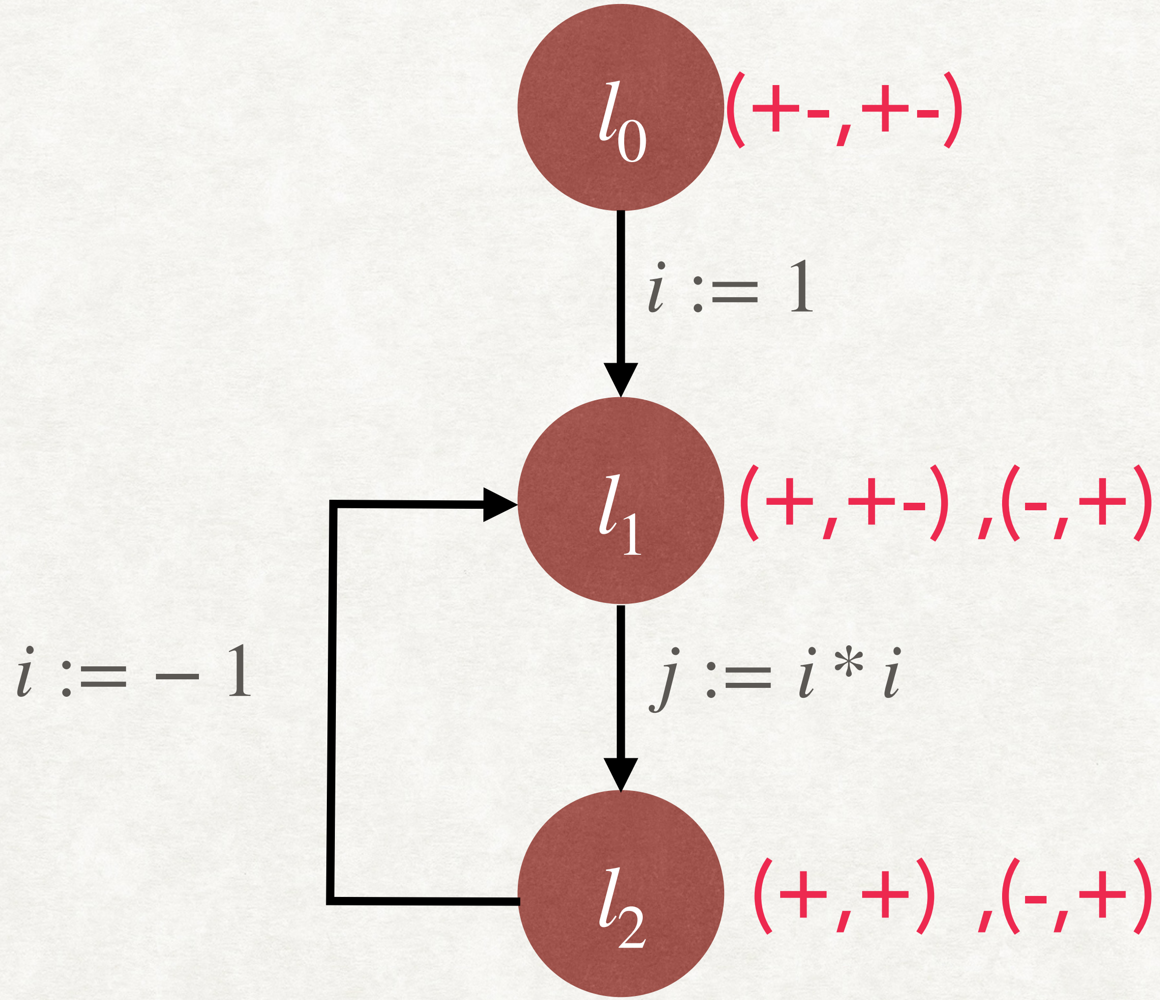
EXAMPLE - COLLECTING SEMANTICS



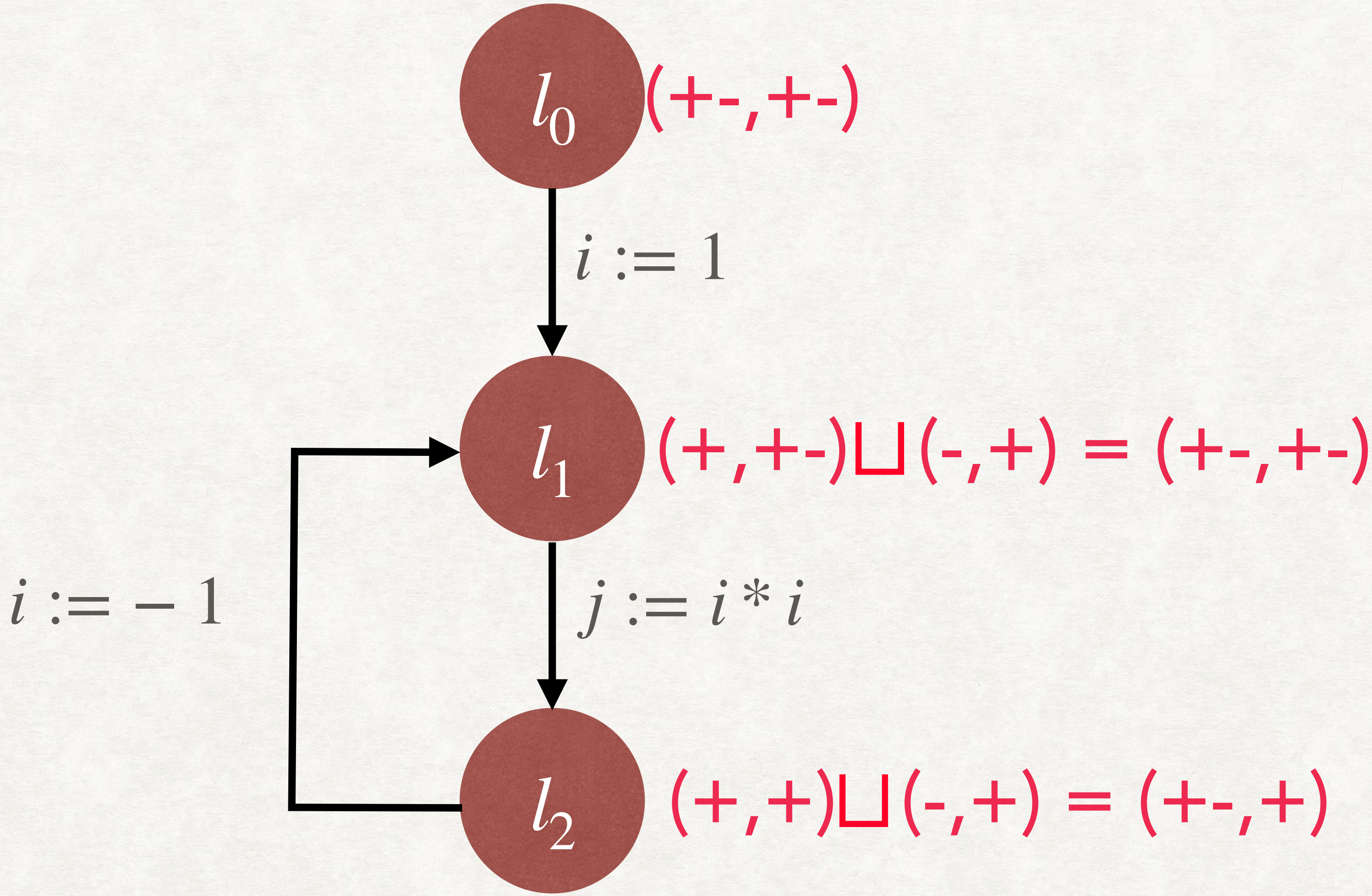
EXAMPLE - COLLECTING SEMANTICS



EXAMPLE - ABSTRACT JOP



EXAMPLE - ABSTRACT JOP



SOUNDNESS OF ABSTRACT INTERPRETATION

DEFINITION

A abstract interpretation consisting of

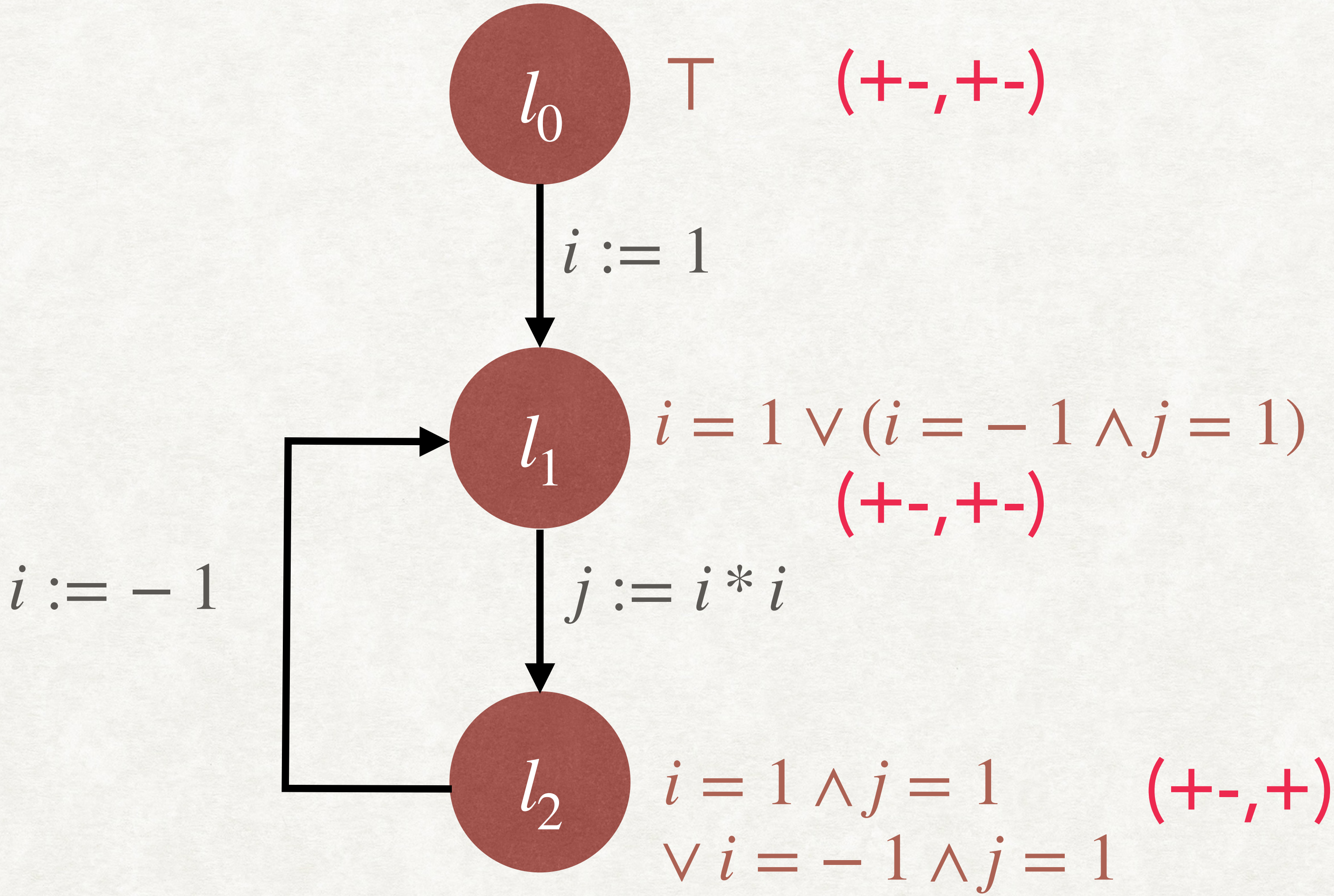
- the abstract domain $(D, \leq)$,
- abstraction, concretization functions (α, γ),
- and abstract transfer functions $\hat{F}_D$

is sound,

if for all $d_0 \in D$, for all programs Γ,
assuming that $\hat{\mu}(l_0) = d_0$, and $\mu(l_0) = c_0$ where $c_0 \subseteq \gamma(d_0)$,
the γ image of the abstract JOP $\hat{\mu}$ at all locations in Γ over approximates the collecting semantics μ,
that is, or all locations l, $\gamma(\hat{\mu}(l)) \supseteq \mu(l)$.

SOUNDNESS OF ABSTRACT INTERPRETATION

EXAMPLE



FROM ABSTRACT INTERPRETATION TO VERIFICATION

- In order to show the validity of the Hoare Triple $\{P\}c\{Q\}$, we instantiate a sound AI $(D, \leq, \alpha, \gamma, \hat{F}_D)$ with $\hat{\mu}(l_0) = d_0$ such that $d_0 = \alpha(P)$ and compute the resulting JOP $\hat{\mu}$ at all locations.
- If $\gamma(\hat{\mu}(l_e)) \subseteq Q$, then the Hoare Triple is valid.
    - Since $\alpha(P) = d_0$, by definition of Galois connection, $P \subseteq \gamma(d_0)$.
    - Hence, by definition of soundness of AI, $\mu(l_e) \subseteq \gamma(\hat{\mu}(l_e))$, where μ is the collecting semantics assuming $\mu(l_0) = P$.

SOUNDNESS OF ABSTRACT INTERPRETATION

SUFFICIENT CONDITIONS

- An abstract interpretation $(D, \leq, \alpha, \gamma, \hat{F}_D)$ is sound if:
  - $(D, \leq)$ is complete lattice.
  - $(\mathbb{P}(State), \subseteq) \underset{\gamma}{\overset{\alpha}{\rightleftarrows}} (D, \leq)$
  - Every abstract transfer function in $\hat{F}_D$ is a consistent abstraction of the corresponding concrete transfer function.