Fixpoint Guided Abstraction Refinements

Patrick Cousot

Jean François Raskin

Pierre Ganty

Now at UCLA!

• University of Brussels, Belgium • ENS of Paris, France •

Motivation

$$M \models \varphi$$

Model

Property

Verification Algorithm, returns $\models$ or $\not\models$

In some cases, because of **undecidability** or the **the state explosion problem**

The verification algorithm is not **applicable**

A possible solution is to **abstract** the verification test

Motivation

$$M \models \varphi$$

Model

Property

Verification Algorithm, returns $\models$ or $\not\models$

Abstract Verification

$$M \models^{A} \varphi$$

returns $\models$, $\not\models$, $\overset{?}{\models}$

Objective: Compute automatically A such that $M \models^{A} \varphi$ returns $\models$ or $\not\models$.

Abstraction Refinement Algorithm
Data: M, φ
while $\left(M \models^{A} \varphi \text{ } \mathbf{return} \overset{?}{\models}\right)$ **do**
  $A \Leftarrow A'$ where A' is "finer than" A

Which A' to choose ?
Can we **guide the refinement** using some relevant information ?

Motivation

$$M \models \varphi$$

Model

Property

Verification Algorithm, returns $\models$ or $\not\models$

Abstract Verification

$$M \models^{A} \varphi \qquad \text{returns } \models, \not\models, \overset{?}{\models}$$

Objective: Compute automatically A such that $M \models^{A} \varphi$ returns $\models$ or $\not\models$.

Counterexample Guided Abstraction Refinement (CEGAR) since 2000

while $\left(M \models^{A} \varphi \textbf{ return } \overset{?}{\models} \textit{ and } \lightning\right)$ **do**

$\quad A \Leftarrow A'$ where A' is "finer than" A and eliminates $\lightning$

If CEGAR & co fails and this happens, there is **no alternative algorithm**!

Fixpoint Guided Abstraction Refinement (FGAR)

- Problem: Invariant Checking
- Model: Transition Systems

Termination properties:

- always terminate when the invariant does not hold (negative instances)
- sufficient conditions when the invariant holds (positive instances)

→ **if CEGAR terminates then FGAR terminates**

And much more:

- easily combined with acceleration techniques
- formalized in the Galois connections framework (Moore-closed domain)
- termination not improved when using boolean closed domain

Background

Transition Systems

$$(C, T, I)$$

state space

transition relation $T \subseteq C \times C$

initial states $I \subseteq C$

Predicate Transformers

$Y = X \cup \mathit{post}(X)$: the successors of X and X

$Y = X \cap \widetilde{\mathit{pre}}(X)$: $Y \subseteq X$ and $\mathit{post}(Y) \subseteq X$

$\mathit{post}^*(X)$ the states reachable from X

$\widetilde{\mathit{pre}}^*(X)$ the states stuck in X

Fixpoint Checking Problem

Input: (C, T, I) and $S \subseteq C$

Question: $\mathit{post}^*(I) \subseteq S \quad \Leftrightarrow \quad I \subseteq \widetilde{\mathit{pre}}^*(S)$

difficult (or impossible) to evaluate in practice

Computing Overapproximations

$$\langle \wp(C), \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle A, \sqsubseteq \rangle$$

Let $f \in \wp(C) \mapsto \wp(C)$

$\alpha \circ f \circ \gamma$ is the (best) abstract counterpart of f

$\mathit{lfp}^{\sqsubseteq}(\alpha \circ f \circ \gamma)$ is the abstract counterpart of $\mathit{lfp}^{\subseteq}(f)$.

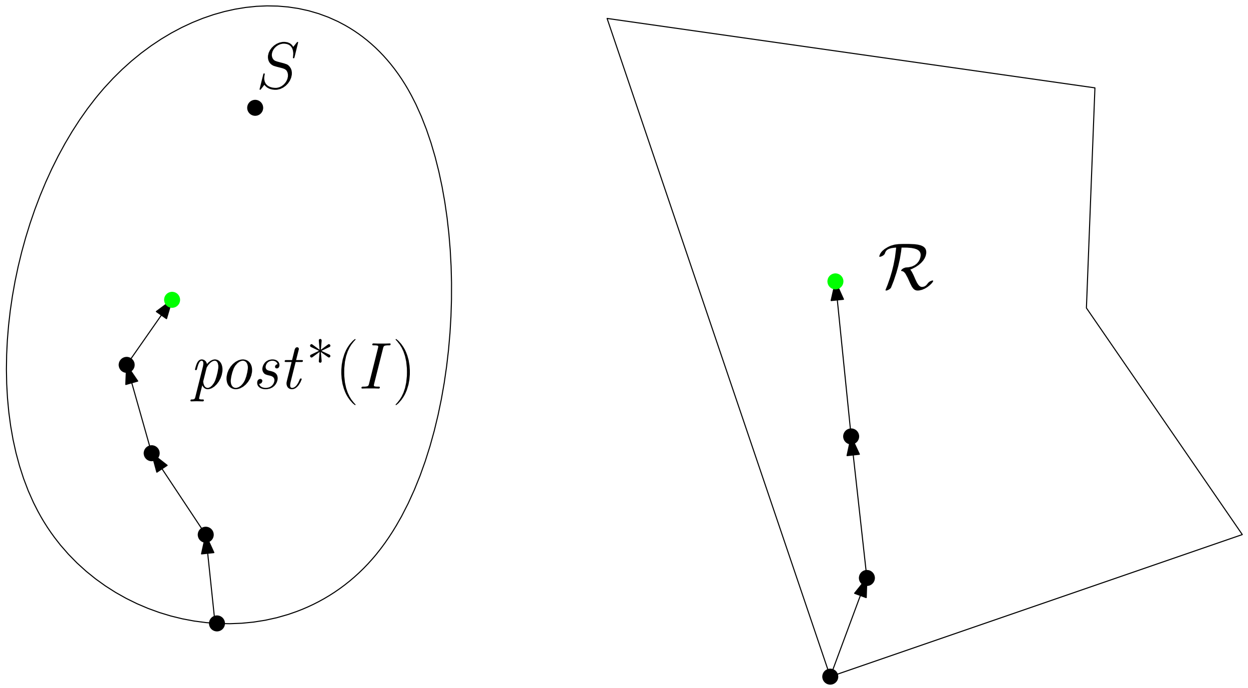
$$\mathit{lfp}^{\sqsubseteq}\lambda X.\, \alpha(I \cup \mathit{post}(\gamma(X))) \text{ and } \mathit{lfp}^{\subseteq}\lambda X.\, I \cup \mathit{post}(X) = \boldsymbol{post^*(I)}$$

$\mathit{gfp}^{\sqsubseteq}(\alpha \circ f \circ \gamma)$ is the abstract counterpart of $\mathit{gfp}^{\subseteq}(f)$.

$$\mathit{gfp}^{\sqsubseteq}\lambda X.\, \alpha(S \cap \widetilde{\mathit{pre}}(\gamma(X))) \text{ and } \mathit{gfp}^{\subseteq}\lambda X.\, S \cap \widetilde{\mathit{pre}}(X) = \boldsymbol{\widetilde{pre}^*(S)}$$

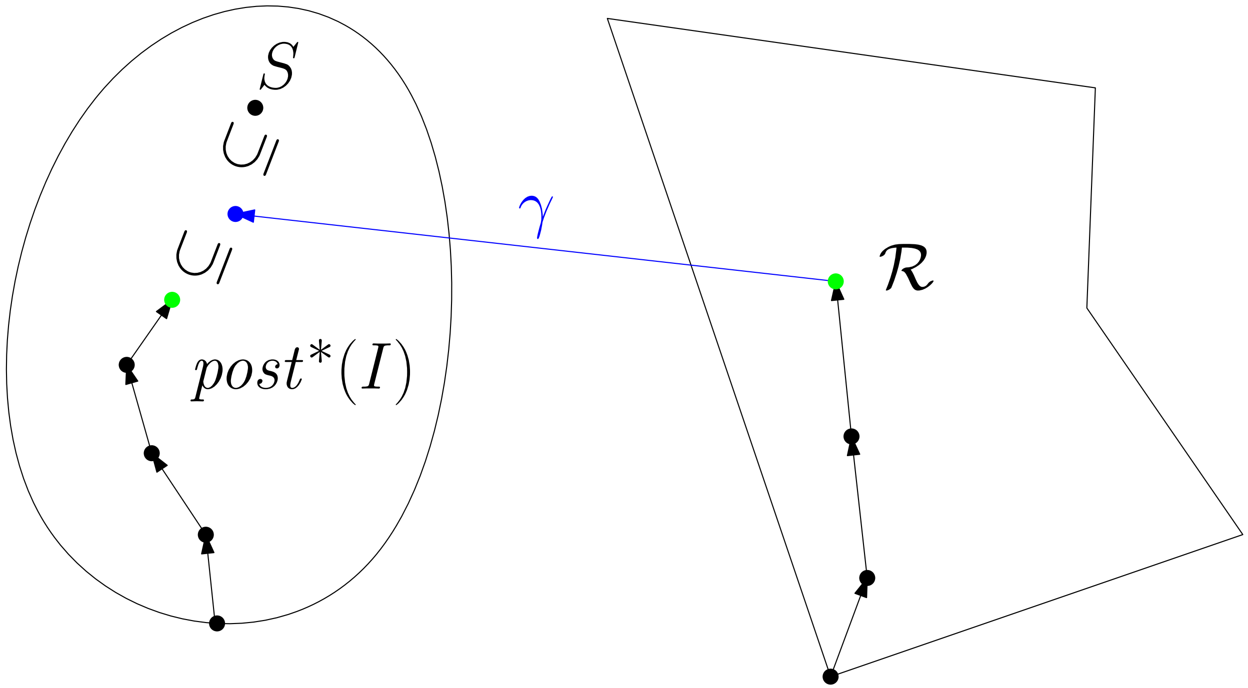
Overapproximations for positive instances

$post^*(I) \subseteq S$



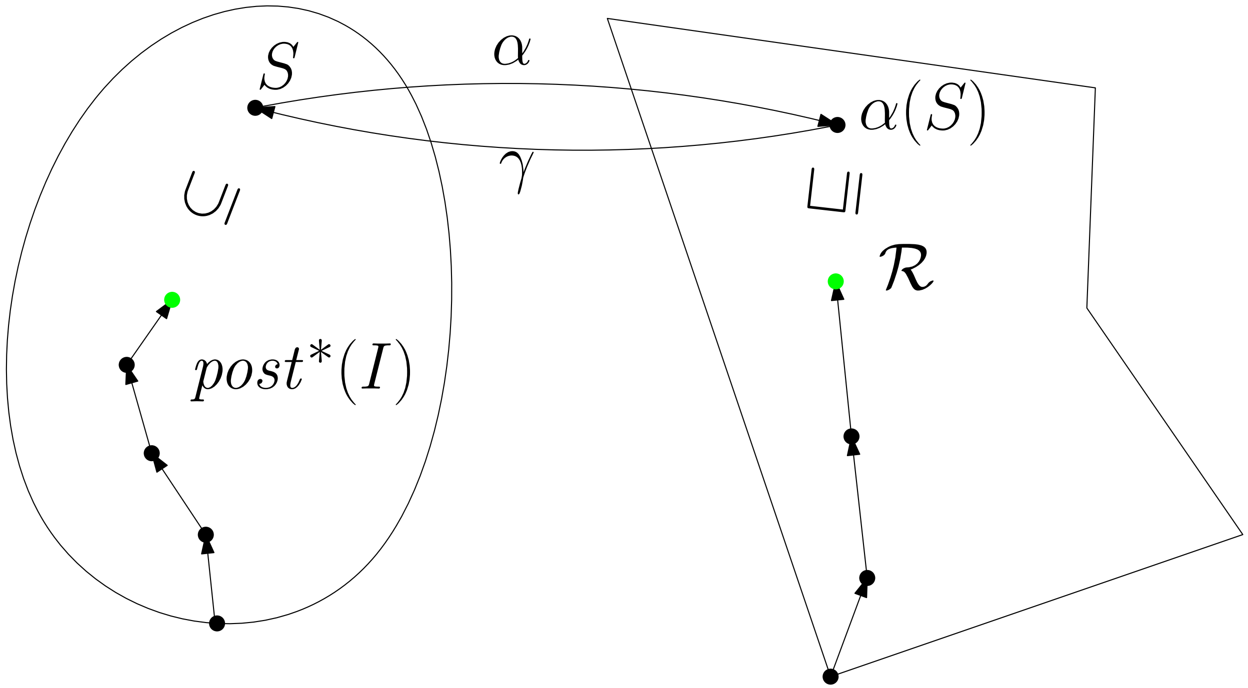
Overapproximations for positive instances

$post^*(I) \subseteq S$



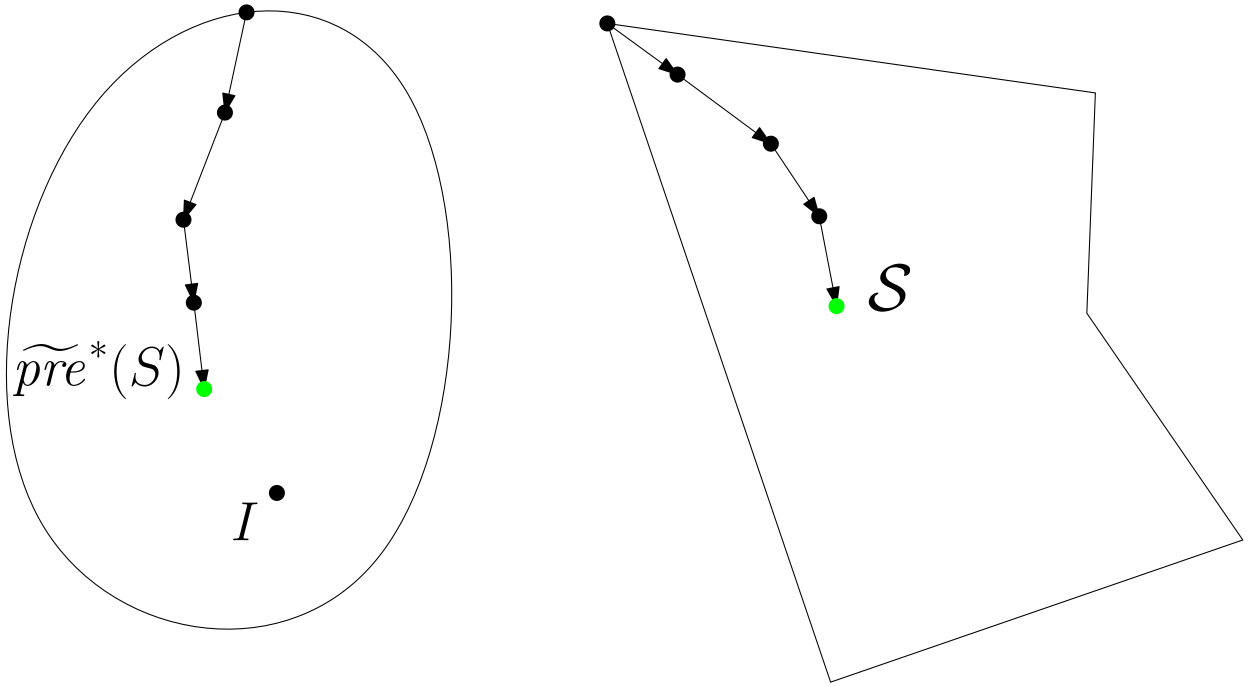
Overapproximations for positive instances

$post^*(I) \subseteq S$



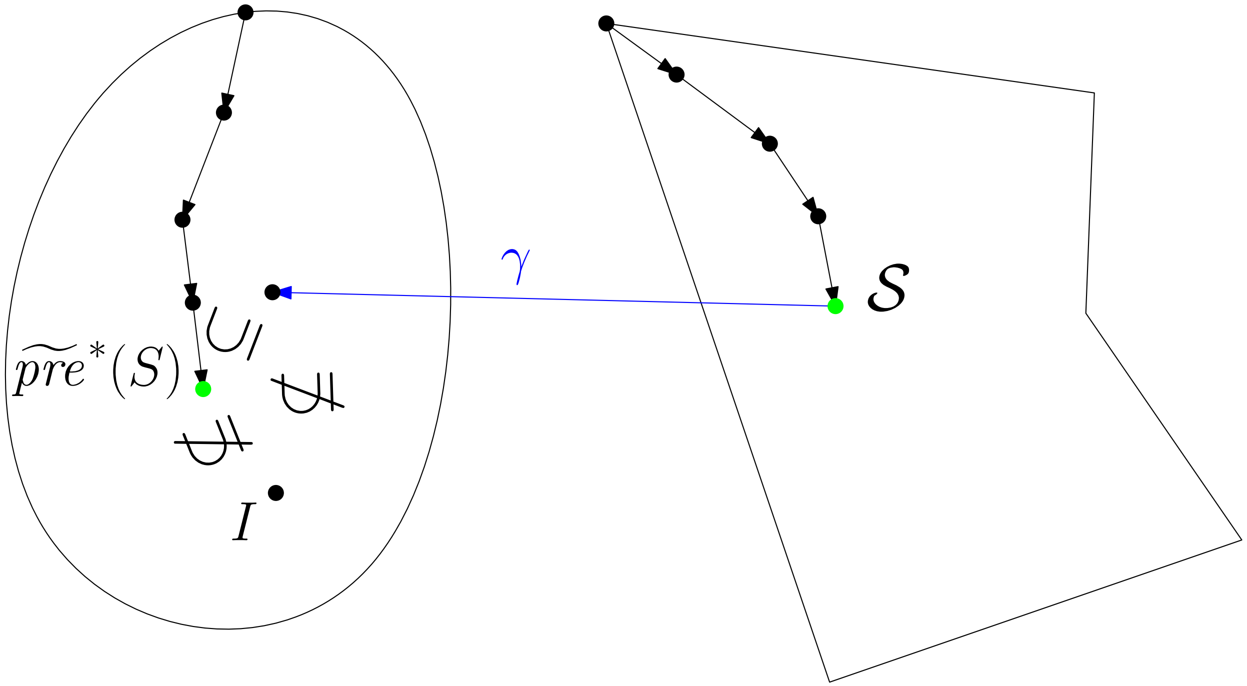
Overapproximations for negative instances

$$I \not\subseteq \widetilde{pre}^*(S)$$



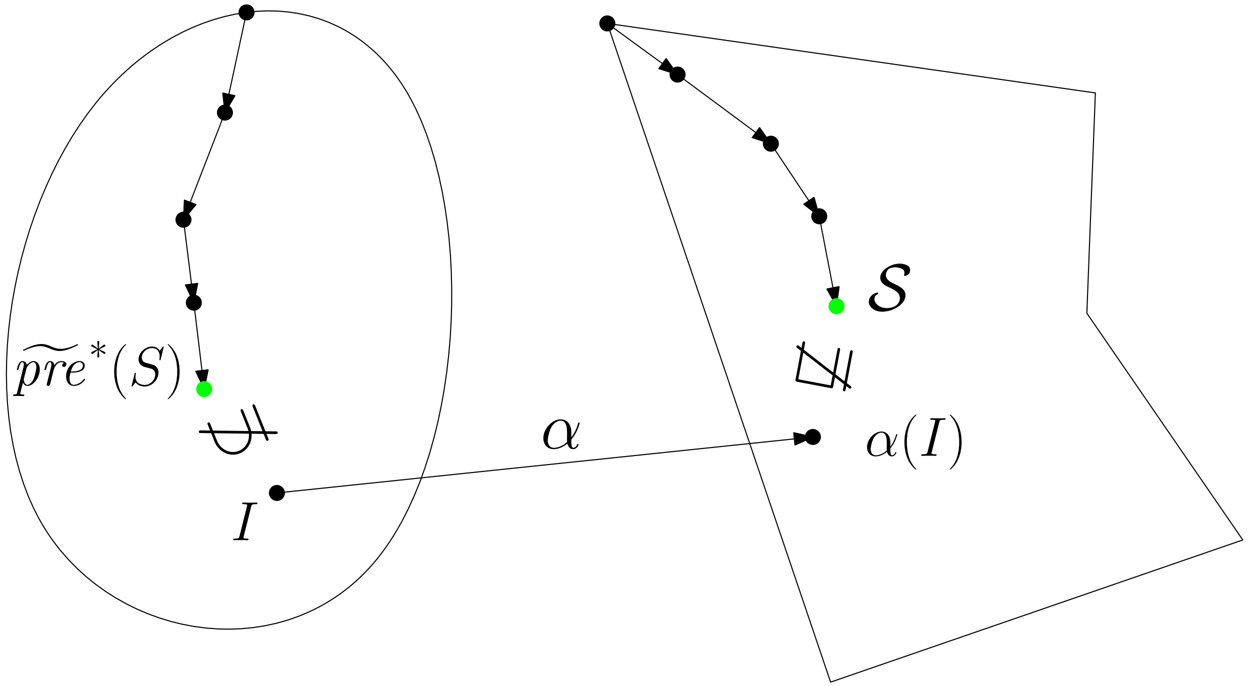
Overapproximations for negative instances

$$I \not\subseteq \widetilde{pre}^*(S)$$



Overapproximations for negative instances

$$I \not\subseteq \widetilde{pre}^*(S)$$



A first draft of the algorithm

Data: (C, T, I) and $S \subseteq C$ and an abstract domain A such that $\gamma \circ \alpha(S) = S$

Compute $\mathcal{R} = lfp^{\sqsubseteq} \lambda X.\, \alpha\big(I \cup post(\gamma(X))\big)$

if $\mathcal{R} \sqsubseteq \alpha(S)$ **then**

  return *positive instance* $(\models)$

else

  Compute $\mathcal{S} = gfp^{\sqsubseteq} \lambda X.\, \alpha\big(S \cap \widetilde{pre}(\gamma(X))\big)$

  if $\alpha(I) \not\sqsubseteq \mathcal{S}$ **then**

    return *negative instance* $(\not\models)$

  else

    return *Analyses are inconclusive* $(\overset{?}{\models})$

  end

end

Observations

- Analyses may be inconclusive.

  Hence, we need a more precise analysis.
  For this, we need a more precise abstract domain.
  Choice is guided by the inconclusive analyses: the fixpoints.

- Analyses are independant.

  Each analysis wants to benefit from the information computed so far.

  Combine the analyses

Combine the analysis

- $\widetilde{pre}^*(S)$

$$\forall R\colon\ post^*(I) \subseteq R$$

$$I \subseteq \widetilde{pre}^*(S) \qquad \Leftrightarrow \qquad I \subseteq \widetilde{pre}^*(S \cap R)$$

- $post^*(I)$

$$\forall S'\colon\ \widetilde{pre}^*(S) \subseteq S' \subseteq S$$

$$post^*(I) \subseteq S \qquad \Leftrightarrow \qquad post^*(I) \subseteq S'$$

$$\mathit{lfp}\lambda X.\, I \cup post(X) \subseteq S'$$

- Compute $R = \mathit{lfp}\lambda X.\, \big((I \cup post(X)) \cap S'\big)$
- Check $I \cup post(R) \subseteq S'$

Combine the analysis

- $\widetilde{pre}^*(S)$

Obtained from the forward analysis: $\gamma(\mathcal{R})$

$$\forall R\colon post^*(I) \subseteq R$$

$$I \subseteq \widetilde{pre}^*(S) \quad \Leftrightarrow \quad I \subseteq \widetilde{pre}^*(S \cap R)$$

- $post^*(I)$

$$\forall S'\colon \widetilde{pre}^*(S) \subseteq S' \subseteq S$$

$$post^*(I) \subseteq S \quad \Leftrightarrow \quad post^*(I) \subseteq S'$$

$$lfp\lambda X.\, I \cup post(X) \subseteq S'$$

- Compute $R = lfp\lambda X.\, \big((I \cup post(X)) \cap S'\big)$
- Check $I \cup post(R) \subseteq S'$

Combine the analysis

- $\widetilde{pre}^*(S)$

Obtained from the forward analysis: $\gamma(\mathcal{R})$

$$\forall R\colon post^*(I) \subseteq R$$

$$I \subseteq \widetilde{pre}^*(S) \quad \Leftrightarrow \quad I \subseteq \widetilde{pre}^*(S \cap R)$$

- $post^*(I)$

Obtained from the backward analysis: $\gamma(\mathcal{S})$

$$\forall S'\colon \widetilde{pre}^*(S) \subseteq S' \subseteq S$$

$$post^*(I) \subseteq S \quad \Leftrightarrow \quad post^*(I) \subseteq S'$$

$$lfp \lambda X.\, I \cup post(X) \subseteq S'$$

- Compute $R = lfp \lambda X. \big((I \cup post(X)) \cap S'\big)$
- Check $I \cup post(R) \subseteq S'$

A second draft of the algorithm

Data: (C, T, I) and $S \subseteq C$ and an abstract domain A such that
$\gamma \circ \alpha(S) = S$

Compute $\mathcal{R} = lfp^{\sqsubseteq} \lambda X.\, \alpha\big(I \cup post(\gamma(X)) \cap S\big)$

if $\alpha\big(I \cup post(\gamma(\mathcal{R}))\big) \sqsubseteq \alpha(S)$ **then**

  return *positive instance* $(\models)$

else

  Compute $\mathcal{S} = gfp^{\sqsubseteq} \lambda X.\, \alpha\big(\gamma(\mathcal{R}) \cap \widetilde{pre}(\gamma(X))\big)$

  if $\alpha(I) \not\sqsubseteq \mathcal{S}$ **then**

    return *negative instance* $(\not\models)$

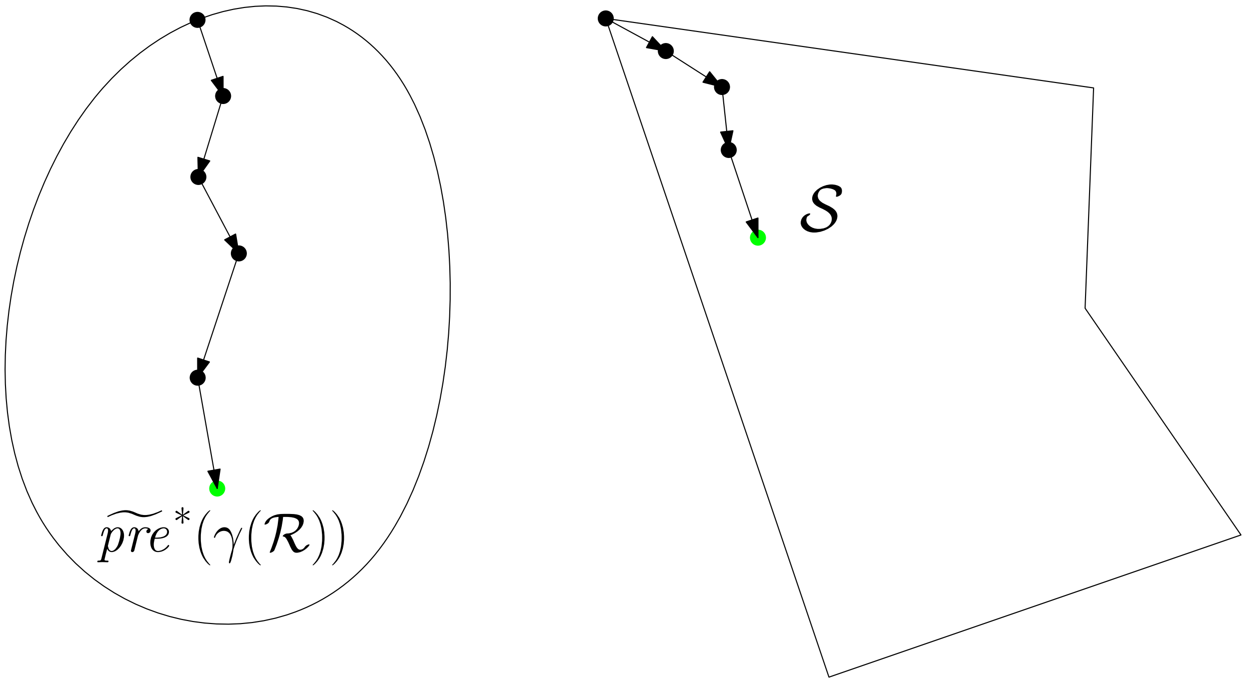
  else

    return *Analyses are inconclusive* $(\overset{?}{\models})$

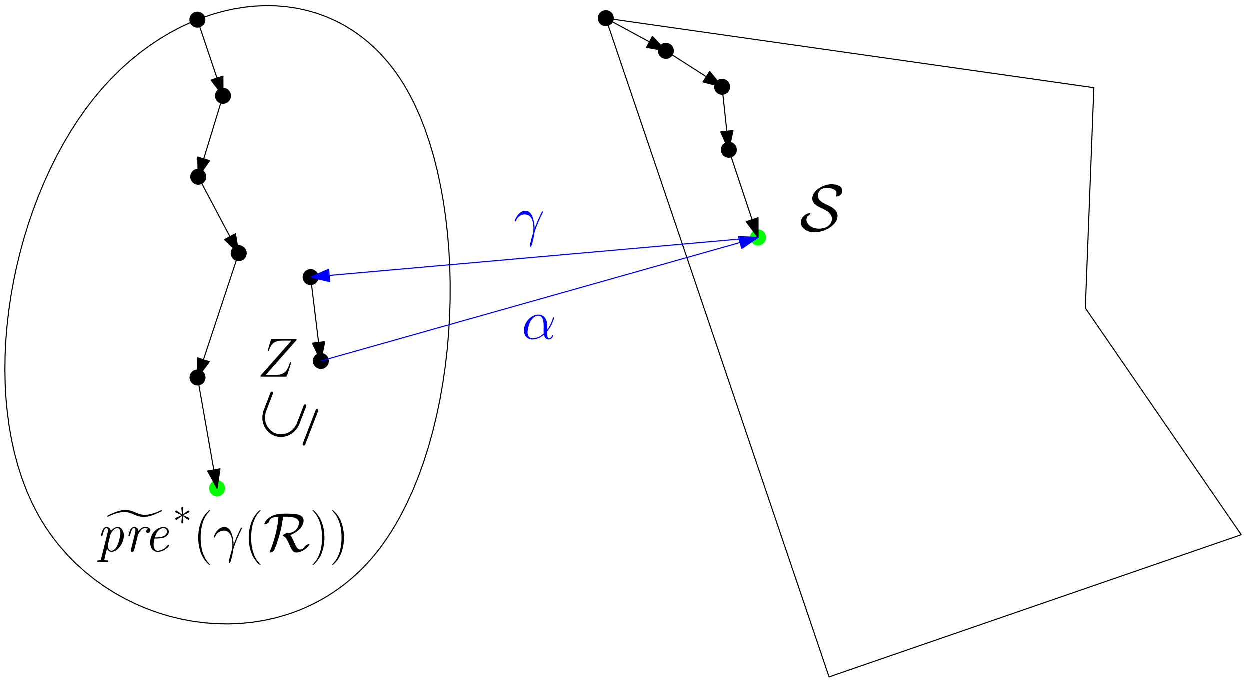
  end

end

Refining the abstract domain

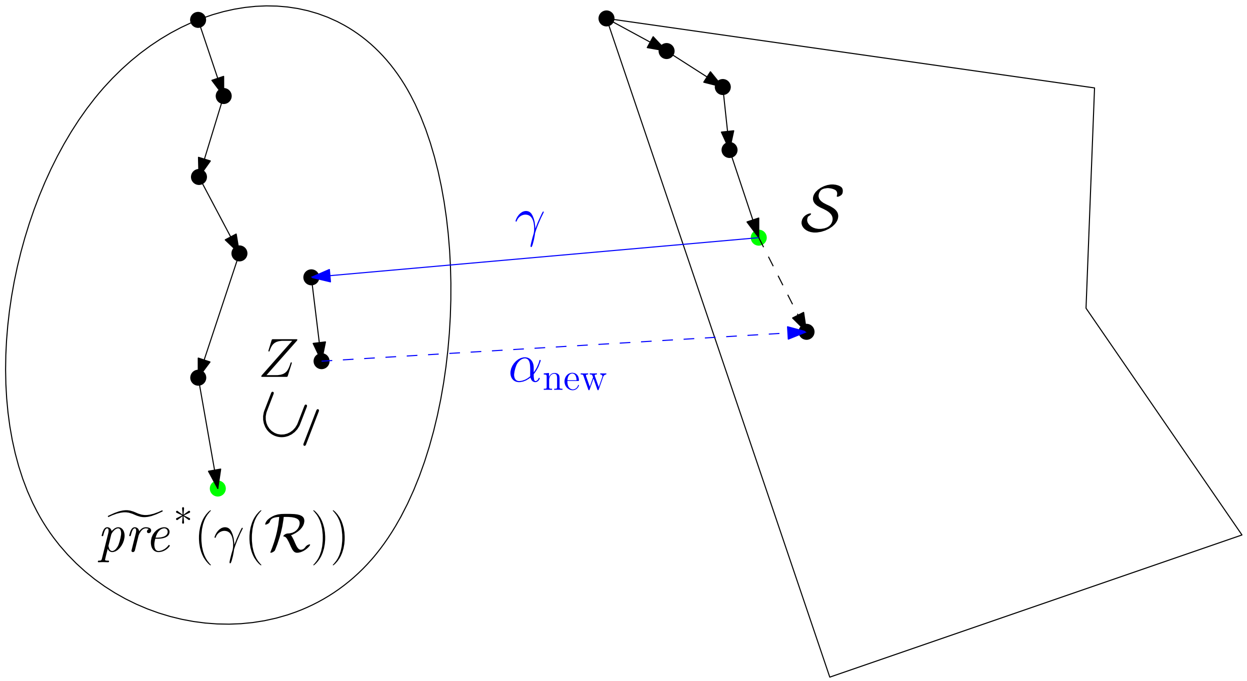


Refining the abstract domain



$$Z = \gamma(\mathcal{S}) \cap \widetilde{pre}(\gamma(\mathcal{S}))$$

Refining the abstract domain



$$Z = \gamma(\mathcal{S}) \cap \widetilde{pre}(\gamma(\mathcal{S}))$$

$$\gamma_{\text{new}} \circ \alpha_{\text{new}}(Z) = Z$$

FGAR

Data: (C, T, I) and $S \subseteq C$ and an abstract domain A_0 such that
$\gamma_0 \circ \alpha_0(S) = S$

$Z_0 = S$

for $i = 0, 1, 2, 3, \ldots$ **do**

  Compute $\mathcal{R}_i = lfp^{\sqsubseteq} \lambda X.\, \alpha_i\Big(\big(I \cup post(\gamma_i(X))\big) \cap Z_i\Big)$

  if $\alpha_i\big(I \cup post(\gamma_i(\mathcal{R}_i))\big) \sqsubseteq \alpha_i(Z_i)$ **then**

    return *positive instance* $(\models)$

  else

    Compute $\mathcal{S}_i = gfp^{\sqsubseteq} \lambda X.\, \alpha_i\big(\gamma_i(\mathcal{R}_i) \cap \widetilde{pre}(\gamma_i(X))\big)$

    if $\alpha_i(I) \sqsubseteq \mathcal{S}_i$ **then**

      Let $Z_{i+1} = \gamma_i(\mathcal{S}_i) \cap \widetilde{pre}(\gamma_i(\mathcal{S}_i))$

      Let A_{i+1} be s.t. $\gamma_{i+1}(A_{i+1}) \supseteq \{Z_{i+1}\} \cup \gamma_i(A_i)$

    else

      return *negative instance* $(\not\models)$

    end

  end

end

FGAR

Data: (C, T, I) and $S \subseteq C$ and an abstract domain A_0 such that
$\gamma_0 \circ \alpha_0(S) = S$

$Z_0 = S$

for $i = 0, 1, 2, 3, \ldots$ **do**

Compute $\mathcal{R}_i = \mathit{lfp}^{\sqsubseteq} \lambda X.\, \alpha_i\Big(\big(I \cup \mathit{post}(\gamma_i(X))\big) \cap Z_i\Big)$

if $\alpha_i\big(I \cup \mathit{post}(\gamma_i(\mathcal{R}_i))\big) \sqsubseteq \alpha_i(Z_i)$ **then**

return *positive instance* ($\models$)

else

Compute $\mathcal{S}_i = \mathit{gfp}^{\sqsubseteq} \lambda X.\, \alpha_i\big(\gamma_i(\mathcal{R}_i) \cap \widetilde{\mathit{pre}}(\gamma_i(X))\big)$

if $\alpha_i(I) \sqsubseteq \mathcal{S}_i$ **then**

Let $Z_{i+1} = \gamma_i(\mathcal{S}_i) \cap \widetilde{\mathit{pre}}(\gamma_i(\mathcal{S}_i))$

Let A_{i+1} be s.t. $\gamma_{i+1}(A_{i+1}) \supseteq \{Z_{i+1}\} \cup \gamma_i(A_i)$

else

return *negative instance* ($\not\models$)

end

end

end

Data: (C, T, I) and $S \subseteq C$ and an abstract domain A_0 such that $\gamma_0 \circ \alpha_0(S) = S$

$Z_0 = S$

for $i = 0, 1, 2, 3, \ldots$ **do**

  Compute $\mathcal{R}_i = \mathit{lfp}^{\sqsubseteq} \lambda X.\, \alpha_i\Big(\big(I \cup \mathit{post}(\gamma_i(X))\big) \cap Z_i\Big)$

  if $\alpha_i\big(I \cup \mathit{post}(\gamma_i(\mathcal{R}_i))\big) \sqsubseteq \alpha_i(Z_i)$ **then**

    return *positive instance* ($\models$)

  else

    Compute $\mathcal{S}_i = \mathit{gfp}^{\sqsubseteq} \lambda X.\, \alpha_i\big(\gamma_i(\mathcal{R}_i) \cap \widetilde{\mathit{pre}}(\gamma_i(X))\big)$

    if $\alpha_i(I) \sqsubseteq \mathcal{S}_i$ **then**

      Let $Z_{i+1} = \gamma_i(\mathcal{S}_i) \cap \widetilde{\mathit{pre}}(\gamma_i(\mathcal{S}_i))$

      Let A_{i+1} be s.t. $\gamma_{i+1}(A_{i+1}) \supseteq \{Z_{i+1}\} \cup \gamma_i(A_i)$

    else

      return *negative instance* ($\not\models$)

    end

  end

end

```
Data: (C, T, I) and S ⊆ C and an abstract domain A_0 such that
      γ_0 ∘ α_0(S) = S
Z_0 = S
for i = 0, 1, 2, 3, ... do
    Compute R_i = lfp^⊑ λX. α_i( (I ∪ post(γ_i(X))) ∩ Z_i )
    if α_i( I ∪ post(γ_i(R_i)) ) ⊑ α_i(Z_i) then
        return positive instance (⊨)
    else
        Compute S_i = gfp^⊑ λX. α_i( γ_i(R_i) ∩ p̃re(γ_i(X)) )
        if α_i(I) ⊑ S_i then
            Let Z_{i+1} = γ_i(S_i) ∩ p̃re(γ_i(S_i))
            Let A_{i+1} be s.t. γ_{i+1}(A_{i+1}) ⊇ {Z_{i+1}} ∪ γ_i(A_i)
        else
            return negative instance (⊭)
        end
    end
end
```

Z_0

Data: (C, T, I) and $S \subseteq C$ and an abstract domain A_0 such that $\gamma_0 \circ \alpha_0(S) = S$

$Z_0 = S$

for $i = 0, 1, 2, 3, \ldots$ **do**

  Compute $\mathcal{R}_i = \mathit{lfp}^{\sqsubseteq} \lambda X.\, \alpha_i\Big(\big(I \cup \mathit{post}(\gamma_i(X))\big) \cap Z_i\Big)$

  if $\alpha_i\big(I \cup \mathit{post}(\gamma_i(\mathcal{R}_i))\big) \sqsubseteq \alpha_i(Z_i)$ **then**

    return *positive instance* ($\models$)

  else

    Compute $\mathcal{S}_i = \mathit{gfp}^{\sqsubseteq} \lambda X.\, \alpha_i\big(\gamma_i(\mathcal{R}_i) \cap \widetilde{\mathit{pre}}(\gamma_i(X))\big)$

    if $\alpha_i(I) \sqsubseteq \mathcal{S}_i$ **then**

      Let $Z_{i+1} = \gamma_i(\mathcal{S}_i) \cap \widetilde{\mathit{pre}}(\gamma_i(\mathcal{S}_i))$

      Let A_{i+1} be s.t. $\gamma_{i+1}(A_{i+1}) \supseteq \{Z_{i+1}\} \cup \gamma_i(A_i)$

    else

      return *negative instance* ($\not\models$)

    end

  end

end

$$Z_0 \supseteq \gamma(\mathcal{R}_0)$$

```
Data: (C, T, I) and S ⊆ C and an abstract domain A_0 such that
      γ_0 ∘ α_0(S) = S
Z_0 = S
for i = 0, 1, 2, 3, . . . do
    Compute R_i = lfp^⊑ λX. α_i((I ∪ post(γ_i(X))) ∩ Z_i)
    if α_i(I ∪ post(γ_i(R_i))) ⊑ α_i(Z_i) then
        return positive instance (⊨)
    else
        Compute S_i = gfp^⊑ λX. α_i(γ_i(R_i) ∩ p̃re(γ_i(X)))
        if α_i(I) ⊑ S_i then
            Let Z_{i+1} = γ_i(S_i) ∩ p̃re(γ_i(S_i))
            Let A_{i+1} be s.t. γ_{i+1}(A_{i+1}) ⊇ {Z_{i+1}} ∪ γ_i(A_i)
        else
            return negative instance (⊭)
        end
    end
end
```

$$Z_0 \supseteq \gamma(\mathcal{R}_0) \supseteq \gamma(\mathcal{S}_0)$$

Data: (C, T, I) and $S \subseteq C$ and an abstract domain A_0 such that $\gamma_0 \circ \alpha_0(S) = S$

$Z_0 = S$

for $i = 0, 1, 2, 3, \ldots$ **do**

- Compute $\mathcal{R}_i = lfp^{\sqsubseteq} \lambda X.\, \alpha_i\Big(\big(I \cup post(\gamma_i(X))\big) \cap Z_i\Big)$
- **if** $\alpha_i\big(I \cup post(\gamma_i(\mathcal{R}_i))\big) \sqsubseteq \alpha_i(Z_i)$ **then**
  - **return** *positive instance* $(\models)$
- **else**
  - Compute $\mathcal{S}_i = gfp^{\sqsubseteq} \lambda X.\, \alpha_i\big(\gamma_i(\mathcal{R}_i) \cap \widetilde{pre}(\gamma_i(X))\big)$
  - **if** $\alpha_i(I) \sqsubseteq \mathcal{S}_i$ **then**
    - Let $Z_{i+1} = \gamma_i(\mathcal{S}_i) \cap \widetilde{pre}(\gamma_i(\mathcal{S}_i))$
    - Let A_{i+1} be s.t. $\gamma_{i+1}(A_{i+1}) \supseteq \{Z_{i+1}\} \cup \gamma_i(A_i)$
  - **else**
    - **return** *negative instance* $(\not\models)$
  - **end**
- **end**

end

$$Z_0 \supseteq \gamma(\mathcal{R}_0) \supseteq \gamma(\mathcal{S}_0) \supseteq Z_1$$

Data: (C, T, I) and $S \subseteq C$ and an abstract domain A_0 such that $\gamma_0 \circ \alpha_0(S) = S$

$Z_0 = S$

for $i = 0, 1, 2, 3, \ldots$ **do**

  Compute $\mathcal{R}_i = \mathit{lfp}^{\sqsubseteq}\lambda X.\, \alpha_i\Big(\big(I \cup \mathit{post}(\gamma_i(X))\big) \cap Z_i\Big)$

  if $\alpha_i\big(I \cup \mathit{post}(\gamma_i(\mathcal{R}_i))\big) \sqsubseteq \alpha_i(Z_i)$ **then**

    return *positive instance* ($\models$)

  else

    Compute $\mathcal{S}_i = \mathit{gfp}^{\sqsubseteq}\lambda X.\, \alpha_i\big(\gamma_i(\mathcal{R}_i) \cap \widetilde{\mathit{pre}}(\gamma_i(X))\big)$

    if $\alpha_i(I) \sqsubseteq \mathcal{S}_i$ **then**

      Let $Z_{i+1} = \gamma_i(\mathcal{S}_i) \cap \widetilde{\mathit{pre}}(\gamma_i(\mathcal{S}_i))$

      Let A_{i+1} be s.t. $\gamma_{i+1}(A_{i+1}) \supseteq \{Z_{i+1}\} \cup \gamma_i(A_i)$

    else

      return *negative instance* ($\not\models$)

    end

  end

end

$Z_0 \supseteq \gamma(\mathcal{R}_0) \supseteq \gamma(\mathcal{S}_0) \supseteq Z_1 \supseteq \gamma(\mathcal{R}_1) \supseteq \gamma(\mathcal{S}_1) \cdots \gamma(\mathcal{R}_i) \supseteq \gamma(\mathcal{S}_i) \supseteq Z_{i+1}$

- If $I \not\subseteq \widetilde{\mathit{pre}}^*(S)$ you eventually add $Z_{i+1} \not\supseteq I$ and the backward analysis concludes: $\alpha_{i+1}(I) \not\sqsubseteq \mathcal{S}$.
- If $\mathit{post}^*(I) \subseteq S$ you possibly add $I \subseteq Z_{i+1}$ s.t. $\mathit{post}(Z_{i+1}) \subseteq Z_{i+1}$ and the forward analysis concludes: $\alpha\big(I \cup \mathit{post}(\gamma(\mathcal{R}))\big) \sqsubseteq \alpha(Z_{i+1})$

Properties of FGAR

Correctness

Termination

- Always terminate for negative instances
- Sufficient termination conditions for positive instances
    - Descending Chain Condition on $\langle \wp(C), \subseteq \rangle$
    - At ith iteration, $\widetilde{pre}^*(Z_i) = \bigcap_{j=0}^{k} \widetilde{pre}^j(Z_i)$ where $k \in \mathbb{N}$
    - CEGAR terminates

Abstract Domain

The abstract domains of FGAR (Moore-closed) are more general than the abstract domains of CEGAR (partitions)

Btw, partitions does not improve FGAR termination

Integration of Acceleration Techniques

$$Z = \gamma(\mathcal{S}) \cap \widetilde{pre}[R](\gamma(\mathcal{S})) \text{ where } T \subseteq R \subseteq T^*$$

Conclusions

FGAR is an **alternative to** CEGAR which

- do **invariant verification** (CEGAR does more)
- do not rely on **counterexamples**
- is theoretical (but has been instantiated and experimented SAS'05, VMCAI'06, ICATPN'07, special PN2007 of Funda. Inf.) full details in my PhD thesis.
- easily extends to infinite abstract domains using extrapolation techniques.

Future Work

- Effective and manipulable representations of enriched abstract domains
- Define a similar algorithm more friendly towards the classical abstract domain.