The Power of Parameterization in Coinductive Proof

Chung-Kil Hur
Microsoft Research Cambridge

Georg Neis, Derek Dreyer, Viktor Vafeiadis
MPI-SWS

POPL 2013
23 Jan 2013

Two Principles for Coinduction

- Tarski's Fixed Point Theorem
  + Simple & Robust
  - Inconvenient to use

- Syntactically Guarded Coinduction
  - Complex & Fragile due to "Guardedness Checking"
  + More convenient to use

Our Contribution

- Parameterized Coinduction

  + Simple & Robust

  + Most convenient to use

Key Idea

Semantic Guardedness

- Parameterized Coinduction

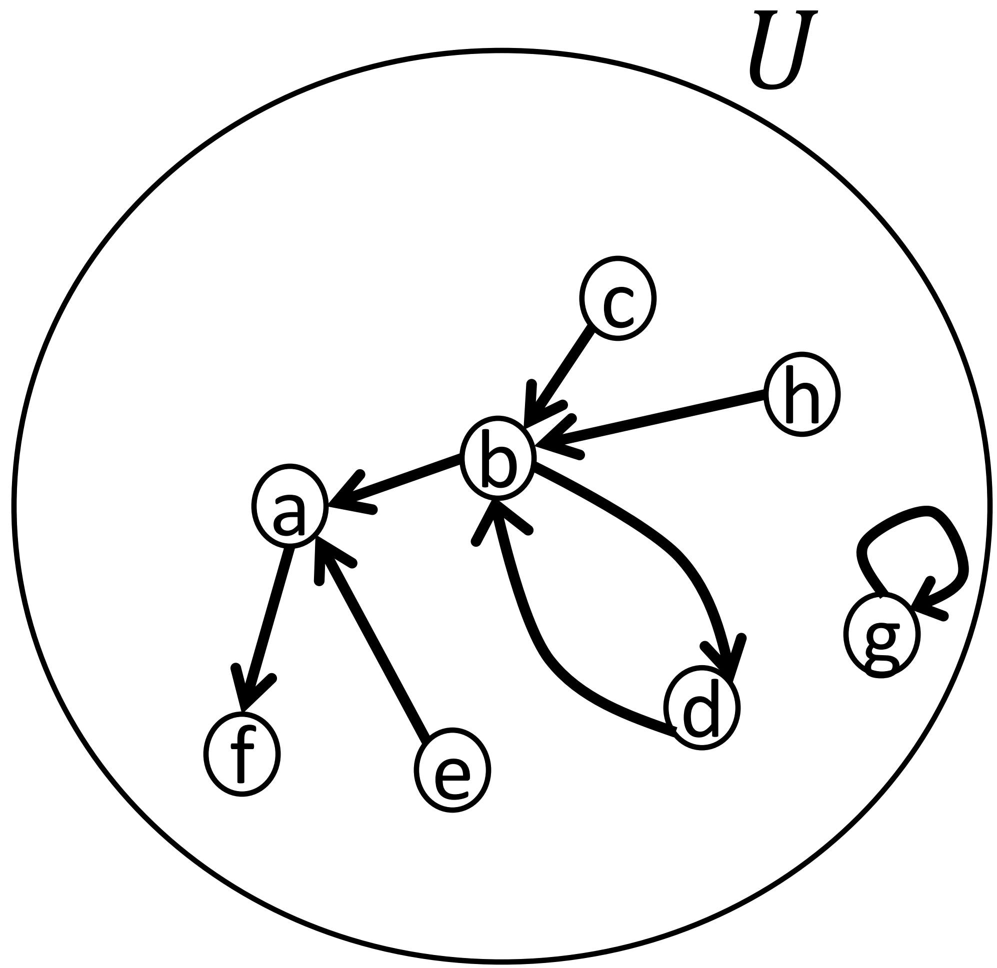
  + Simple & Robust

  + Most convenient to use

Talk Outline

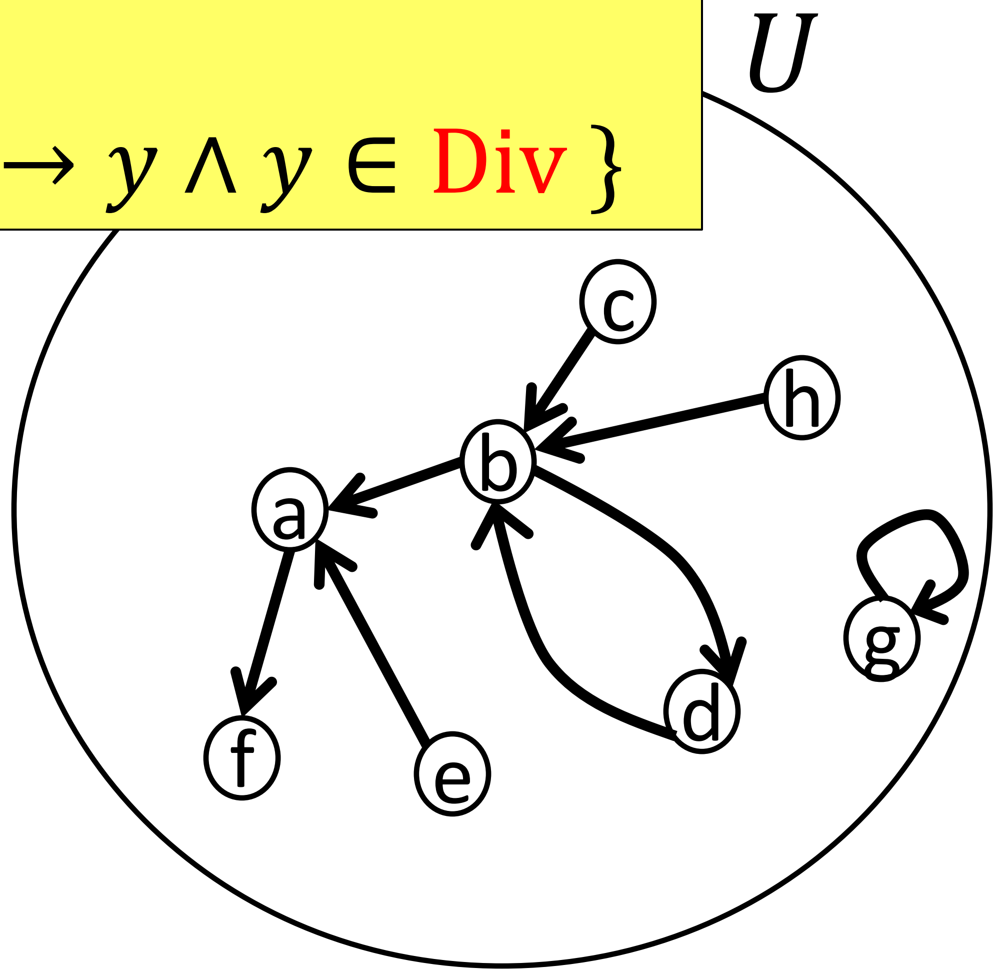
- Previous Approaches
  - Tarski’s Fixed Point Theorem
  - Syntactically Guarded Coinduction
- Our Approach
  - Parameterized Coinduction

Example of Recursively Defined Set: Divergent Nodes in STS



Example of Recursively Defined Set: Divergent Nodes in STS

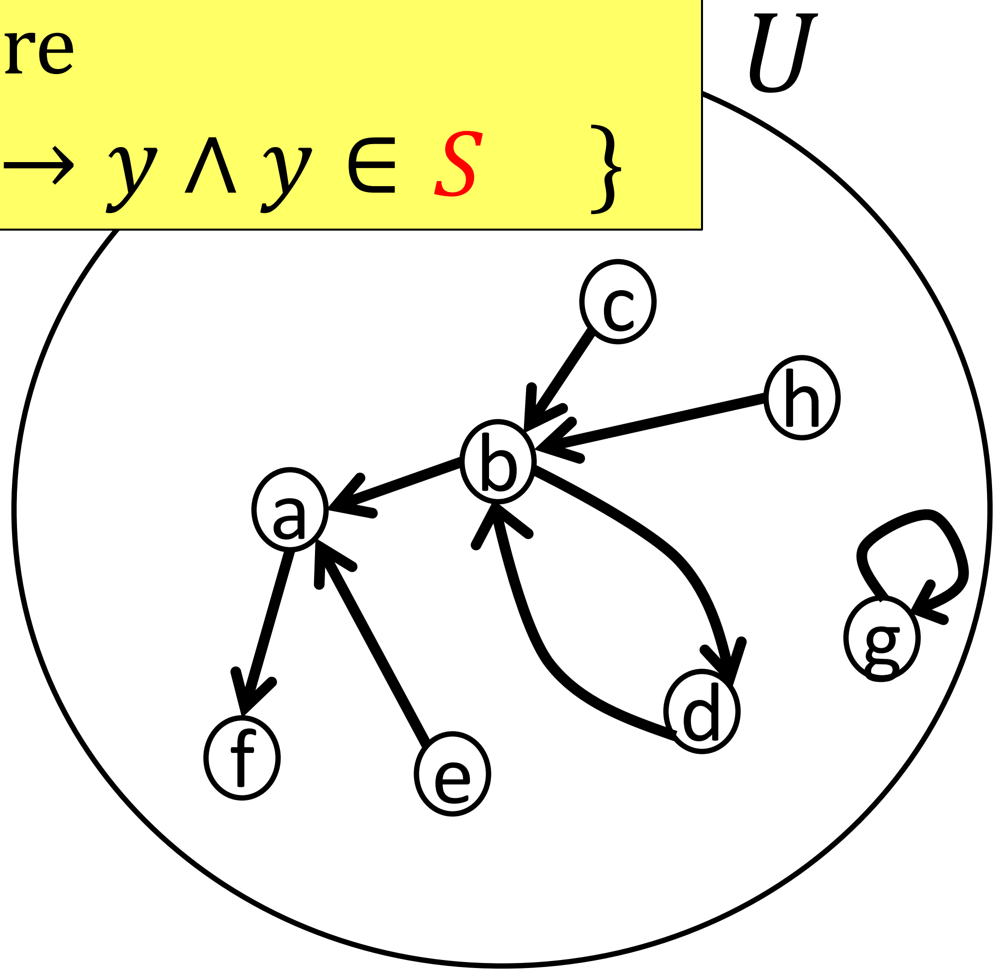
$$\mathrm{Div} = \{\, x \mid \exists y.\ x \to y \wedge y \in \mathrm{Div} \,\}$$



Example of Recursively Defined Set: Divergent Nodes in STS

$$\mathrm{Div} = f(\mathrm{Div}) \quad \text{where}$$

$$f(S) = \{\, x \mid \exists y.\ x \to y \land y \in S \,\}$$



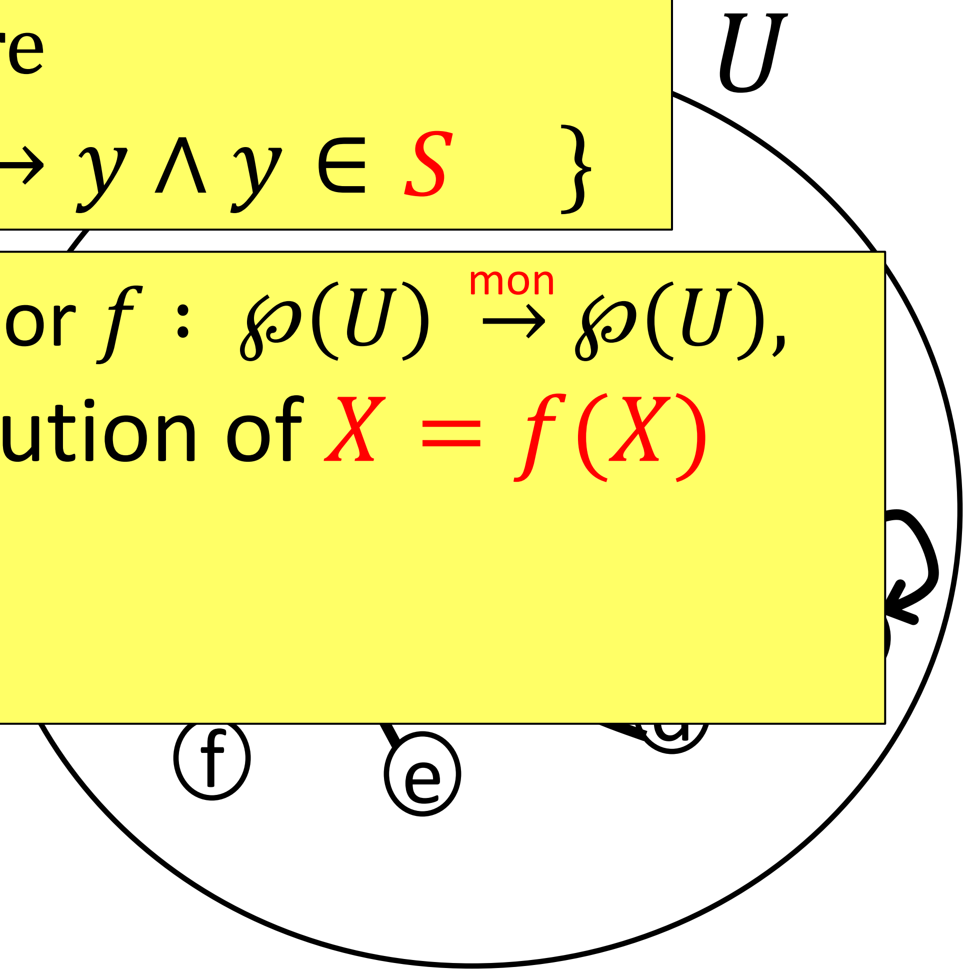
Example of Recursively Defined Set: Divergent Nodes in STS

$\mathrm{Div} = f(\mathrm{Div})$ where

$f(S) = \{\, x \mid \exists y.\ x \to y \wedge y \in S \,\}$

(Tarski's Theorem) For $f : \wp(U) \xrightarrow{\text{mon}} \wp(U)$,

νf is the greatest solution of $X = f(X)$



Example of Recursively Defined Set: Divergent Nodes in STS

$\mathrm{Div} = f(\mathrm{Div})$ where

$$f(S) = \{\, x \mid \exists y.\ x \to y \land y \in S \,\}$$

U

(Tarski's Theorem) For $f : \wp(U) \xrightarrow{\text{mon}} \wp(U)$,

νf is the greatest solution of $X = f(X)$

$$\nu f = \bigcup \{\, S \mid \boxed{S \subseteq f(S)} \,\}$$

S is consistent

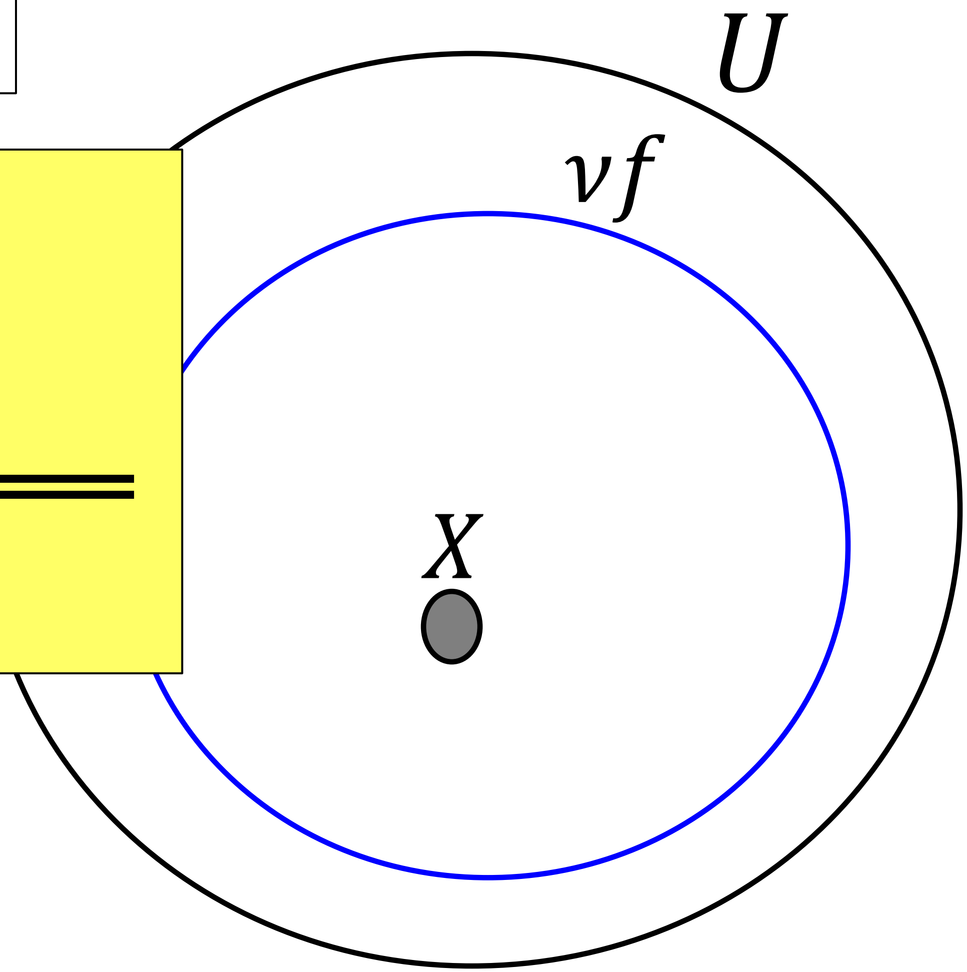
e

Tarski's Principle

$$f : \wp(U) \xrightarrow{\text{mon}} \wp(U)$$

(Tarski's Principle)

$$\frac{\qquad}{X \subseteq \nu f}$$

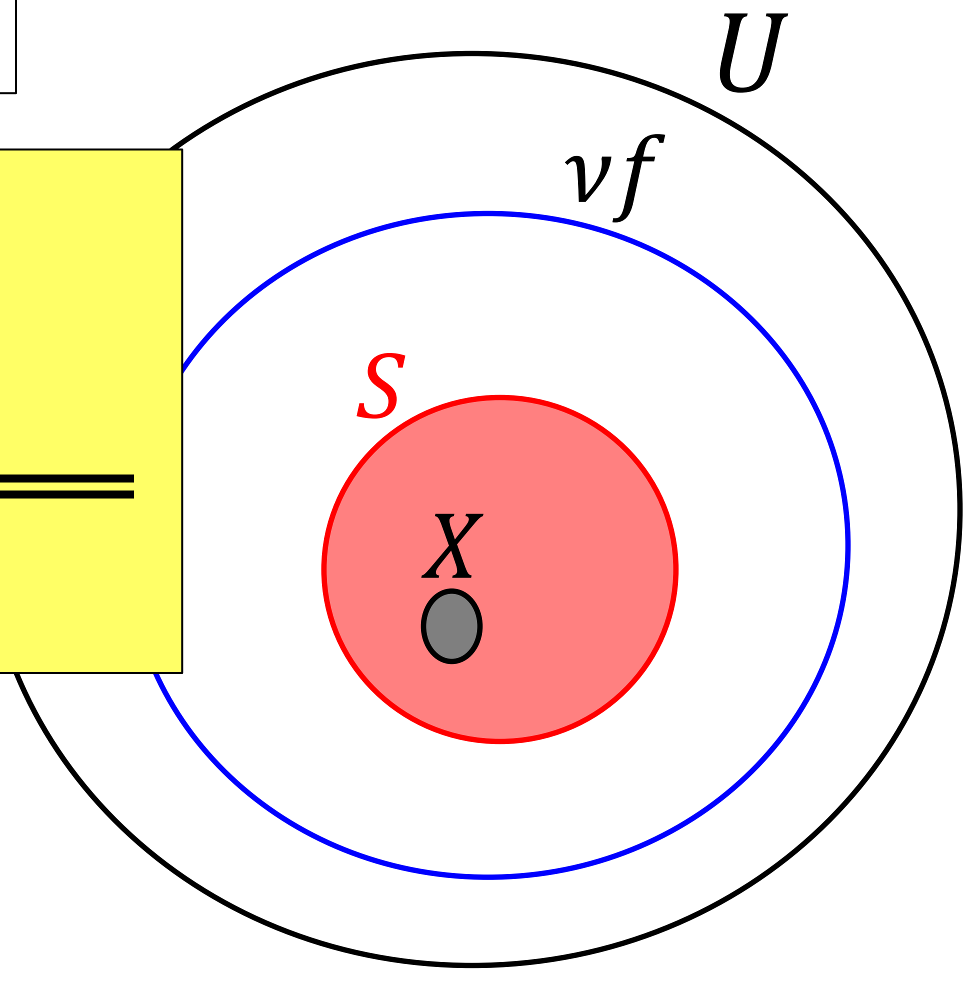


Tarski's Principle

$$f : \wp(U) \xrightarrow{\text{mon}} \wp(U)$$

(Tarski's Principle)

$$\frac{\exists\, S.\ X \subseteq S}{X \subseteq \nu f}$$

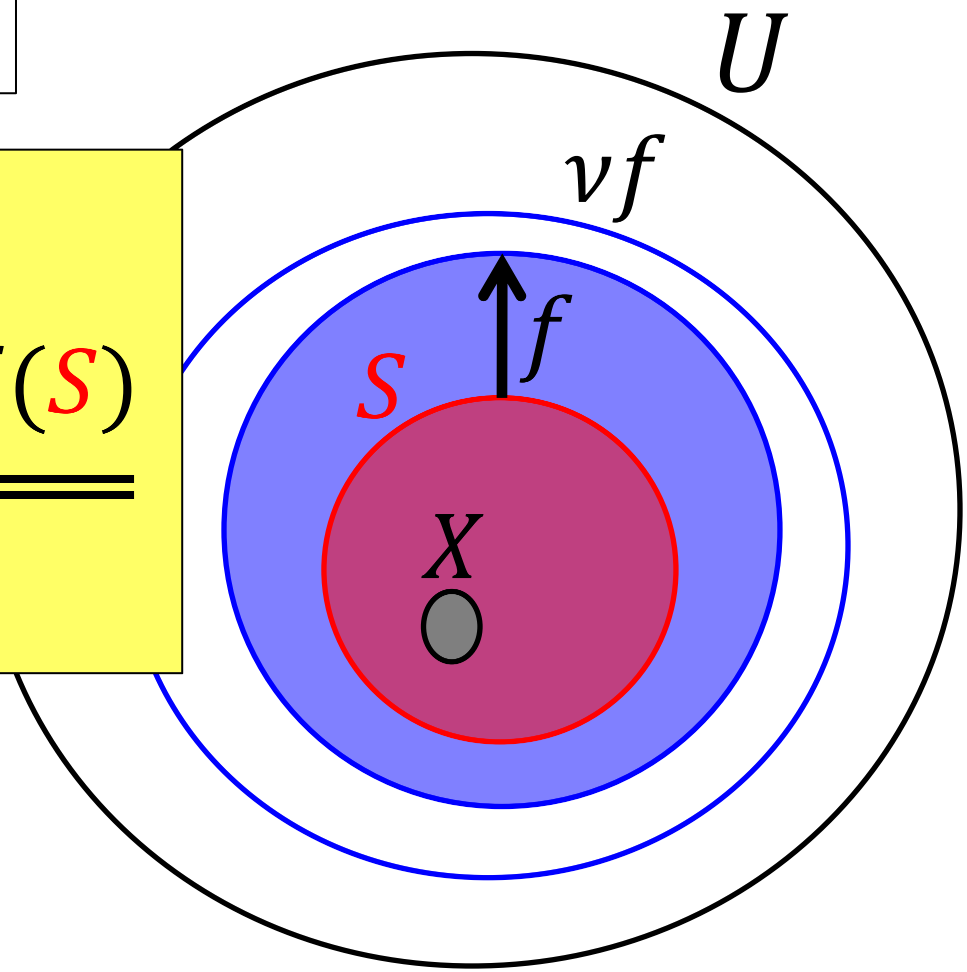


Tarski's Principle

$$f : \wp(U) \xrightarrow{\text{mon}} \wp(U)$$

(Tarski's Principle)

$$\frac{\exists\, S.\ X \subseteq S \ \wedge\ S \subseteq f(S)}{X \subseteq \nu f}$$

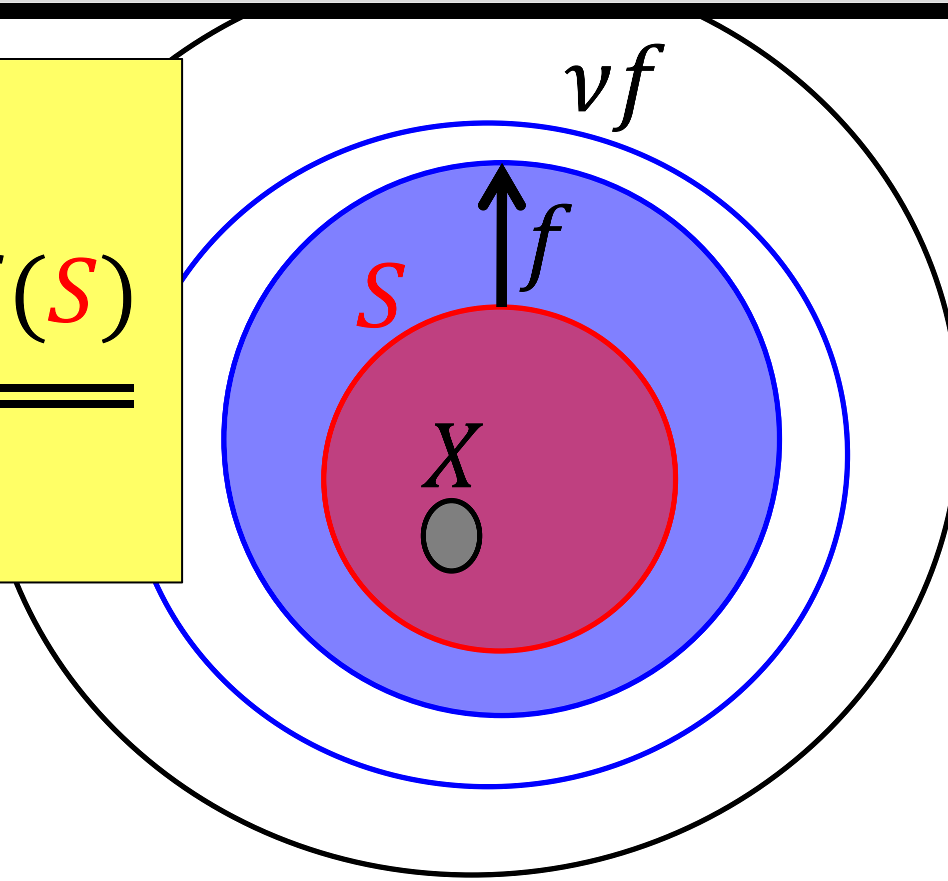


Tarski's Principle

Sound by Definition: $\nu f = \bigcup \{ S \mid S \subseteq f(S) \}$

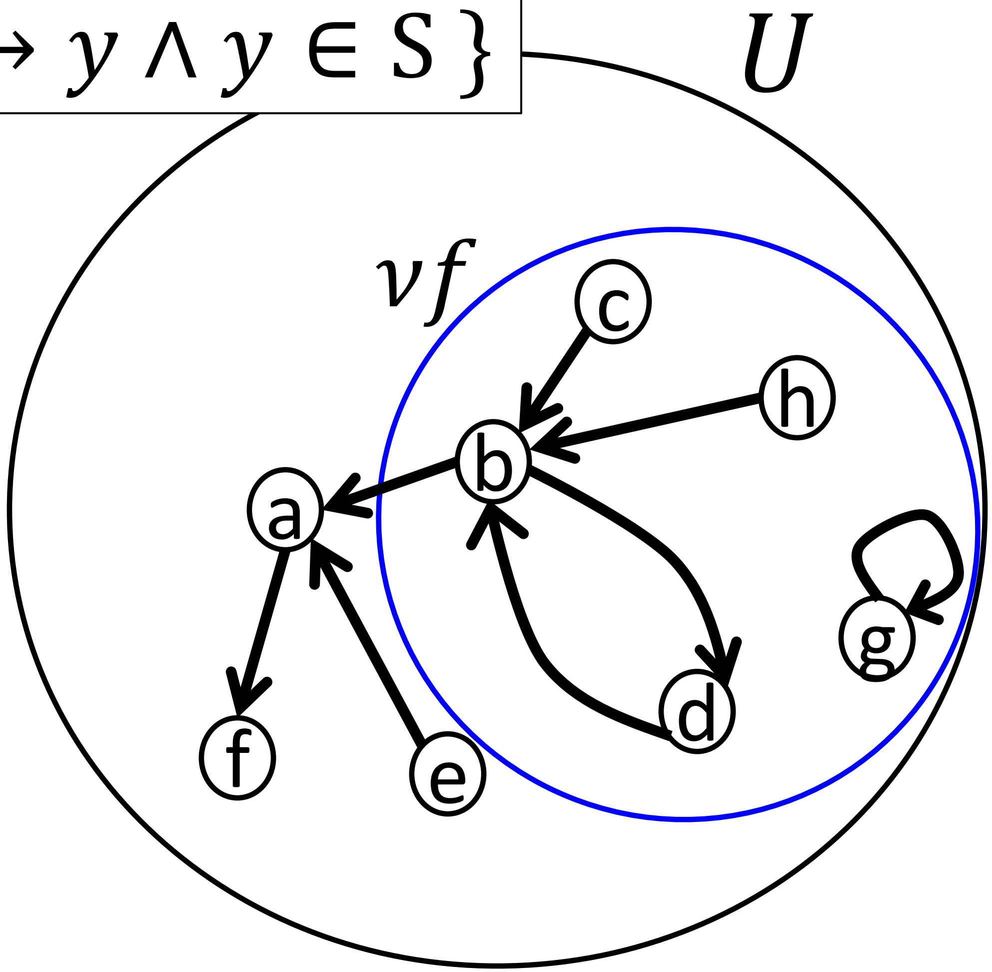
(Tarski's Principle)

$$\frac{\exists\, S.\ X \subseteq S \ \wedge\ S \subseteq f(S)}{X \subseteq \nu f}$$



Tarski Principle: Example

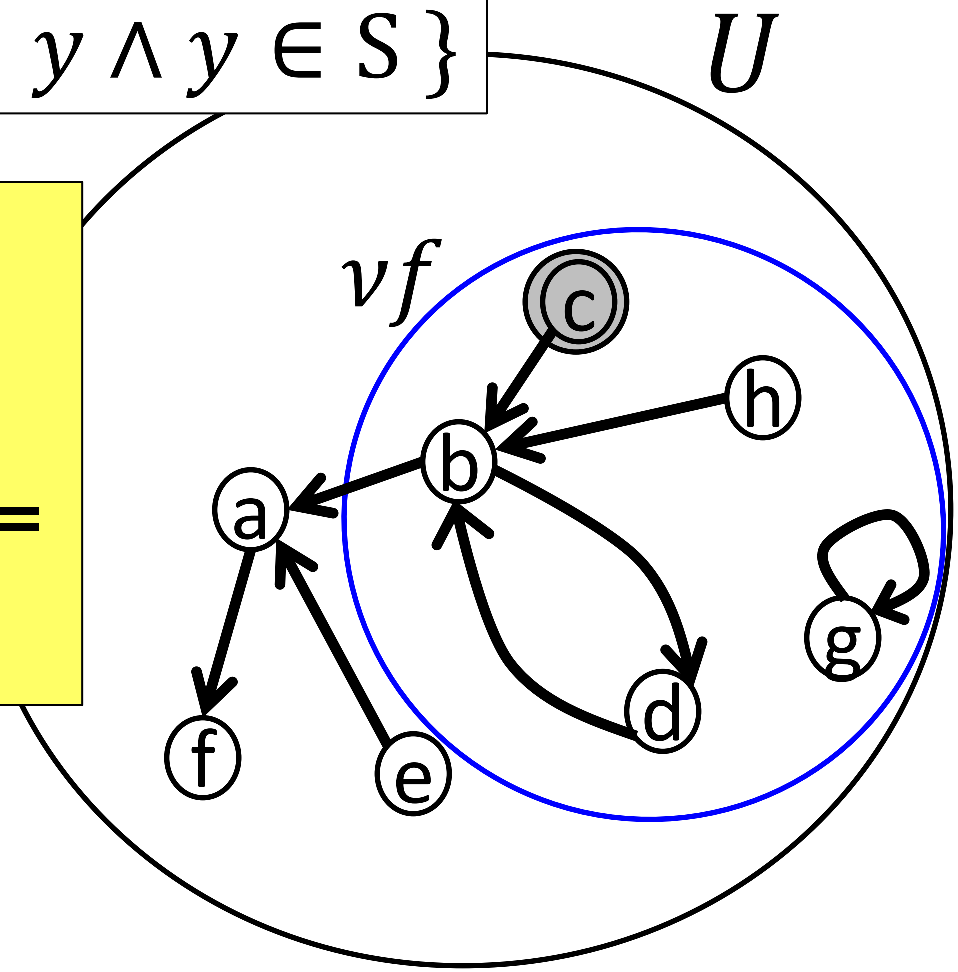
$$f(S) = \{\, x \mid \exists y.\ x \to y \wedge y \in S \,\}$$



Tarski Principle: Example

$$f(S) = \{\, x \mid \exists y.\ x \rightarrow y \wedge y \in S \,\}$$

$$\frac{}{\{c\} \subseteq \nu f}$$

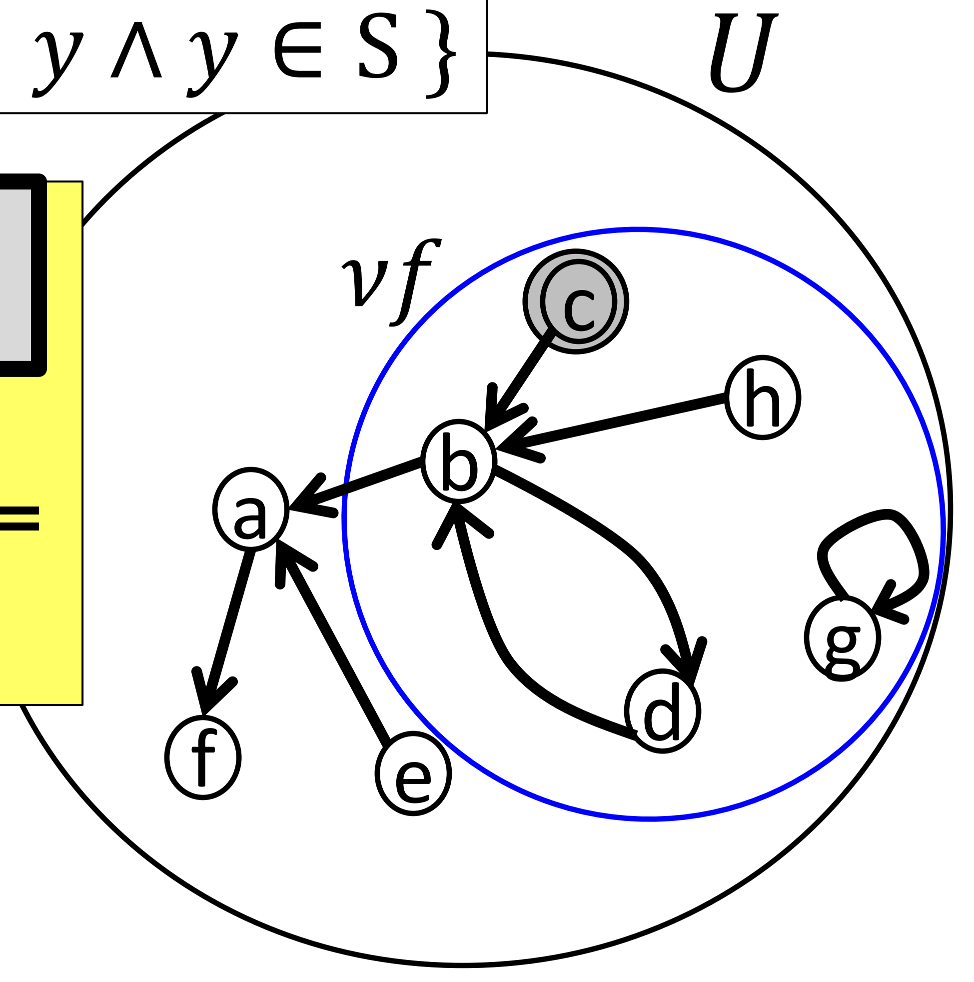


Tarski Principle: Example

$f(S) = \{\, x \mid \exists y.\ x \to y \land y \in S \,\}$

$\boxed{\{c\} \not\subseteq f(\{c\}) = \emptyset}$

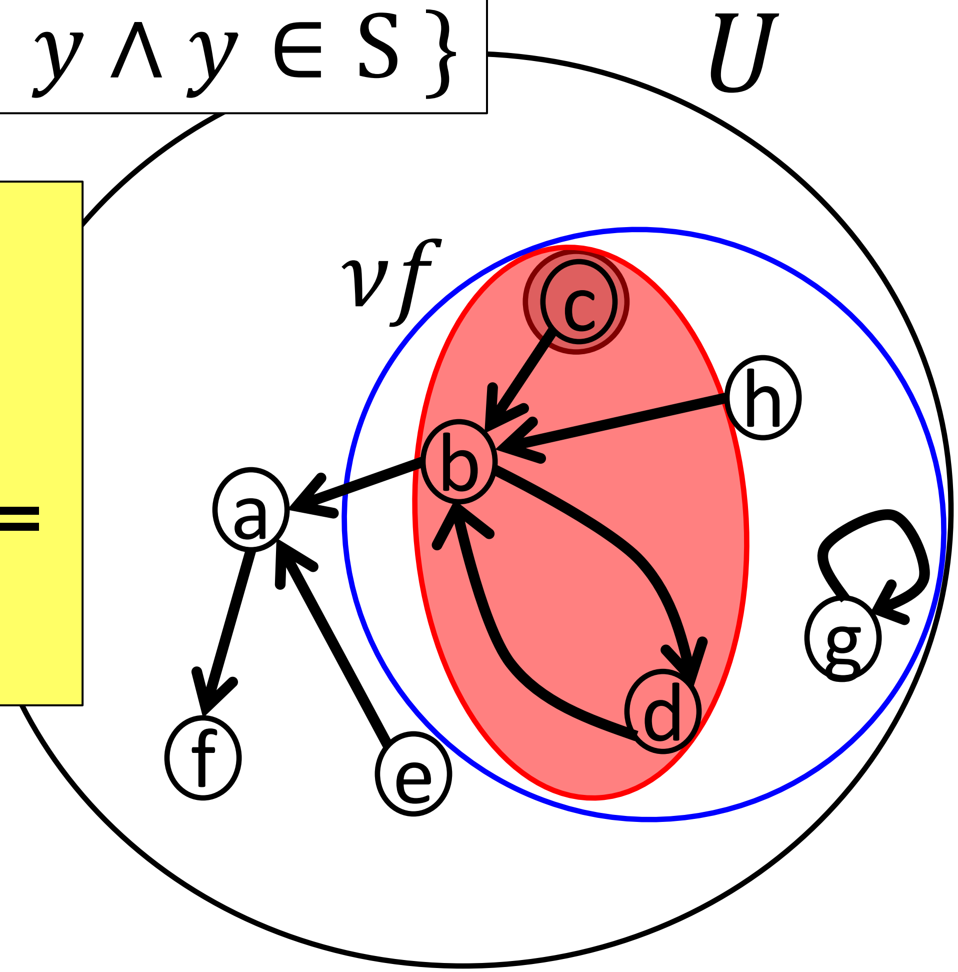
$$\frac{\{c\} \not\subseteq f(\{c\}) = \emptyset}{\{c\} \subseteq \nu f}$$



Tarski Principle: Example

$$f(S) = \{\, x \mid \exists y.\ x \to y \land y \in S \,\}$$

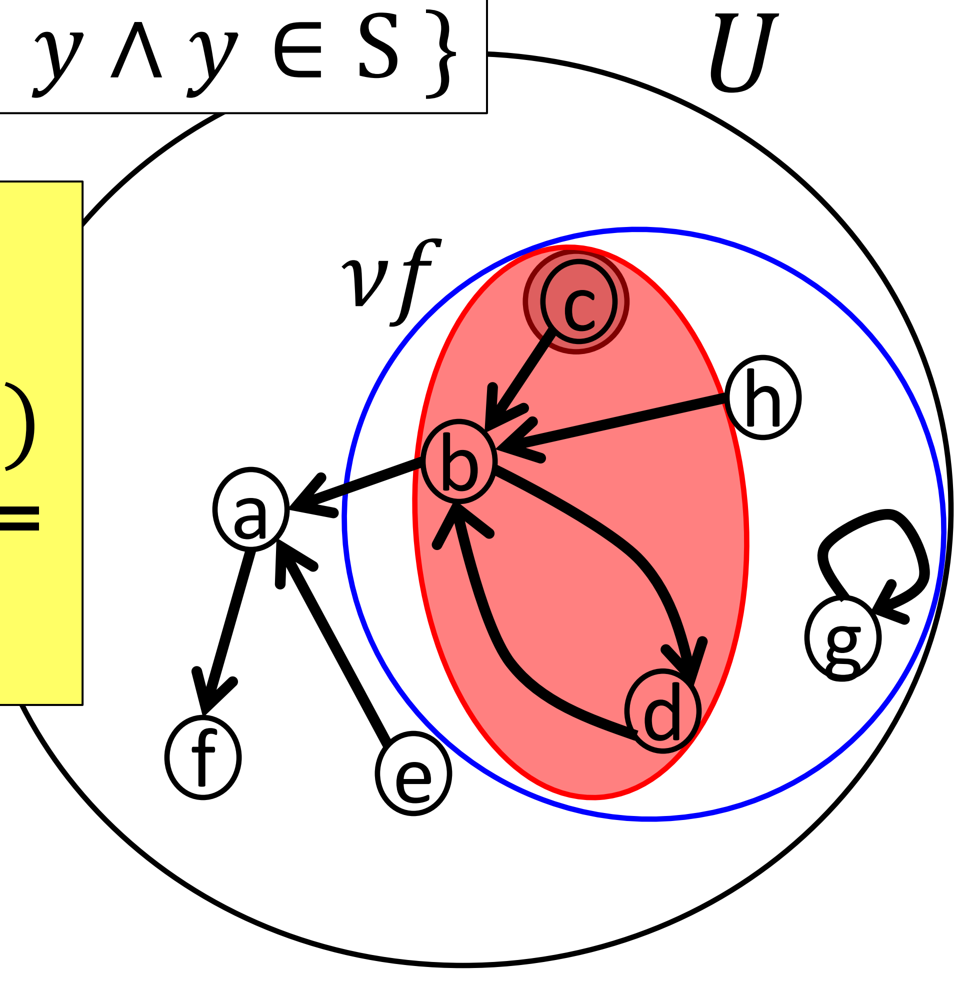
$$\frac{\{\mathrm{c}\} \subseteq {\color{red}\{\mathrm{c}, \mathrm{b}, \mathrm{d}\}}}{\{\mathrm{c}\} \subseteq \nu f}$$



Tarski Principle: Example

$$f(S) = \{\, x \mid \exists y.\ x \to y \land y \in S \,\}$$

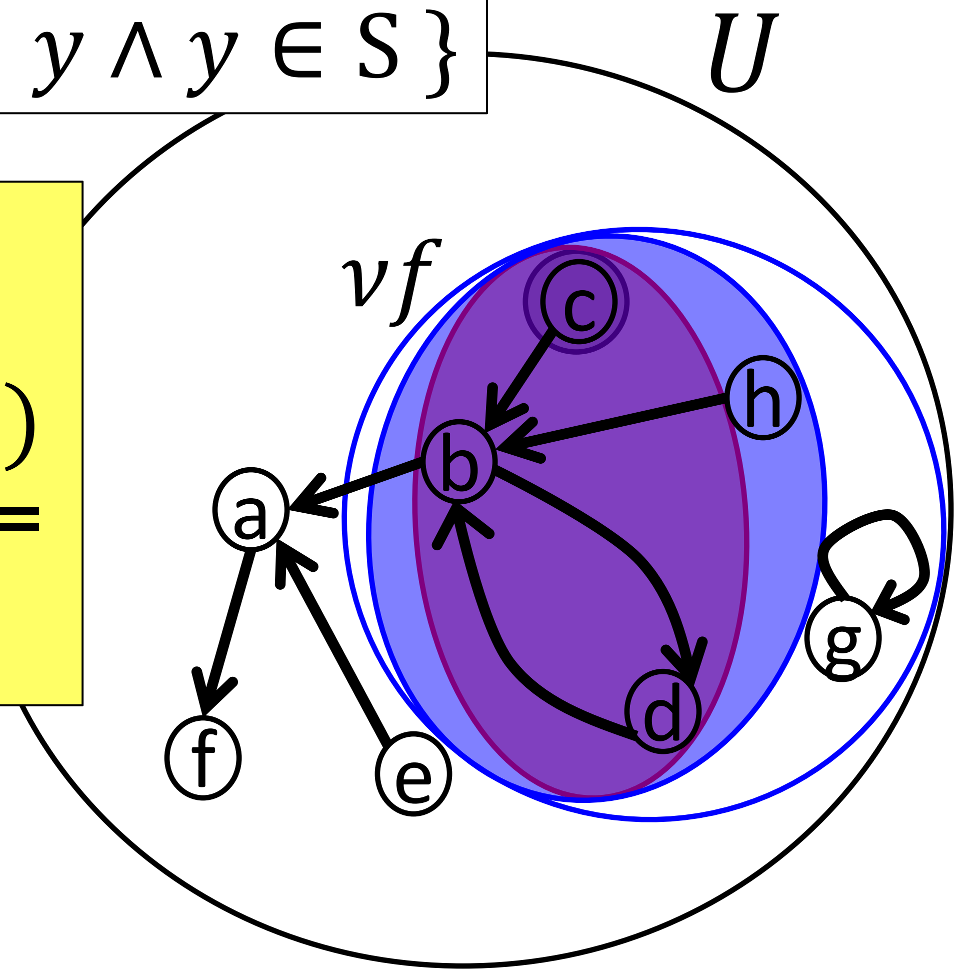
$$\frac{\{c\} \subseteq \{c, b, d\} \qquad \{c, b, d\} \subseteq f(\{c, b, d\})}{\{c\} \subseteq \nu f}$$



Tarski Principle: Example

$$f(S) = \{\, x \mid \exists y.\ x \to y \land y \in S \,\}$$

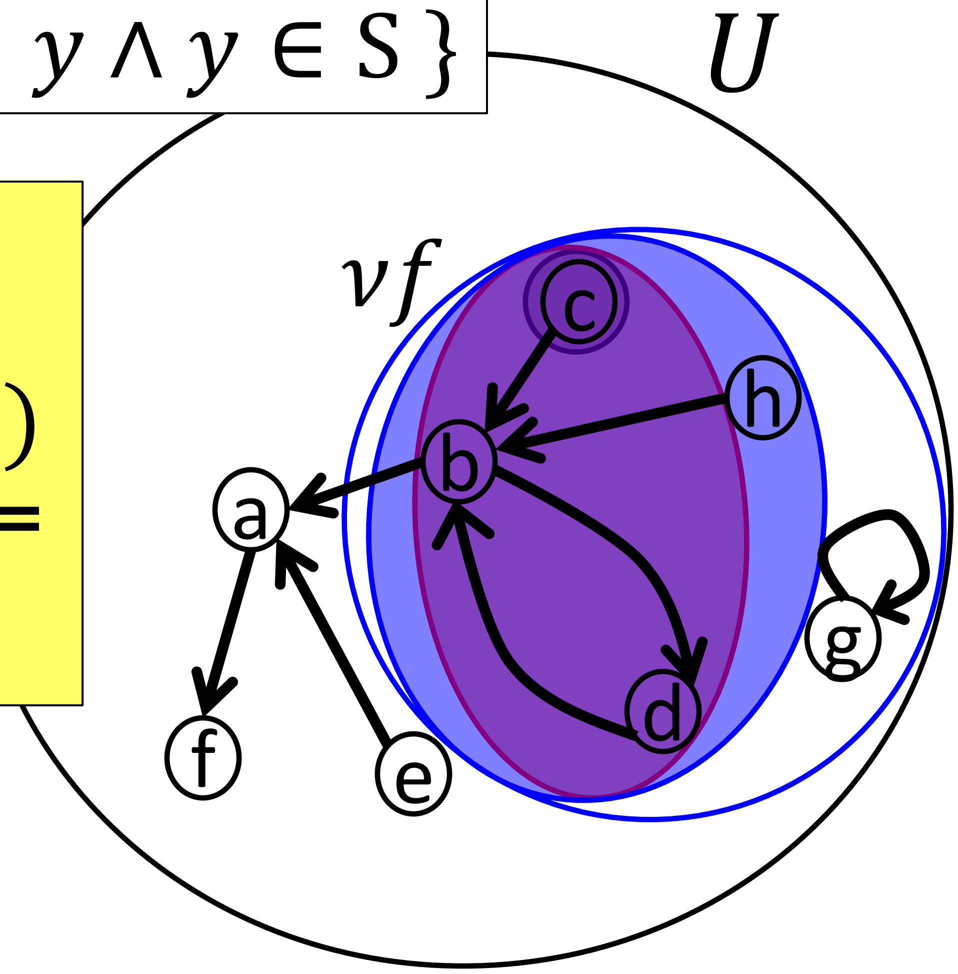
$$\dfrac{\{c\} \subseteq \{c, b, d\} \qquad \{c, b, d\} \subseteq f(\{c, b, d\})}{\{c\} \subseteq \nu f}$$



Tarski Principle: Example

$$f(S) = \{\, x \mid \exists y.\ x \to y \land y \in S \,\}$$

$$\frac{\{c\} \subseteq \{c, b, d\} \qquad \{c, b, d\} \subseteq f(\{c, b, d\})}{\{c\} \subseteq \nu f}$$

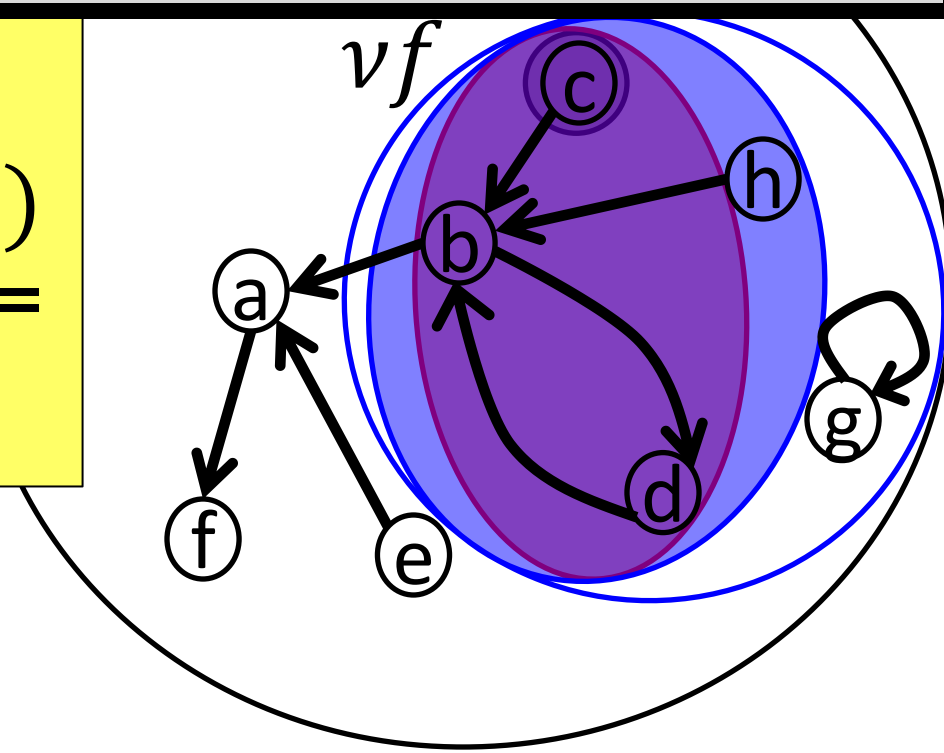


Tarski's Principle

+ Simple & Easy to Understand

- Have to Find a Consistent Set Up Front

$$\frac{\{c\} \subseteq \{c,b,d\} \qquad \{c,b,d\} \subseteq f(\{c,b,d\})}{\{c\} \subseteq \nu f}$$



Talk Outline

- Previous Approaches
  - Tarski's Fixed Point Theorem
  - Syntactically Guarded Coinduction
- Our Approach
  - Parameterized Coinduction

Syntactically Guarded Coinduction

$$f : \wp(U) \xrightarrow{\text{mon}} \wp(U)$$

(Guarded Coinduction)

$$\frac{}{X \subseteq \nu f}$$

Syntactically Guarded Coinduction

$$f : \wp(U) \xrightarrow{\text{mon}} \wp(U)$$

(Guarded Coinduction)

$$\frac{X \subseteq \nu f \implies X \subseteq \nu f}{X \subseteq \nu f}$$

Syntactically Guarded Coinduction

$$f : \wp(U) \xrightarrow{\text{mon}} \wp(U)$$

(Guarded Coinduction)

$$\frac{\text{pf: } X \subseteq \nu f \implies X \subseteq \nu f}{X \subseteq \nu f} \text{ (pf is guarded)}$$

Syntactically Guarded Coinduction

- Guardedness rules out trivial proofs.

f

(Gu

$$\frac{\text{pf: } X \subseteq \nu f \implies X \subseteq \nu f}{X \subseteq \nu f}\quad \text{(pf is } \boxed{\text{guarded}}\text{)}$$

Syntactically Guarded Coinduction

f

- Guardedness rules out trivial proofs.
- Its definition is syntactic and complex.

(Gu

$$\frac{\text{pf: } X \subseteq \nu f \implies X \subseteq \nu f}{X \subseteq \nu f} \quad \text{(pf is } \boxed{\text{guarded}}\text{)}$$

Syntactically Guarded Coinduction

f :

- Guardedness rules out trivial proofs.
- Its definition is syntactic and complex.
- The assumption can be used only after making progress.

(Gu

$$\frac{\text{pf: } X \subseteq \nu f \implies X \subseteq \nu f}{X \subseteq \nu f} \quad (\text{pf is } \boxed{\text{guarded}})$$

Syn   duction

f

What cofix does in Coq

ofs.

- Its definition is syntactic and complex.
- The assumption can be used only after making progress.

(Gu

$$\frac{\text{pf: } X \subseteq \nu f \implies X \subseteq \nu f}{X \subseteq \nu f} \quad \text{(pf is } \boxed{\text{guarded}}\text{)}$$

Syn… …duction

Permits Incremental Reasoning

…ofs.

- Its definition is syntactic and complex.
- The assumption can be used only after making progress.

f …

(Gu…

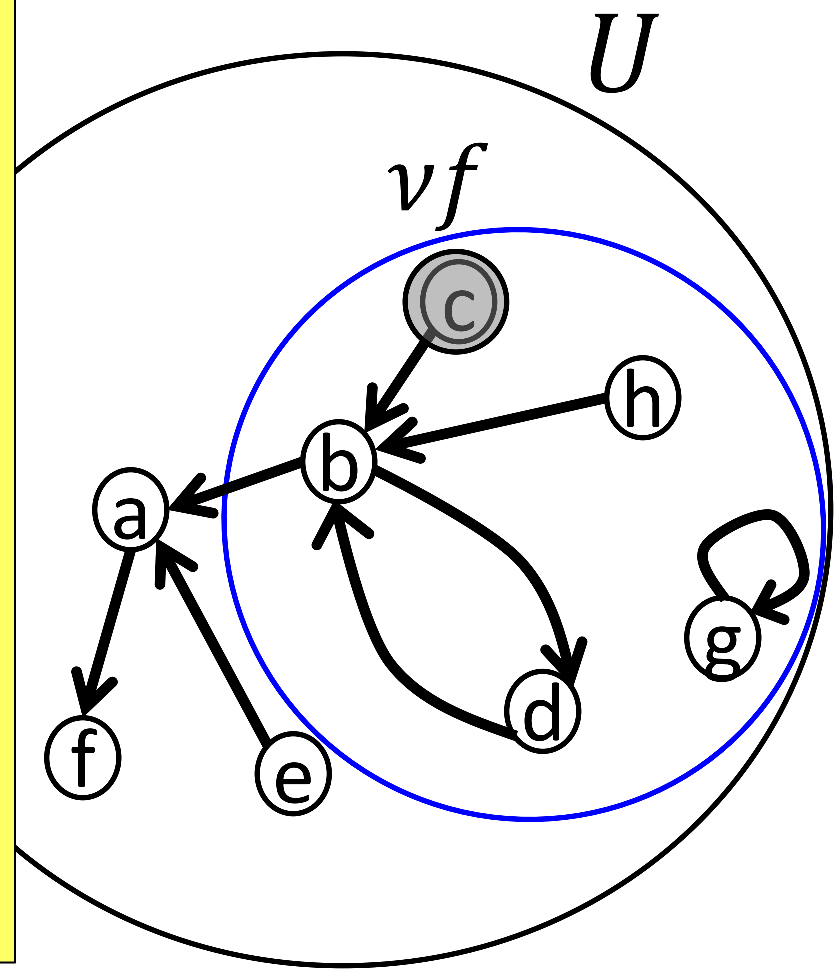
pf: $X \subseteq \nu f \implies X \subseteq \nu f$

$$\frac{\quad}{X \subseteq \nu f}$$

(pf is $\boxed{\text{guarded}}$)

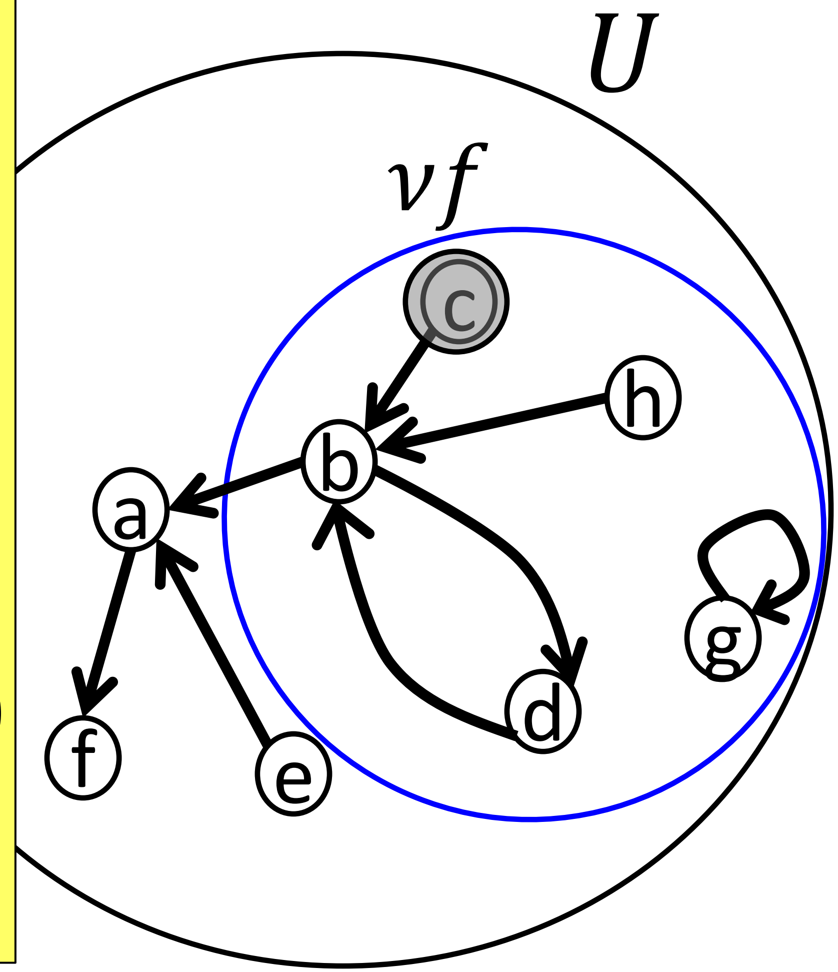
Example: Guarded Coinduction

$\{\mathrm{c}\} \subseteq \nu f$



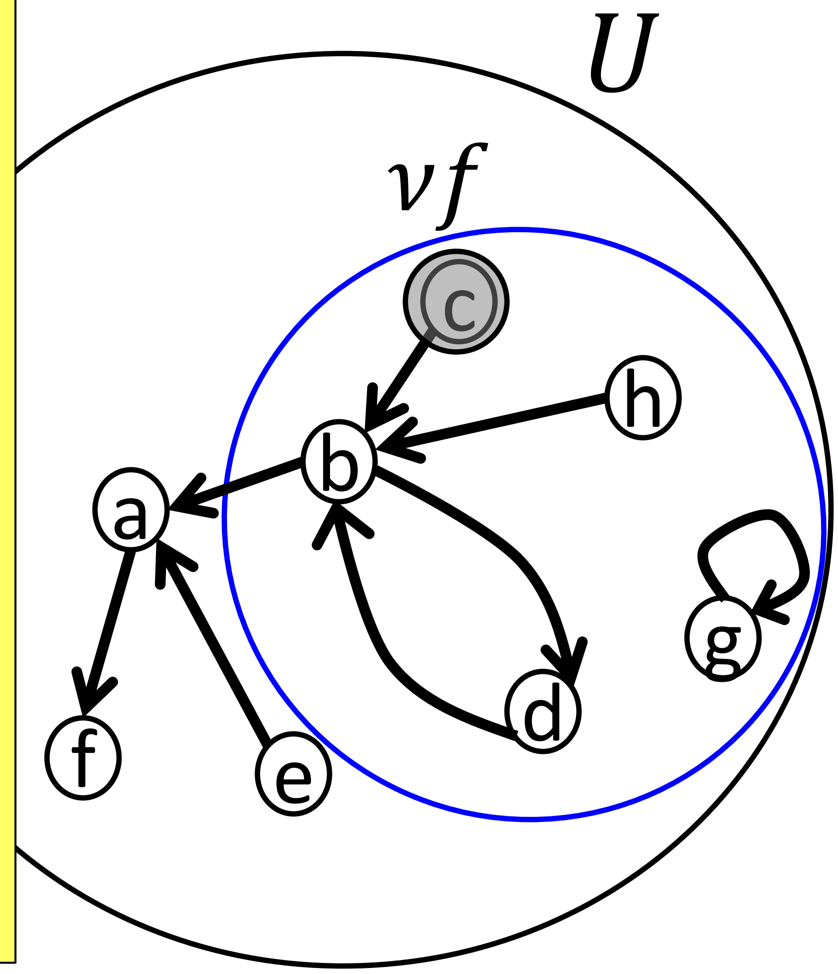
Example: Guarded Coinduction

$$\frac{\{c\} \subseteq f(\nu f)}{\{c\} \subseteq \nu f}$$



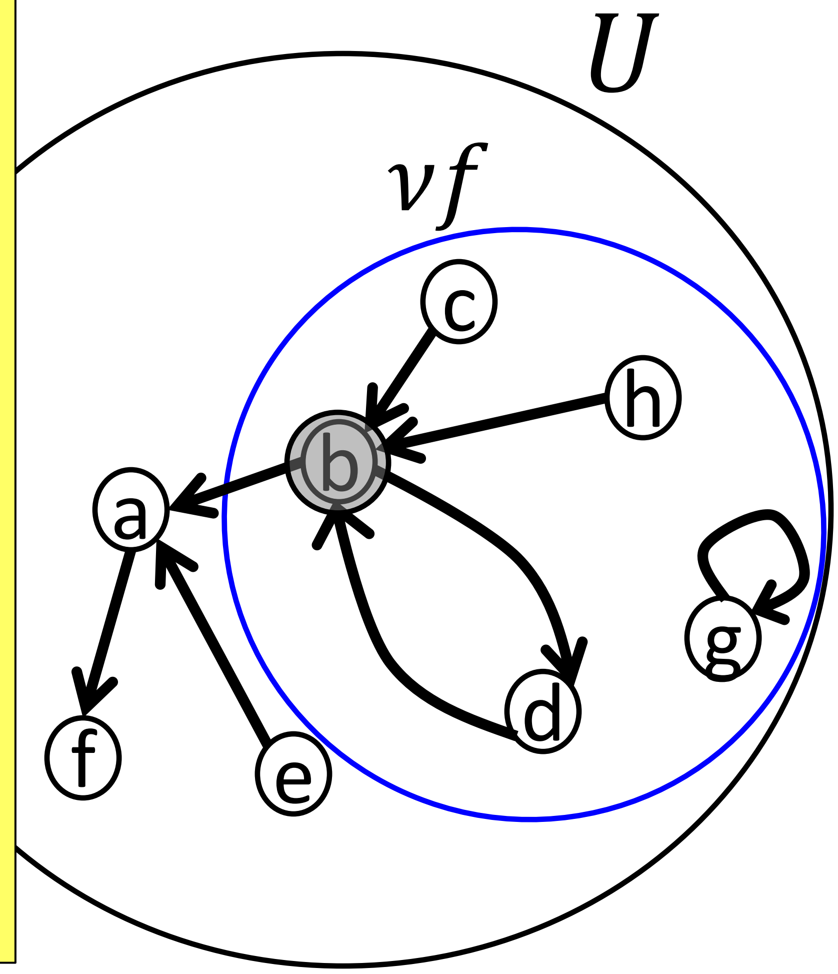
Example: Guarded Coinduction

$$\frac{\exists y.\ \mathrm{c} \to y \land y \in \nu f}{\{\mathrm{c}\} \subseteq \nu f}$$



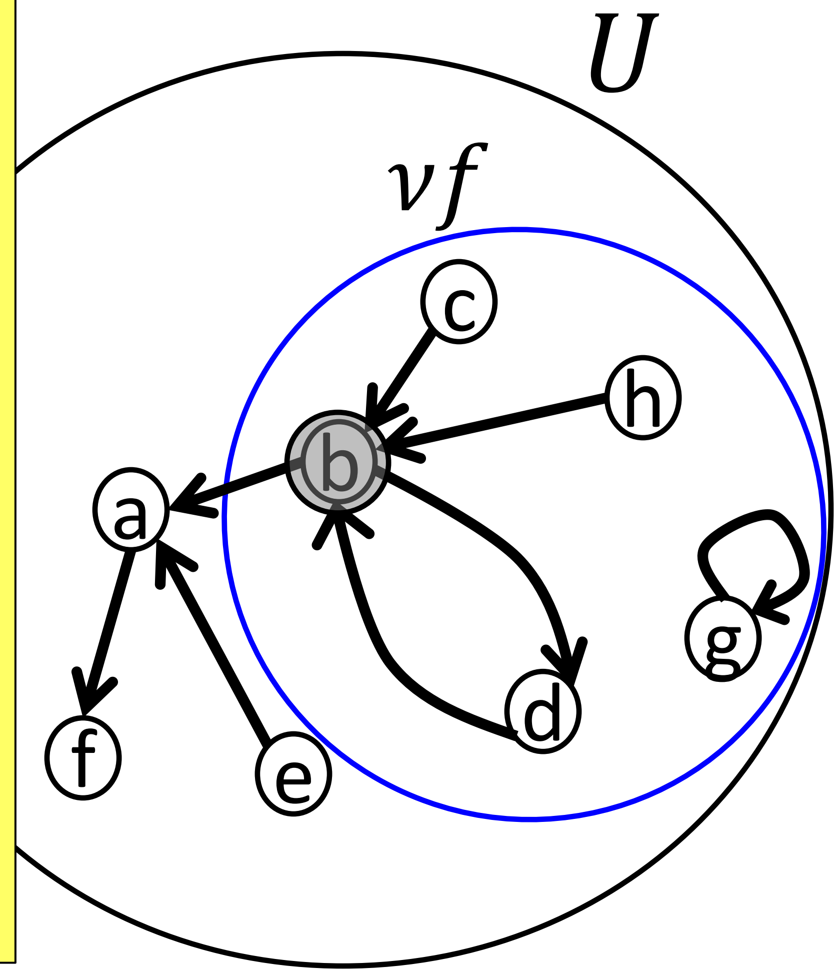
Example: Guarded Coinduction

$$\frac{\exists y.\ \mathrm{c} \rightarrow y \land y \in \nu f}{\{\mathrm{c}\} \subseteq \nu f}$$



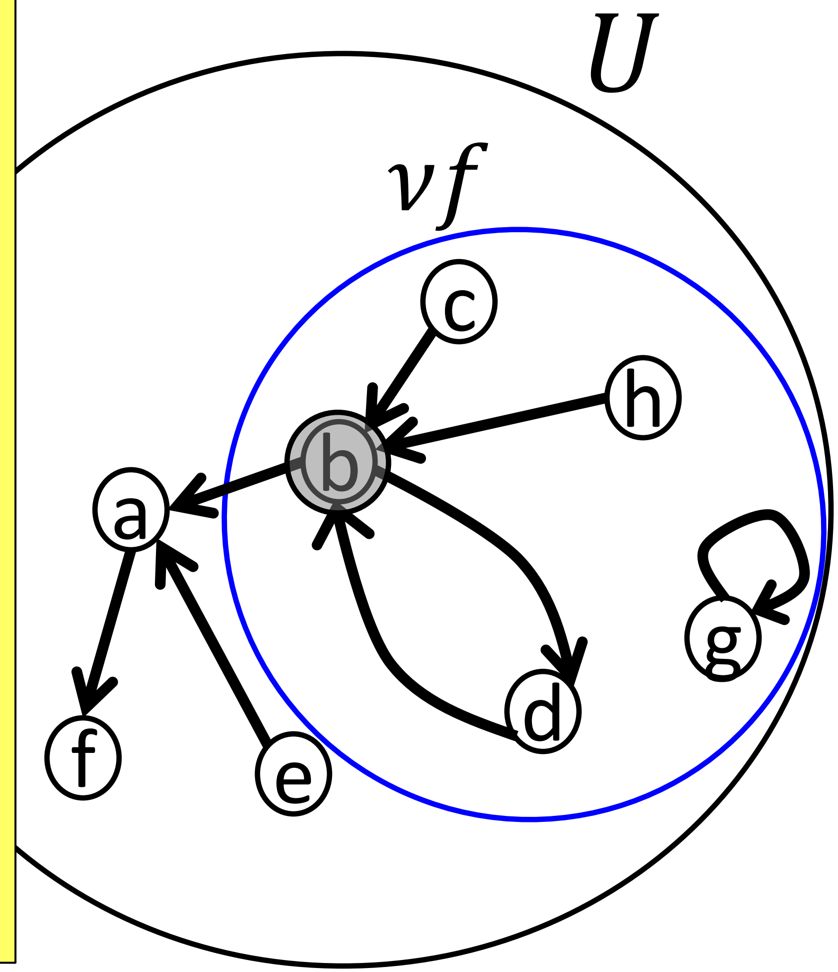
Example: Guarded Coinduction

$$\frac{b \in \nu f}{\{c\} \subseteq \nu f}$$



Example: Guarded Coinduction

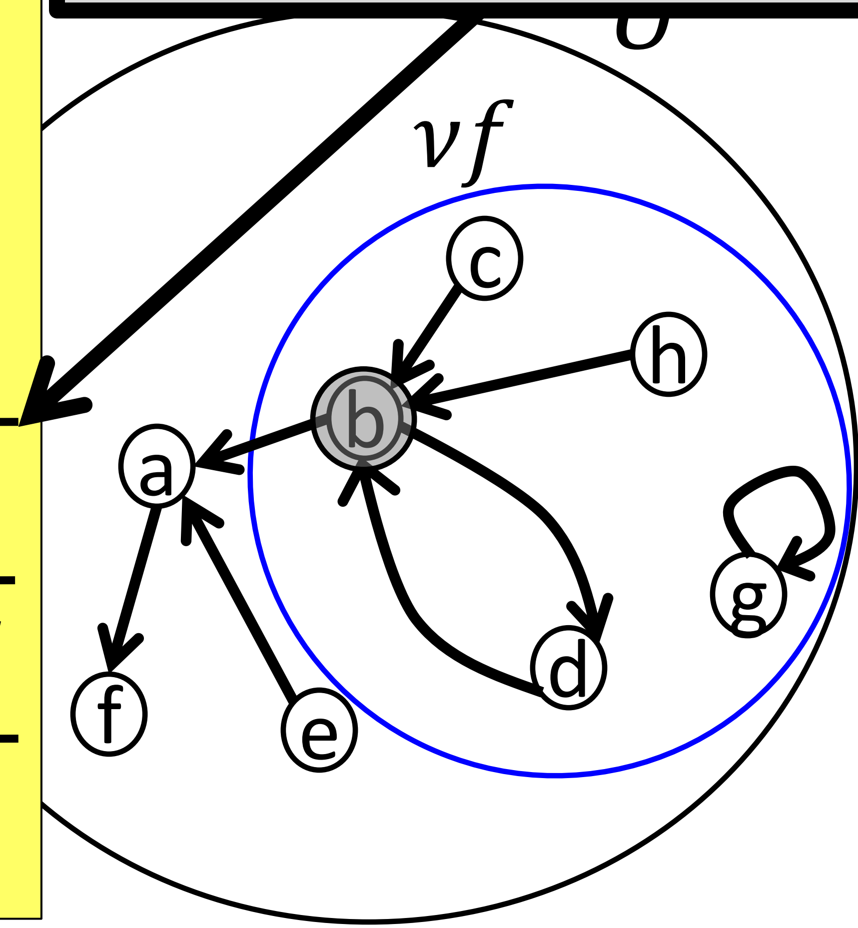
$$\dfrac{\dfrac{\{\mathrm{b}\} \subseteq \nu f}{\mathrm{b} \in \nu f}}{\{\mathrm{c}\} \subseteq \nu f}$$



Example: Guarde

By
Guarded Coinduction

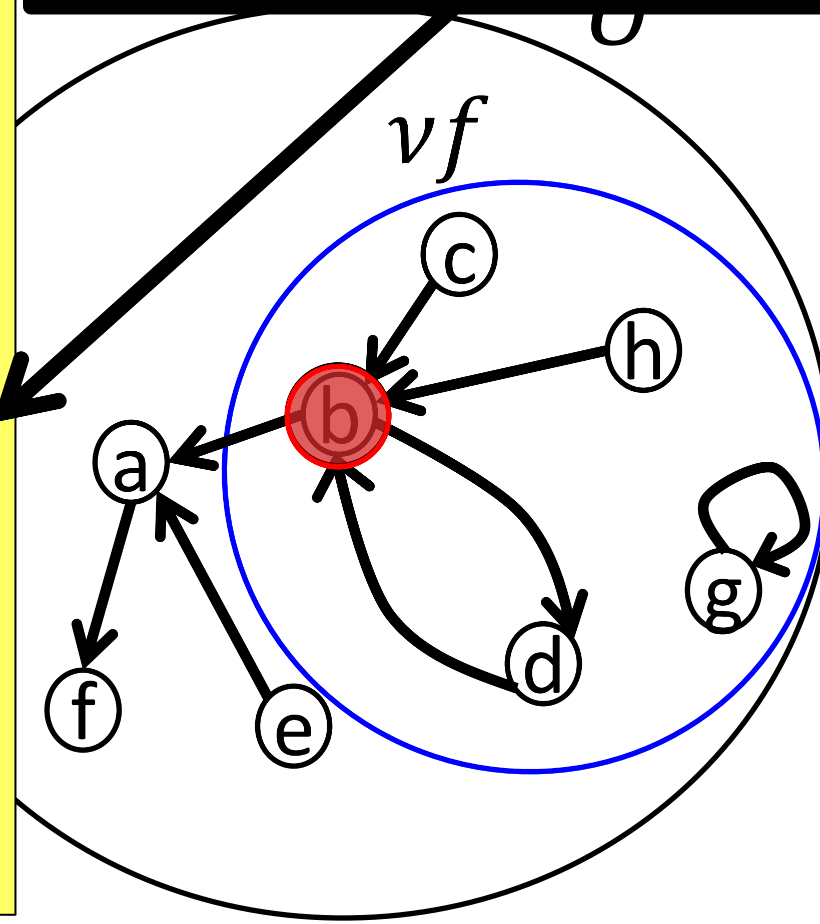
$$\begin{array}{c}
\color{red}{\{b\} \subseteq \nu f} \Rightarrow \{b\} \subseteq \nu f \\
\hline
\{b\} \subseteq \nu f \\
\hline
b \in \nu f \\
\hline
\{c\} \subseteq \nu f
\end{array}$$



Example: Guarde

By
Guarded Coinduction

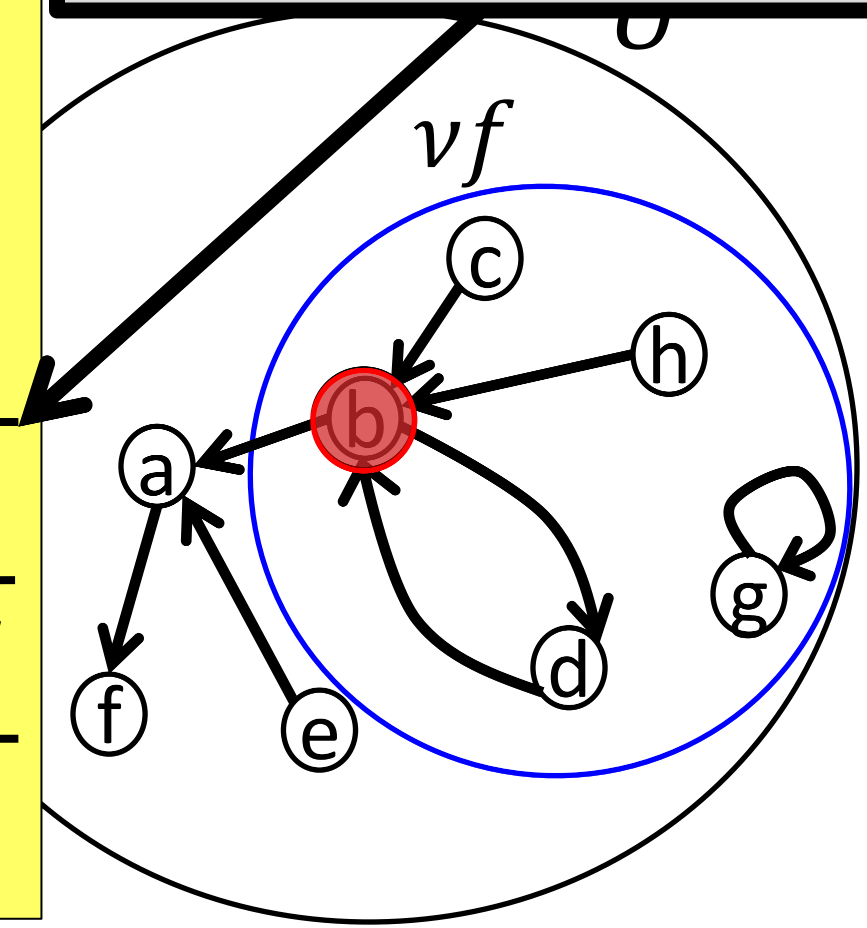
$$\frac{\dfrac{\dfrac{{\color{red}\{b\} \subseteq \nu f} \Rightarrow \{b\} \subseteq \nu f}{\{b\} \subseteq \nu f}}{b \in \nu f}}{\{c\} \subseteq \nu f}$$



Trivial Proof is
NOT Guarded

By
Guarded Coinduction

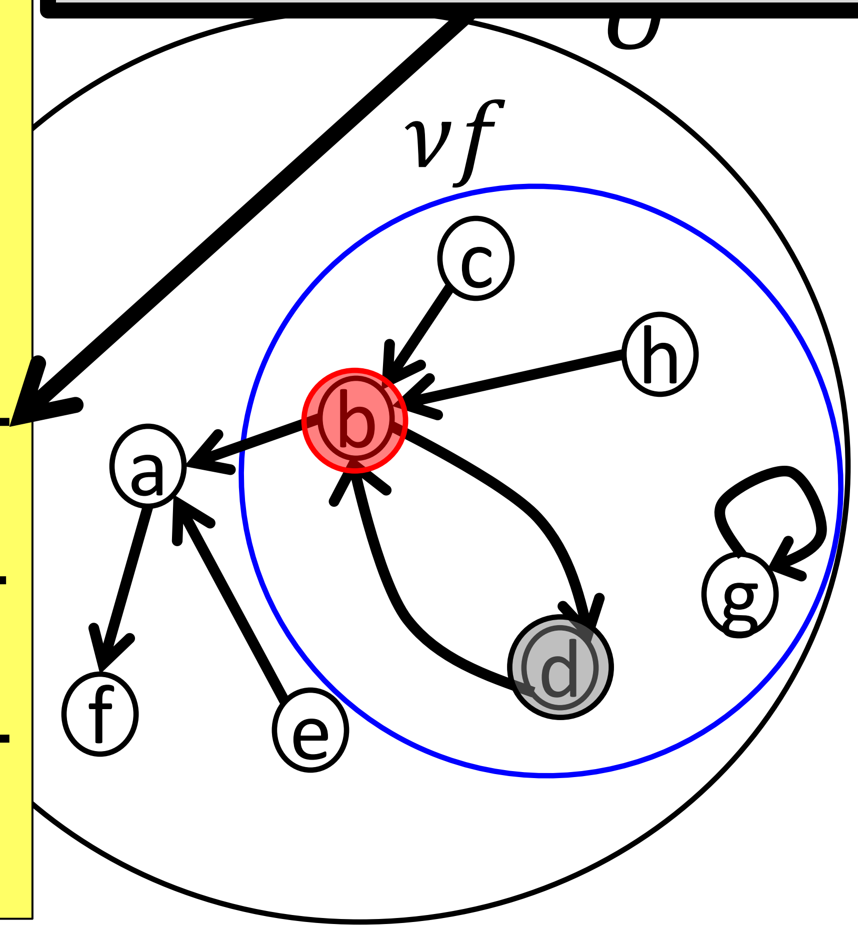
$$\frac{\dfrac{\dfrac{\{b\} \subseteq \nu f \Rightarrow \{b\} \subseteq \nu f}{\{b\} \subseteq \nu f}}{b \in \nu f}}{\{c\} \subseteq \nu f}$$



Example: Guarde

By
Guarded Coinduction

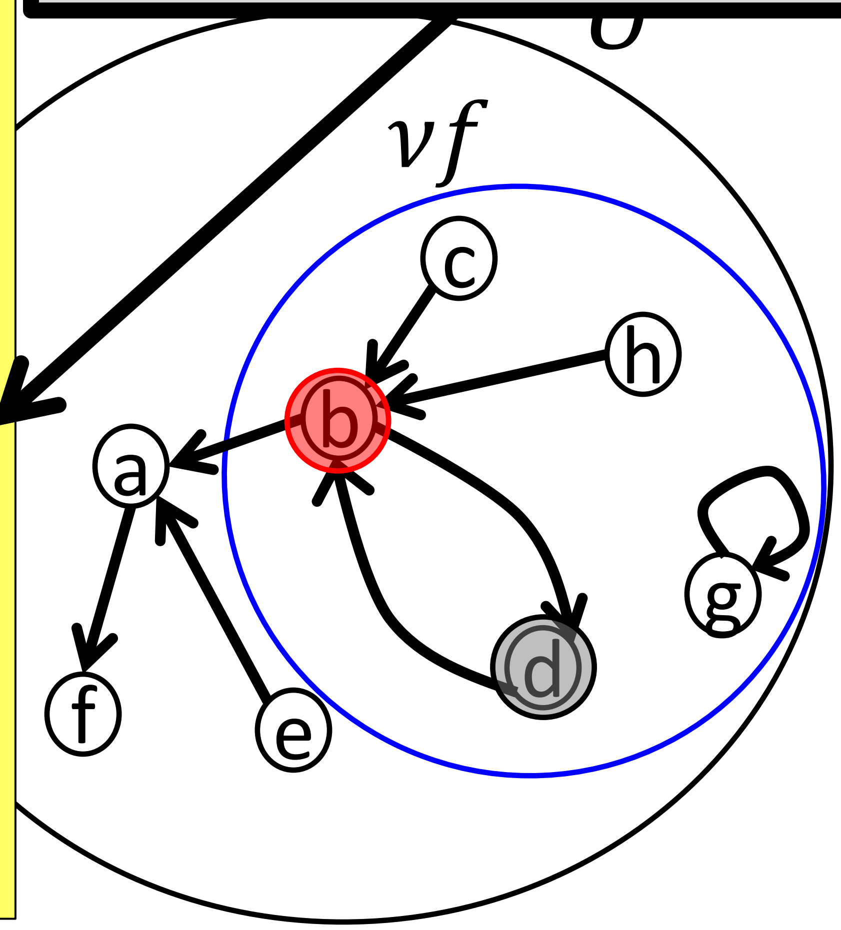
$$\dfrac{\dfrac{\dfrac{\{b\} \subseteq \nu f \Rightarrow \{b\} \subseteq \nu f}{\{b\} \subseteq \nu f}}{b \in \nu f}}{\{c\} \subseteq \nu f}$$



Example: Guarde

By
Guarded Coinduction

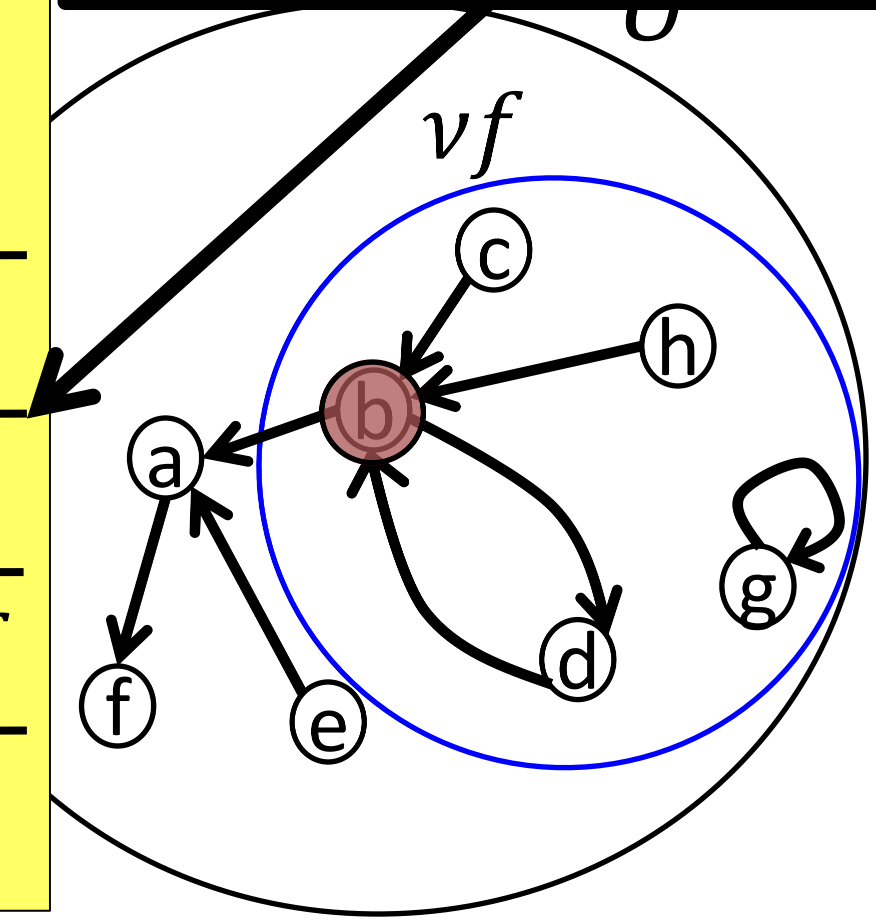
$$\dfrac{\dfrac{\dfrac{\dfrac{\color{red}{\{\mathrm{b}\} \subseteq \nu f} \Rightarrow \{\mathrm{d}\} \subseteq \nu f}{\color{red}{\{\mathrm{b}\} \subseteq \nu f} \Rightarrow \{\mathrm{b}\} \subseteq \nu f}}{\{\mathrm{b}\} \subseteq \nu f}}{\mathrm{b} \in \nu f}}{\{\mathrm{c}\} \subseteq \nu f}$$



Example: Guarde

By
Guarded Coinduction

$$
\begin{array}{c}
\{b\} \subseteq \nu f \Rightarrow \{d\} \subseteq \nu f \\
\hline
\{b\} \subseteq \nu f \Rightarrow \{b\} \subseteq \nu f \\
\hline
\{b\} \subseteq \nu f \\
\hline
b \in \nu f \\
\hline
\{c\} \subseteq \nu f
\end{array}
$$



Example: Guarde

By
Guarded Coinduction

$\{b\} \subseteq \nu f \Rightarrow \{b\} \subseteq \nu f$

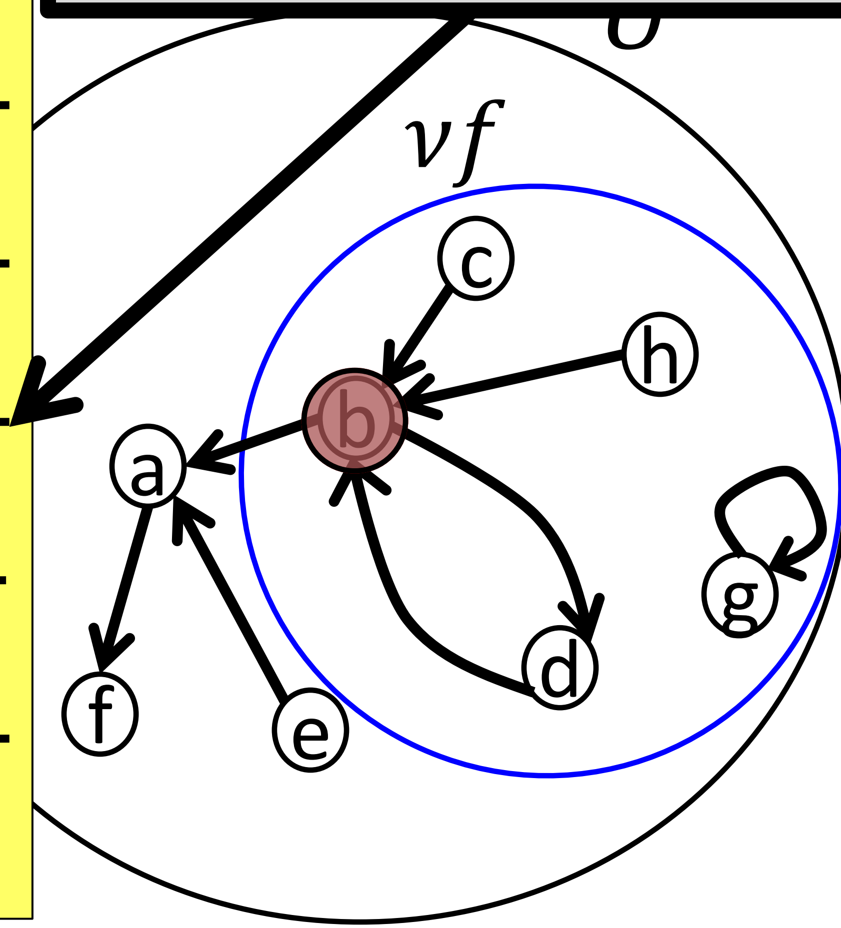
$\{b\} \subseteq \nu f \Rightarrow \{d\} \subseteq \nu f$

$\{b\} \subseteq \nu f \Rightarrow \{b\} \subseteq \nu f$

$\{b\} \subseteq \nu f$

$b \in \nu f$

$\{c\} \subseteq \nu f$



This Proof is Guarded

By Guarded Coinduction

$\{b\} \subseteq \nu f \Rightarrow \{b\} \subseteq \nu f$

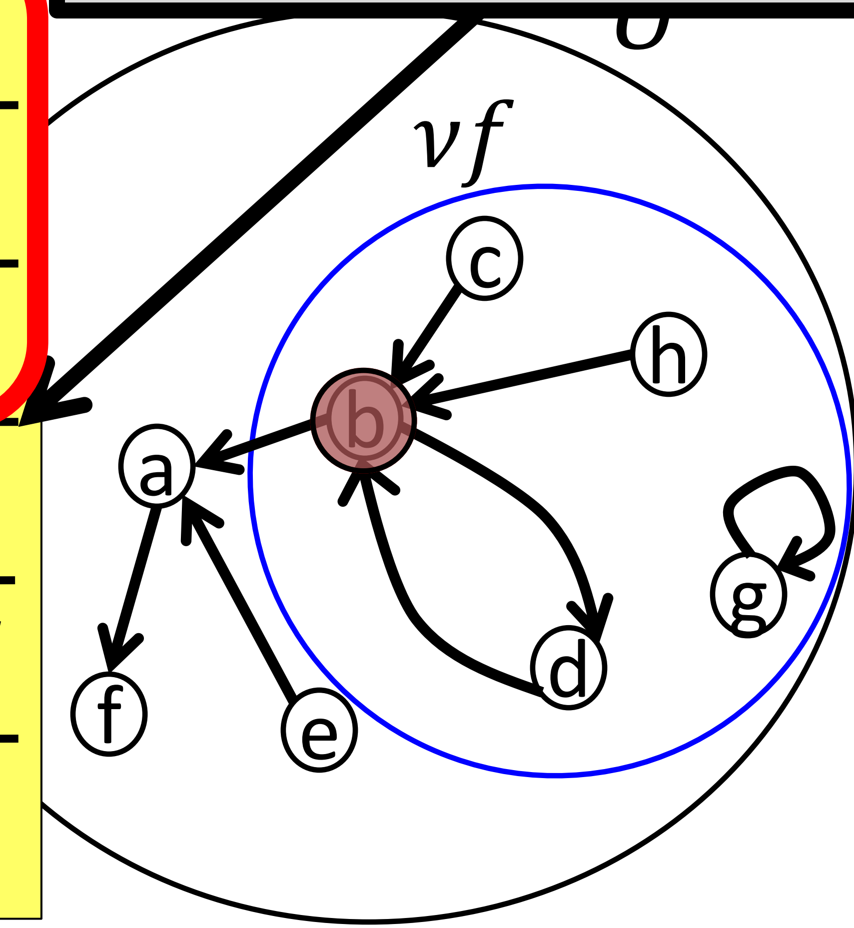
$\{b\} \subseteq \nu f \Rightarrow \{d\} \subseteq \nu f$

$\{b\} \subseteq \nu f \Rightarrow \{b\} \subseteq \nu f$

$\{b\} \subseteq \nu f$

$b \in \nu f$

$\{c\} \subseteq \nu f$



This Proof is
Guarded

de

By
Guarded Coinduction

$$\{b\} \subseteq \nu f \Rightarrow \{b\} \subseteq \nu f$$

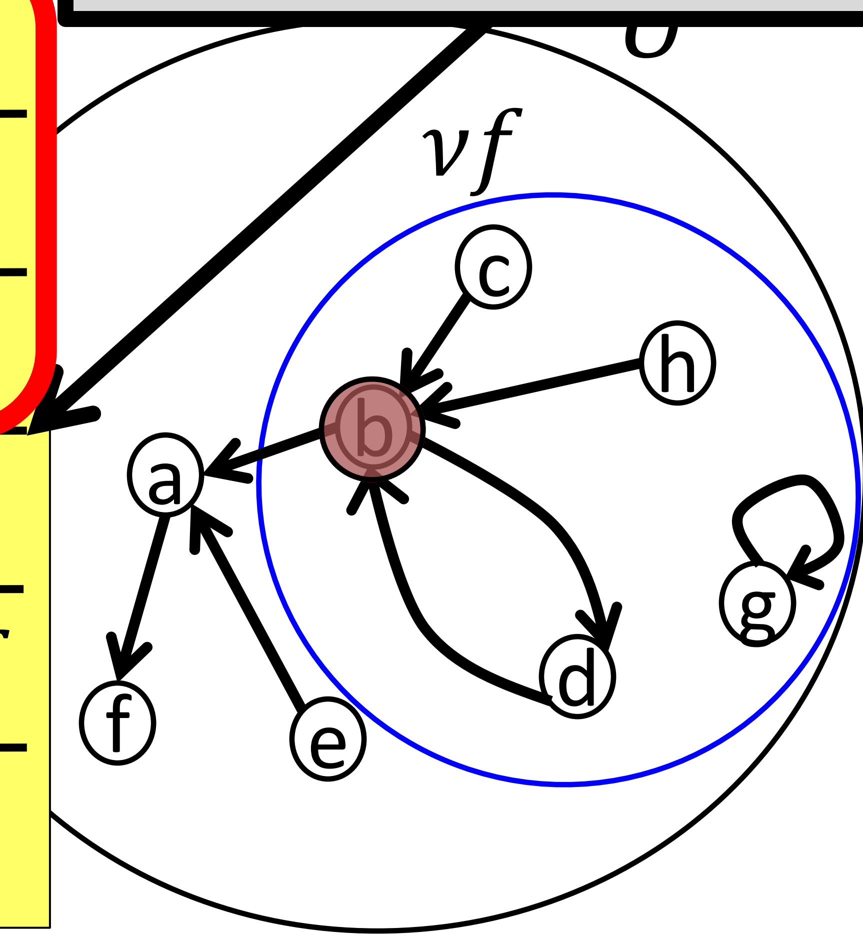
$$\{b\} \subseteq \nu f \Rightarrow \{d\} \subseteq \nu f$$

$$\{b\} \subseteq \nu f \Rightarrow \{b\} \subseteq \nu f$$

$$\{b\} \subseteq \nu f$$

$$b \in \nu f$$

$$\{c\} \subseteq \nu f$$



This Proof is Guarded

By Guarded Coinduction

$\{b\} \subseteq \nu f \Rightarrow \{b\} \subseteq \nu f$

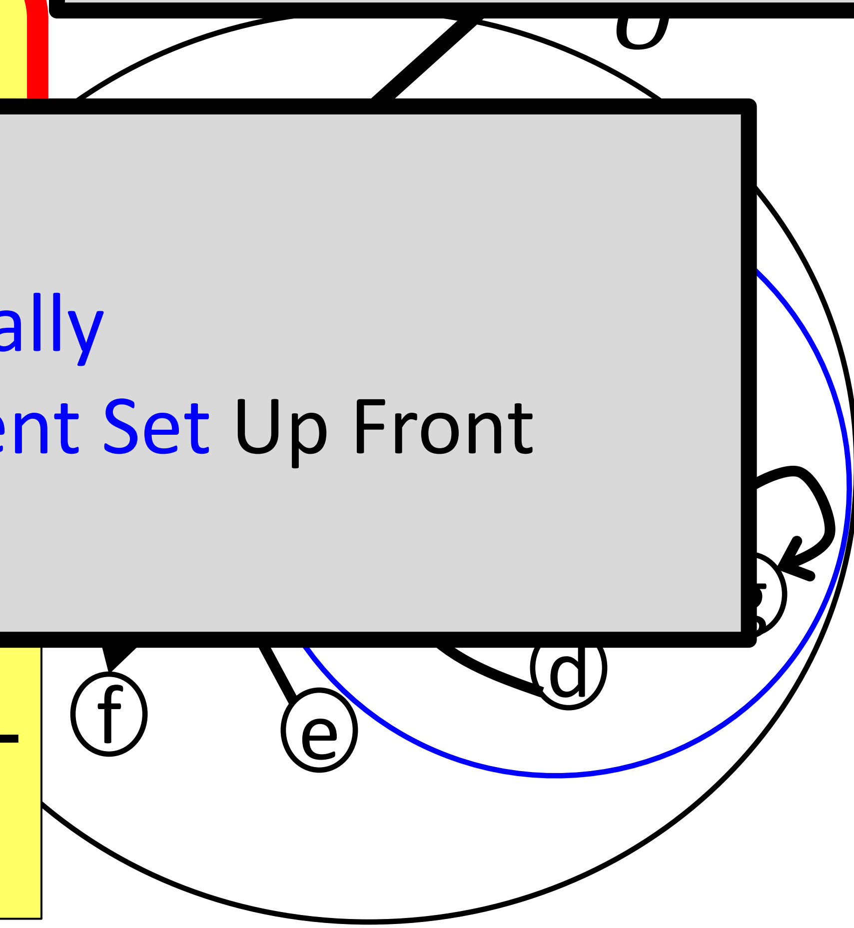
$\{b\}$

$\{b\}$

Prove Incrementally with NO Consistent Set Up Front

$b \in \nu f$

$\{c\} \subseteq \nu f$



Example: Simulation between Programs

$$\mathsf{restart}\ h\ n := \mathsf{if}\ n > \underline{0}\ \mathsf{then}\ (\mathsf{output}\ n; \mathsf{restart}\ h\ (n - \underline{1}))\ \mathsf{else}\ h\ \underline{0}$$

$$\begin{aligned} \mathsf{f}\ n\ := {} & \mathsf{output}\ (\underline{2} * n); \\ & \mathsf{let}\ v = \mathsf{input}()\ \mathsf{in} \\ & \mathsf{if}\ v = \underline{0}\ \mathsf{then}\ \mathsf{restart}\ \mathsf{f}\ (\underline{2} * n)\ \mathsf{else}\ \mathsf{f}\ (v + n) \end{aligned}$$

$$\begin{aligned} \mathsf{g}\ n := {} & \mathsf{let}\ v = (\mathsf{output}\ n; \mathsf{input}()) * \underline{2}\ \mathsf{in} \\ & (\mathsf{if}\ v \neq \underline{0}\ \mathsf{then}\ \mathsf{g}\ \mathsf{else}\ \mathsf{restart}\ \mathsf{g})\ (v + n) \end{aligned}$$

Example: Simulation between Programs

$$\boxed{\mathsf{restart}}\ h\ n := \mathsf{if}\ n > \underline{0}\ \mathsf{then}\ (\mathsf{output}\ n; \boxed{\mathsf{restart}}\ h\ (n - \underline{1}))\ \mathsf{else}\ h\ \underline{0}$$

$$\begin{aligned}\boxed{\mathsf{f}}\ n\ :=\ & \mathsf{output}\ (\underline{2} * n);\\ & \mathsf{let}\ v = \mathsf{input}()\ \mathsf{in}\\ & \mathsf{if}\ v = \underline{0}\ \mathsf{then}\ \boxed{\mathsf{restart}}\ \boxed{\mathsf{f}}\ (\underline{2} * n)\ \mathsf{else}\ \boxed{\mathsf{f}}\ (v + n)\end{aligned}$$

$$\begin{aligned}\boxed{\mathsf{g}}\ n := {} & \mathsf{let}\ v = (\mathsf{output}\ n; \mathsf{input}()) * \underline{2}\ \mathsf{in}\\ & (\mathsf{if}\ v \neq \underline{0}\ \mathsf{then}\ \boxed{\mathsf{g}}\ \mathsf{else}\ \boxed{\mathsf{restart}}\ \boxed{\mathsf{g}})\ (v + n)\end{aligned}$$

Proof using Tarski's Principle

```
nductive fgsim : exp -> exp -> Prop :=
_fgsim1: forall (n m: nat) (EQ: n = 2 * m), fgsim
   (f @ n)
   (g @ m)
_fgsim2: forall (m: nat), fgsim
   (Let "v" := (Eoutput (m + (m + 0));; Einput) ~* 2
    in (If 1 ~<= "v" then f else restart @ f) @ ("v" ~+ (m + (m + 0))))
   (Eoutput (2 ~* m);;
    (Let "v" := Einput
     in If "v" ~<= 0 then (restart @ g) @ (2 ~* m) else g @ ("v" ~+ m)))
_fgsim3: forall (m: nat), fgsim
   (Let "v" := ((<>);; Einput) ~* 2
    in (If 1 ~<= "v" then f else restart @ f) @ ("v" ~+ (m + (m + 0))))
   ((<>);;
    (Let "v" := Einput
     in If "v" ~<= 0 then (restart @ g) @ (2 ~* m) else g @ ("v" ~+ m)))
_fgsim4: forall (m: nat), fgsim
   (Let "v" := Einput ~* 2
    in (If 1 ~<= "v" then f else restart @ f) @ ("v" ~+ (m + (m + 0))))
   (Let "v" := Einput
    in If "v" ~<= 0 then (restart @ g) @ (2 ~* m) else g @ ("v" ~+ m))
_fgsim5: forall (m m0: nat), fgsim
   (Let "v" := m0 ~* 2
    in (If 1 ~<= "v" then f else restart @ f) @ ("v" ~+ (m + (m + 0))))
   (Let "v" := m0
    in If "v" ~<= 0 then (restart @ g) @ (2 ~* m) else g @ ("v" ~+ m))
_fgsim6: forall (m m0: nat), fgsim
   (Let "v" := m0 * 1 + m0
    in (If 1 ~<= "v" then f else restart @ f) @ ("v" ~+ (m + (m + 0)))
   (If m0 ~<= 0 then (restart @ g) @ (2 ~* m) else g @ (m0 ~+ m))
_fgsim7: forall (m m0: nat), fgsim
   ((If 1 ~<= (m0 * 1 + m0) then f else restart @ f) @
    ((m0 * 1 + m0) ~+ (m + (m + 0))))
   (If ble_nat m0 0 then (restart @ g) @ (2 ~* m) else g @ (m0 ~+ m))
_fgsim8: forall (m m0: nat), fgsim
   ((If 1 ~<= (m0 * 1 + m0) then f else restart @ f) @
    (m0 * 1 + m0 + (m + (m + 0))))
   (If ble_nat m0 0 then (restart @ g) @ (2 ~* m) else g @ (m0 ~+ m))
| _fgsim9: forall (m m0: nat), fgsim
   ((If ble_nat 1 (m0 * 1 + m0) then f else restart @ f) @
    (m0 * 1 + m0 + (m + (m + 0))))
   (If ble_nat m0 0 then (restart @ g) @ (2 ~* m) else g @ (m0 ~+ m))
| _fgsim10: forall (m: nat), fgsim
   ((restart @ f) @ m) ((restart @ g) @ m)
| _fgsim11: forall (m: nat), fgsim
   (Efix "_" "n"
      (If "n" ~<= 0 then f @ 0
       else Eoutput "n";; (restart @ f) @ ("n" ~- 1)) @ m)
   (Efix "_" "n"
      (If "n" ~<= 0 then g @ 0
       else Eoutput "n";; (restart @ g) @ ("n" ~- 1)) @ m)
| _fgsim12: forall (m: nat), fgsim
   (If m ~<= 0 then f @ 0 else Eoutput m;; (restart @ f) @ (m ~- 1))
   (If m ~<= 0 then g @ 0 else Eoutput m;; (restart @ g) @ (m ~- 1))
| _fgsim13: forall (m: nat), fgsim
   (If ble_nat m 0 then f @ 0 else Eoutput m;; (restart @ f) @ (m ~- 1)
   (If ble_nat m 0 then g @ 0 else Eoutput m;; (restart @ g) @ (m ~- 1)
| _fgsim14: forall (m: nat), fgsim
   (Eoutput (S m);; (restart @ f) @ (S m ~- 1))
   (Eoutput (S m);; (restart @ g) @ (S m ~- 1))
| _fgsim15: forall (m: nat), fgsim
   ((<>);; (restart @ f) @ (S m ~- 1))
   ((<>);; (restart @ g) @ (S m ~- 1))
| _fgsim16: forall (m: nat), fgsim
   ((restart @ f) @ (S m ~- 1))
   ((restart @ g) @ (S m ~- 1)).
```

```
Lemma rsp_simulated_tarski:
  forall n m (EQ: n = 2 * m), similarity (f @ n) (g @ m).
Proof.
  i; eapply (@simul_tarski fgsim); [clear n m EQ|by eauto].
  i; destruct PR; subst; try by unfold_rfg; simul_step 1; fold_rfg.
  - by simul_step 0; fold_rfg; eauto.
  - by simul_step 0; fold_rfg; eauto.
  - by destruct m0; simul_step 2; eauto.
  - by destruct m; simul_step 1; eauto.
Qed.
```

Proof using Tarski's Principle

```
nductive fgsim : exp -> exp -> Prop :=
 _fgsim1: forall (n m: nat) (EQ: n = 2 * m), fgsim
    (f @ n)
    (g @ m)
 _fgsim2: forall (m: nat), fgsim
    (Let "v" := (Eoutput (m + (m + 0));; Einput) ~* 2
     in (If 1 ~<= "v" then f else restart @ f) @ ("v" ~+ (m + (m + 0))))
    (Eoutput (2 ~* m);;
     (Let "v" := Einput
      in If "v" ~<= 0 then (restart @ g) @ (2 ~* m) else g @ ("v" ~+ m)))
 _fgsim3: forall (m: nat), fgsim
    (Let "v" := ((<>);; Einput) ~* 2
     in (If 1 ~<= "v" then f else restart @ f) @ ("v" ~+ (m + (m + 0))))
    ((<>);;
     (Let "v" := Einput
      in If "v" ~<= 0 then (restart @ g) @ (2 ~* m) else g @ ("v" ~+ m)))
 _fgsim4: forall (m: nat), fgsim
    (Let "v" := Einput ~* 2
     in (If 1 ~<= "v" then f else restart @ f) @ ("v" ~+ (m + (m + 0))))
    (Let "v" := Einput
     in If "v" ~<= 0 then (restart @ g) @ (2 ~* m) else g @ ("v" ~+ m))
 _fgsim5: forall (m m0: nat), fgsim
    (Let "v" := m0 ~* 2
     in (If 1 ~<= "v" then f else restart @ f) @ ("v" ~+ (m + (m + 0))))
    (Let "v" := m0
     in If "v" ~<= 0 then (restart @ g) @ (2 ~* m) else g @ ("v" ~+ m))
 _fgsim6: forall (m m0: nat), fgsim
    (Let "v" := m0 * 1 + m0
     in (If 1 ~<= "v" then f else restart @ f) @ ("v" ~+ (m + (m + 0)))
    (If m0 ~<= 0 then (restart @ g) @ (2 ~* m) else g @ (m0 ~+ m))
 _fgsim7: forall (m m0: nat), fgsim
    ((If 1 ~<= (m0 * 1 + m0) then f else restart @ f) @
     ((m0 * 1 + m0) ~+ (m + (m + 0))))
    (If ble_nat m0 0 then (restart @ g) @ (2 ~* m) else g @ (m0 ~+ m))
 _fgsim8: forall (m m0: nat), fgsim
    ((If 1 ~<= (m0 * 1 + m0) then f else restart @ f) @
     (m0 * 1 + m0 + (m + (m + 0))))
    (If ble_nat m0 0 then (restart @ g) @ (2 ~* m) else g @ (m0 ~+ m))
| _fgsim9: forall (m m0: nat), fgsim
    ((If ble_nat 1 (m0 * 1 + m0) then f else restart @ f) @
     (m0 * 1 + m0 + (m + (m + 0))))
    (If ble_nat m0 0 then (restart @ g) @ (2 ~* m) else g @ (m0 ~+ m))
| _fgsim10: forall (m: nat), fgsim
    ((restart @ f) @ m) ((restart @ g) @ m)
| _fgsim11: forall (m: nat), fgsim
    (Efix "_" "n"
       (If "n" ~<= 0 then f @ 0
        else Eoutput "n";; (restart @ f) @ ("n" ~- 1)) @ m)
    (Efix "_" "n"
       (If "n" ~<= 0 then g @ 0
        else Eoutput "n";; (restart @ g) @ ("n" ~- 1)) @ m)
| _fgsim12: forall (m: nat), fgsim
    (If m ~<= 0 then f @ 0 else Eoutput m;; (restart @ f) @ (m ~- 1))
    (If m ~<= 0 then g @ 0 else Eoutput m;; (restart @ g) @ (m ~- 1))
| _fgsim13: forall (m: nat), fgsim
    (If ble_nat m 0 then f @ 0 else Eoutput m;; (restart @ f) @ (m ~- 1)
    (If ble_nat m 0 then g @ 0 else Eoutput m;; (restart @ g) @ (m ~- 1)
| _fgsim14: forall (m: nat), fgsim
    (Eoutput (S m);; (restart @ f) @ (S m ~- 1))
    (Eoutput (S m);; (restart @ g) @ (S m ~- 1))
| _fgsim15: forall (m: nat), fgsim
    ((<>);; (restart @ f) @ (S m ~- 1))
    ((<>);; (restart @ g) @ (S m ~- 1))
| _fgsim16: forall (m: nat), fgsim
    ((restart @ f) @ (S m ~- 1))
    ((restart @ g) @ (S m ~- 1)).
```

```
emma rsp_simulated_tarski:
  forall n m (EQ: n = 2 * m), similarity (f @ n) (g @ m).
Proof.
  i; eapply (@simul_tarski fgsim); [clear n m EQ|by eauto].
  i; destruct PR; subst; try by unfold_rfg; simul_step 1; fold_rfg.
  - by simul_step 0; fold_rfg; eauto.
  - by simul_step 0; fold_rfg; eauto.
  - by destruct m0; simul_step 2; eauto.
  - by destruct m; simul_step 1; eauto.
ed.
```

Proof using Guarded Coinduction

```
Lemma rsp_simulated_cofix:
  forall n m (EQ: n = 2 * m), similarity (f @ n) (g @ m).
Proof.
  cofix CIH.
  intros; subst; do 6 csimul_step 1; do 2 csimul_step 0.
  destruct m0; csimul_step 2; [|by eauto].
  fold_rfg; generalize (m+(m+0)).
  cofix CIH'.
  intros; do 3 csimul_step 1.
  destruct n; csimul_step 1; [by eauto|].
  do 3 csimul_step 1; eauto.
End.
```

Proof using Guarded Coinduction

Thanks to

Incremental Proof + Simple Automation

```
Lemma rsp_simulated_cofix:
  forall n m (EQ: n = 2 * m), similarity (f @ n) (g @ m).
Proof.
  cofix CIH.
  intros; subst; do 6 csimul_step 1; do 2 csimul_step 0.
  destruct m0; csimul_step 2; [|by eauto].
  fold_rfg; generalize (m+(m+0)).
  cofix CIH'.
  intros; do 3 csimul_step 1.
  destruct n; csimul_step 1; [by eauto|].
  do 3 csimul_step 1; eauto.
Qed.
```

Proof using Guarded Coinduction

Thanks to

Incremental Proof + Simple Automation

BUT!!!

```
Lemma rsp_simulated_cofix:
  forall n m (EQ: n = 2 * m), similarity (f @ n) (g @ m).
Proof.
  cofix CIH.
  intros; subst; do 6 csimul_step 1; do 2 csimul_step 0.
  destruct m0; csimul_step 2; [|by eauto].
  fold_rfg; generalize (m+(m+0)).
  cofix CIH'.
  intros; do 3 csimul_step 1.
  destruct n; csimul_step 1; [by eauto|].
  do 3 csimul_step 1; eauto.
End.
```

Problems with Syntactic Guardedness

```
Variable Node: Type.
Variable R: relation Node.

CoInductive infpath x : Prop :=
_infpath y (STEP: R x y) (INF: infpath y).
Hint Constructors infpath.

Lemma Park's_principle:
  forall (P: Node -> Prop),
  (forall x, P x -> exists y, clos_trans_1n _ R x y /\ P y) ->
  forall x, P x -> infpath x.
Proof.
  cofix CIH; intros P M x Px.
  destruct (M _ Px) as (y & C & Py); clear Px.
  induction C; eauto.
Qed.

-\---  x.v              56%  (51,0)        (Coq Script(0-) Holes)
No more subgoals.
(dependent evars: ?250 using , ?260 using , ?270 using ,)
```

Problems with Syntactic Guardedness

```
Variable Node: Type.
Variable R: relation Node.

CoInductive infpath x : Prop :=
_infpath y (STEP: R x y) (INF: infpath y).
Hint Constructors infpath.

Lemma Park's_principle:
  forall (P: Node -> Prop),
  (forall x, P x -> exists y, clos_trans_1n _ R x y /\ P y) ->
  forall x, P x -> infpath x.
Proof.
  cofix CIH; intros P M x Px.
  destruct (M _ Px) as (y & C & Py); clear Px.
  induction C; eauto.
Qed.

-\---  x.v            56%  (51,0)       (Coq Script(0-) Holes)
No more subgoals.
(dependent evars: ?250 using , ?260 using , ?270 using ,)
```

Problems with Syntactic Guardedness

```
Variable Node: Type.
Variable R: relation Node.

CoInductive infpath x : Prop :=
_infpath y (STEP: R x y) (INF: infpath y).
Hint Constructors infpath.

Lemma Park's_principle:
  forall (P: Node -> Prop),
  (forall x, P x -> exists y, clos_trans_1n _ R x y /\ P y) ->
  forall x, P x -> infpath x.
Proof.
  cofix CIH; intros P M x Px.
  destruct (M _ Px) as (y & C & Py); clear Px.
  induction C; eauto.
Qed.

-\---  x.v              56% (51,0)        (Coq Script(0-) Holes)
No more subgoals.
(dependent evars: ?250 using , ?260 using , ?270 using ,)
```

Problems with Syntactic Guardedness

```
Variable Node: Type.
Variable R: relation Node.

CoInductive infpath x : Prop :=
_infpath y (STEP: R x y) (INF: infpath y).
Hint Constructors infpath.

Lemma Park's_principle:
  forall (P: Node -> Prop),
  (forall x, P x -> exists y, clos_trans_1n _ R x y /\ P y) ->
  forall x, P x -> infpath x.
Proof.
  cofix CIH. intros P M x Px.
  destruct (M _ Px) as (y & C & Py); clear Px.
  induction C; eauto.
Qed.

-\---  x.v             56%  (51,0)       (Coq Script(0-) Holes)
No more subgoals.
(dependent evars: ?250 using , ?260 using , ?270 using ,)
```

Problems with Syntactic Guardedness

```
Variable Node: Type.
Variable R: relation Node.

CoInductive infpath x : Prop :=
_infpath y (STEP: R x y) (INF: infpath y).
Hint Constructors infpath.

Lemma Park's_principle:
  forall (P: Node -> Prop),
  (forall x, P x -> exists y, clos_trans_1n _ R x y /\ P y) ->
  forall x, P x -> infpath x.
Proof.
  cofix CIH; intros P M x Px.
  destruct (M _ Px) as (y & C & Py); clear Px.
  induction C; eauto.
Qed.

-\---  x.v          56% (51,0)       (Coq Script(0-) Holes)
No more subgoals.
(dependent evars: ?250 using , ?260 using , ?270 using ,)
```

Problems with Syntactic Guardedness

```
Variable Node: Type.
Variable R: relation Node.

CoInductive infpath x : Prop :=
_infpath y (STEP: R x y) (INF: infpath y).
Hint Co

Lemma P
  foral
  (fora                                                y) ->
  foral
Proof.
  cofix CIH; intros P M x Px.
  destruct (M _ Px) as (y & C & Py); clear Px.
  induction C; eauto.
Qed.
```

The proof is

NOT Syntactically Guarded

```
-\---  x.v          56% (52,0)       (Coq Script(0-) Holes)
Error:
Recursive definition of CIH is ill-formed.
```

Problems with Syntactic Guardedness

Guardedness NOT Expressed in the Logic

1. Far from complete
2. Bad interaction with automation
3. Hard to debug
4. Slow

```
Proof.
  cofix CIH; intros P M x Px.
  destruct (M _ Px) as (y & C & Py); clear Px.
  induction C; eauto.
Qed.

-\---  x.v          56%  (52,0)      (Coq Script(0-) Holes)
Error:
Recursive definition of CIH is ill-formed.
```

Problems with Syntactic Guardedness

Guardedness NOT Expressed in the Logic

1. Far from complete
2. Bad interaction with automation
3. Hard to debug
4. Slow

```
Proof.
  cofix CIH; intros P M x Px.
  destruct (M _ Px) as (y & C & Py); clear Px.
  induction C; eauto.
Qed.

-\---  x.v          56%  (52,0)      (Coq Script(0-) Holes)
Error:
Recursive definition of CIH is ill-formed.
```

Summary:
Pros and Cons of Two Principles

- Tarski's Fixed Point Theorem
  + Simple & Robust
  - Inconvenient to use

- Syntactically Guarded Coinduction
  - Complex & Fragile due to "Guardedness Checking"
  + More Convenient to use

Talk Outline

- ❑ Previous Approaches
  - ➢ Tarski's Fixed Point Theorem
  - ➢ Syntactically Guarded Coinduction
- ❑ Our Approach
  - ➢ Parameterized Coinduction

Syntactically Guarded Coinduction

$$f : \wp(U) \xrightarrow{\text{mon}} \wp(U)$$

Syntactically Guarded Coinduction

$$\frac{\text{pf: } X \subseteq \nu f \implies X \subseteq \nu f}{X \subseteq \nu f} \text{ (pf is guarded)}$$

Semantically Guarded Coinduction

Guardedness Expressed Directly Within the Logic

Semantically Guarded Coinduction

$$\frac{X \subseteq \nu f \overset{\mathrm{G}}{\Longrightarrow} X \subseteq \nu f}{X \subseteq \nu f}$$

Guarded Implication

$$Y \subseteq \nu f \overset{\mathrm{G}}{\Longrightarrow} X \subseteq \nu f$$

Guarded Implication

Do NOT want to EXTEND the logic

$$Y \subseteq \nu f \overset{\mathrm{G}}{\Longrightarrow} X \subseteq \nu f$$

Guarded Implication

Instead, Define G_f

Called "Parameterized Greatest Fixed Point" of f

$$Y \subseteq \nu f \overset{\mathrm{G}}{\Longrightarrow} X \subseteq \nu f$$

$$\Downarrow$$

$$X \subseteq \mathrm{G}_f(Y)$$

Guarded Implication

Greatest Fixed Point of f

Under Guarded Assumption of $Y \subseteq \nu f$

$$Y \subseteq \nu f \overset{\mathrm{G}}{\Longrightarrow} X \subseteq \nu f$$

$$\Downarrow$$

$$X \subseteq \boxed{\mathrm{G}_f(Y)}$$

Guarded Implication

Greatest Fixed Point of f

Under Guarded Assumption of $Y \subseteq \nu f$

$$Y \subseteq \nu f \overset{\mathrm{G}}{\Longrightarrow} X \subseteq \nu f$$

$$\Downarrow$$

$$X \subseteq \boxed{\mathrm{G}_f(Y)}$$

$$\mathrm{G}_f(Y) \overset{\text{def}}{=} \nu(\lambda S.\, f(Y \cup S))$$

Properties of G_f

(Init) $\mathrm{G}_f(\emptyset) = \nu f$

(Unfolding) $\mathrm{G}_f(Y) = f(Y \cup \mathrm{G}_f(Y))$

(Semantically Guarded Coinduction)

$$X \subseteq \mathrm{G}_f(Y) \iff X \subseteq \mathrm{G}_f(X \cup Y)$$

Properties of G_f

(Init) $\mathrm{G}_f(\emptyset) = \nu f$

(Unfolding) $\mathrm{G}_f(Y) = f(Y \cup \mathrm{G}_f(Y))$

(Semantically Guarded Coinduction)

$$X \subseteq \mathrm{G}_f(Y) \iff X \subseteq \mathrm{G}_f(X \cup Y)$$

Properties of $\mathbf{G}_f$

(Init) $\mathrm{G}_f(\emptyset) = \nu f$

(Unfolding) $\mathrm{G}_f(Y) = f(Y \cup \mathrm{G}_f(Y))$

(Semantically Guarded Coinduction)

$$X \subseteq \mathrm{G}_f(Y) \iff X \subseteq \mathrm{G}_f(X \cup Y)$$

Properties of G_f

(Init) $\mathrm{G}_f(\emptyset) = \nu f$

(Unfolding) $\mathrm{G}_f(Y) = f(\boxed{Y \cup}\, \mathrm{G}_f(Y))$

(Semantically Guarded Coinduction)

$$X \subseteq \mathrm{G}_f(Y) \iff X \subseteq \mathrm{G}_f(X \cup Y)$$

Properties of G_f

(Init) $\quad \mathrm{G}_f(\emptyset) = \nu f$

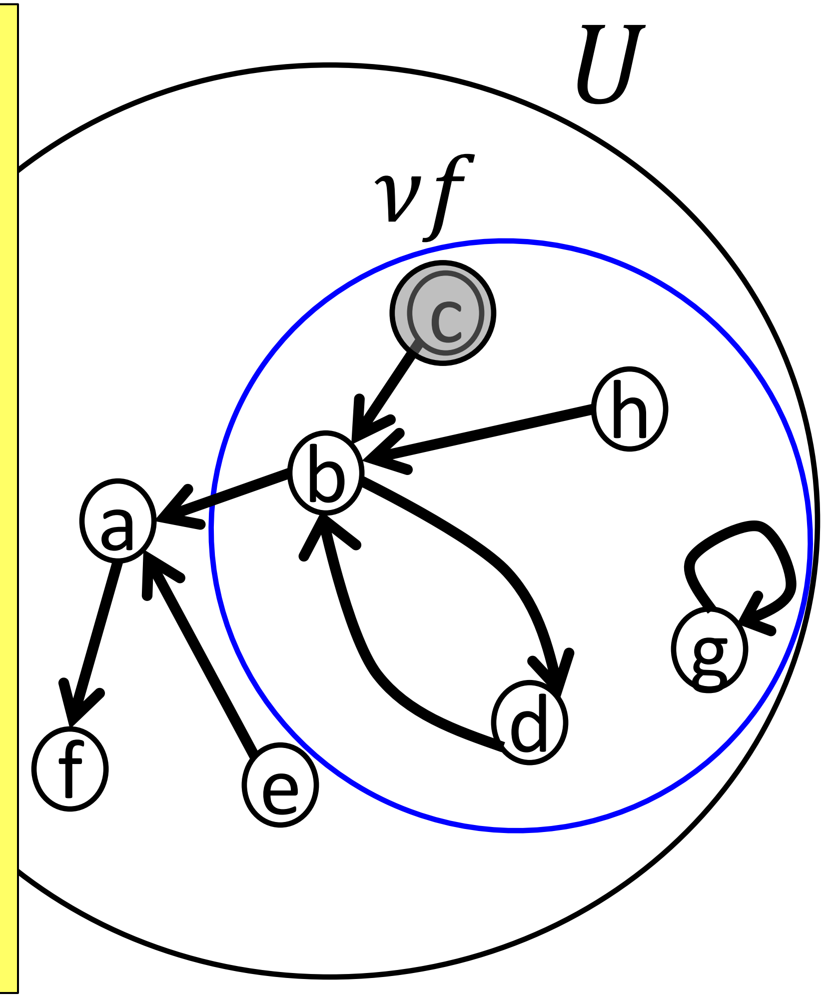
(Unfolding) $\quad \mathrm{G}_f(Y) = f(Y \cup \mathrm{G}_f(Y))$

(Semantically Guarded Coinduction)

$$X \subseteq \mathrm{G}_f(Y) \iff X \subseteq \mathrm{G}_f(X \cup Y)$$

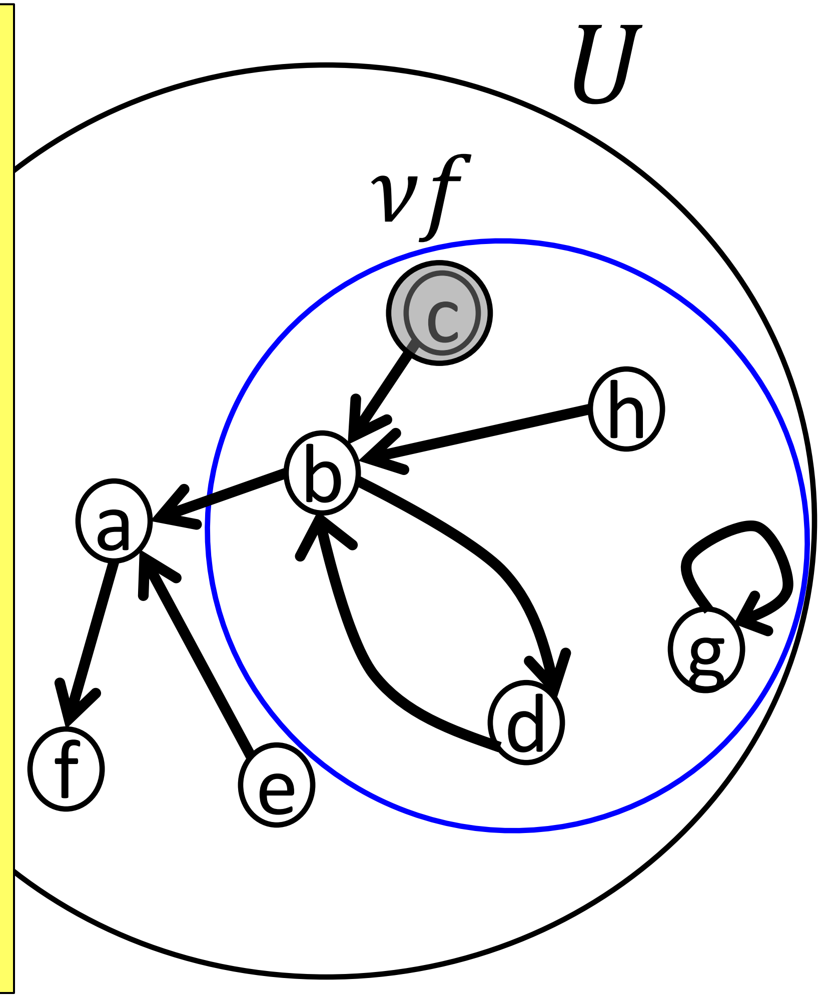
Example: Parameterized Coinduction

$\{c\} \subseteq \nu f$



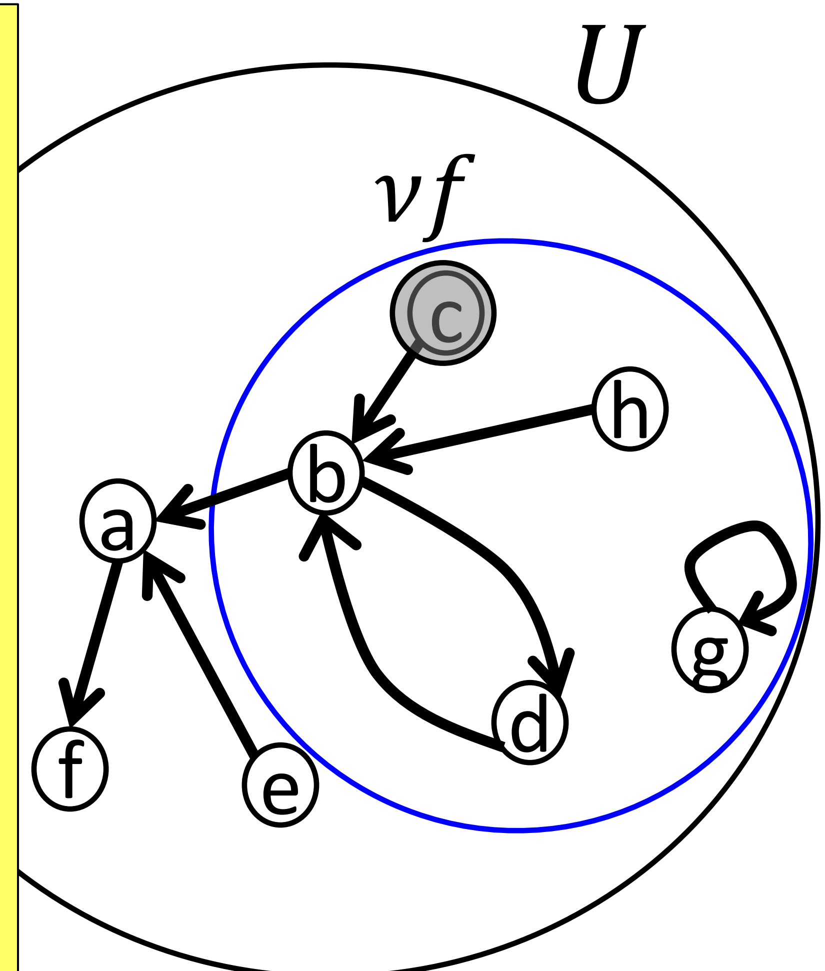
Example: Parameterized Coinduction

$\{c\} \subseteq G_f(\emptyset)$



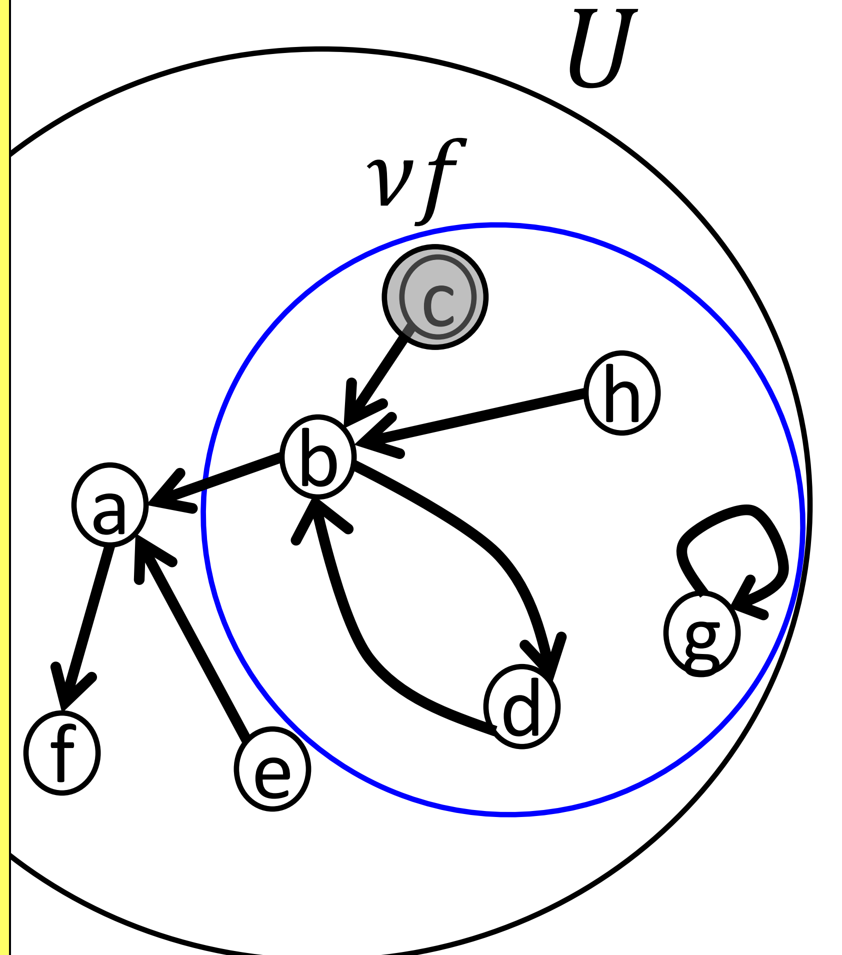
Example: Parameterized Coinduction

$$\frac{\{c\} \subseteq f(\emptyset \cup G_f(\emptyset))}{\{c\} \subseteq G_f(\emptyset)}$$



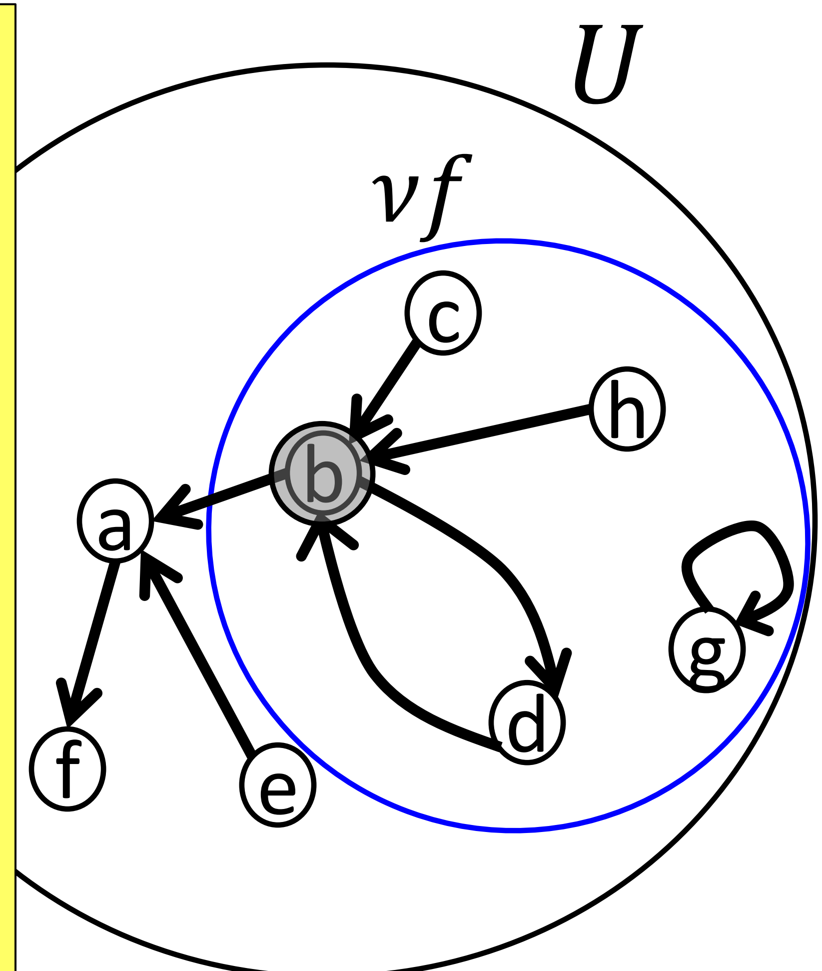
Example: Parameterized Coinduction

$$\frac{\exists y.\, \mathrm{c} \to y \land y \in (\emptyset \cup \mathrm{G}_f(\emptyset))}{\{\mathrm{c}\} \subseteq \mathrm{G}_f(\emptyset)}$$



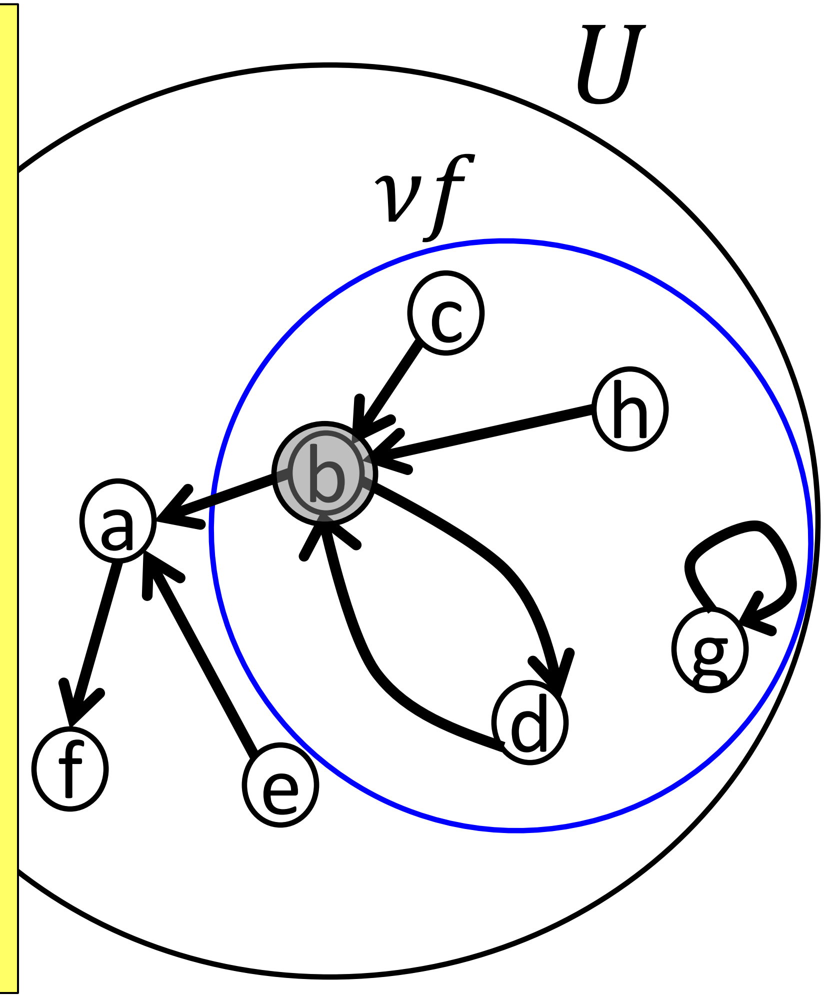
Example: Parameterized Coinduction

$$\frac{\exists y.\, \mathrm{c} \to y \land y \in (\emptyset \cup \mathrm{G}_f(\emptyset))}{\{\mathrm{c}\} \subseteq \mathrm{G}_f(\emptyset)}$$



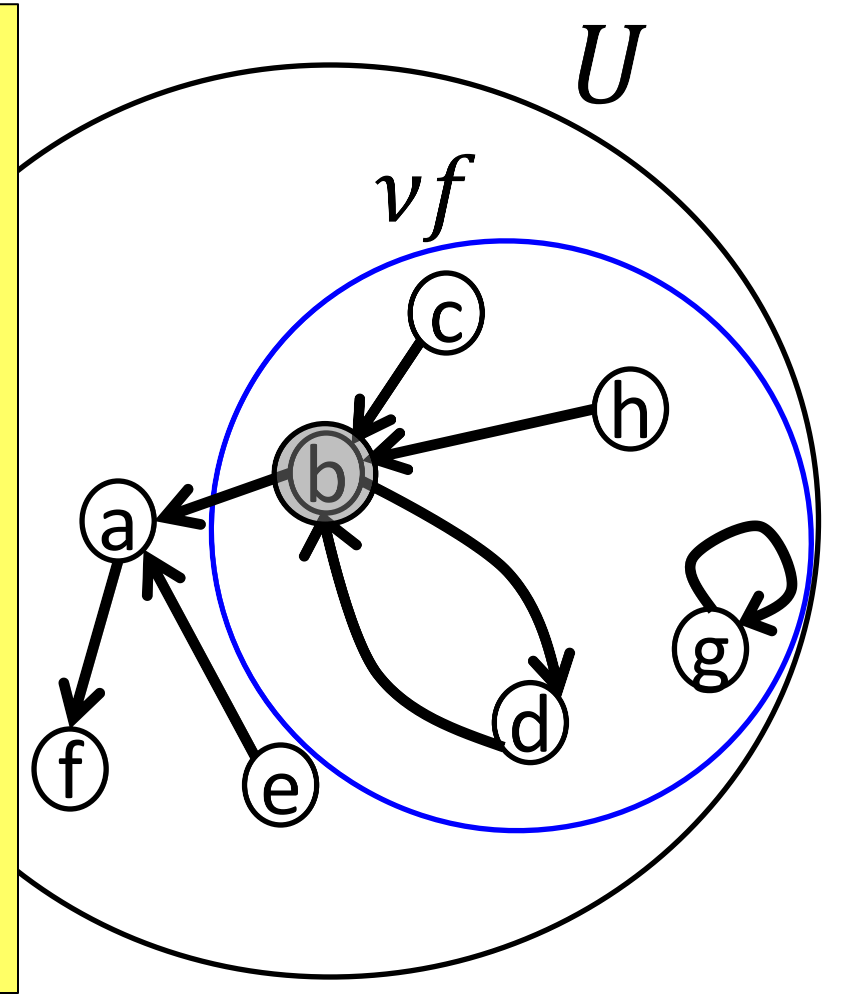
Example: Parameterized Coinduction

$$\frac{b \in (\emptyset \cup G_f(\emptyset))}{\{c\} \subseteq G_f(\emptyset)}$$



Example: Parameterized Coinduction

$$\frac{\dfrac{\{b\} \subseteq G_f(\emptyset)}{b \in (\emptyset \cup G_f(\emptyset))}}{\{c\} \subseteq G_f(\emptyset)}$$



Examp…
Parameterized C…

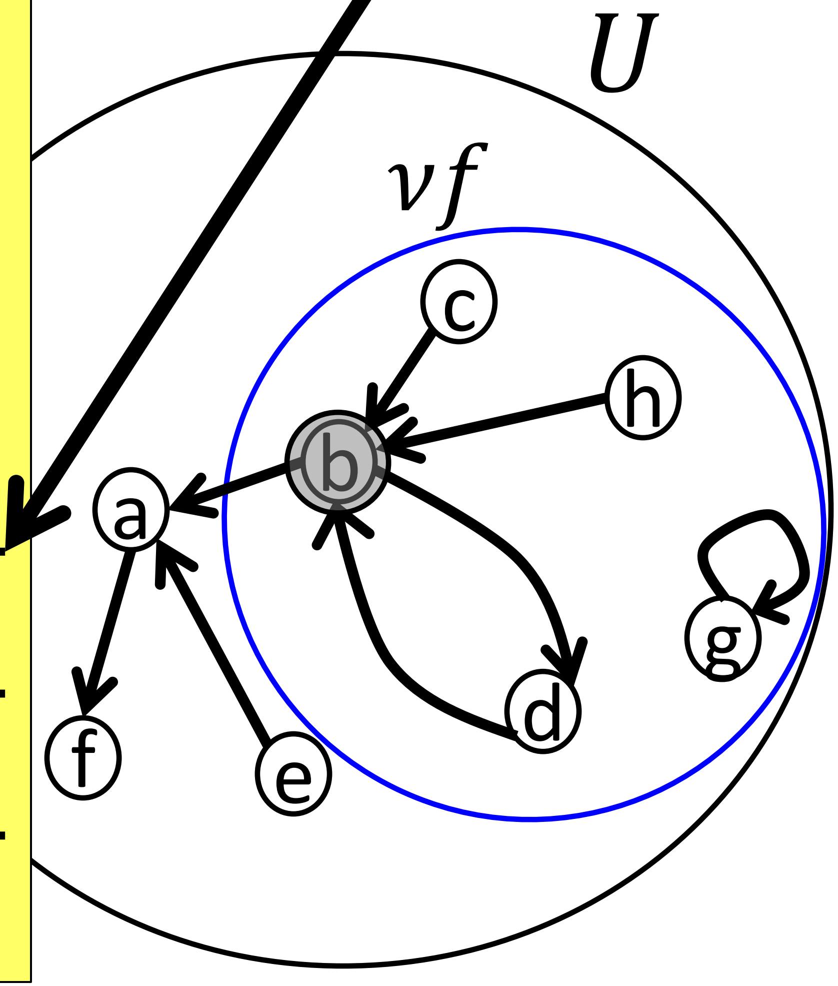
By Semantically Guarded Coinduction

$$\{b\} \subseteq G_f(\{b\})$$

$$\{b\} \subseteq G_f(\emptyset)$$

$$b \in (\emptyset \cup G_f(\emptyset))$$

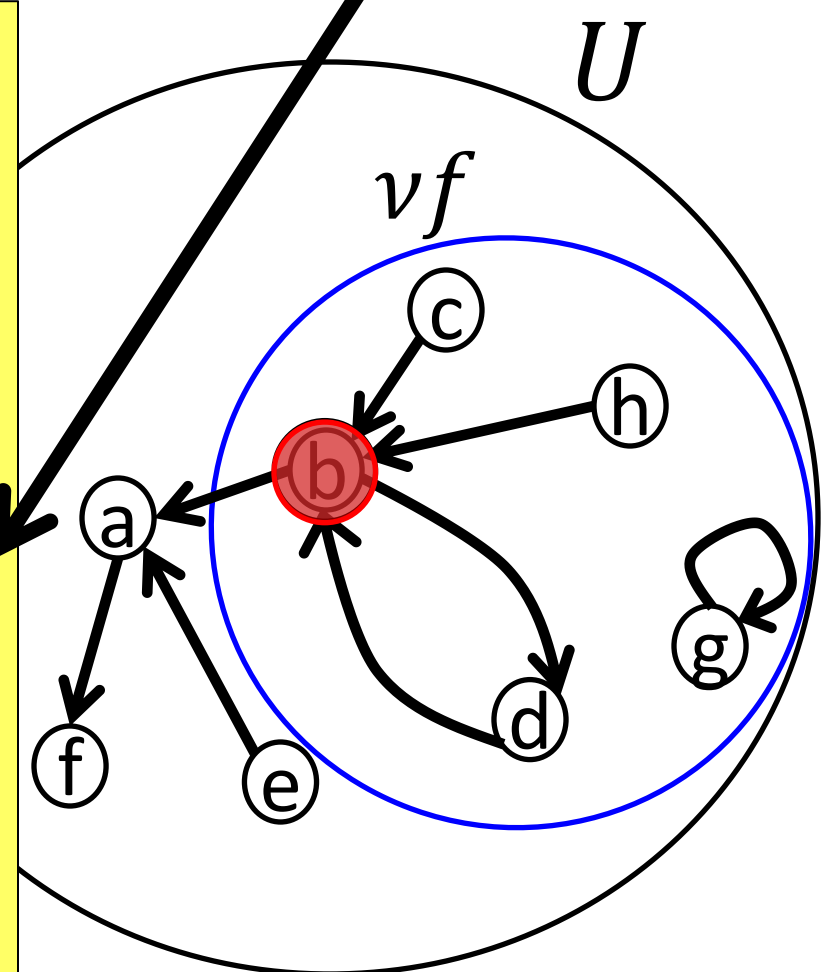
$$\{c\} \subseteq G_f(\emptyset)$$



Examp
Parameterized

By Semantically
Guarded Coinduction

$$\dfrac{\dfrac{\dfrac{\{b\} \subseteq G_f(\{b\})}{\{b\} \subseteq G_f(\emptyset)}}{b \in (\emptyset \cup G_f(\emptyset))}}{\{c\} \subseteq G_f(\emptyset)}$$



NO Trivial Proof

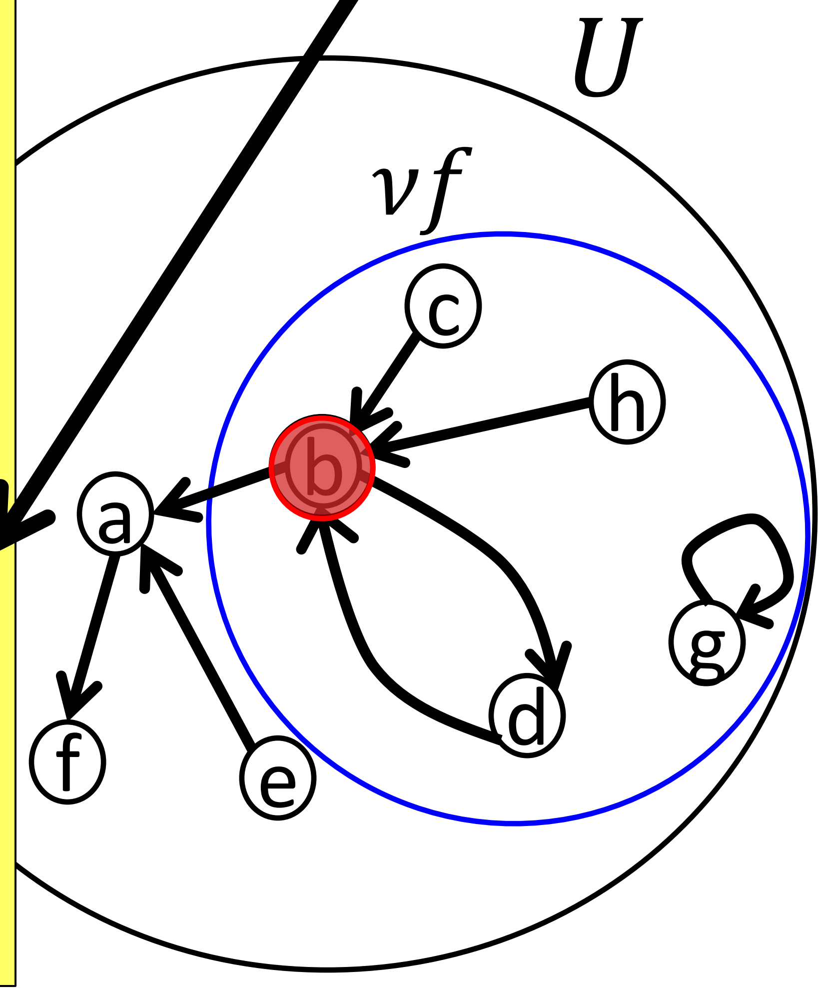
By Semantically Guarded Coinduction

$$\{b\} \subseteq G_f(\{b\})$$

$$\{b\} \subseteq G_f(\emptyset)$$

$$b \in (\emptyset \cup G_f(\emptyset))$$

$$\{c\} \subseteq G_f(\emptyset)$$



Examp
Parameterized C

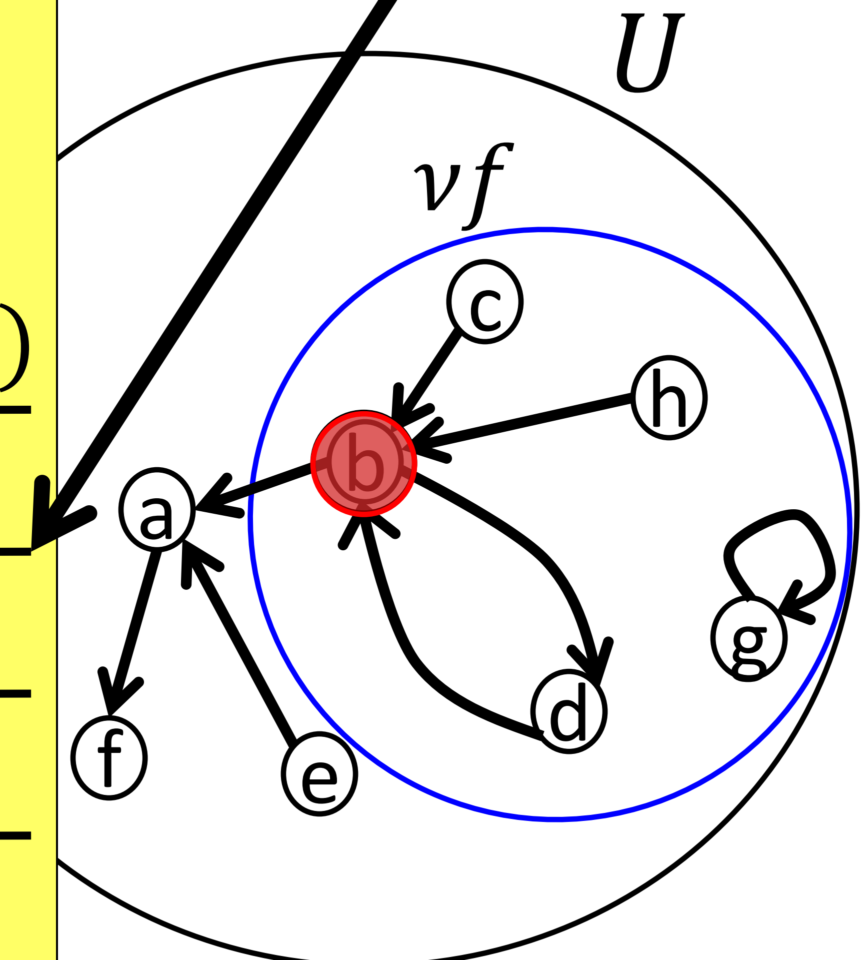
By Semantically
Guarded Coinduction

$$\{b\} \subseteq f(\{b\} \cup G_f(\{b\}))$$

$$\frac{\{b\} \subseteq G_f(\{b\})}{\{b\} \subseteq G_f(\emptyset)}$$

$$b \in (\emptyset \cup G_f(\emptyset))$$

$$\{c\} \subseteq G_f(\emptyset)$$



Examp… Parameterized C…

By Semantically Guarded Coinduction

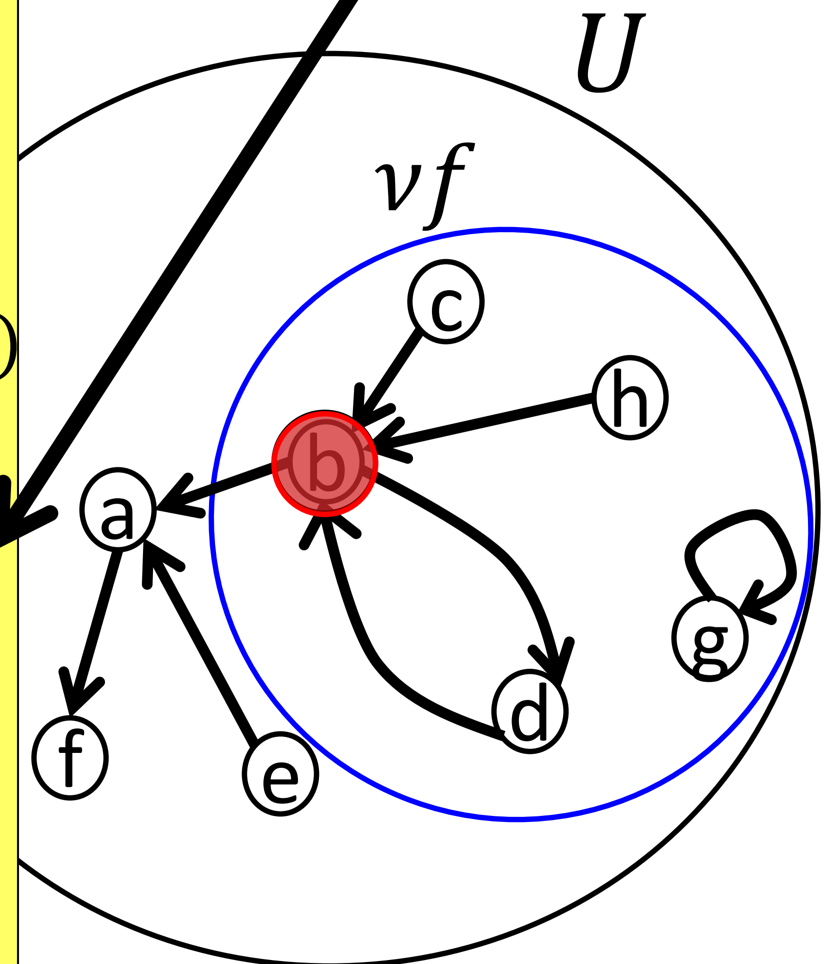
$$\exists y.\, \mathrm{b} \to y \land y \in (\{\mathrm{b}\} \cup \mathrm{G}_f(\{\mathrm{b}\}))$$

$$\overline{\{\mathrm{b}\} \subseteq \mathrm{G}_f(\{\mathrm{b}\})}$$

$$\overline{\{\mathrm{b}\} \subseteq \mathrm{G}_f(\emptyset)}$$

$$\overline{\mathrm{b} \in (\emptyset \cup \mathrm{G}_f(\emptyset))}$$

$$\overline{\{\mathrm{c}\} \subseteq \mathrm{G}_f(\emptyset)}$$



Examp
Parameterized C

By Semantically
Guarded Coinduction

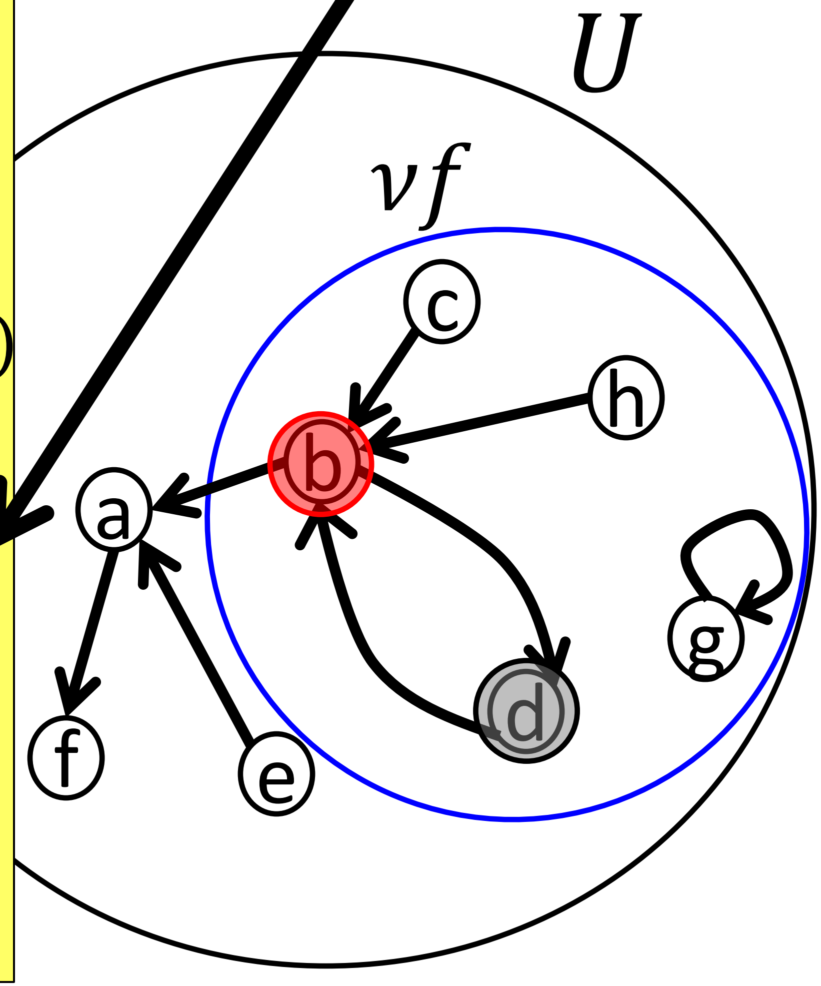
$$\exists y.\, \mathrm{b} \to y \wedge y \in (\{\mathrm{b}\} \cup G_f(\{\mathrm{b}\}))$$

$$\{\mathrm{b}\} \subseteq G_f(\{\mathrm{b}\})$$

$$\{\mathrm{b}\} \subseteq G_f(\emptyset)$$

$$\mathrm{b} \in (\emptyset \cup G_f(\emptyset))$$

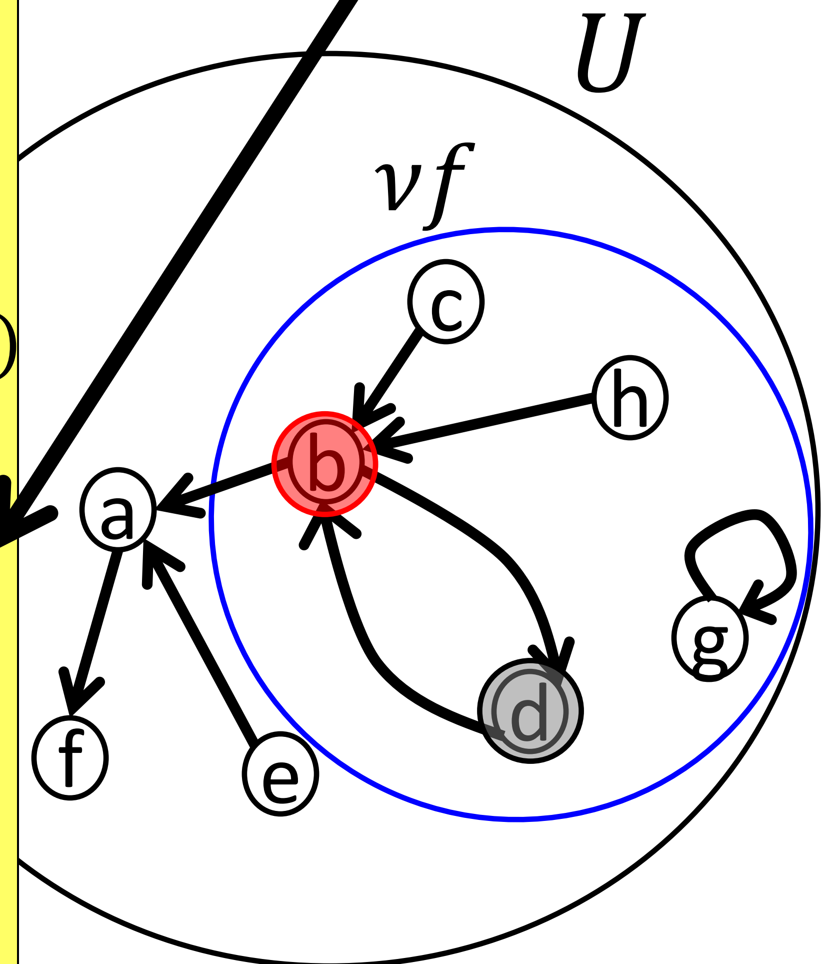
$$\{\mathrm{c}\} \subseteq G_f(\emptyset)$$



Examp… Parameterized C…

By Semantically Guarded Coinduction

$$\frac{\dfrac{\dfrac{\dfrac{d \in (\{b\} \cup G_f(\{b\}))}{\{b\} \subseteq G_f(\{b\})}}{\{b\} \subseteq G_f(\emptyset)}\quad b \in (\emptyset \cup G_f(\emptyset))}{\{c\} \subseteq G_f(\emptyset)}}{}$$



Examp

Parameterized

By Semantically Guarded Coinduction

$$\{d\} \subseteq G_f(\{b\})$$

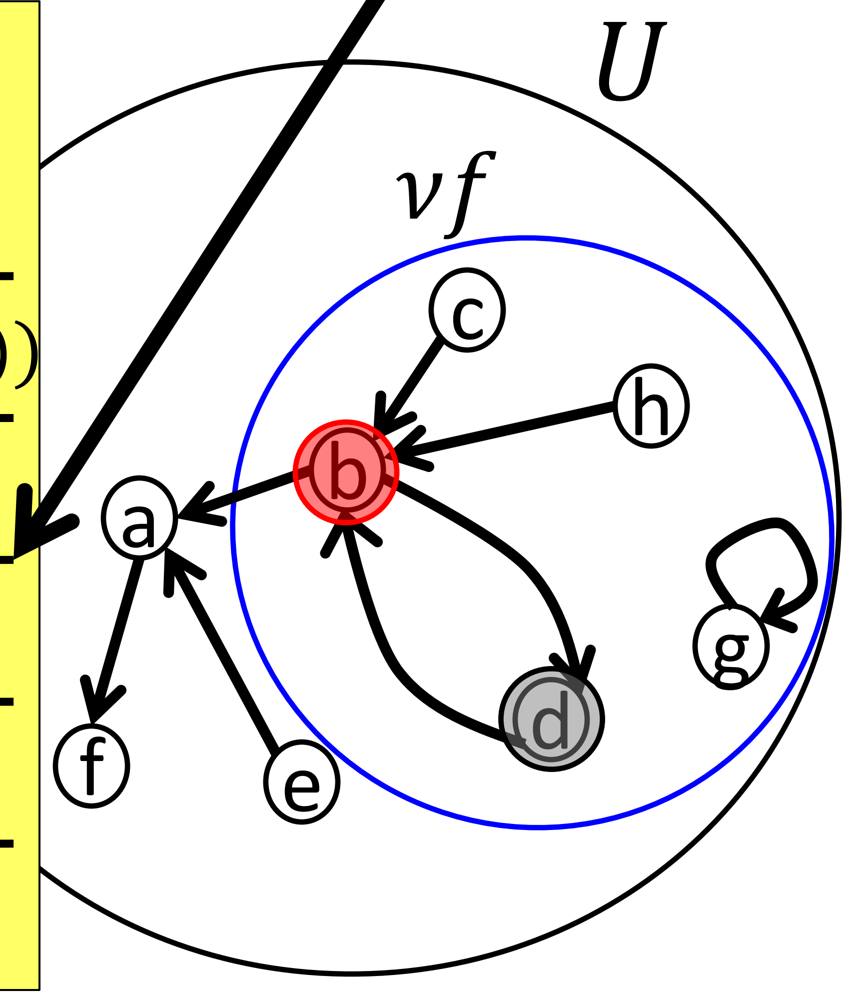
$$d \in (\{b\} \cup G_f(\{b\}))$$

$$\{b\} \subseteq G_f(\{b\})$$

$$\{b\} \subseteq G_f(\emptyset)$$

$$b \in (\emptyset \cup G_f(\emptyset))$$

$$\{c\} \subseteq G_f(\emptyset)$$



Examp… Parameterized C…

By Semantically Guarded Coinduction

$$\{d\} \subseteq f(\{b\} \cup G_f(\{b\}))$$

$$\{d\} \subseteq G_f(\{b\})$$

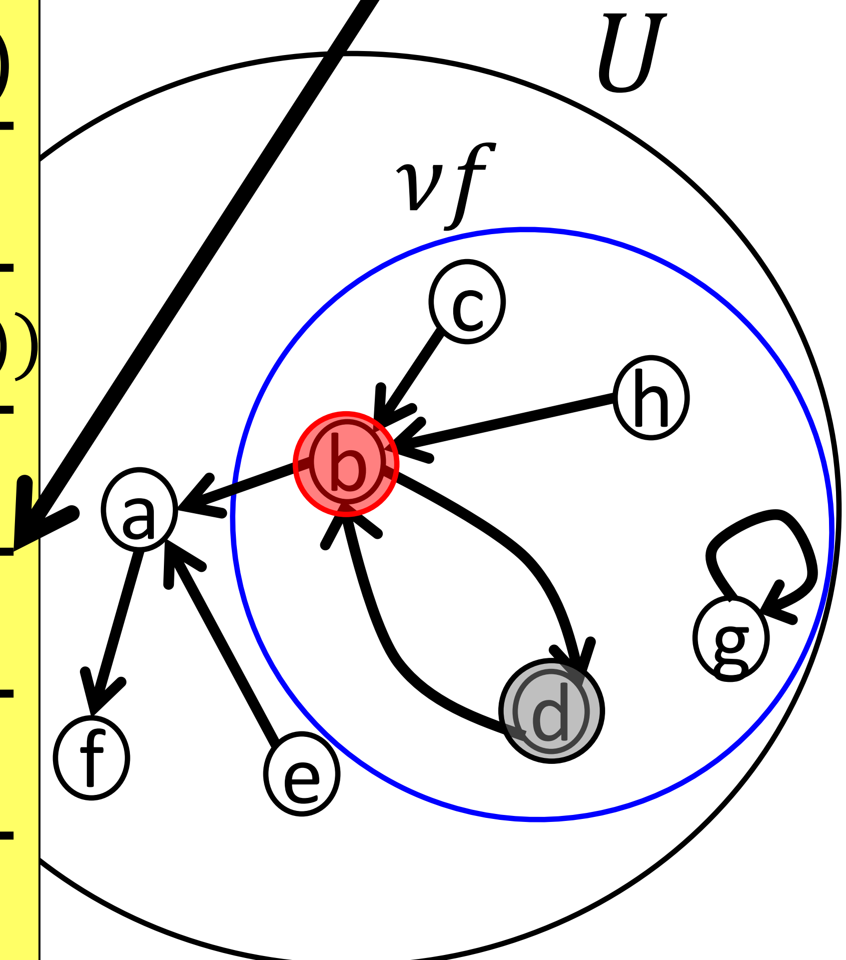
$$d \in (\{b\} \cup G_f(\{b\}))$$

$$\{b\} \subseteq G_f(\{b\})$$

$$\{b\} \subseteq G_f(\emptyset)$$

$$b \in (\emptyset \cup G_f(\emptyset))$$

$$\{c\} \subseteq G_f(\emptyset)$$



Examp… Parameterized C…

By Semantically Guarded Coinduction

$\exists y.\, \mathrm{d} \rightarrow y \land y \in (\{\mathrm{b}\} \cup \mathrm{G}_f(\{\mathrm{b}\}))$

$\{\mathrm{d}\} \subseteq \mathrm{G}_f(\{\mathrm{b}\})$

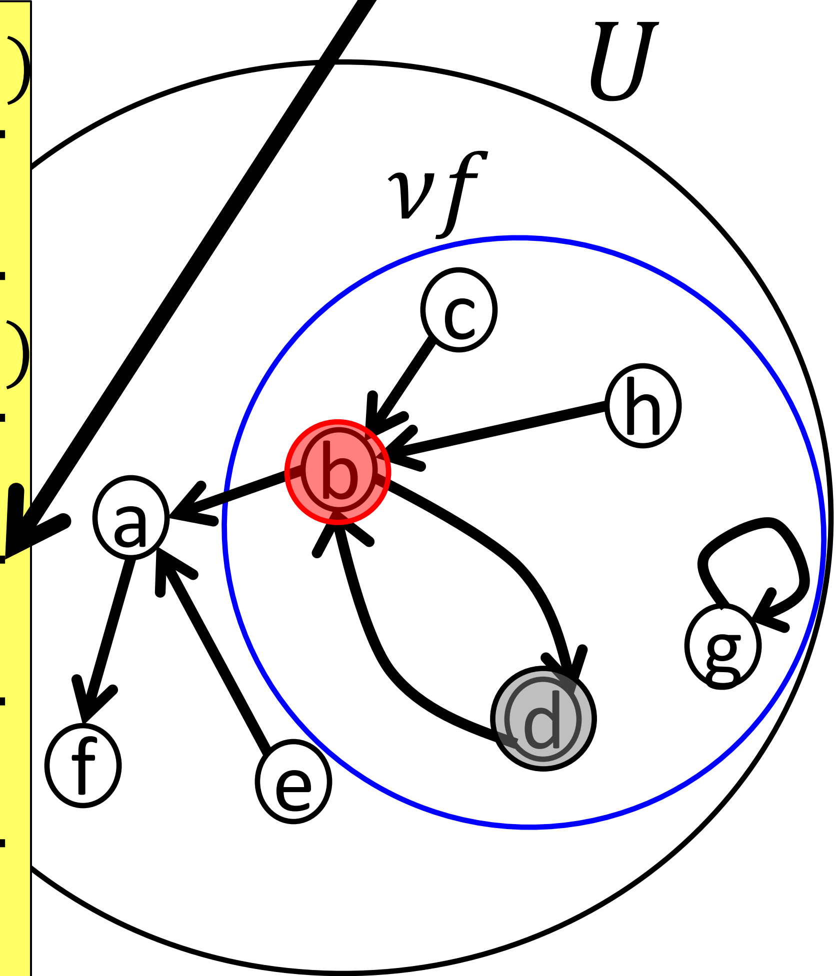
$\mathrm{d} \in (\{\mathrm{b}\} \cup \mathrm{G}_f(\{\mathrm{b}\}))$

$\{\mathrm{b}\} \subseteq \mathrm{G}_f(\{\mathrm{b}\})$

$\{\mathrm{b}\} \subseteq \mathrm{G}_f(\emptyset)$

$\mathrm{b} \in (\emptyset \cup \mathrm{G}_f(\emptyset))$

$\{\mathrm{c}\} \subseteq \mathrm{G}_f(\emptyset)$



Examp…
Parameterized …

By Semantically
Guarded Coinduction

$$\exists y.\, \mathrm{d} \rightarrow y \wedge y \in (\{\mathrm{b}\} \cup \mathrm{G}_f(\{\mathrm{b}\}))$$

$$\{\mathrm{d}\} \subseteq \mathrm{G}_f(\{\mathrm{b}\})$$

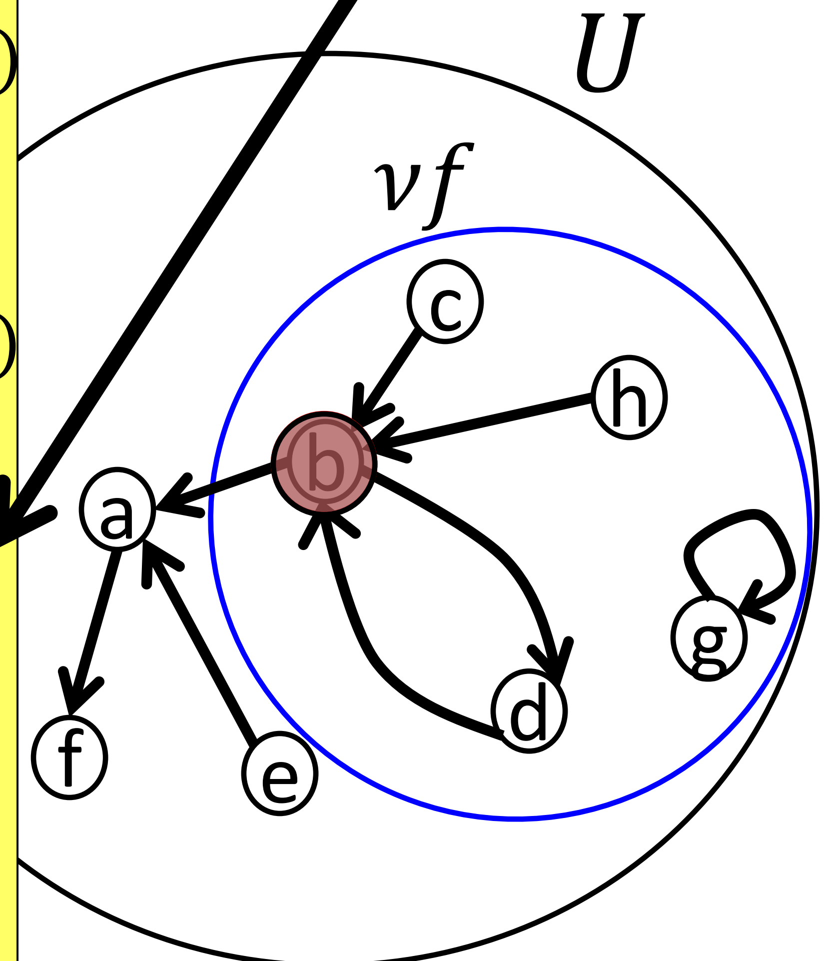
$$\mathrm{d} \in (\{\mathrm{b}\} \cup \mathrm{G}_f(\{\mathrm{b}\}))$$

$$\{\mathrm{b}\} \subseteq \mathrm{G}_f(\{\mathrm{b}\})$$

$$\{\mathrm{b}\} \subseteq \mathrm{G}_f(\emptyset)$$

$$\mathrm{b} \in (\emptyset \cup \mathrm{G}_f(\emptyset))$$

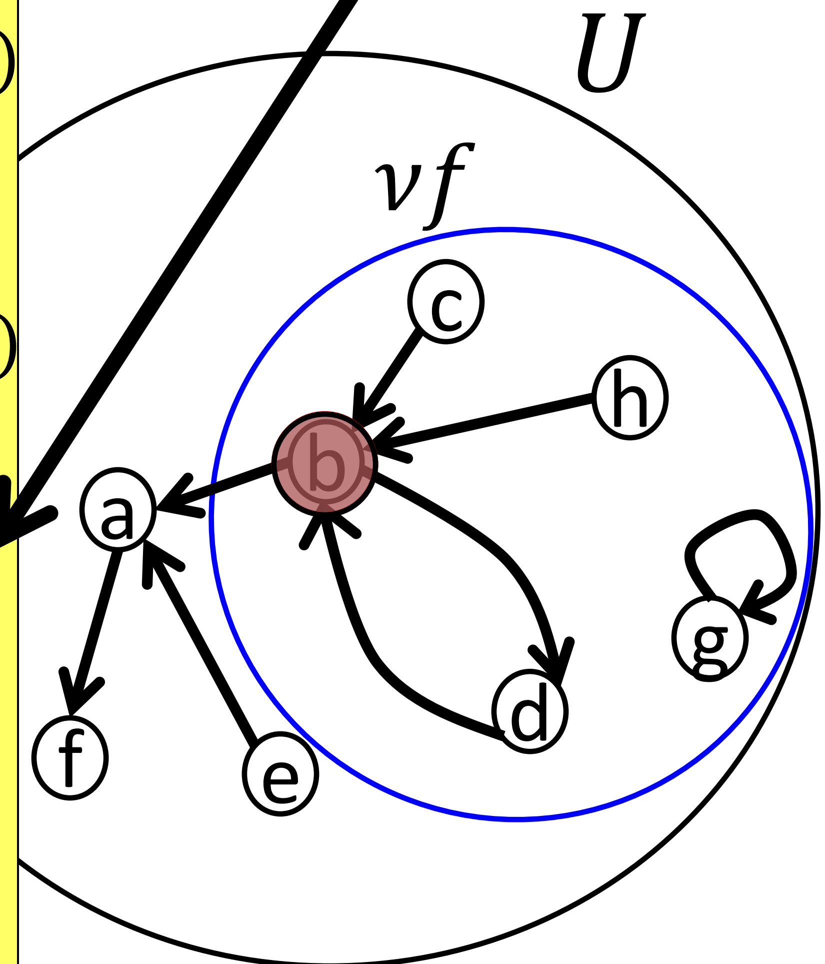
$$\{\mathrm{c}\} \subseteq \mathrm{G}_f(\emptyset)$$



Examp… Parameterized C…

By Semantically
Guarded Coinduction

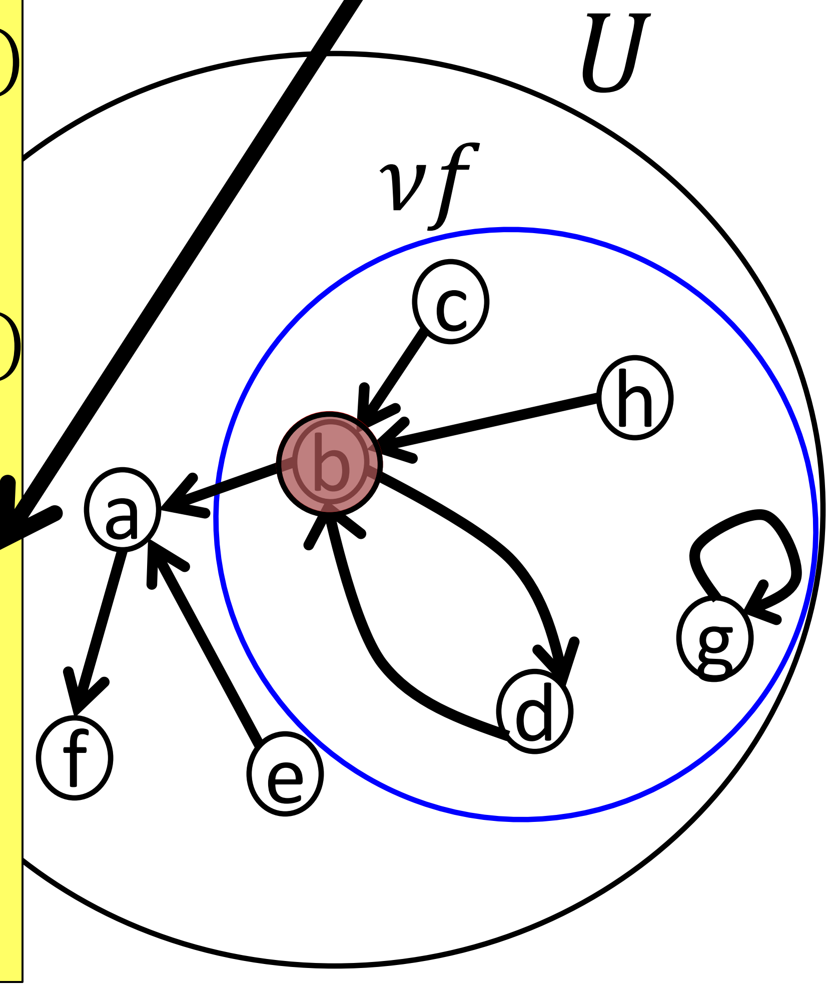
$$
\begin{array}{r}
b \in (\{b\} \cup G_f(\{b\})) \\
\hline
\{d\} \subseteq G_f(\{b\}) \\
\hline
d \in (\{b\} \cup G_f(\{b\})) \\
\hline
\{b\} \subseteq G_f(\{b\}) \\
\hline
\{b\} \subseteq G_f(\emptyset) \\
\hline
b \in (\emptyset \cup G_f(\emptyset)) \\
\hline
\{c\} \subseteq G_f(\emptyset)
\end{array}
$$



Examp… Parameterized C…

By Semantically Guarded Coinduction

$$
\begin{array}{c}
b \in (\{b\} \cup G_f(\{b\})) \\
\hline
\{d\} \subseteq G_f(\{b\}) \\
\hline
d \in (\{b\} \cup G_f(\{b\})) \\
\hline
\{b\} \subseteq G_f(\{b\}) \\
\hline
\{b\} \subseteq G_f(\emptyset) \\
\hline
b \in (\emptyset \cup G_f(\emptyset)) \\
\hline
\{c\} \subseteq G_f(\emptyset)
\end{array}
$$



Examp…
Parameterized C…

By Semantically
Guarded Coinduction

$b \in (\{b\} \cup G_f(\{b\}))$

$\{d\} \subseteq G_f(\{b\})$

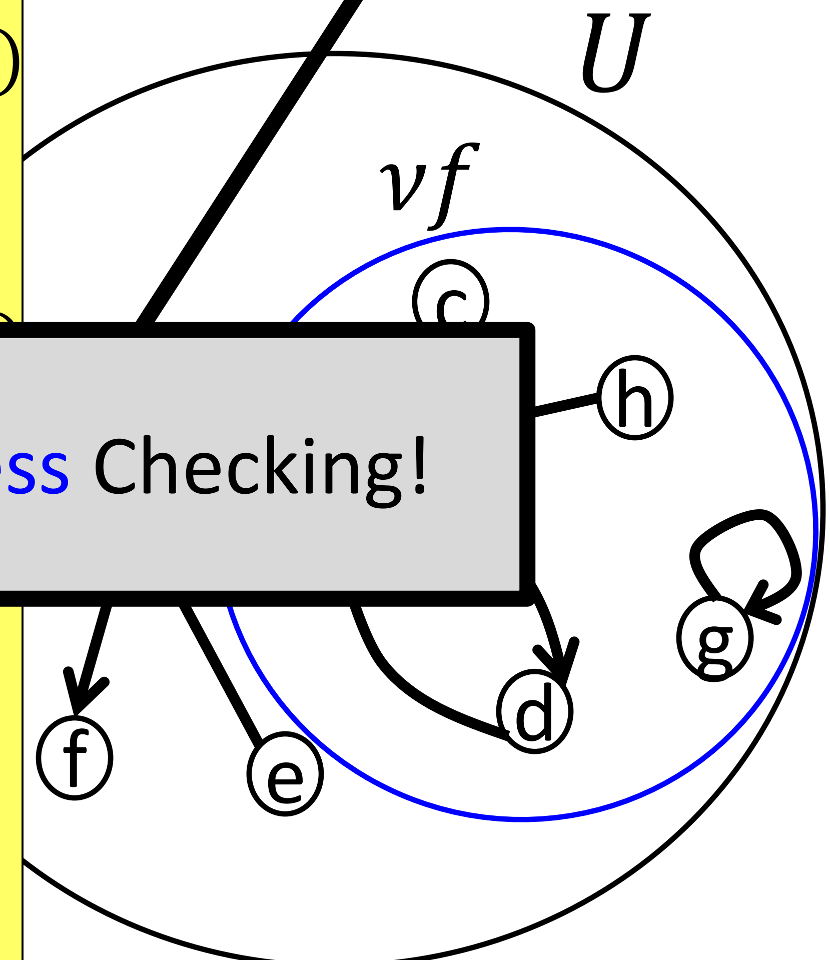
$\{b\} \subseteq G$…

$\{b\} \subseteq G_f(\emptyset)$

$b \in (\emptyset \cup G_f(\emptyset))$

$\{c\} \subseteq G_f(\emptyset)$

NO Guardedness Checking!



Paco: A Coq Library for Parameterized Coinduction

http://plv.mpi-sws.org/paco/

Syntactic Guardedness

```
Lemma rsp_simulated_cofix:
  forall n m (EQ: n = 2 * m), similarity (f @ n) (g @ m).
Proof.
  cofix CIH.
  intros; subst; do 6 csimul_step 1; do 2 csimul_step 0.
  destruct m0; csimul_step 2; [|by eauto].
  fold_rfg; generalize (m+(m+0)).
  cofix CIH'.
  intros; do 3 csimul_step 1.
  destruct n; csimul_step 1; [by eauto|].
  do 3 csimul_step 1; eauto.
Qed.
```

Semantic Guardedness

```
Lemma rsp_simulated_paco:
  forall n m (EQ: n = 2 * m), paco2 sim bot2 (f @ n) (g @ m).
Proof.
  pcofix CIH.
  intros; subst; do 6 psimil_step 1; do 2 psimil_step 0.
  destruct m0; psimil_step 2; [|by eauto].
  left; fold_rfg; generalize (m+(m+0)).
  pcofix CIH'.
  intros; do 3 psimil_step 1.
  destruct n; psimil_step 1; [by eauto|].
  do 3 psimil_step 1; eauto.
Qed.
```

Paco: A Coq Library for Parameterized Coinduction

http://plv.mpi-sws.org/paco/

Syntactic Guardedness

```
Lemma rsp_simulated_cofix:
  forall n m (EQ: n = 2 * m), similarity (f @ n) (g @ m).
Proof.
  cofix CIH.
  intros; subst; do 6 csimul_step 1; do 2 csimul_step 0.
  destruct m0; csimul_step 2; [|by eauto].
  fold rfg; generalize (m+(m+0)).
  cofix CIH'.
  intros; do 3 csimul_step 1.
  destruct n; csimul_step 1; [by eauto|].
  do 3 csimul_step 1; eauto.
Qed.
```

Semantic Guardedness

```
Lemma rsp_simulated_paco:
  forall n m (EQ: n = 2 * m), paco2 sim bot2 (f @ n) (g @ m).
Proof.
  pcofix CIH.
  intros; subst; do 6 psimil_step 1; do 2 psimil_step 0.
  destruct m0; psimil_step 2; [|by eauto].
  left; fold rfg; generalize (m+(m+0)).
  pcofix CIH'.
  intros; do 3 psimil_step 1.
  destruct n; psimil_step 1; [by eauto|].
  do 3 psimil_step 1; eauto.
Qed.
```

Paco: A Coq Library for Parameterized Coinduction

http://plv.mpi-sws.org/paco/

Guardedness Checking
Needed!

tic Guardedness

d_paco:

n = 2 * m), paco2 sim bot2 (f @ n) (g @ m).

```
Proof.
cofix CIH.
  intros; subst; do 6 csimul_step 1; do 2 cs
  destruct m0; csimul_ste  2; [|by eauto].
  fold rfg; generaliz  (m+(m+0)).
cofix CIH'.
  intros; do   csimul_step 1.
  destruct  n; csimul_step 1; [by eauto|].
  do   csimul_step 1; eauto.
Qed.
```

Guardedness Checking
NOT Needed!

```
Proof
  intros; do 3   imil_step 1.
  destruct  , psimil_step 1; [by eauto|].
  do   psimil_step 1; eauto.
Qed.
```

Failed Proof using Syntactically Guarded Coinduction

```
Variable Node: Type.
Variable R: relation Node.

CoInductive infpath x : Prop :=
_infpath y (STEP: R x y) (INF: infpath y).
Hint Constructors infpath.

Lemma Park's_principle:
  forall (P: Node -> Prop),
  (forall x, P x -> exists y, clos_trans_1n _ R x y /\ P y) ->
  forall x, P x -> infpath x.
Proof.
  cofix CIH; intros P M x Px.
  destruct (M _ Px) as (y & C & Py); clear Px.
  induction C; eauto.
Qed.

-\---  x.v            56% (52,0)      (Coq Script(0-) Holes)
Error:
Recursive definition of CIH is ill-formed.
```

Successful Proof using Semantically Guarded Coinduction

```
Variable Node: Type.
Variable R: relation Node.

Inductive step (X: Node -> Prop) (x: Node) : Prop :=
| step_intro: forall y, R x y -> X y -> step X x.
Hint Constructors step.

Definition infpath := paco1 step bot1.
Lemma step_mon : monotone1 step. Proof. pmonauto. Qed.
Hint Resolve step_mon : paco.

Lemma Park's_principle:
  forall (P: Node -> Prop),
  (forall x, P x -> exists y, clos_trans_1n _ R x y /\ P y) ->
  forall x, P x -> infpath x.
Proof.
  pcofix CIH. intros P M x Px.
  destruct (M _ Px) as (y & C & Py); clear Px.
  induction C; pfold; eauto.
Qed.

-\---  y.v           50% (56,0)      (Coq Script(0-) Holes)
Park's_principle is defined
```

Successful Proof using Semantically Guarded Coinduction

```
Variable Node: Type.
Variable R: relation Node.

Inductive step (X: Node -> Prop) (x: Node) : Prop :=
| step_intro: forall y, R x y -> X y -> step X x.
Hint Constructors step.

Definition infpath := paco1 step bot1.
Lemma step_mon : monotone1 step. Proof. pmonauto. Qed.
Hint Resolve step_mon : paco.

Lemma Park's_principle:
  forall (P: Node -> Prop),
  (forall x, P x -> exists y, clos_trans_1n _ R x y /\ P y) ->
  forall x, P x -> infpath x.
Proof.
  pcofix CIH. intros P M x Px.
  destruct (M _ Px) as (y & C & Py); clear Px.
  induction C; pfold; eauto.
Qed.

-\---  y.v          50% (56,0)       (Coq Script(0-) Holes)
Park's_principle is defined
```

Successful Proof using Semantically Guarded Coinduction

Possible because of

NO Syntactic Guardedness Checking

```
Definition infpath := paco1 step bot1.
Lemma step_mon : monotone1 step. Proof. pmonauto. Qed.
Hint Resolve step_mon : paco.

Lemma Park's_principle:
  forall (P: Node -> Prop),
  (forall x, P x -> exists y, clos_trans_1n _ R x y /\ P y) ->
  forall x, P x -> infpath x.
Proof.
  pcofix CIH. intros P M x Px.
  destruct (M _ Px) as (y & C & Py); clear Px.
  induction C; pfold; eauto.
Qed.

-\---  y.v           50% (56,0)      (Coq Script(0-) Holes)
Park's_principle is defined
```

Syntactic Guardedness is NOT Compositional

$$X \subseteq \nu f$$

Syntactic Guardedness is NOT Compositional

$$\frac{\begin{array}{c} X \subseteq \nu f \Rightarrow Y \subseteq \nu f \\ Y \subseteq \nu f \Rightarrow X \subseteq \nu f \end{array}}{X \subseteq \nu f}$$

Syntactic Guardedness is NOT Compositional

pf$_1$: $X \subseteq \nu f \Rightarrow Y \subseteq \nu f$ (pf$_1$ is guarded)

pf$_2$: $Y \subseteq \nu f \Rightarrow X \subseteq \nu f$ (pf$_2$ is guarded)

$$\frac{\qquad\qquad\qquad\qquad}{X \subseteq \nu f}$$

Not Expressible in the logic

$$\mathrm{pf}_1 : X \subseteq \nu f \Rightarrow Y \subseteq \nu f \text{ (pf}_1 \text{ is guarded)}$$

$$\mathrm{pf}_2 : Y \subseteq \nu f \Rightarrow X \subseteq \nu f \text{ (pf}_2 \text{ is guarded)}$$

$$\overline{\qquad\qquad\qquad\qquad}$$

$$X \subseteq \nu f$$

Make Proofs Transparent

pf_1: $X \subseteq \nu f \Rightarrow Y \subseteq \nu f$

pf_2: $Y \subseteq \nu f \Rightarrow X \subseteq \nu f$

$$\frac{}{X \subseteq \nu f}$$

Make Proofs Transparent

$$\frac{\dfrac{\mathrm{pf}_1 : X \subseteq \nu f \Rightarrow Y \subseteq \nu f \qquad \mathrm{pf}_2 : Y \subseteq \nu f \Rightarrow X \subseteq \nu f}{\mathrm{pf}_2 \circ \mathrm{pf}_1 : X \subseteq \nu f \Rightarrow X \subseteq \nu f}}{X \subseteq \nu f}$$

Make Proofs Transparent

$$\mathrm{pf}_1 : X \subseteq \nu f \Rightarrow Y \subseteq \nu f$$

$$\mathrm{pf}_2 : Y \subseteq \nu f \Rightarrow X \subseteq \nu f$$

$$\frac{\mathrm{pf}_2 \circ \mathrm{pf}_1 : X \subseteq \nu f \Rightarrow X \subseteq \nu f}{X \subseteq \nu f}$$

($\mathrm{pf}_2 \circ \mathrm{pf}_1$ is guarded)

Two Problems with Transparent Proofs

1. Slowdown
2. NO Abstraction

$$\mathrm{pf}_1 : X \subseteq \nu f \Rightarrow Y \subseteq \nu f$$

$$\mathrm{pf}_2 : Y \subseteq \nu f \Rightarrow X \subseteq \nu f$$

$$\frac{\dfrac{}{\mathrm{pf}_2 \circ \mathrm{pf}_1 : X \subseteq \nu f \Rightarrow X \subseteq \nu f}}{X \subseteq \nu f}$$

($\mathrm{pf}_2 \circ \mathrm{pf}_1$ is guarded)

Semantic Guardedness is Compositional

$$\frac{Y \subseteq \mathrm{G}_f(X) \qquad X \subseteq \mathrm{G}_f(Y)}{X \subseteq \mathrm{G}_f(\emptyset)}$$

Possible because Guardedness can be expressed in the Logic!

$$\frac{Y \subseteq \mathrm{G}_f(X) \qquad X \subseteq \mathrm{G}_f(Y)}{X \subseteq \mathrm{G}_f(\emptyset)}$$

Problems with Syntactic Guardedness

1. Non-Compositional
2. Far from complete
3. Bad interaction with automation
4. Hard to debug
5. Slow

~~Problems with~~ Semantic Guardedness

1. Non-Compositional
2. Far from complete
3. Bad interaction with automation
4. Hard to debug
5. Slow

What else is in the paper?

- Related Work
  - Glynn Winskel (1989)
  - Lawrence Moss (2001)
- Combination with Up-To Techniques
- Mechanization in Coq
  – Using Mendler-style recursion