ABSTRACT FORWARD PROPAGATE

KILDALL'S ALGORITHM

AbstractForwardPropagate(Γ_c,P)
  S := $\{l_0\}$;
  $\hat{\mu}(l_0)$:= α(P);
  $\hat{\mu}(l)$:= $\bot$, for $l \in L \backslash \{l_0\}$;
  while S $\neq \emptyset$ do{
    l := Choose S;
    S := S \ $\{l\}$;
    foreach $(l, c, l') \in T$ do{
      F := $\hat{sp}(\hat{\mu}(l), c)$;
      if $\neg(F \leq \hat{\mu}(l'))$ then{
        $\hat{\mu}(l') := \hat{\mu}(l') \sqcup F$;
        S := S $\cup \{l'\}$;
      }
    }
  }

- Does this algorithm actually calculate the abstract JOP?
- What are the conditions under which it is guaranteed to terminate?

A Unified Approach to Global Program Optimization. Gary Kildall. POPL 1973.

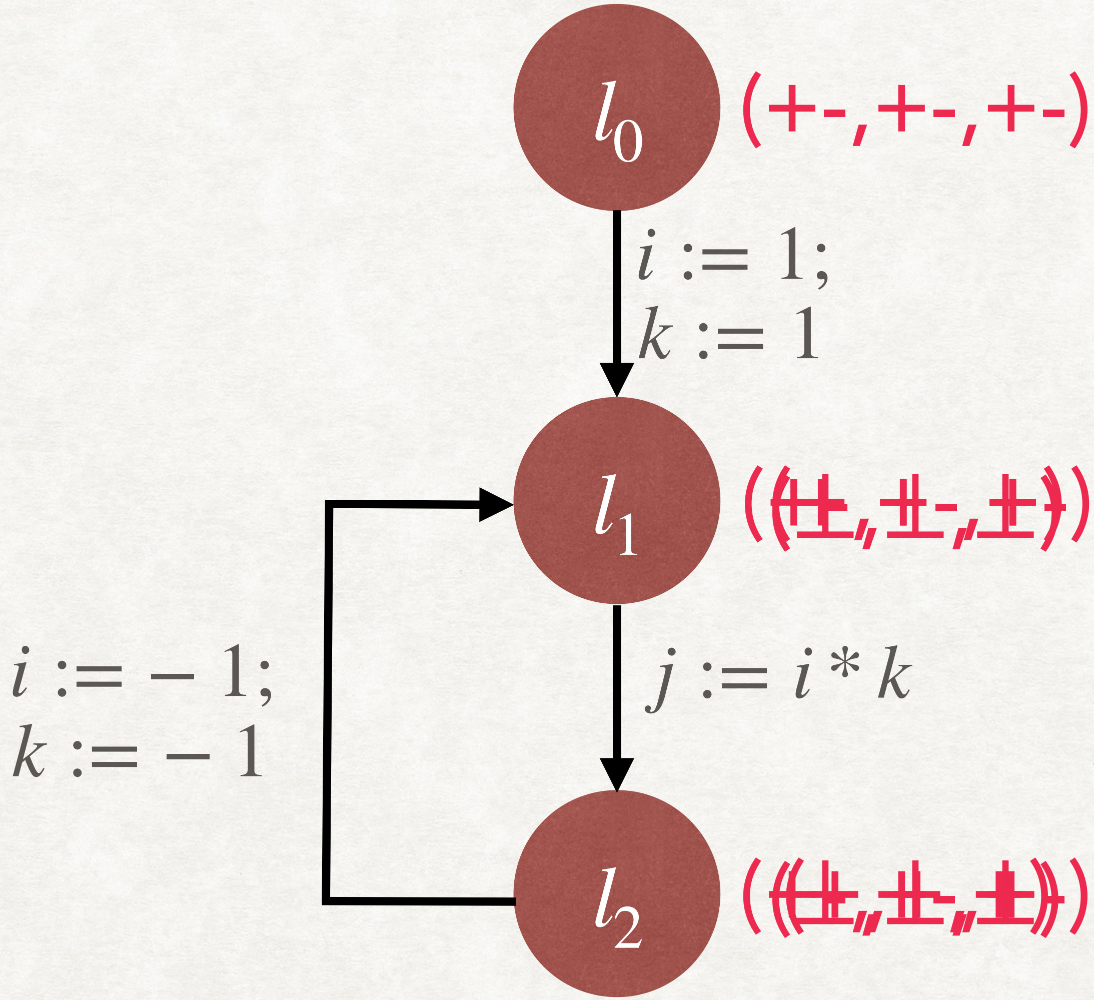
ABSTRACT FORWARD PROPAGATE

KILDALL'S ALGORITHM

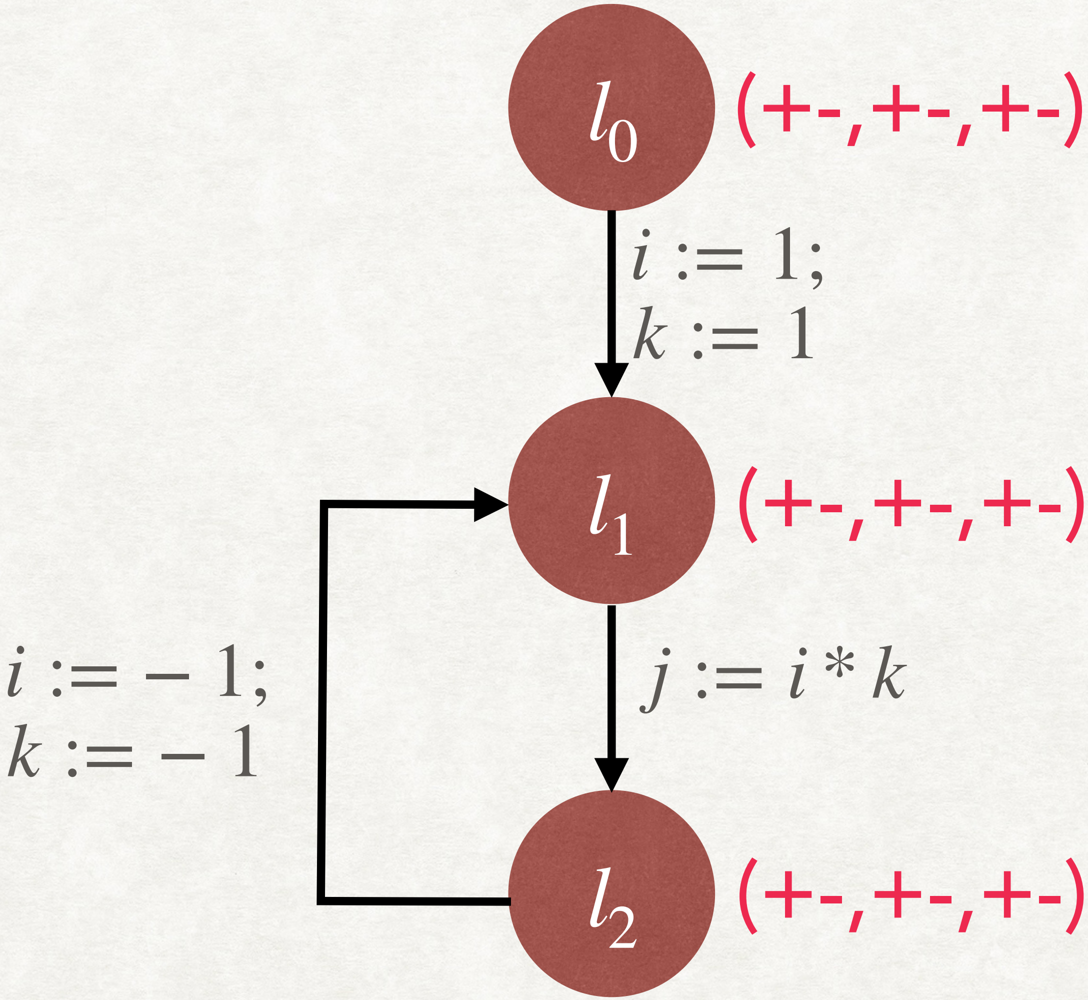
```
AbstractForwardPropagate(Γ_c,P)
  S := {l_0};
  μ̂_K(l_0) := α(P);
  μ̂_K(l) := ⊥, for l ∈ L\{l_0};
  while S ≠ ∅ do{
     l := Choose S;
     S := S \ {l};
     foreach (l, c, l') ∈ T do{
        F := f̂_c(μ̂_K(l));
        if ¬(F ≤ μ̂_K(l')) then{
           μ̂_K(l') := μ̂_K(l') ⊔ F;
         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

A Unified Approach to Global Program Optimization. Gary Kildall. POPL 1973.

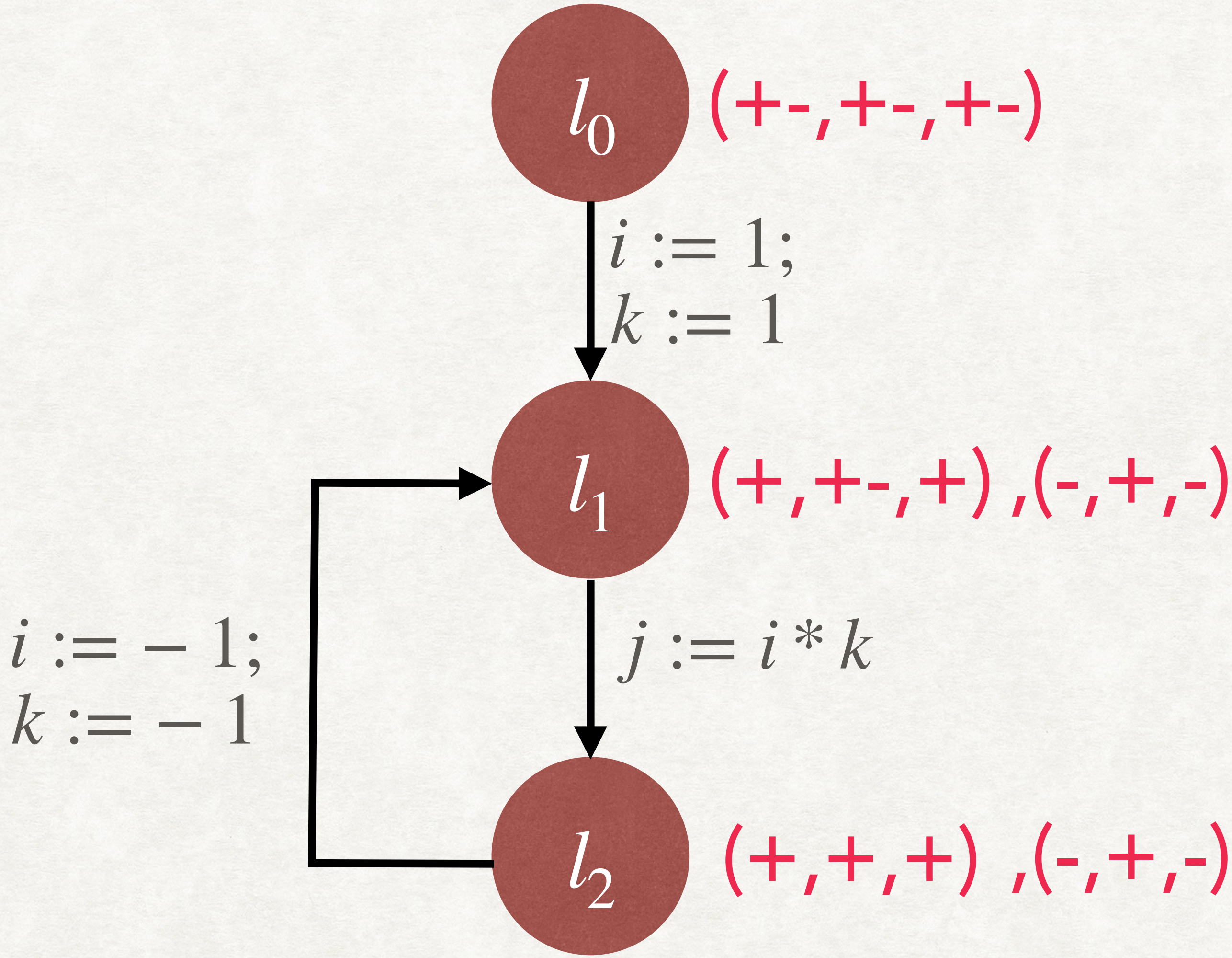
EXAMPLE - KILDALL'S ALGORITHM



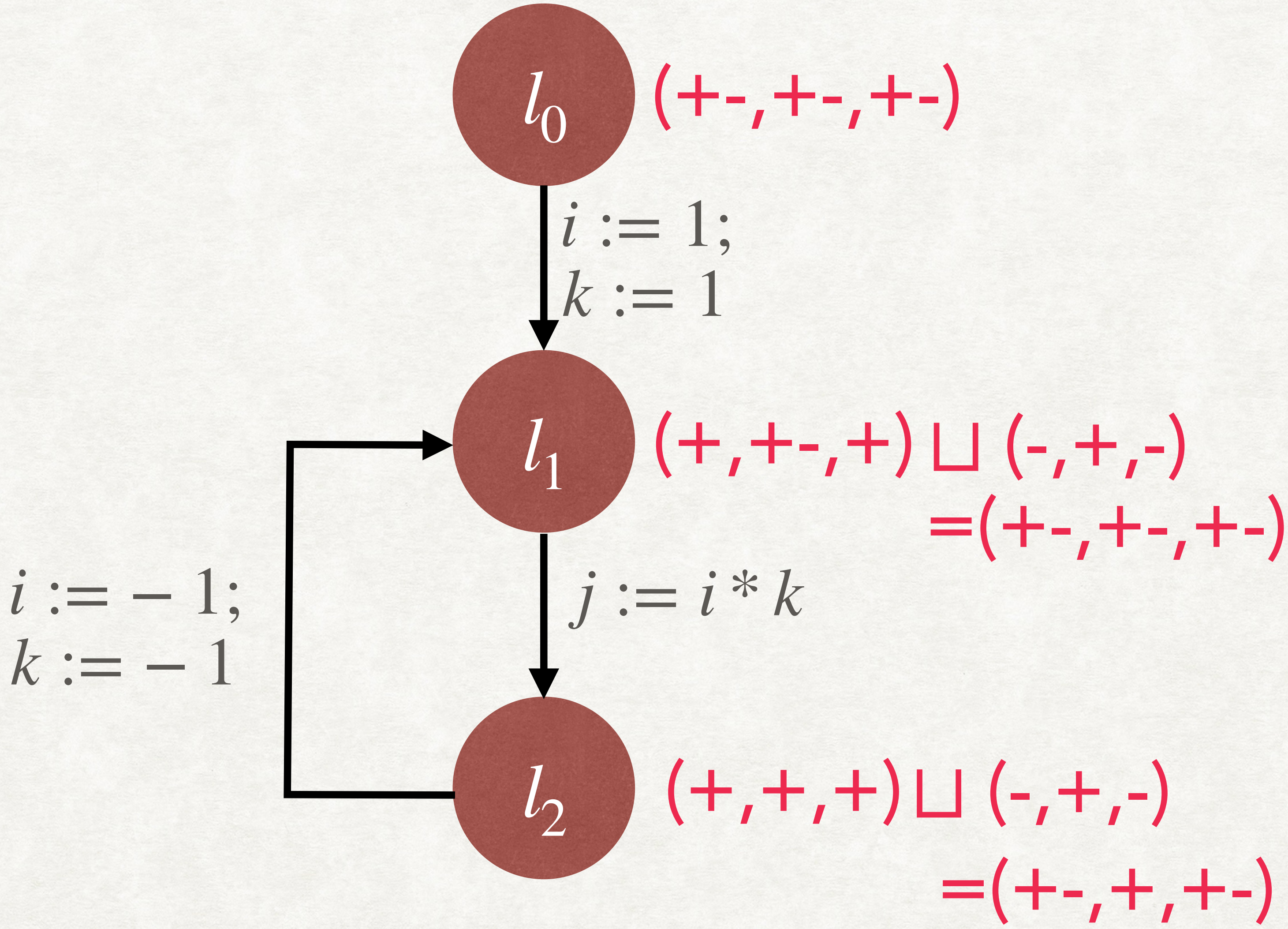
EXAMPLE - KILDALL'S ALGORITHM



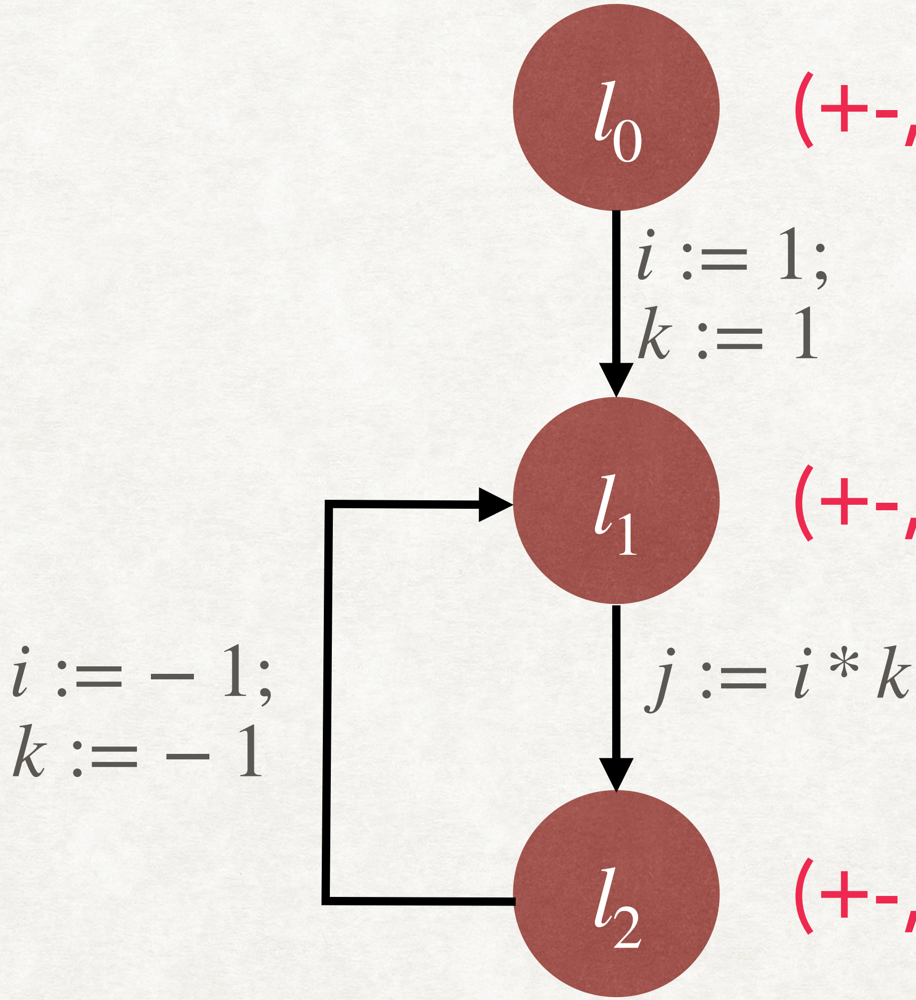
EXAMPLE - ABSTRACT JOP



EXAMPLE - ABSTRACT JOP



EXAMPLE - KILDALL VS ABSTRACT JOP



	$\hat{\mu}_K$	$\hat{\mu}$
l_0	(+-,+-,+-)	(+-,+-,+-)
l_1	(+-,+-,+-)	(+-,+-,+-)
l_2	(+-,+-,+-)	(+-,+,+-)

$\hat{\mu}_K \neq \hat{\mu}$: This is because Kildall's Algorithm applies join eagerly

We will prove that $\hat{\mu}_K \geq \hat{\mu}$

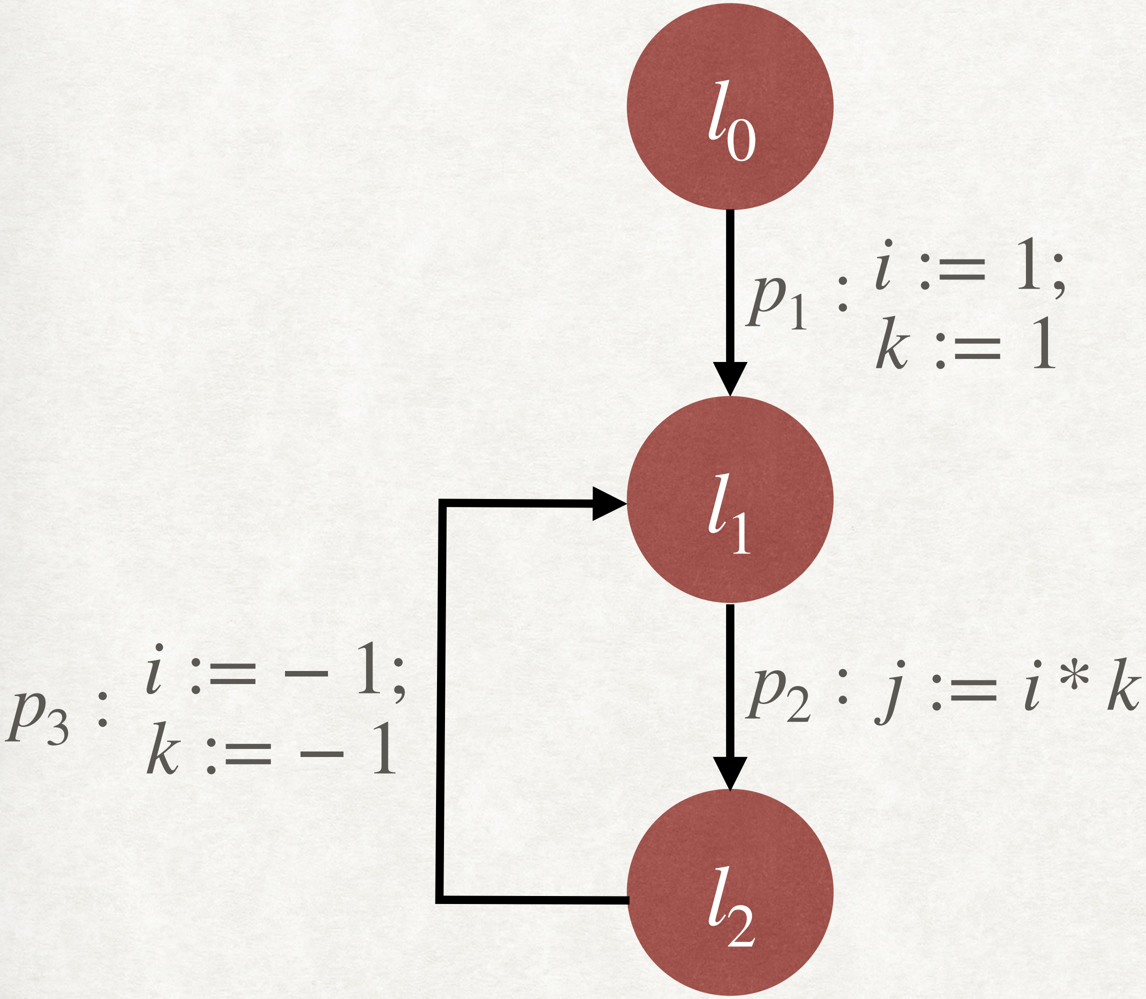
PROPERTIES OF KILDALL'S ALGORITHM

1. The values computed using Kildall's algorithm are an over-approximation of the abstract JOP, if the underlying AI framework is monotonic.
2. In general, Kildall's algorithm computes the least solution to a system of equations.
3. If the abstract domain satisfies the ascending chain condition, then Kildall's algorithm is guaranteed to terminate.

DATAFLOW EQUATIONS

- Program $\Gamma_c = (V, L, l_0, l_e, T)$ induces a system of data flow equations:
  - $X_{l_0} = d_0$
  - For all other locations $l \in L \backslash \{l_0\},\ X_l = \bigsqcup_{(l',c,l)\in T} \hat{f}_c(X_{l'})$
- For collecting semantics, replace d_0 with c_0, ⊔ with ∪ and $\hat{f}_c$ with f_c.

EXAMPLE - DATAFLOW EQUATIONS



$$X_{l_0} = d_0$$

$$X_{l_1} = \hat{f}_{p_1}(X_{l_0}) \sqcup \hat{f}_{p_3}(X_{l_2})$$

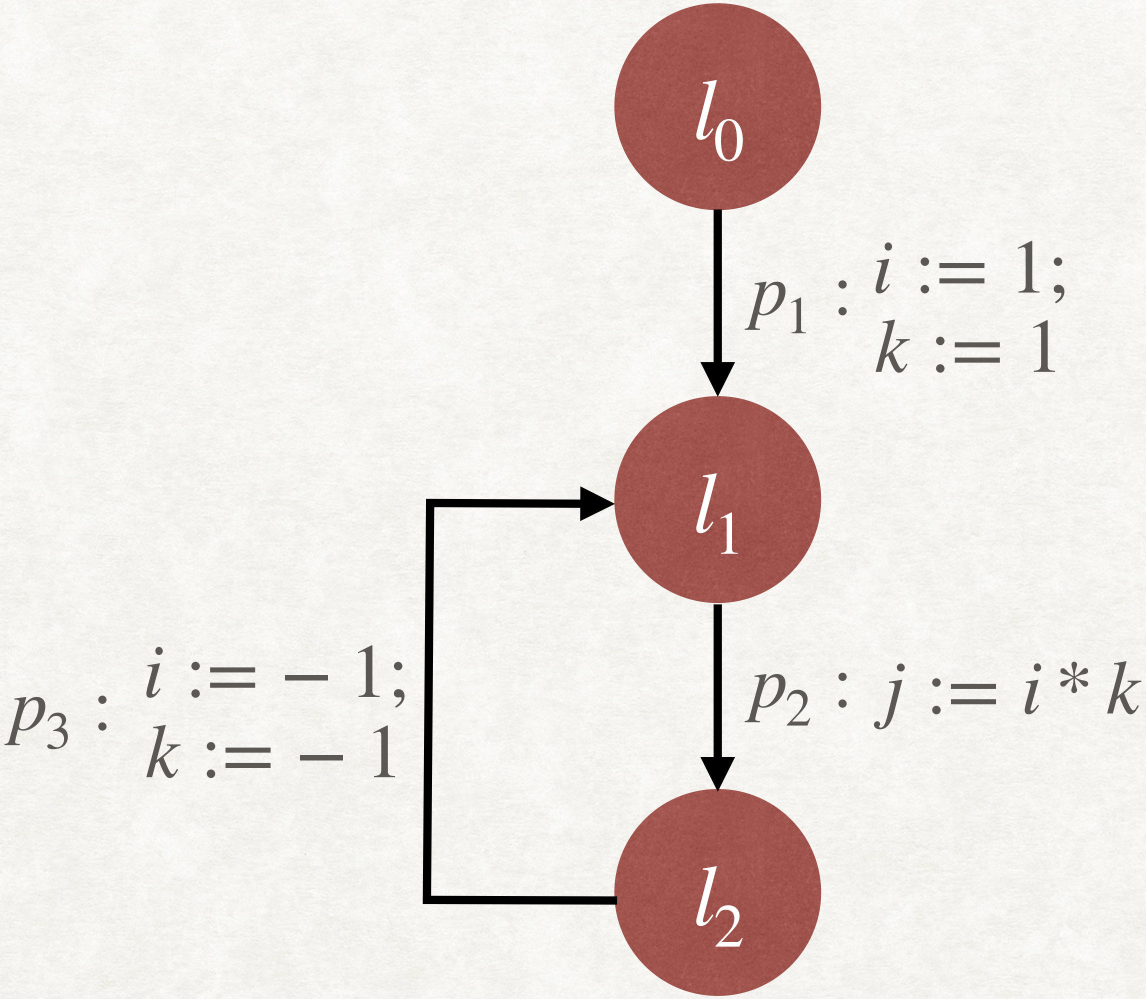
$$X_{l_2} = \hat{f}_{p_2}(X_{l_1})$$

DATAFLOW EQUATIONS AS FUNCTION

- Consider the 'vectorised' lattice $(\bar{D}, \bar{\leq})$, where $\bar{D} = D^{|L|}$.
  - $\bar{d} \mathrel{\bar{\leq}} \bar{d}' \Leftrightarrow \forall l \in L \,.\, \bar{d}(l) \leq \bar{d}'(l)$
  - Homework: Prove that if $(D, \leq)$ is a complete lattice, then $(\bar{D}, \bar{\leq})$ is also a complete lattice.
- We can view the data flow equations as a function $\bar{f} : \bar{D} \to \bar{D}$:
  - $(\bar{f}(\bar{d}))(l_0) = d_0$
  - $(\bar{f}(\bar{d}))(l) = \bigsqcup_{(l',c,l)\in T} \hat{f}_c(\bar{d}(l'))$

DATAFLOW EQUATIONS AS FUNCTION

EXAMPLE



Notice that a fixpoint of $\bar{f}$ is a solution to the dataflow equations

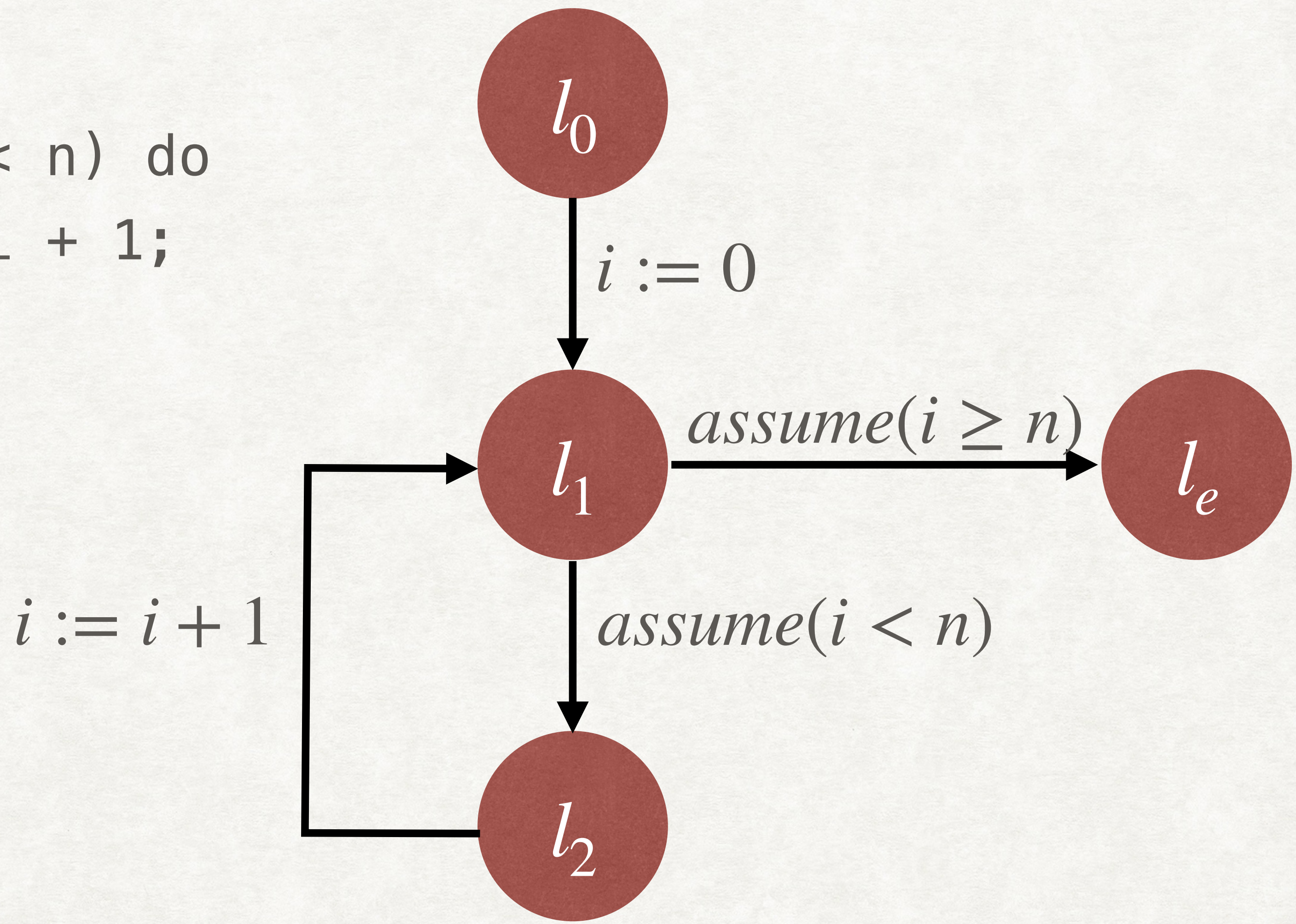
$$\bar{f}(d_{l_0}, d_{l_1}, d_{l_2}) = (d_0, \hat{f}_{p_1}(d_{l_0}) \sqcup \hat{f}_{p_3}(d_{l_2}), \hat{f}_{p_2}(d_{l_1}))$$

DATAFLOW EQUATIONS AS FUNCTION

- If every abstract transfer function $\hat{f} : D \to D$ is monotonic, then the function $\bar{f} : \bar{D} \to \bar{D}$ is also monotonic.
  - Homework: Prove this.
- We have a monotonic function $\bar{f}$ on a complete lattice $\bar{D}$. Hence, we can apply Knaster-Tarski fixpoint theorem.
- The least fixpoint $lfp(\bar{f})$ exists, and is in fact the least solution to the dataflow equations.
- We will show that Kildall's algorithm actually computes $lfp(\bar{f})$.
- Note that we can also use the sequence $\perp, \bar{f}(\perp), \bar{f}^2(\perp), \ldots$ to compute $lfp(\bar{f})$.
  - This method is also called Kleene Iteration.

LFP INTRODUCES THE LEAST OVER APPROXIMATION: EXAMPLE

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



$(+-, +, +, +)$ is a solution to the data flow equations,

And $(+-, +-, +-, +-)$ is also another solution

ABSTRACT JOP $\leq lfp(\bar{f})$ FOR MONOTONIC AI FRAMEWORK

PROOF

- Given AI $(D, \leq, \alpha, \gamma, \hat{F}_D)$, if all functions in $\hat{F}_D$ are monotonic, then Abstract JOP $\leq lfp(\bar{f})$.

Proof: Abstract JOP $\hat{\mu}(l) = \bigsqcup_{\pi \in \Pi_l} \hat{f}_\pi(d_0)$

Let $lfp(\bar{f}) = \bar{d}$. We have to show that $\forall l \in L \,.\, \hat{\mu}(l) \leq \bar{d}(l)$.

We will show that for all locations l, all paths $\pi \in \Pi_l$, $\hat{f}_\pi(d_0) \leq \bar{d}(l)$.

This proves the required result. Why?

We will use induction on length of the paths.

Base Case: Paths π of length 0 are empty and end at l_0. Hence, $\hat{f}_\pi(d_0) = d_0$.

Since $\bar{f}(\bar{d}) = \bar{d}$ and $(\bar{f}(\bar{d}))(l_0) = d_0$, we have $\bar{d}(l_0) = d_0$.

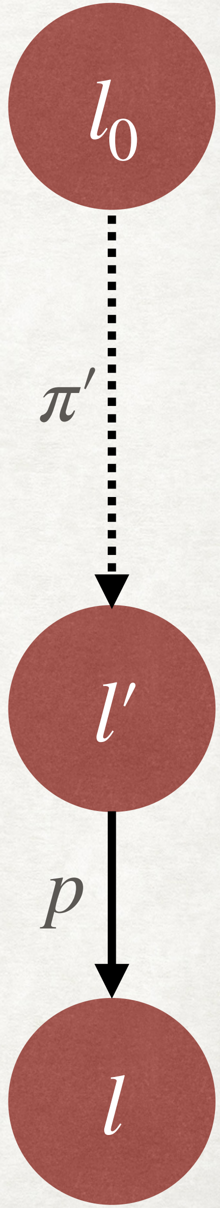
Thus, $\hat{f}_\pi(d_0) \leq \bar{d}(l_0)$

ABSTRACT JOP $\leq lfp(\bar{f})$ FOR MONOTONIC AI FRAMEWORK

PROOF

Inductive Case: Assume that the claim holds for all paths of length n.

Consider a path π of length $n+1$ ending at location l.



ABSTRACT JOP $\leq lfp(\bar{f})$ FOR MONOTONIC AI FRAMEWORK

PROOF

Inductive Case: Assume that the claim holds for all paths of length n.

Consider a path π of length $n + 1$ ending at location l.

Let π' be the prefix of the path of length n, ending at location l'.

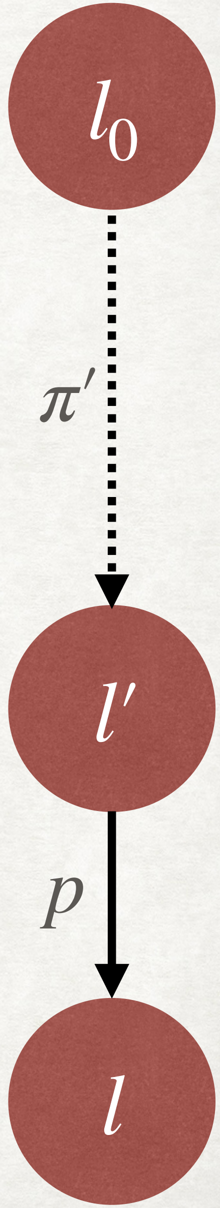
By Inductive Hypothesis, $\hat{f}_{\pi'}(d_0) \leq \bar{d}(l')$.

Since $\hat{f}_p$ is monotonic, $\hat{f}_p(\hat{f}_{\pi'}(d_0)) \leq \hat{f}_p(\bar{d}(l'))$.

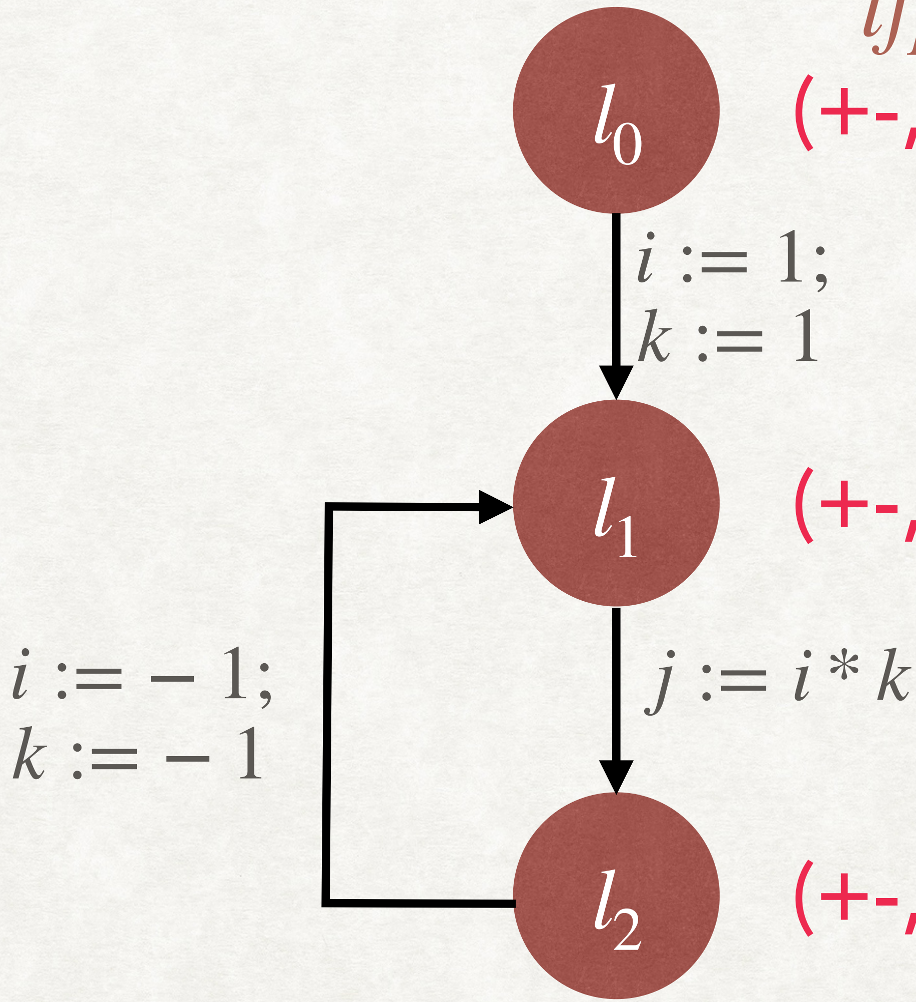
Hence, $\hat{f}_\pi(d_0) \leq \hat{f}_p(\bar{d}(l'))$.

Now $\bar{f}(\bar{d}) = \bar{d}$. Hence, $\bar{d}(l) = \bigsqcup_{(l',p,l)\in T} \hat{f}_p(\bar{d}(l'))$.

Hence, $\hat{f}_p(\bar{d}(l')) \leq \bar{d}(l)$. Thus, $\hat{f}_\pi(d_0) \leq \bar{d}(l)$.



EXAMPLE - LFP VS ABSTRACT JOP



$lfp(\bar{f})$	Abstract JOP
(+-,+-,+-)	(+-,+-,+-)
(+-,+-,+-)	(+-,+-,+-)
(+-,+-,+-)	(+-,+,+-)

$$\bar{f}(d_{l_0}, d_{l_1}, d_{l_2}) = (d_0, \hat{f}_{p_1}(d_{l_0}) \sqcup \hat{f}_{p_3}(d_{l_2}), \hat{f}_{p_2}(d_{l_1}))$$

DISTRIBUTIVE AND INFINITELY DISTRIBUTIVE FUNCTIONS

- Given two posets $(D_1, \leq_1)$ and $(D_2, \leq_2)$, function $f: D_1 \to D_2$ is called distributive if for $x, y \in D_1$ such that $x \sqcup_1 y$ exists, then $f(x) \sqcup_2 f(y)$ also exists, and $f(x \sqcup_1 y) = f(x) \sqcup_2 f(y)$.
- Given two posets $(D_1, \leq_1)$ and $(D_2, \leq_2)$, function $f: D_1 \to D_2$ is called infinitely distributive if for all $X \subseteq D_1$ such that $\sqcup_1 X$ exists, then $\sqcup_2 f(X)$ also exists, and $\sqcup_2 f(X) = f(\sqcup_1 X)$.
- Exercise: If f is distributive, then f is also monotonic.

ABSTRACT JOP = $lfp(\bar{f})$ FOR INFINITELY DISTRIBUTIVE AI FRAMEWORK

PROOF

- Given AI $(D, \leq, \alpha, \gamma, \hat{F}_D)$, if all functions in $\hat{F}_D$ are infinitely distributive, then Abstract JOP = $lfp(\bar{f})$.

Proof: We will show that Abstract JOP ($\hat{\mu}$) is a fixpoint of $\bar{f}$. This is sufficient to prove the result. Why?

ABSTRACT JOP = $lfp(\bar{f})$ FOR INFINITELY DISTRIBUTIVE AI FRAMEWORK

PROOF

- Given AI $(D, \leq, \alpha, \gamma, \hat{F}_D)$, if all functions in $\hat{F}_D$ are infinitely distributive, then Abstract JOP = $lfp(\bar{f})$.

Proof: We will show that Abstract JOP ($\hat{\mu}$) is a fixpoint of $\bar{f}$. This is sufficient to prove the result. Why?

$$
\begin{aligned}
(\bar{f}(\hat{\mu}))(l) &= \bigsqcup_{(l',p,l)\in T} \hat{f}_p(\hat{\mu}(l')) \\
&= \bigsqcup_{(l',p,l)\in T} \hat{f}_p(\bigsqcup_{\pi\in\Pi_{l'}} \hat{f}_\pi(d_0)) \\
&= \bigsqcup_{(l',p,l)\in T} \bigsqcup_{\pi\in\Pi_{l'}} \hat{f}_p \circ \hat{f}_\pi(d_0)
\end{aligned}
$$

ABSTRACT JOP = $lfp(\bar{f})$ FOR INFINITELY DISTRIBUTIVE AI FRAMEWORK

PROOF

$$(\bar{f}(\hat{\mu}))(l) = \bigsqcup_{(l',p,l)\in T} \bigsqcup_{\pi\in\Pi_{l'}} \hat{f}_p \circ \hat{f}_\pi(d_0)$$

And we know that $\hat{\mu}(l) = \bigsqcup_{\pi'\in\Pi_l} \hat{f}_{\pi'}(d_0)$.

Then, due to associativity of $\sqcup$, $(\bar{f}(\hat{\mu}))(l) = \hat{\mu}(l)$.

Thus, $\hat{\mu}$ is a fixpoint of $\bar{f}$. We know from previous result that $\hat{\mu} \leq lfp(\bar{f})$. Thus, $\hat{\mu} = lfp(\bar{f})$.

RECALL: SIGN ABSTRACT DOMAIN

- The abstract transfer functions in sign abstract domain are monotonic, but not infinitely distributive.

Consider p : j := i*k and $d_1 = (+ , + - , +)$, $d_2 = (- , + - , -)$.

Then, $\hat{f}_p(d_1 \sqcup d_2) = ???$

RECALL: SIGN ABSTRACT DOMAIN

- The abstract transfer functions in sign abstract domain are monotonic, but not infinitely distributive.

Consider p : j := i*k and $d_1 = (+ , +- , +)$, $d_2 = (- , +- , -)$.

Then, $\hat{f}_p(d_1 \sqcup d_2) = (+- , +- , +-)$.

RECALL: SIGN ABSTRACT DOMAIN

- The abstract transfer functions in sign abstract domain are monotonic, but not infinitely distributive.

Consider p : j := i*k and $d_1 = (\,+\,,\,+-\,,\,+\,)$, $d_2 = (\,-\,,\,+-\,,\,-\,)$.

Then, $\hat{f}_p(d_1 \sqcup d_2) = (\,+-\,,\,+-\,,\,+-\,)$.

$\hat{f}_p(d_1) \sqcup \hat{f}_p(d_2) = ???$

RECALL: SIGN ABSTRACT DOMAIN

- The abstract transfer functions in sign abstract domain are monotonic, but not infinitely distributive.

Consider p : j := i*k and $d_1 = (+\,, +-\,, +\,)$, $d_2 = (-\,, +-\,, -\,)$.

Then, $\hat{f}_p(d_1 \sqcup d_2) = (+-\,, +-\,, +-\,)$.

$\hat{f}_p(d_1) \sqcup \hat{f}_p(d_2) = (+\,, +\,, +\,) \sqcup (-\,, +\,, -\,) = (+-\,, +\,, +-\,)$

RECALL: SIGN ABSTRACT DOMAIN

- The abstract transfer functions in sign abstract domain are monotonic, but not infinitely distributive.

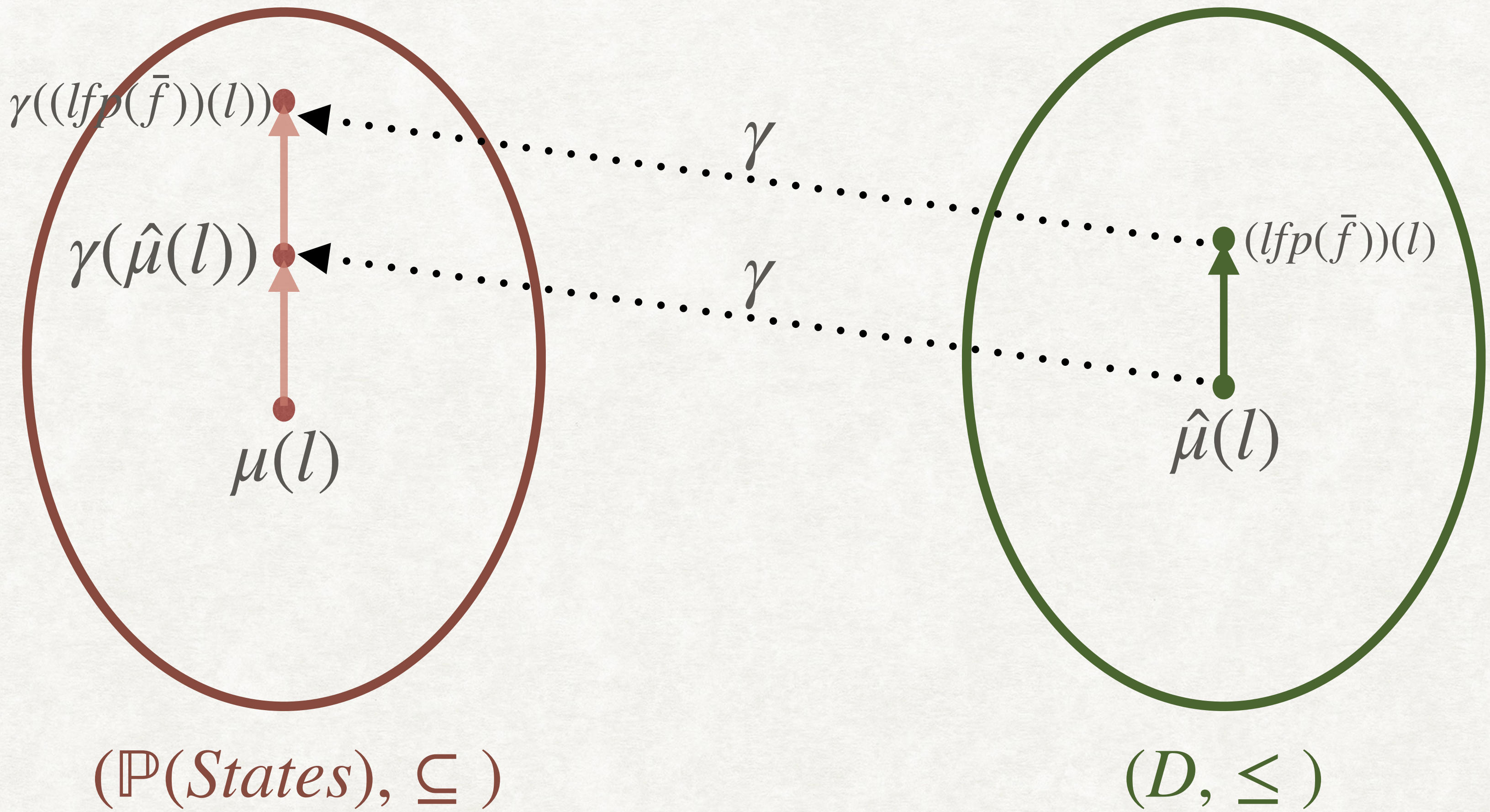
Consider p : j := i*k and $d_1 = (+, +-, +)$, $d_2 = (-, +-, -)$.

Then, $\hat{f}_p(d_1 \sqcup d_2) = (+-, +-, +-)$.

$\hat{f}_p(d_1) \sqcup \hat{f}_p(d_2) = (+, +, +) \sqcup (-, +, -) = (+-, +, +-)$

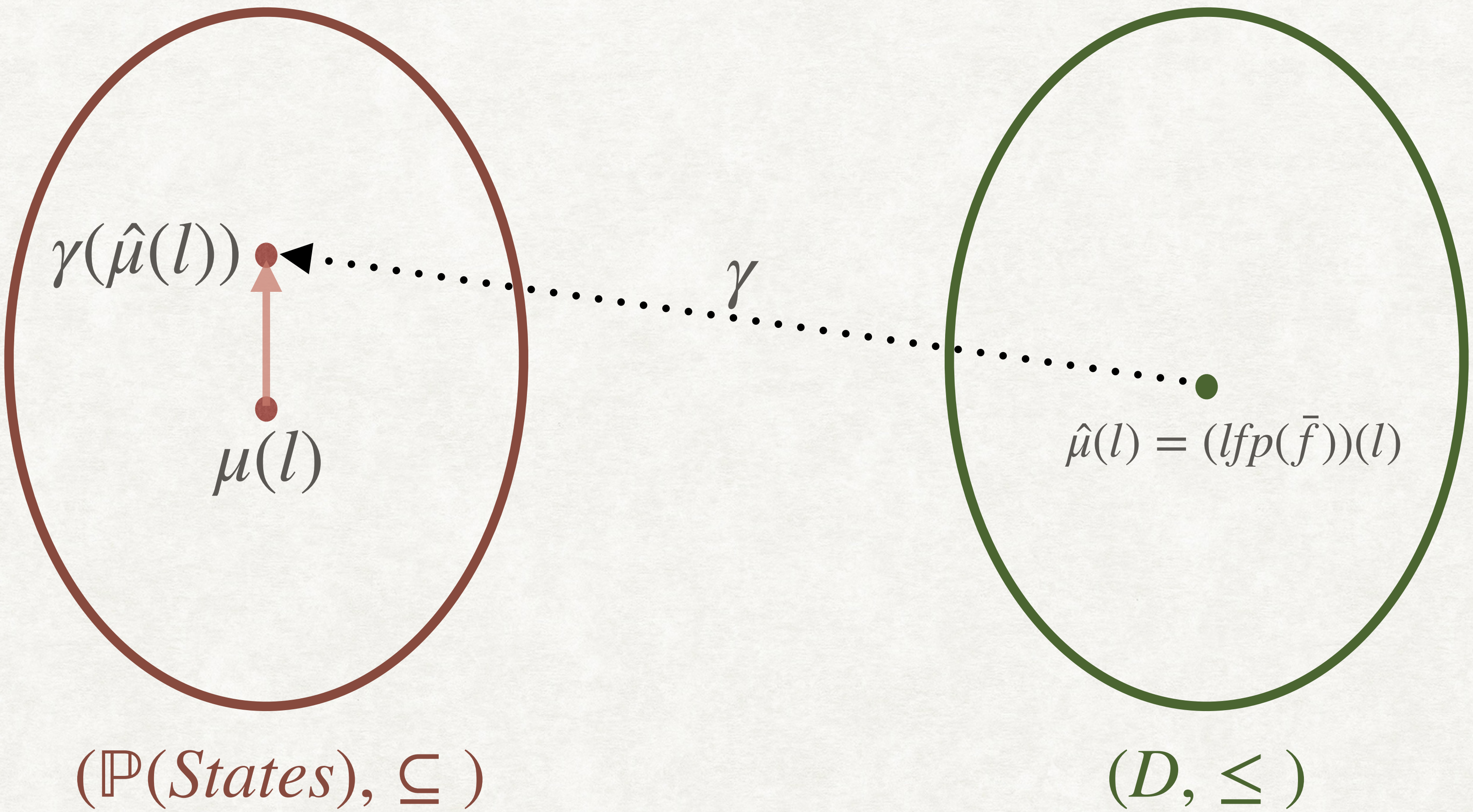
- The concrete transfer functions are always infinitely distributive. Hence, the concrete JOP is the least solution of the concrete data-flow equations.

BIG PICTURE



For Monotonic AI Framework

BIG PICTURE



For Infinitely Distributive AI Framework

KILDALL'S ALGORITHM COMPUTES $lfp(\bar{f})$

PROOF

- First, we will show that $\hat{\mu}_K \leq lfp(\bar{f})$

We will show that $\hat{\mu}_K \leq lfp(\bar{f})$ is a loop invariant of the outer while loop.

At the beginning, $\hat{\mu}_K(l_0) = \alpha(P) \leq d_0$.

Hence, $\forall l \,.\, \hat{\mu}_K(l) \leq (lfp(\bar{f}))(l)$.

Assuming that the claim holds at the beginning of some iteration, let $\hat{\mu}_K = \bar{d}$, $lfp(\bar{f}) = \bar{g}$. We have $\bar{d} \leq \bar{g}$.

```
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'};
      }
    }
  }
```

KILDALL'S ALGORITHM COMPUTES $lfp(\bar{f})$

PROOF

```
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'};
          }
      }
  }
```

KILDALL'S ALGORITHM COMPUTES $lfp(\bar{f})$

PROOF

For some successor l' of l,
$\hat{\mu}_K(l') = d(l') \sqcup \hat{f}_c(d(l))$.

Now, $\bar{d}(l) \leq \bar{g}(l) \Rightarrow \hat{f}_c(\bar{d}(l)) \leq \hat{f}_c(\bar{g}(l))$.

Further, $\bar{g}(l') = \bigsqcup_{(l,c,l')\in T} \hat{f}_c(\bar{g}(l))$

Hence, $\bar{g}(l') \geq \hat{f}_c(\bar{g}(l)) \geq \hat{f}_c(\bar{d}(l))$

We also know that $\bar{g}(l') \geq \bar{d}(l')$.

Thus, $\bar{g}(l') \geq \bar{d}(l') \sqcup \hat{f}_c(\bar{d}(l))$.

Hence, $\bar{g}(l') \geq \hat{\mu}_K(l')$.

```
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'};
        }
     }
  }
```