Gradual Verification

(with Implicit Dynamic Frames)

Johannes Bader, KIT, Karlsruhe, Germany

Prof. Jonathan Aldrich, CMU, Pittsburgh, USA

Prof. Éric Tanter, University of Chile, Santiago, Chile

Prof. Gregor Snelting, KIT, Karlsruhe, Germany

```
int getFour(int i)
    requires ?; // haven't figured that one out, yet
    ensures  result = 4;
{
    i = i + 1;
    return i;
}
```

Motivation

- Program verification (against some specification)
- Two flavors: dynamic & static

```
// spec: callable only if (this.balance >= amount)
void withdrawCoins(int amount)
{
   // business logic
   this.balance -= amount;
}
```

Dynamic Verification

- runtime checks
- testing techniques
- guarantee compliance **at runtime**

```
// spec: callable only if (this.balance >= amount)
void withdrawCoins(int amount)
{
   // business logic
   this.balance -= amount;
}
```

Dynamic Verification

- runtime checks
- testing techniques
- guarantee compliance **at runtime**

```
void withdrawCoins(int amount)
{
   assert this.balance >= amount;
   // business logic
   this.balance -= amount;
}
```

Dynamic Verification – Drawbacks

• runtime checks	runtime overhead
• testing techniques	additional efforts
• guarantee compliance **at runtime**	pot. late detection

```
void withdrawCoins(int amount)
{
    assert this.balance >= amount;
    // business logic
    this.balance -= amount;
}
```

Static Verification

- declarative
- formal logic
- guarantee compliance **in advance**

```
void withdrawCoins(int amount)
    requires this.balance >= amount;
{
    // business logic
    this.balance -= amount;
}
```

Static Verification – Drawbacks

- declarative
- formal logic
- guarantee compliance **in advance**

limited expressiveness

decidability

annotation overhead

```
void withdrawCoins(int amount)
    requires this.balance >= amount;
{
    // business logic
    this.balance -= amount;
}
```

Static Verification – Drawbacks

- declarative
- formal logic
- guarantee compliance **in advance**

limited expressiveness

decidability

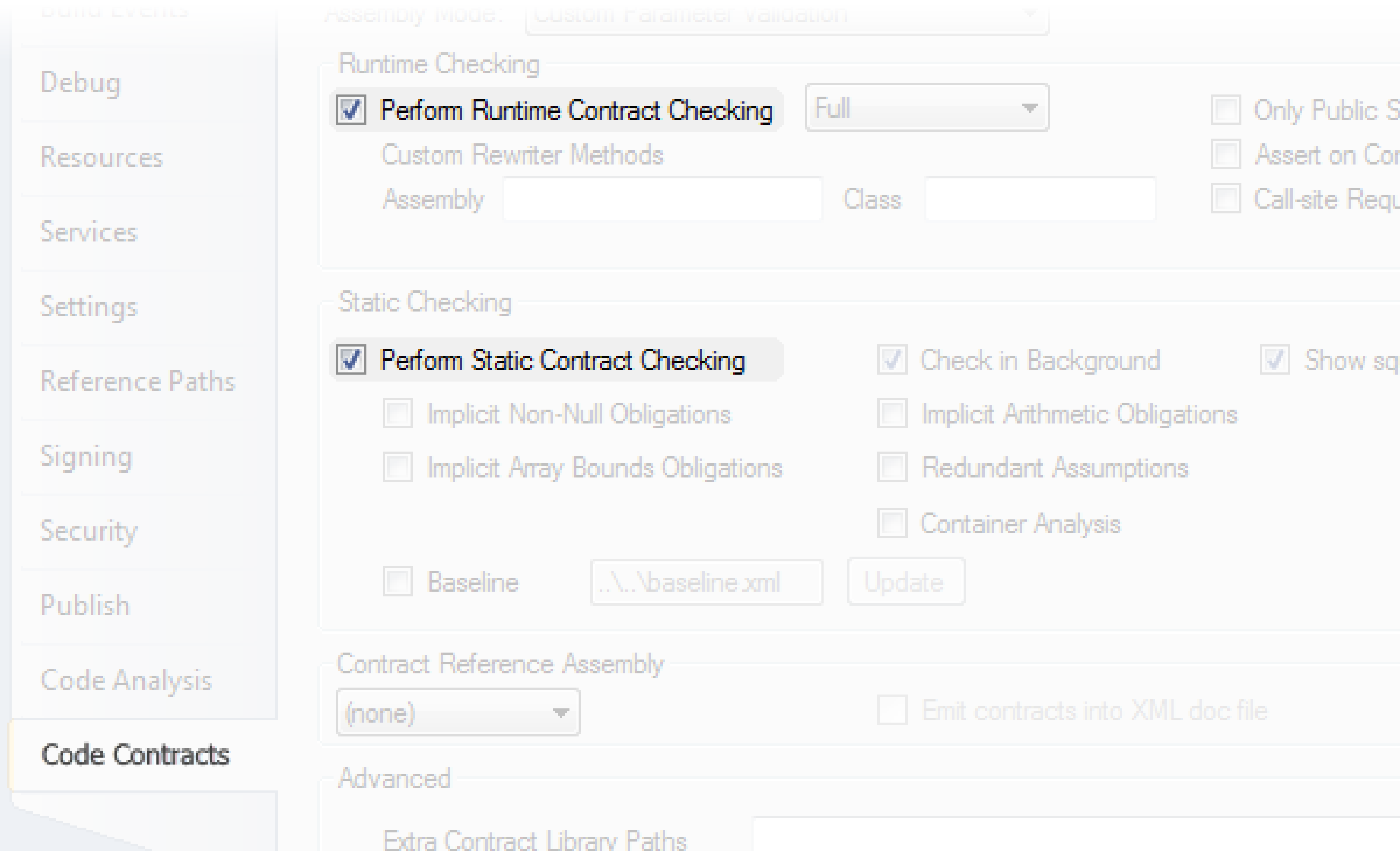
annotation overhead

```
void withdrawCoins(int amount)
   requires this.balance >= amount;
   ensures  this.balance == old(this.balance) – amount;
{
   // business logic
   this.balance -= amount;
}
```

Solution? Static + Dynamic

- “relaxed” static verification (warnings on failure)
- turn contracts into runtime assertions (“patch”)

Solution? Static + Dynamic



Assembly Mode: Custom Parameter Validation

Runtime Checking

☑ Perform Runtime Contract Checking   Full   □ Only Public S

Custom Rewriter Methods   □ Assert on Co

Assembly   Class   □ Call-site Requ

Static Checking

☑ Perform Static Contract Checking   ☑ Check in Background   ☑ Show sq

□ Implicit Non-Null Obligations   □ Implicit Arithmetic Obligations

□ Implicit Array Bounds Obligations   □ Redundant Assumptions

□ Container Analysis

□ Baseline   ..\..\baseline.xml   Update

Contract Reference Assembly

(none)   □ Emit contracts into XML doc file

Advanced

Extra Contract Library Paths

Solution! Static ⊕ Dynamic

“Static Checking Where Possible,
Dynamic Checking When Needed”

Solution! Static ⊕ Dynamic

“Static Checking Where Possible,
Dynamic Checking When Needed”

```
void withdrawCoins(int amount)
    requires this.balance >= amount;
{
    …
}
…
acc.balance = 100;
acc.withdrawCoins(50); // can prove acc.balance >= 50
acc.withdrawCoins(30); // can’t prove acc.balance >= 30
acc.withdrawCoins(30); // can’t prove acc.balance >= 30
```

Solution! Static ⊕ Dynamic

“Static Checking Where Possible,
Dynamic Checking When Needed”

```
void withdrawCoins(int amount)
    requires this.balance >= amount;
{
    …
}
…
acc.balance = 100;
acc.withdrawCoins(50); // statically guaranteed
acc.withdrawCoins(30); // dynamically guaranteed
acc.withdrawCoins(30); // dynamically guaranteed ↯
```

Solution! Static ⊕ Dynamic

“Static Typing Where Possible,
Dynamic Typing When Needed” (Erik Meijer)

```
void withdrawCoins(int amount)
   requires this.balance >= amount;
{
   …
}
…
acc.balance = 100;
acc.withdrawCoins(50); // statically guaranteed
acc.withdrawCoins(30); // dynamically guaranteed
acc.withdrawCoins(30); // dynamically guaranteed ↯
```

Solution! Static ⊕ Dynamic

“Static Typing Where Possible,
Dynamic Typing When Needed” (Erik Meijer)

= Gradual Typing

Solution! Static ⊕ Dynamic

“Static Typing Where Possible,
Dynamic Typing When Needed” (Erik Meijer)

= Gradual Typing

Solution! Static ⊕ Dynamic

“Static Typing Where Possible,
Dynamic Typing When Needed” (Erik Meijer)

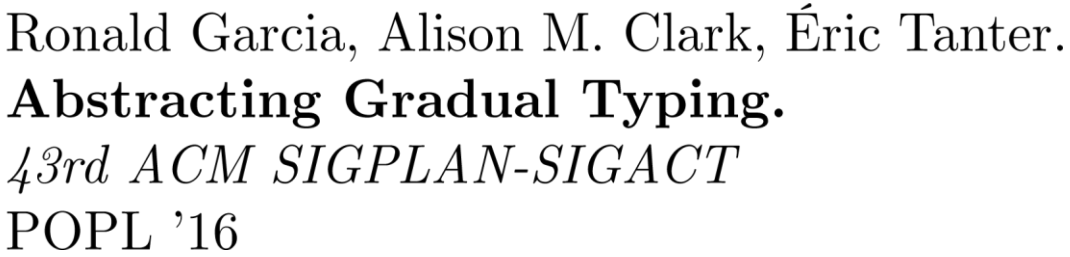
= Gradual Typing

Solution! Static ⊕ Dynamic

"Static Typing Where Possible,
Dynamic Typing When Needed" (Erik Meijer)

= Gradual Typing

- draw on recent advances in gradual typing



Ronald Garcia, Alison M. Clark, Éric Tanter.
Abstracting Gradual Typing.
43rd ACM SIGPLAN-SIGACT
POPL '16

- adapt methodology to verification setting

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

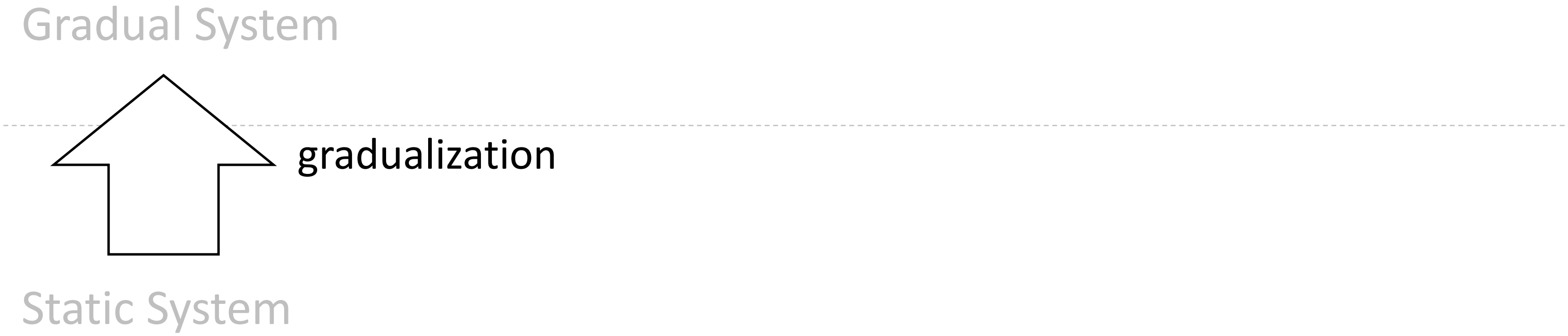
Gradual System

Static System

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

- define gradual (type) system in terms of a pre-existing static one = “**gradualization**”



Gradual System

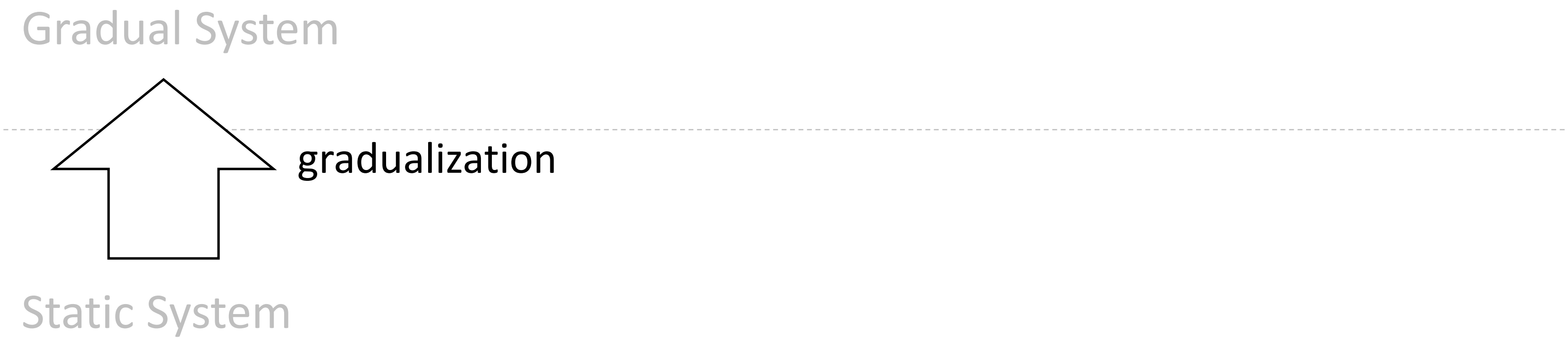
gradualization

Static System

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

- define gradual (type) system in terms of a pre-existing static one = "**gradualization**"
- parts of a semantics affected by gradualization are expressible as **predicates/functions operating on types**



Gradual System

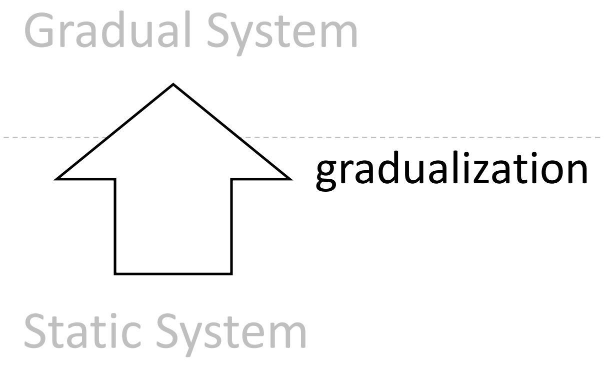
gradualization

Static System

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

- define gradual (type) system in terms of a pre-existing static one = "**gradualization**"
- parts of a semantics affected by gradualization are expressible as **predicates/functions operating on types**



Gradual System

gradualization

Static System

$$\begin{tikzcd}\tau_1 \arrow[rr, bend left, "f"'] & & \tau_2\end{tikzcd}$$

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

- define gradual (type) system in terms of a pre-existing static one = "**gradualization**"
- parts of a semantics affected by gradualization are expressible as **predicates/functions operating on types**

Gradual System

gradualization

Static System

$$\tau_1 \xrightarrow{f} \tau_2$$

$$f(\tau) = \tau \sqcap \texttt{int}$$

`typeof(if <bool> then <τ> else 42)`

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\begin{tikzcd} \widetilde{\tau_1} \arrow[r, bend left, "\widetilde{f}"'] & \widetilde{\tau_2} \end{tikzcd}$$

Gradual System

$\widetilde{\tau} \; ::= \; \tau \mid ?$

Static System

$$\begin{tikzcd} \tau_1 \arrow[r, bend left, "f"'] & \tau_2 \end{tikzcd}$$

$f(\tau) = \tau \sqcap \texttt{int}$

$\mathrm{typeof}(\texttt{if <bool> then <}\tau\texttt{> else 42})$

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\begin{tikzcd}\widetilde{\tau}_1 \arrow[r, bend left, "\widetilde{f}"'] & \widetilde{\tau}_2\end{tikzcd}$$

Gradual System

$$\widetilde{\tau} \;::=\; \tau \mid ?$$

Static System

$$\begin{tikzcd}\tau_1 \arrow[r, bend left, "f"'] & \tau_2\end{tikzcd}$$

$$f(\tau) = \tau \sqcap \texttt{int}$$

$$\text{typeof}(\texttt{if <bool> then <}\tau\texttt{> else 42})$$

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

Gradual System

$\widetilde{\tau} \ ::= \ \tau \mid ?$

$$\begin{tikzcd}
\widetilde{\tau}_1 \arrow[r, bend left, "\widetilde{f}"'] & \widetilde{\tau}_2
\end{tikzcd}$$

Static System

$$\begin{tikzcd}
\tau_1 \arrow[r, bend left, "f"'] & \tau_2
\end{tikzcd}$$

$f(\tau) = \tau \sqcap \texttt{int}$

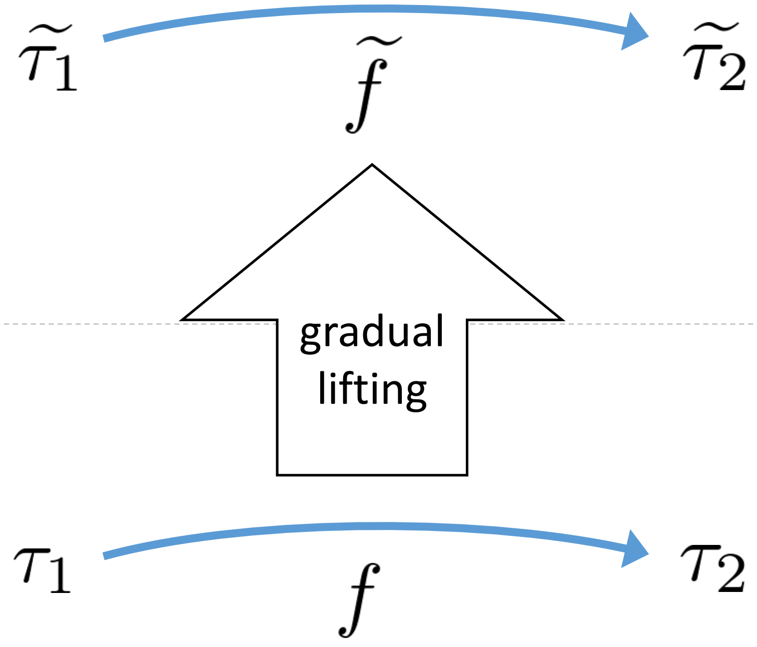
typeof(if <bool> then <τ> else 42)

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

Gradual System

$\widetilde{\tau} \ ::= \ \tau \mid ?$



Static System

$f(\tau) = \tau \sqcap \texttt{int}$

$\text{typeof}(\texttt{if <bool> then <}\tau\texttt{> else 42})$

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

Gradual System

$$\begin{aligned} \gamma(\tau) &= \{ \tau \} \\ \gamma(?) &= \textsc{Types} \end{aligned}$$

Static System

$$\begin{tikzcd}
\widetilde{\tau_1} \arrow[rr, bend left, "\widetilde{f}"] \arrow[d, dotted, "\gamma"'] & & \widetilde{\tau_2} \\
\overline{\tau_1} \arrow[rr, dotted, bend left, "\overline{f}"'] \arrow[urr, phantom, "\text{gradual lifting}"] & & \overline{\tau_2} \arrow[u, dotted, "\alpha"]
\end{tikzcd}$$

$$f(\tau) = \tau \sqcap \texttt{int}$$

$\texttt{typeof}(\texttt{if <bool> then <}\tau\texttt{> else 42})$

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

Gradual System

$$\gamma(\tau) = \{\ \tau\ \}$$

$$\gamma(?) = \textsc{Types}$$

Static System

$$\begin{tikzcd}
\widetilde{\tau_1} \arrow[rr, bend left, "\widetilde{f}"] \arrow[d, dotted, "\gamma"'] & & \widetilde{\tau_2} \\
\overline{\tau_1} \arrow[rr, dotted, bend left, "\overline{f}"'] \arrow[rru, phantom, "\Uparrow\ \text{gradual lifting}"] & & \overline{\tau_2} \arrow[u, dotted, "\alpha"]
\end{tikzcd}$$

$$f(\tau) = \tau \sqcap \texttt{int}$$

$$\texttt{typeof}(\texttt{if <bool> then <}\tau\texttt{> else 42})$$

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

Gradual System

$$\begin{aligned} \gamma(\tau) &= \{\ \tau\ \} \\ \gamma(?) &= \textsc{Types} \end{aligned}$$

Static System

$$\begin{tikzcd}[column sep=huge, row sep=huge]
\widetilde{\tau_1} \arrow[rr, bend left, "\widetilde{f}"] \arrow[d, dashed, "\gamma"'] & & \widetilde{\tau_2} \\
\overline{\tau_1} \arrow[rr, dashed, bend left, "\overline{f}"'] \arrow[r, phantom, ""] & \arrow[u, phantom, "\text{gradual lifting}"] & \overline{\tau_2} \arrow[u, dashed, "\alpha"']
\end{tikzcd}$$

$f(\tau) = \tau \sqcap \texttt{int}$

$\texttt{typeof}(\texttt{if <bool> then <}\tau\texttt{> else 42})$

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

Gradual System

$$\gamma(\tau) = \{\ \tau\ \}$$
$$\gamma(?) = \textsc{Types}$$

Static System

$$\begin{tikzcd}
\widetilde{\tau_1} \arrow[r, bend left, "\widetilde{f}"] \arrow[d, dotted, "\gamma"'] & \widetilde{\tau_2} \arrow[dr, dotted, "\gamma"] & \\
\overline{\tau_1} \arrow[r, dotted, bend left, "\overline{f}"'] & \overline{\tau_2} \arrow[u, dotted, "\alpha"] \arrow[r, phantom, "\subseteq"] & \overline{\tau_2}'
\end{tikzcd}$$

$$f(\tau) = \tau \sqcap \texttt{int}$$

$$\text{typeof}(\texttt{if <bool> then <}\tau\texttt{> else 42})$$

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

Gradual System

$$\begin{aligned} \gamma(\tau) &= \{\ \tau\ \} \\ \gamma(?) &= \textsc{Types} \end{aligned}$$

Static System

$$\begin{tikzcd}
\widetilde{\tau_1} \arrow[rr, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & & \widetilde{\tau_2} \arrow[rrd, dotted, "\gamma"] & & \\
\overline{\tau_1} \arrow[rr, dotted, bend left, "\overline{f}"'] & & \overline{\tau_2} \arrow[u, dotted, "\alpha"] & \subseteq & \overline{\tau_2}'
\end{tikzcd}$$

$f(\tau) = \tau \sqcap \mathtt{int}$

typeof(`if <bool> then <`τ`> else 42`)

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

Gradual System

$$\begin{aligned} \gamma(\tau) &= \{\ \tau\ \} \\ \gamma(?) &= \textsc{Types} \end{aligned}$$

Static System

$$\begin{tikzcd}
\widetilde{\tau_1} \arrow[r, bend left, "\widetilde{f}"] \arrow[d, dotted, "\gamma"'] & \widetilde{\tau_2} \arrow[dr, dotted, "\gamma"] & \\
\overline{\tau_1} \arrow[r, dotted, bend left, "\overline{f}"'] & \overline{\tau_2} \arrow[u, dotted, "\alpha"] \arrow[r, phantom, "\subseteq"] & \overline{\tau_2}'
\end{tikzcd}$$

$$f(\tau) = \tau \sqcap \texttt{int}$$

typeof(`if <bool> then <`τ`> else 42`)

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \texttt{int}$$
$$\widetilde{f}(?) = ?$$

Gradual System

$$\begin{aligned} \gamma(\tau) &= \{\ \tau\ \} \\ \gamma(?) &= \textsc{Types} \end{aligned}$$

$$\begin{tikzcd}
\widetilde{\tau_1} \arrow[r, bend left, "\widetilde{f}"] \arrow[d, dotted, "\gamma"'] & \widetilde{\tau_2} \arrow[dr, dotted, "\gamma"] & \\
\overline{\tau_1} \arrow[r, dotted, bend left, "\overline{f}"'] & \overline{\tau_2} \arrow[r, phantom, "\subseteq"] & \overline{\tau_2}'
\end{tikzcd}$$

Static System

$$f(\tau) = \tau \sqcap \texttt{int}$$

`typeof(if <bool> then <`τ`> else 42)`

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \texttt{int}$$
$$\widetilde{f}(?) = ?$$

Gradual System

$$\gamma(\tau) \;= \{\, \tau \,\}$$
$$\gamma(?) \;= \textsc{Types}$$

$$\begin{tikzcd}
{} \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dashed, "\gamma"'] & {} \arrow[d, dashed, "\gamma"] \\
{} \arrow[r, bend left, dashed, "\overline{f}"'] & {}
\end{tikzcd}$$

Static System

$$f(\tau) = \tau \sqcap \texttt{int}$$

typeof$($`if <bool> then <`τ`> else 42`$)$

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \texttt{int}$$
$$\widetilde{f}(?) = ?$$

$$\begin{tikzcd}
{} \arrow[rr, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & & {} \arrow[d, dotted, "\gamma"] \\
{} \arrow[rr, dotted, bend left, "\overline{f}"'] & & {}
\end{tikzcd}$$

Gradual System

$$\gamma(\tau) \;= \{\ \tau\ \}$$
$$\gamma(?) \;= \textsc{Types}$$

Static System

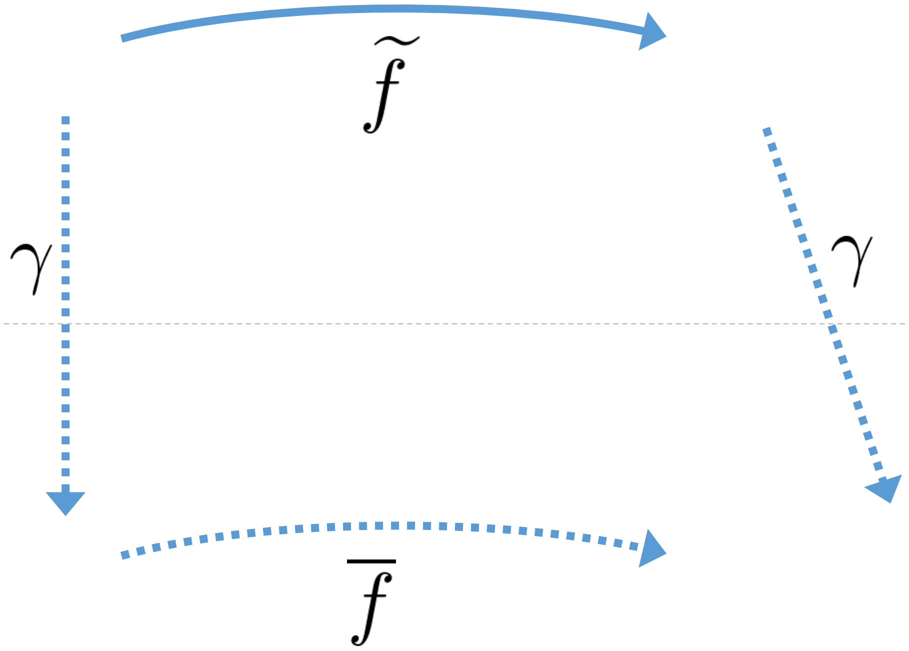
$$f(\tau) = \tau \sqcap \texttt{int}$$

$\mathrm{typeof}(\texttt{if <bool> then <}\tau\texttt{> else 42})$

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \texttt{int}$$
$$\widetilde{f}(?) = ?$$



$\widetilde{f}$

γ

γ

$\overline{f}$

Gradual System

$$\gamma(\tau) \;= \{\ \tau\ \}$$
$$\gamma(?) \;= \textsc{Types}$$

Static System

$$f(\tau) = \tau \sqcap \texttt{int}$$

typeof(`if <bool> then <`τ`> else 42`)

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \texttt{int}$$
$$\widetilde{f}(?) = ?$$

$$\begin{tikzcd}
\tau \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dashed, "\gamma"'] & {} \arrow[d, dashed, "\gamma"] \\
{} \arrow[r, bend left, dashed, "\overline{f}"'] & {}
\end{tikzcd}$$

Gradual System

$$\gamma(\tau) = \{\ \tau\ \}$$
$$\gamma(?) = \textsc{Types}$$

Static System

$$f(\tau) = \tau \sqcap \texttt{int}$$

typeof(if <bool> then <τ> else 42)

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \texttt{int}$$
$$\widetilde{f}(?) = ?$$

Gradual System

$$\gamma(\tau) \;\; = \{\ \tau\ \}$$
$$\gamma(?) \;\; = \textsc{Types}$$

$$\begin{tikzcd}
\tau \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & \tau \sqcap \texttt{int} \arrow[d, dotted, "\gamma"] \\
{} \arrow[r, dotted, bend left, "\overline{f}"'] & {}
\end{tikzcd}$$

Static System

$$f(\tau) = \tau \sqcap \texttt{int}$$

```
typeof(if <bool> then <τ> else 42)
```

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \texttt{int}$$
$$\widetilde{f}(?) = ?$$

$$\begin{tikzcd}
\tau \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dashed, "\gamma"'] & \tau \sqcap \texttt{int} \arrow[d, dashed, "\gamma"] \\
\{\tau\} \arrow[r, dashed, bend left, "\overline{f}"'] & {}
\end{tikzcd}$$

Gradual System

$$\gamma(\tau) \;= \{\ \tau\ \}$$
$$\gamma(?) \;= \textsc{Types}$$

Static System

$$f(\tau) = \tau \sqcap \texttt{int}$$

typeof(`if <bool> then <`τ`> else 42`)

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \mathtt{int}$$

$$\widetilde{f}(?) = ?$$

Gradual System

$$\gamma(\tau) \ = \{\ \tau\ \}$$

$$\gamma(?) \ = \textsc{Types}$$

$$\begin{tikzcd}
\tau \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & \tau \sqcap \mathtt{int} \arrow[d, dotted, "\gamma"] \\
\{\tau\} \arrow[r, dotted, bend left, "\overline{f}"'] & \{\tau \sqcap \mathtt{int}\} \subseteq \{\tau \sqcap \mathtt{int}\}
\end{tikzcd}$$

Static System

$$f(\tau) = \tau \sqcap \mathtt{int}$$

typeof(`if <bool> then <`τ`> else 42`)

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \texttt{int}$$

$$\widetilde{f}(?) = ?$$

Gradual System

$$\begin{aligned} \gamma(\tau) &= \{\ \tau\ \} \\ \gamma(?) &= \textsc{Types} \end{aligned}$$

$$\begin{tikzcd} {} \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & {} \arrow[d, dotted, "\gamma"] \\ {} \arrow[r, dotted, bend left, "\overline{f}"'] & {} \end{tikzcd}$$

Static System

$$f(\tau) = \tau \sqcap \texttt{int}$$

typeof(`if <bool> then <`τ`> else 42`)

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \texttt{int}$$
$$\widetilde{f}(?) = ?$$

Gradual System

$$\gamma(\tau) \; = \{\ \tau\ \}$$
$$\gamma(?) \; = \textsc{Types}$$

$$\begin{tikzcd}
{} \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & {} \arrow[d, dotted, "\gamma"] \\
{} \arrow[r, dotted, bend left, "\overline{f}"'] & {}
\end{tikzcd}$$

Static System

$$f(\tau) = \tau \sqcap \texttt{int}$$

$\text{typeof}(\texttt{if <bool> then <}\tau\texttt{> else 42})$

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \texttt{int}$$
$$\widetilde{f}(?) = ?$$

Gradual System

$$\gamma(\tau) \ = \{\ \tau\ \}$$
$$\gamma(?) \ = \textsc{Types}$$

$$\begin{tikzcd}
{} \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & {} \arrow[d, dotted, "\gamma"] \\
{} \arrow[r, dotted, bend left, "\overline{f}"'] & {}
\end{tikzcd}$$

Static System

$$f(\tau) = \tau \sqcap \texttt{int}$$

typeof(`if <bool> then <`τ`> else 42`)

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \texttt{int}$$
$$\widetilde{f}(?) = ?$$

Gradual System

$$\begin{aligned} \gamma(\tau) &= \{ \tau \} \\ \gamma(?) &= \textsc{Types} \end{aligned}$$

$$\begin{tikzcd}
? \arrow[r, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & {} \arrow[d, dotted, "\gamma"] \\
{} \arrow[r, dotted, "\overline{f}"'] & {}
\end{tikzcd}$$

Static System

$$f(\tau) = \tau \sqcap \texttt{int}$$

$\text{typeof}(\texttt{if <bool> then <}\tau\texttt{> else 42})$

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \texttt{int}$$

$$\widetilde{f}(?) = ?$$

$$\begin{tikzcd}
? \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & ? \arrow[d, dotted, "\gamma"] \\
{} \arrow[r, dotted, bend left, "\overline{f}"'] & {}
\end{tikzcd}$$

Gradual System

$$\gamma(\tau) = \{\ \tau\ \}$$

$$\gamma(?) = \textsc{Types}$$

Static System

$$f(\tau) = \tau \sqcap \texttt{int}$$

typeof(if <bool> then <τ> else 42)

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \texttt{int}$$
$$\widetilde{f}(?) = ?$$

Gradual System

$$\begin{aligned} \gamma(\tau) &= \{\ \tau\ \} \\ \gamma(?) &= \textsc{Types} \end{aligned}$$

$$\begin{tikzcd}
? \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & ? \arrow[d, dotted, "\gamma"] \\
\textsc{Types} \arrow[r, dotted, bend left, "\overline{f}"'] & {}
\end{tikzcd}$$

Static System

$$f(\tau) = \tau \sqcap \texttt{int}$$

$\text{typeof}(\texttt{if <bool> then <}\tau\texttt{> else 42})$

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

$$\widetilde{f}(\tau) = \tau \sqcap \texttt{int}$$
$$\widetilde{f}(?) = ?$$

Gradual System

$$\gamma(\tau) \;= \{\ \tau\ \}$$
$$\gamma(?) \;= \textsc{Types}$$

$$\begin{tikzcd}
? \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & ? \arrow[d, dotted, "\gamma"] \\
\textsc{Types} \arrow[r, dotted, bend left, "\overline{f}"'] & \ldots \subseteq \textsc{Types}
\end{tikzcd}$$

Static System

$$f(\tau) = \tau \sqcap \texttt{int}$$

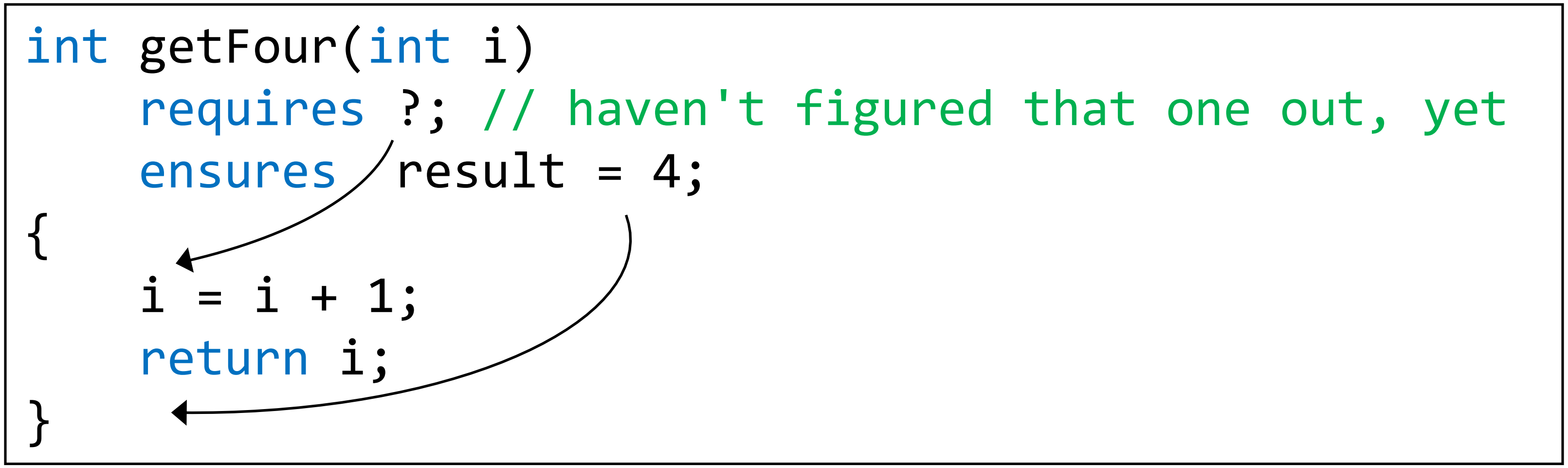
typeof(if <bool> then <τ> else 42)

How does this relate to Verification?

```
int getFour(int i)
    requires ?; // haven't figured that one out, yet
    ensures  result = 4;
{
    i = i + 1;
    return i;
}
```

Types restrict, which **values** are valid for a certain variable

How does this relate to Verification?



```
int getFour(int i)
    requires ?; // haven't figured that one out, yet
    ensures result = 4;
{
    i = i + 1;
    return i;
}
```

Types restrict, which **values** are valid for a certain variable

Formulas restrict, which **program states** are valid for a certain point during execution

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

Gradual System

$\widetilde{\tau} \; ::= \; \tau \mid ?$

$$\begin{tikzcd}
\widetilde{\tau_1} \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dashed, "\gamma"'] & \widetilde{\tau_2} \arrow[dr, dashed, "\gamma"] & \\
\overline{\tau_1} \arrow[r, dashed, bend left, "\overline{f}"'] & \overline{\tau_2} \arrow[r, phantom, "\subseteq"] & \overline{\tau_2}'
\end{tikzcd}$$

Static System

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

Gradual System

$\widetilde{\tau} \ ::= \ \tau \mid ?$

$$\begin{tikzcd}
\widetilde{\tau_1} \arrow[r, bend left, "\widetilde{f}"] \arrow[d, dotted, "\gamma"'] & \widetilde{\tau_2} \arrow[d, dotted, "\gamma"] \\
\overline{\tau_1} \arrow[r, dotted, bend left, "\overline{f}"'] & \overline{\tau_2} \subseteq \overline{\tau_2}'
\end{tikzcd}$$

Static System

Abstracting Gradual Typing

Ronald Garcia, Alison M. Clark, and Éric Tanter

Gradual System

$\widetilde{\tau} \ ::= \ \tau \mid ?$

Static System

$$\begin{tikzcd}
\widetilde{\tau_1} \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & \widetilde{\tau_2} \arrow[dr, dotted, "\gamma"] & \\
\overline{\tau_1} \arrow[r, dotted, bend left, "\overline{f}"'] & \overline{\tau_2} \arrow[r, phantom, "\subseteq"] & \overline{\tau_2}'
\end{tikzcd}$$

Abstracting Gradual ~~Typing~~ Verification

Ronald Garcia, Alison M. Clark, and Éric Tanter

Gradual System

$$\widetilde{\phi} \;::=\; \phi \mid ?$$

Static System

$$\begin{tikzcd}
\widetilde{\phi_1} \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & \widetilde{\phi_2} \arrow[d, dotted, "\gamma"] \\
\overline{\phi_1} \arrow[r, dotted, bend left, "\overline{f}"'] & \overline{\phi_2} \subseteq \overline{\phi_2}'
\end{tikzcd}$$

Abstracting Gradual ~~Typing~~ Verification

Ronald Garcia, Alison M. Clark, and Éric Tanter

Gradual System

$$\widetilde{\phi} \ ::= \ \phi \mid ?$$

Static System

$$\begin{tikzcd}
\widetilde{\phi_1} \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & \widetilde{\phi_2} \arrow[d, dotted, "\gamma"] \\
\overline{\phi_1} \arrow[r, dotted, bend left, "\overline{f}"'] & \overline{\phi_2} \subseteq \overline{\phi_2}'
\end{tikzcd}$$

Abstracting Gradual ~~Typing~~ Verification

Ronald Garcia, Alison M. Clark, and Éric Tanter

Gradual System

$\widetilde{\phi} \ ::= \ \phi \mid ?$

Static System

$$\begin{tikzcd}
\widetilde{\phi_1} \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & \widetilde{\phi_2} \arrow[dr, dotted, "\gamma"] & \\
\overline{\phi_1} \arrow[r, dotted, bend left, "\overline{f}"'] & \overline{\phi_2} \arrow[r, phantom, "\subseteq"] & \overline{\phi_2}'
\end{tikzcd}$$

Gradualization – Overview

Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

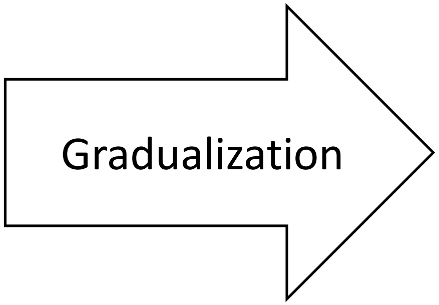
Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \models \phi$

Soundness



Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Semantics

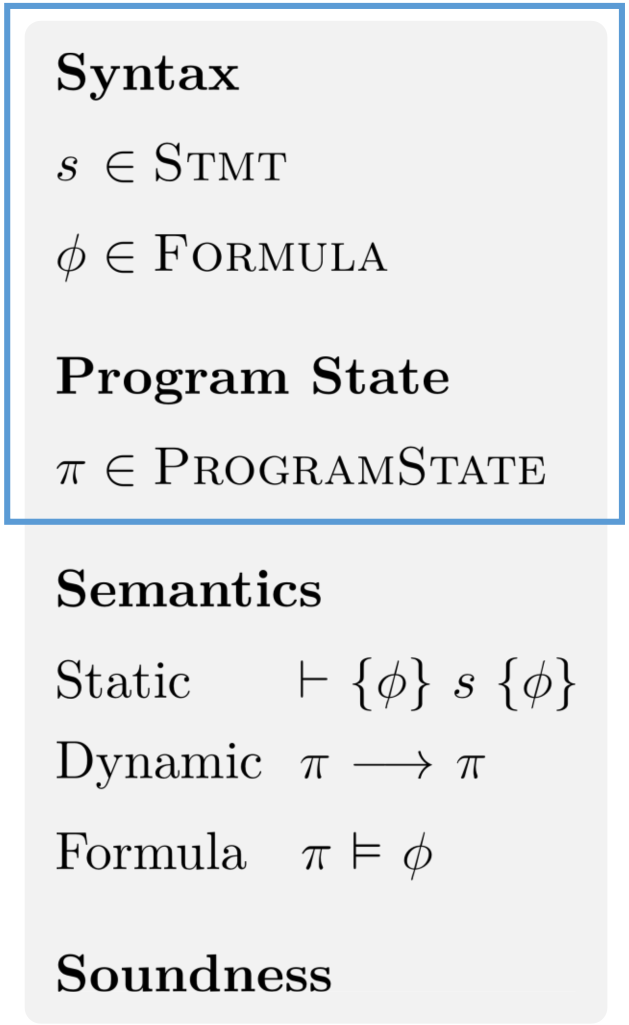
Static $\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$

Dynamic $\widetilde{\pi} \mathrel{\widetilde{\longrightarrow}} \widetilde{\pi}$

Formula $\widetilde{\pi} \mathrel{\widetilde{\models}} \widetilde{\phi}$

Soundness

Gradualization – Starting Point



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\quad \vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\quad \pi \longrightarrow \pi$

Formula $\quad \pi \vDash \phi$

Soundness

$$\begin{array}{lcl} s & ::= & \texttt{skip} \mid x \texttt{ := } e \mid \texttt{assert}\ \phi \mid s_1 \texttt{;}\ s_2 \\ \phi & ::= & \texttt{true} \mid (e_1 \texttt{ = } e_2) \mid \phi_1 \wedge \phi_2 \end{array}$$

Gradualization – Starting Point

Syntax

$s \in \textsc{Stmt}$ $\quad s \;::=\; \texttt{skip} \mid x \texttt{ := } e \mid \texttt{assert } \phi \mid s_1 \texttt{; } s_2$

$\phi \in \textsc{Formula}$ $\quad \phi \;::=\; \texttt{true} \mid (e_1 \texttt{ = } e_2) \mid \phi_1 \wedge \phi_2$

Program State

$\pi \in \textsc{ProgramState}$ $\quad = (\textsc{Var} \rightharpoonup \mathbb{N}_0) \times \textsc{Stmt}$

$\langle [\mathrm{x} \mapsto 6, \mathrm{y} \mapsto 3], \texttt{x := y; assert (x = 3)} \rangle$

Semantics

Static $\quad \vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\quad \pi \longrightarrow \pi$

Formula $\quad \pi \vDash \phi$

Soundness

Gradualization – Starting Point

Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

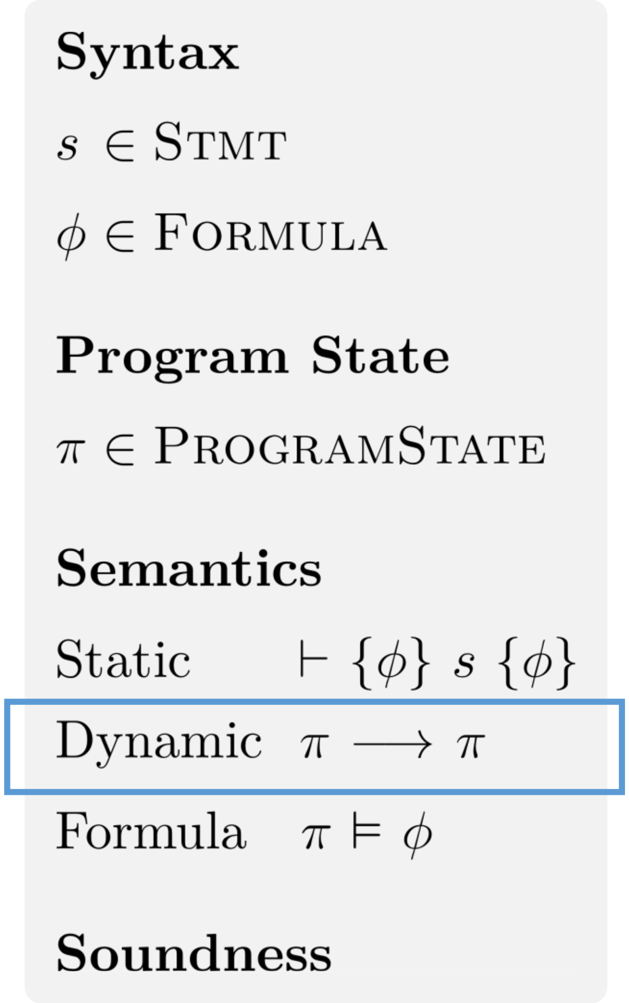
Soundness

$$\frac{}{\vdash \{\phi\}\ \texttt{skip}\ \{\phi\}}\ \textsc{HSkip}$$

$$\frac{}{\vdash \{\phi[e/x]\}\ x\ \texttt{:=}\ e\ \{\phi\}}\ \textsc{HAssign}$$

$$\vdots$$

Gradualization – Starting Point



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness

$$\langle[\mathtt{x} \mapsto 6, \mathtt{y} \mapsto 3], \mathtt{x\ :=\ y;\ assert\ (x\ =\ 3)}\rangle$$

$$\longrightarrow^{*}$$

$$\langle[\mathtt{x} \mapsto 3, \mathtt{y} \mapsto 3], \mathtt{skip}\rangle$$

Gradualization – Starting Point

Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness

$$\langle [\mathrm{x} \mapsto 6, \mathrm{y} \mapsto 3], \texttt{x := y; assert (x = 3)} \rangle$$

$$\longrightarrow^{*}$$

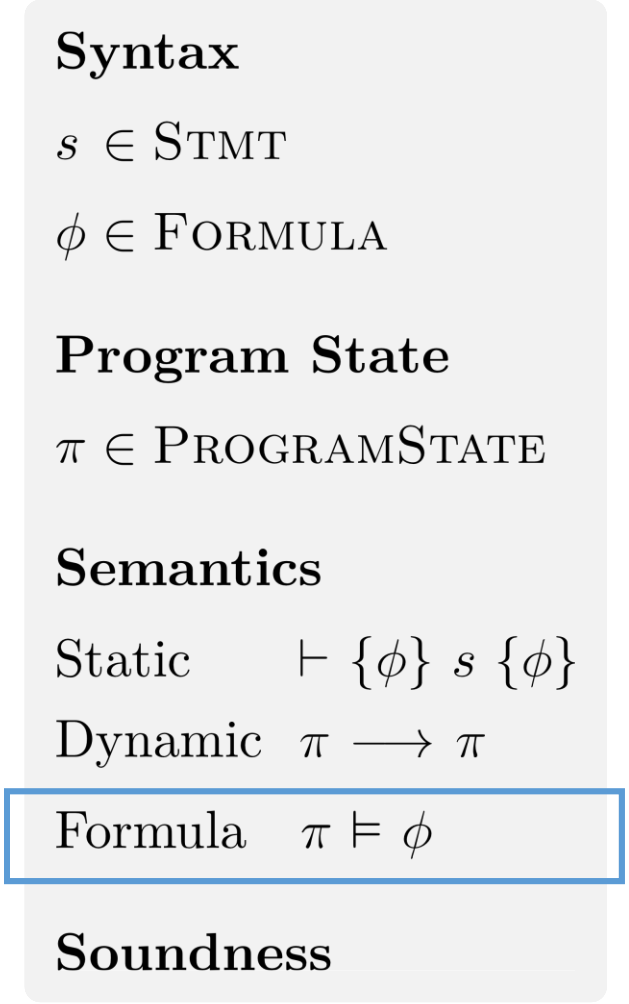
$$\langle [\mathrm{x} \mapsto 3, \mathrm{y} \mapsto 3], \texttt{skip} \rangle$$

$$\langle [\mathrm{x} \mapsto 6, \mathrm{y} \mapsto 3], \texttt{x := y; assert (x = 3)} \rangle$$

$$\xrightarrow{\texttt{x := y; assert (x = 3)}}$$

$$\langle [\mathrm{x} \mapsto 3, \mathrm{y} \mapsto 3], \texttt{skip} \rangle$$

Gradualization – Starting Point



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \models \phi$

Soundness

$$\langle [\mathrm{x} \mapsto 3], s\rangle \models (\texttt{x = 3})$$

$$\langle [\mathrm{x} \mapsto 4, \mathrm{y} \mapsto 4], s\rangle \models (\texttt{y = x})$$

Gradualization – Starting Point

Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness

$$\langle [\mathrm{x} \mapsto 3], s \rangle \vDash (\mathtt{x}\ =\ \mathtt{3})$$

$$\langle [\mathrm{x} \mapsto 4, \mathrm{y} \mapsto 4], s \rangle \vDash (\mathtt{y}\ =\ \mathtt{x})$$

Natural definition of implication

$$\phi_1 \Rightarrow \phi_2 \overset{\text{def}}{\iff} \forall \pi .\ \pi \vDash \phi_1 \implies \pi \vDash \phi_2$$

Gradualization – Starting Point

Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness

$$\langle[\mathrm{x} \mapsto 3], s\rangle \vDash (\mathtt{x\ =\ 3})$$

$$\langle[\mathrm{x} \mapsto 4, \mathrm{y} \mapsto 4], s\rangle \vDash (\mathtt{y\ =\ x})$$

Natural definition of implication

$$\phi_1 \Rightarrow \phi_2 \overset{\text{def}}{\iff} \forall\pi.\ \pi \vDash \phi_1 \implies \pi \vDash \phi_2$$

$$(\mathtt{x\ =\ 3}) \wedge (\mathtt{y\ =\ 2}) \Rightarrow (\mathtt{y\ =\ 2})$$

Gradualization – Starting Point

Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness

$$\langle [\mathrm{x} \mapsto 3], s\rangle \vDash (\mathtt{x\ =\ 3})$$

$$\langle [\mathrm{x} \mapsto 4, \mathrm{y} \mapsto 4], s\rangle \vDash (\mathtt{y\ =\ x})$$

Natural definition of implication

$$\phi_1 \Rightarrow \phi_2 \overset{\text{def}}{\Longleftrightarrow} \forall \pi.\ \pi \vDash \phi_1 \implies \pi \vDash \phi_2$$

$$(\mathtt{x\ =\ 3}) \wedge (\mathtt{y\ =\ 2}) \Rightarrow (\mathtt{y\ =\ 2})$$

$$(\mathtt{a\ =\ b}) \wedge (\mathtt{b\ =\ c}) \Rightarrow (\mathtt{a\ =\ c})$$

Gradualization – Starting Point

Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness

$$\frac{\vdash \{\phi\}\ s\ \{\phi'\}}{\vDash \{\phi\}\ s\ \{\phi'\}}\ \textsc{Soundness}$$

Gradualization – Starting Point

Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness

Semantical validity of Hoare triples

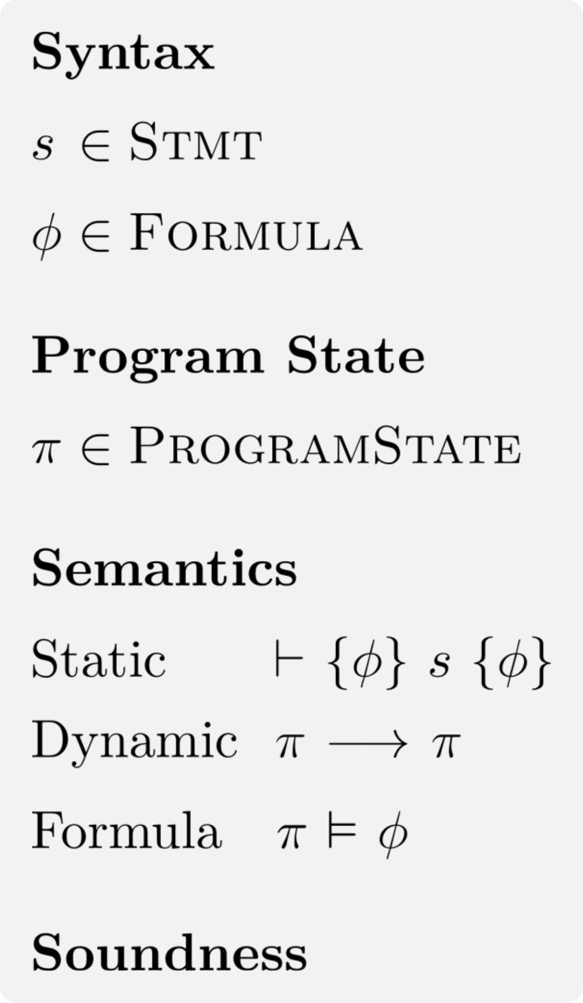
$$\vDash \{\phi\}\ s\ \{\phi'\}$$

$$\overset{\text{def}}{\Longleftrightarrow}$$

$$\forall \pi, \pi'.\ \pi \xrightarrow{s} \pi' \land \pi \vDash \phi \implies \pi' \vDash \phi'$$

$$\dfrac{\vdash \{\phi\}\ s\ \{\phi'\}}{\vDash \{\phi\}\ s\ \{\phi'\}}\ \textsc{Soundness}$$

Gradualization – Overview



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

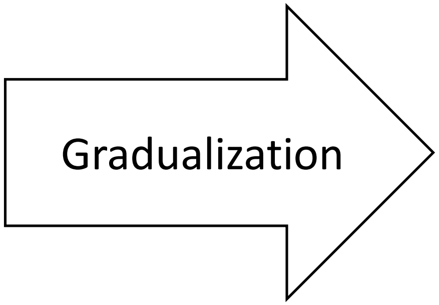
Semantics

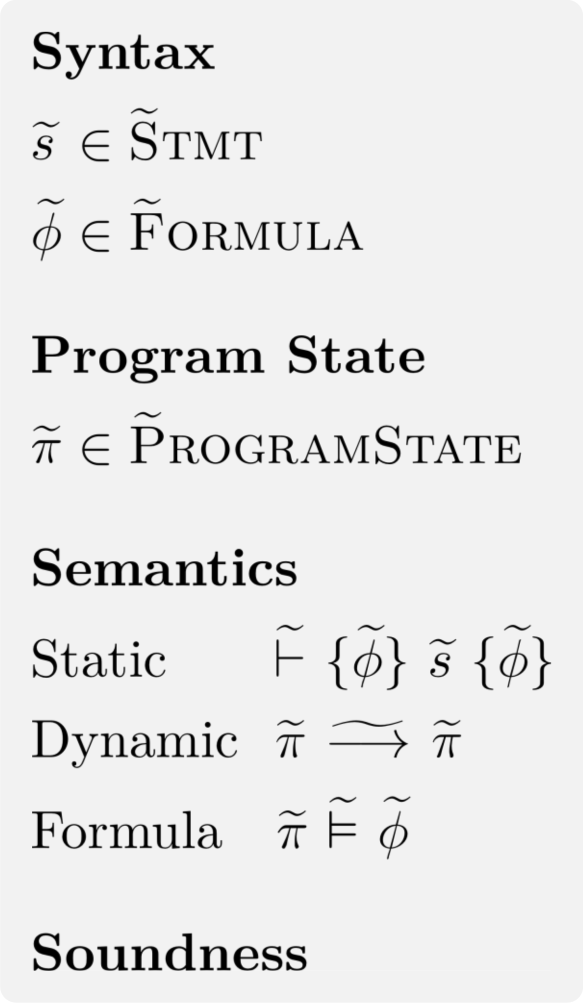
Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness





Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Semantics

Static $\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$

Dynamic $\widetilde{\pi} \mathrel{\widetilde{\longrightarrow}} \widetilde{\pi}$

Formula $\widetilde{\pi} \mathrel{\widetilde{\vDash}} \widetilde{\phi}$

Soundness

Gradualization – Approach

Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Semantics

Static $\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$

Dynamic $\widetilde{\pi} \ \widetilde{\longrightarrow}\ \widetilde{\pi}$

Formula $\widetilde{\pi}\ \widetilde{\vDash}\ \widetilde{\phi}$

Soundness

Gradualization – Approach

Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Semantics

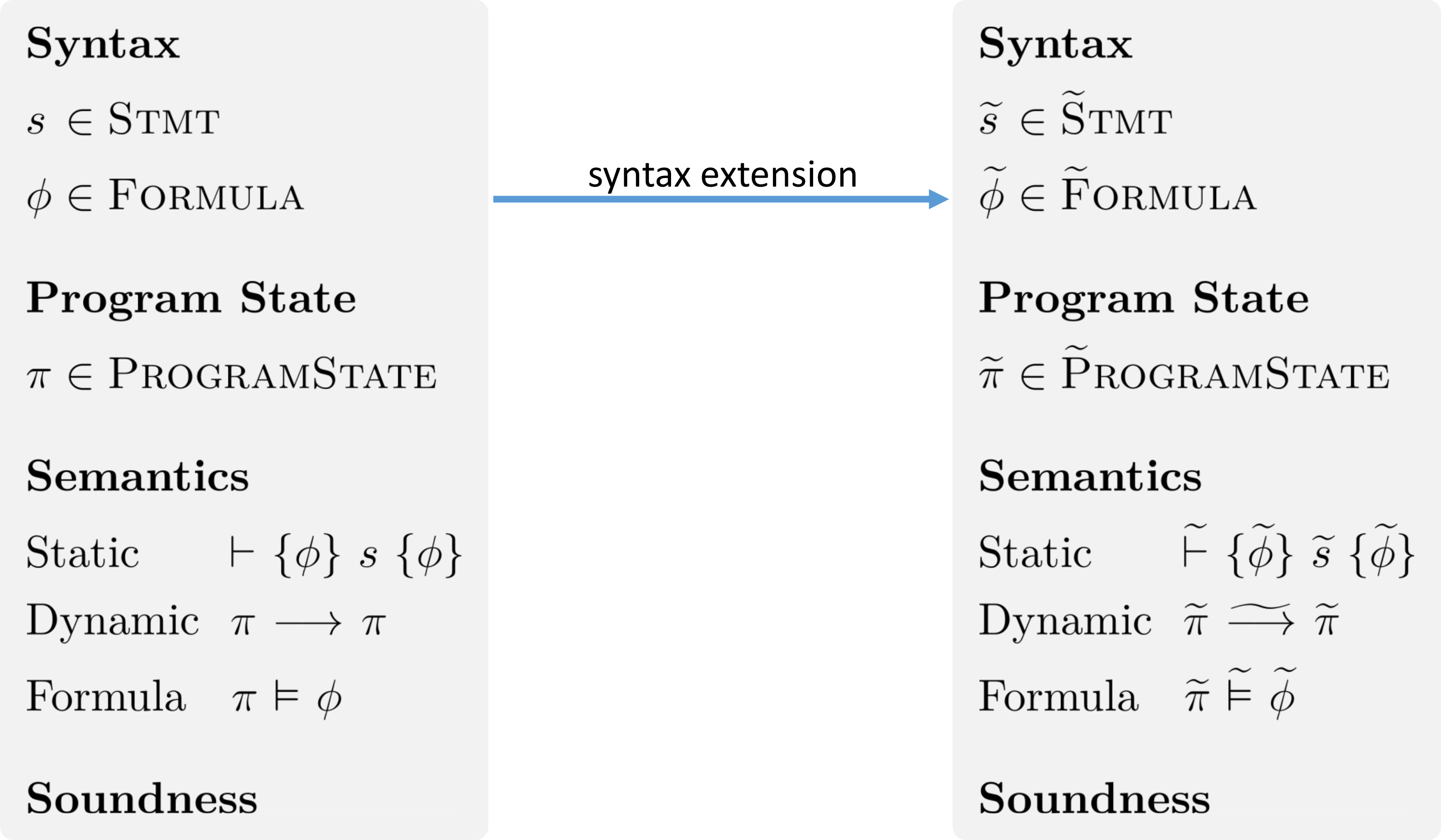
Static $\widetilde{\vdash}\ \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$

Dynamic $\widetilde{\pi}\ \widetilde{\longrightarrow}\ \widetilde{\pi}$

Formula $\widetilde{\pi}\ \widetilde{\vDash}\ \widetilde{\phi}$

Soundness

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{S}}\textsc{tmt}$

$\widetilde{\phi} \in \widetilde{\textsc{F}}\textsc{ormula}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{P}}\textsc{rogramState}$

Semantics

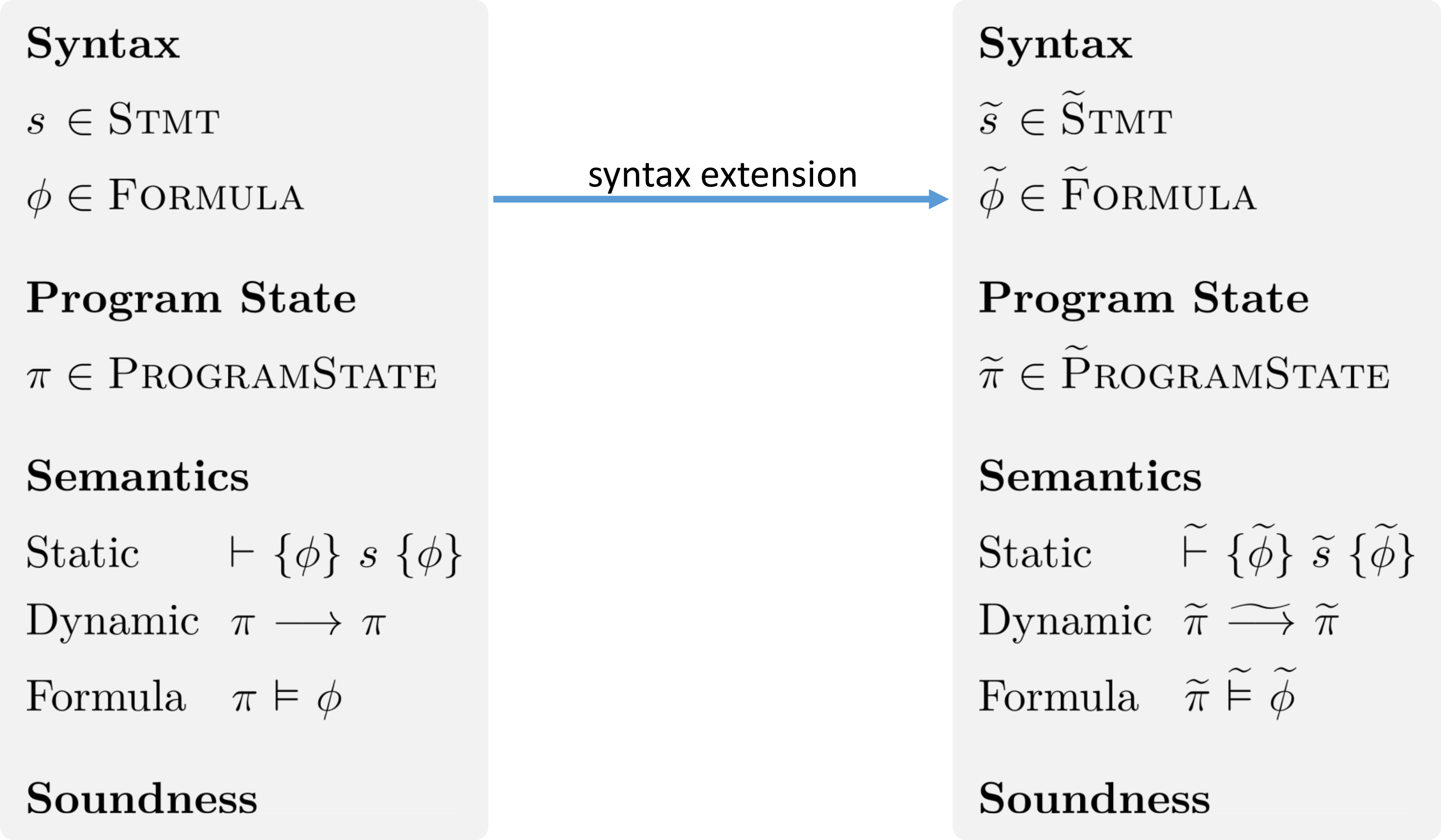
Static $\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$

Dynamic $\widetilde{\pi} \mathrel{\widetilde{\longrightarrow}} \widetilde{\pi}$

Formula $\widetilde{\pi} \mathrel{\widetilde{\vDash}} \widetilde{\phi}$

Soundness

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Semantics

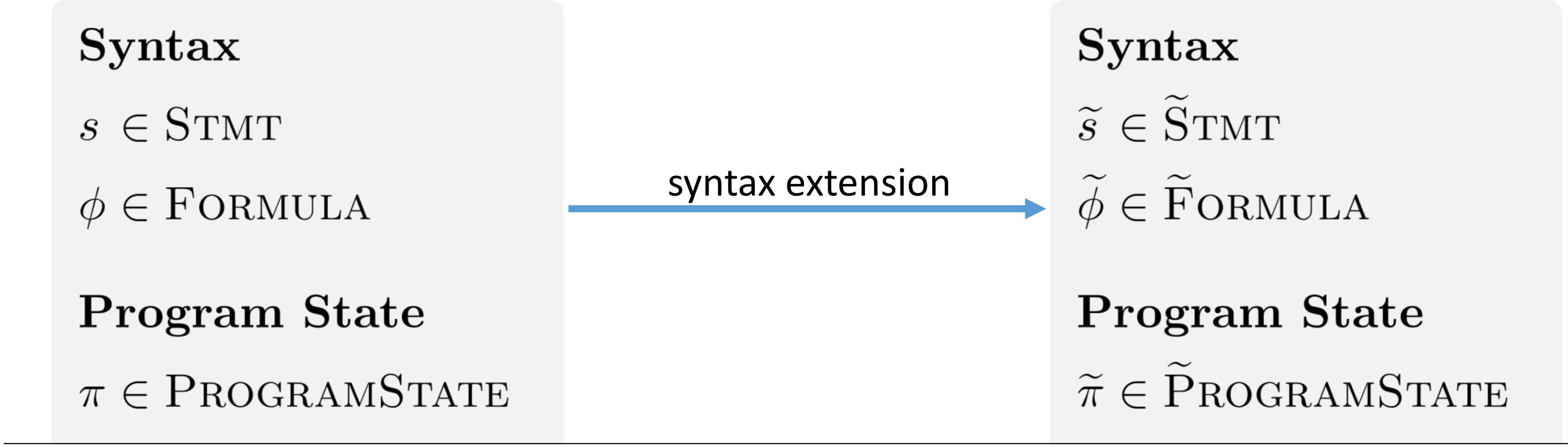
Static $\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$

Dynamic $\widetilde{\pi} \overset{\sim}{\longrightarrow} \widetilde{\pi}$

Formula $\widetilde{\pi}\ \widetilde{\vDash}\ \widetilde{\phi}$

Soundness

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

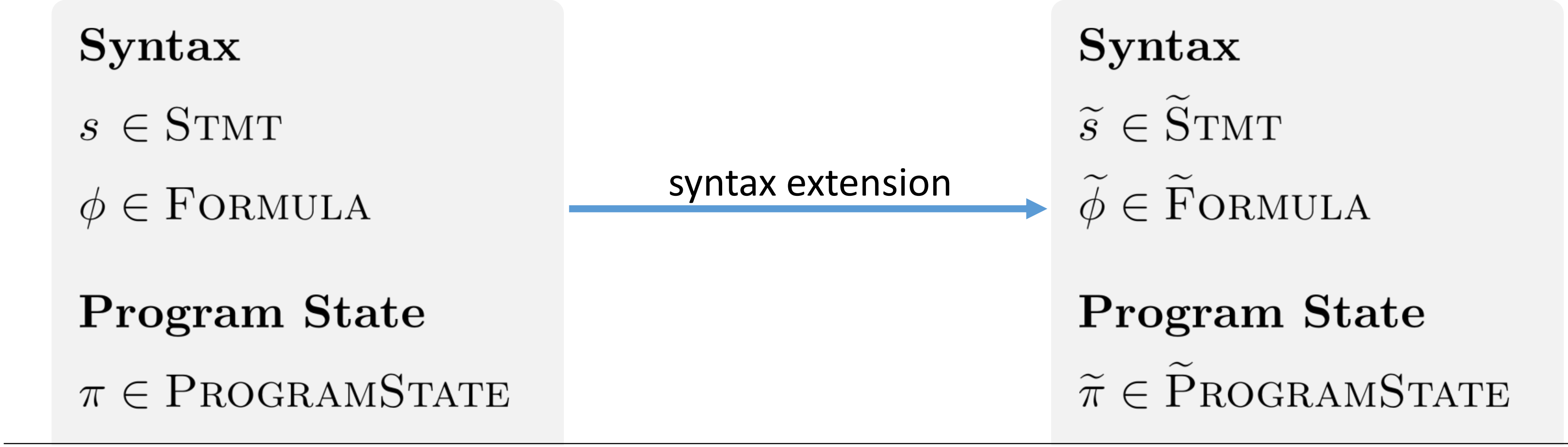
Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Rules

Implementation

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Rules

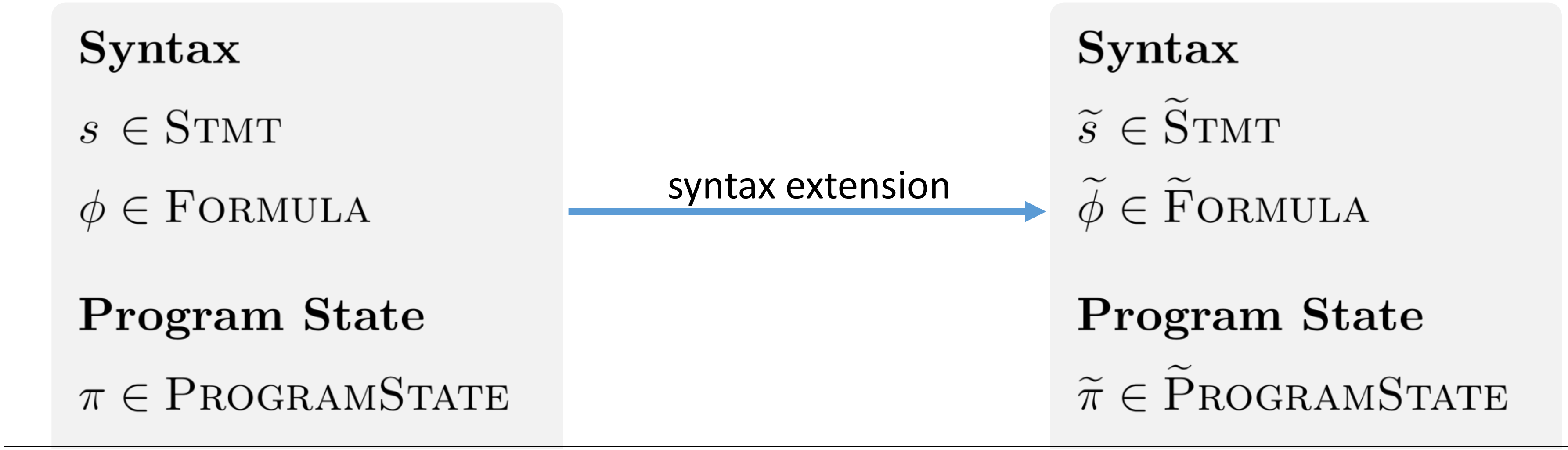
$\textsc{Formula} \subset \widetilde{\textsc{Formula}}$

$? \in \widetilde{\textsc{Formula}}$

$? \notin \textsc{Formula}$

Implementation

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Rules

$\textsc{Formula} \subset \widetilde{\textsc{Formula}}$

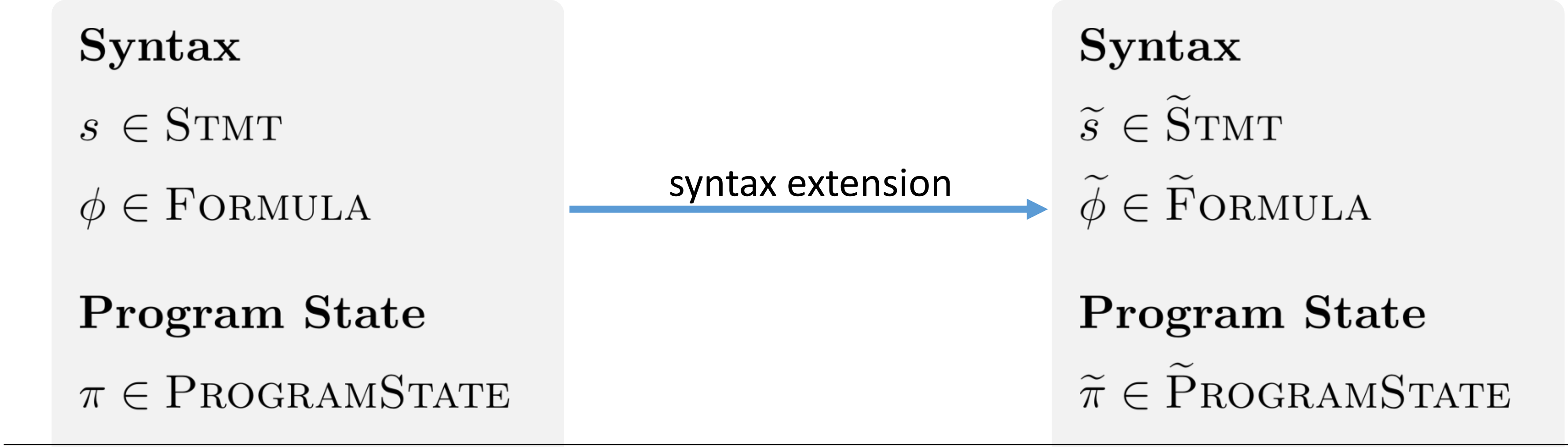
$? \in \widetilde{\textsc{Formula}}$

$? \notin \textsc{Formula}$

Implementation

$\widetilde{\phi} \ ::= \ \phi \mid ?$

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Rules

$\textsc{Formula} \subset \widetilde{\textsc{Formula}}$

$\texttt{?} \in \widetilde{\textsc{Formula}}$

$\texttt{?} \notin \textsc{Formula}$

Implementation

$\widetilde{\phi} \; ::= \; \phi \mid \texttt{?}$

$\widetilde{\phi} \; ::= \; \phi \mid \phi \land \texttt{?}$

$\text{where} \quad \texttt{?} \overset{\text{def}}{=} \texttt{true} \land \texttt{?}$

Gradualization – Approach

Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

$$\xrightarrow{\text{syntax extension}}$$

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Rules

$\textsc{Formula} \subset \widetilde{\textsc{Formula}}$

$? \in \widetilde{\textsc{Formula}}$

$? \notin \textsc{Formula}$

```
(x = 3) ∧ (y = ...)
(x = 3) ∧ ?
```

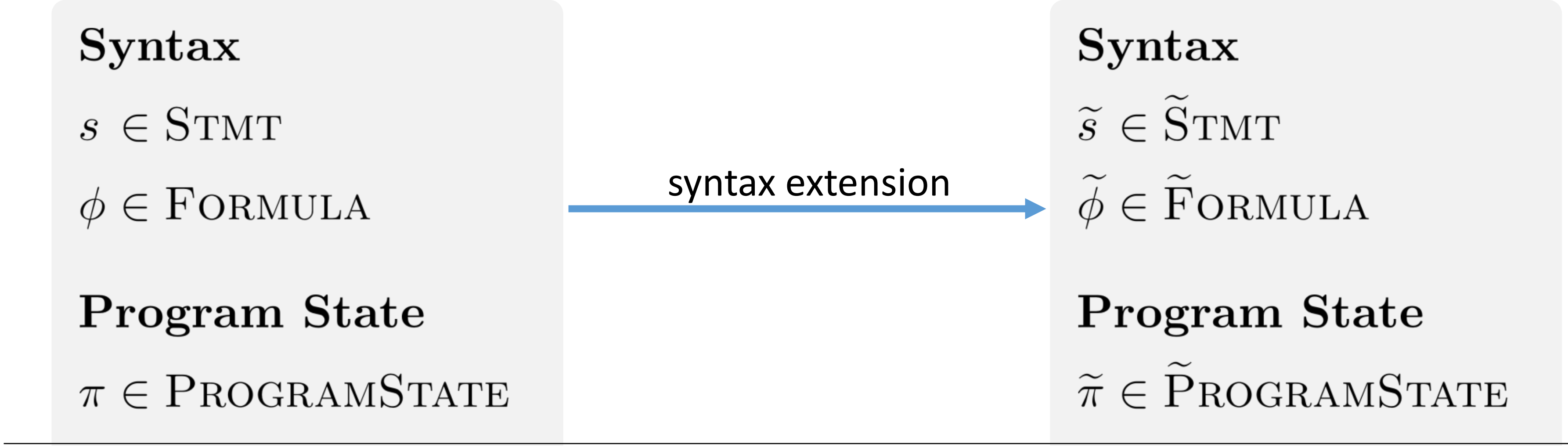
Implementation

$\widetilde{\phi} \ ::= \ \phi \mid ?$

$\widetilde{\phi} \ ::= \ \phi \mid \phi \ \land \ ?$

where $\ ? \overset{\text{def}}{=} \texttt{true} \ \land \ ?$

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Rules

$\textsc{Formula} \subset \widetilde{\textsc{Formula}}$

$? \in \widetilde{\textsc{Formula}}$

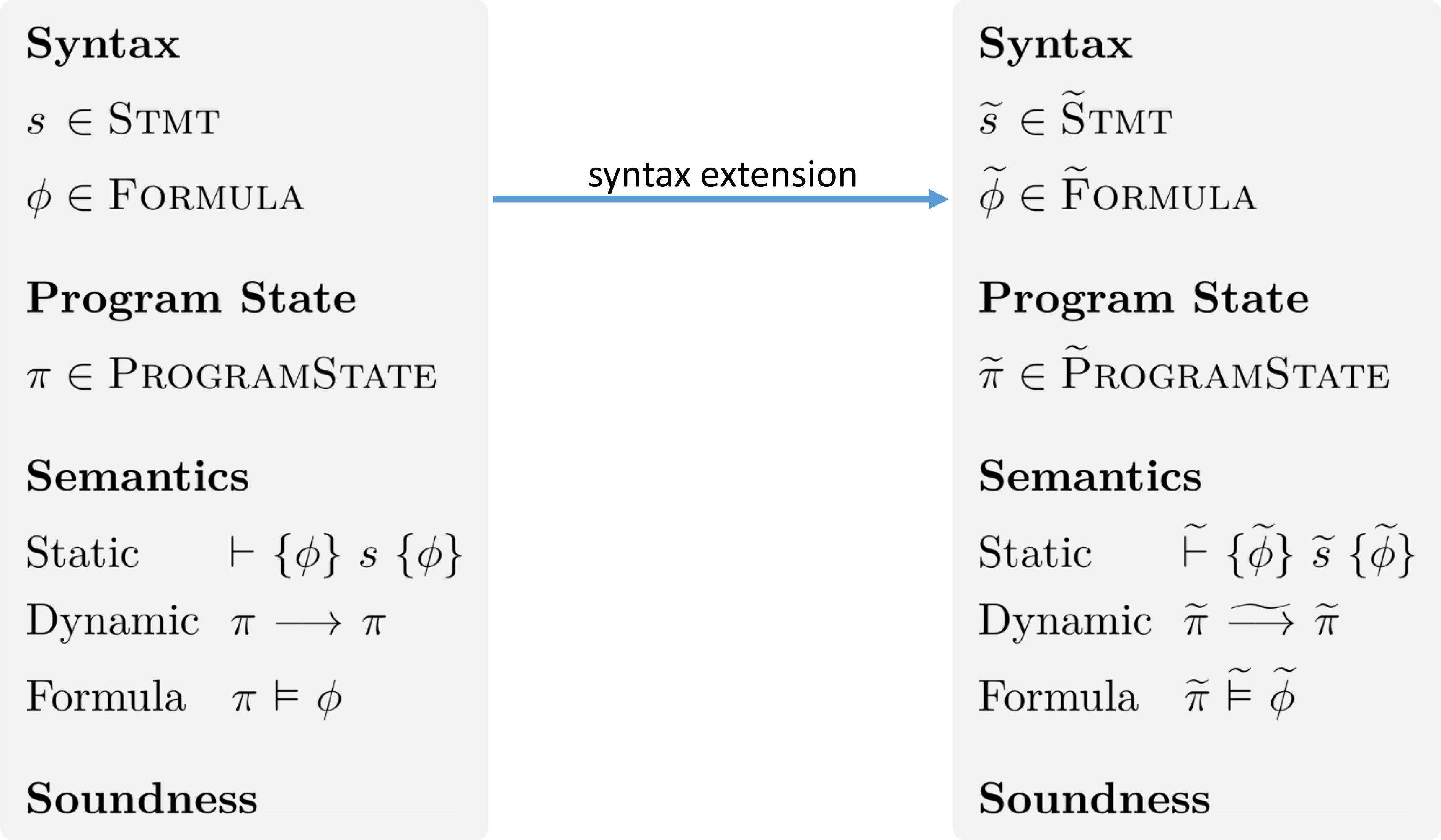
$? \notin \textsc{Formula}$

Implementation

$$\widetilde{\phi} \;::=\; \phi \mid ?$$

$$\begin{aligned} \gamma(\phi) &= \{\ \phi\ \} \\ \gamma(?) &= \textsc{SatFormula} \\ &= \{\ \phi \mid \exists\pi.\ \pi \models \phi\ \} \end{aligned}$$

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \models \phi$

Soundness

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Semantics

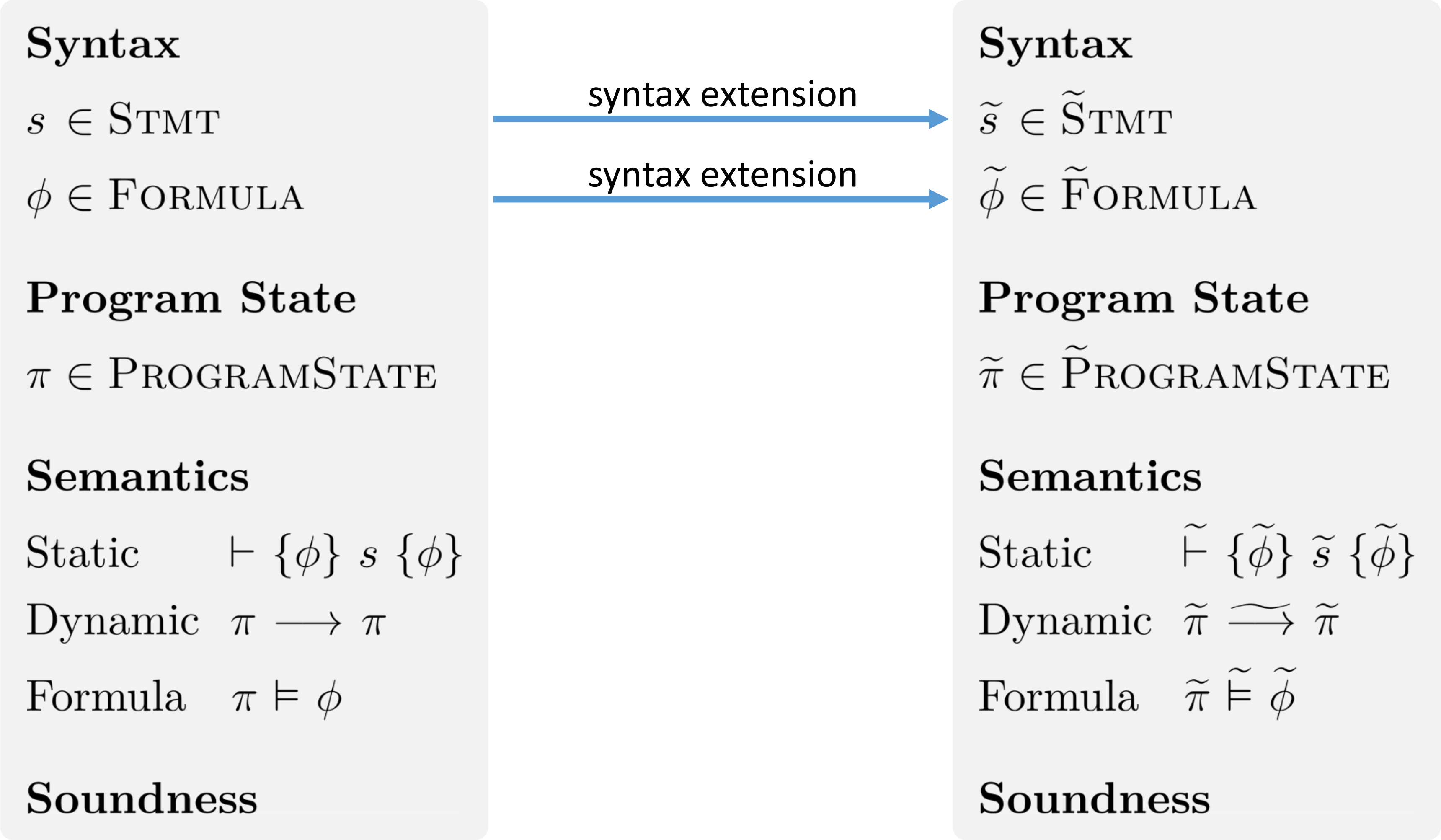
Static $\widetilde{\vdash}\ \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$

Dynamic $\widetilde{\pi}\ \widetilde{\longrightarrow}\ \widetilde{\pi}$

Formula $\widetilde{\pi}\ \widetilde{\models}\ \widetilde{\phi}$

Soundness

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \models \phi$

Soundness

syntax extension

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Semantics

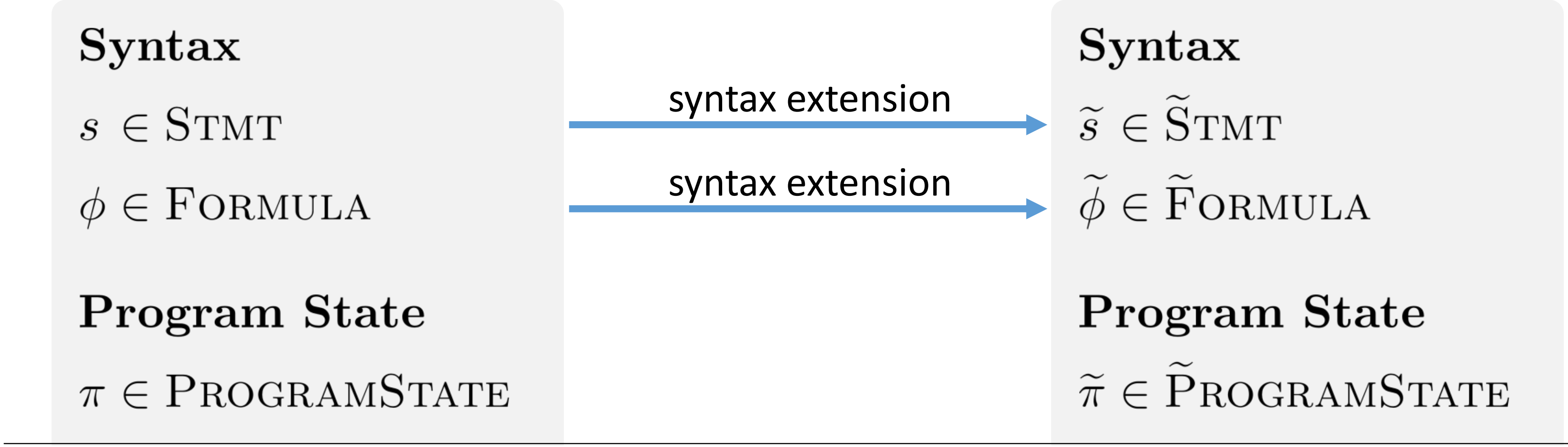
Static $\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$

Dynamic $\widetilde{\pi} \mathrel{\widetilde{\longrightarrow}} \widetilde{\pi}$

Formula $\widetilde{\pi} \mathrel{\widetilde{\models}} \widetilde{\phi}$

Soundness

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

syntax extension

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

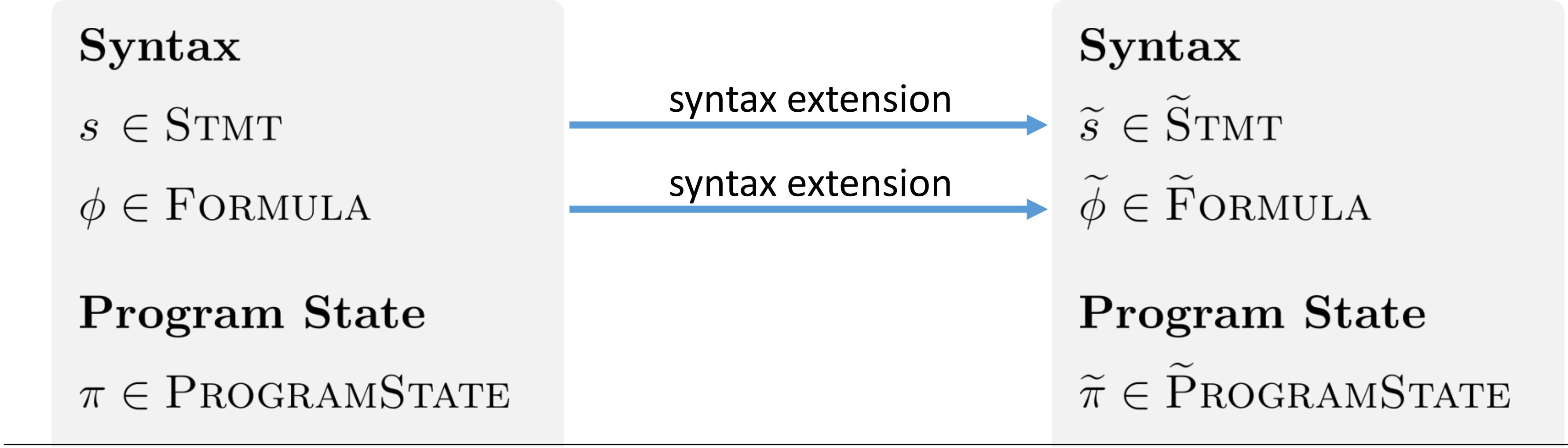
Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Rules

Implementation

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

syntax extension

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

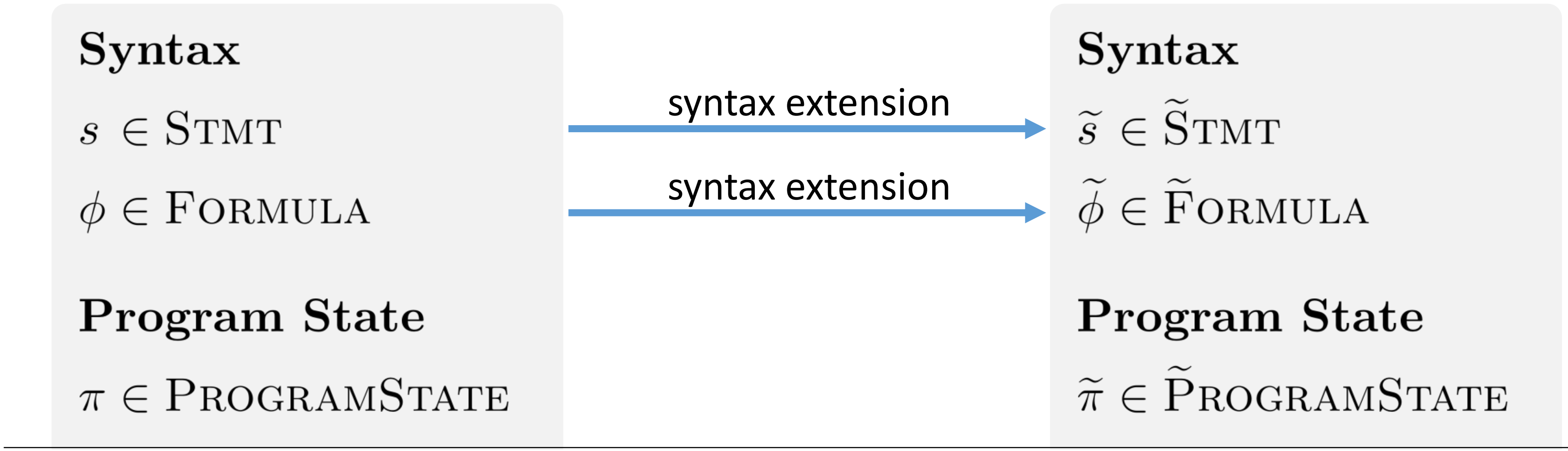
$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Rules

$\textsc{Stmt} \subseteq \widetilde{\textsc{Stmt}}$

Implementation

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

syntax extension

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

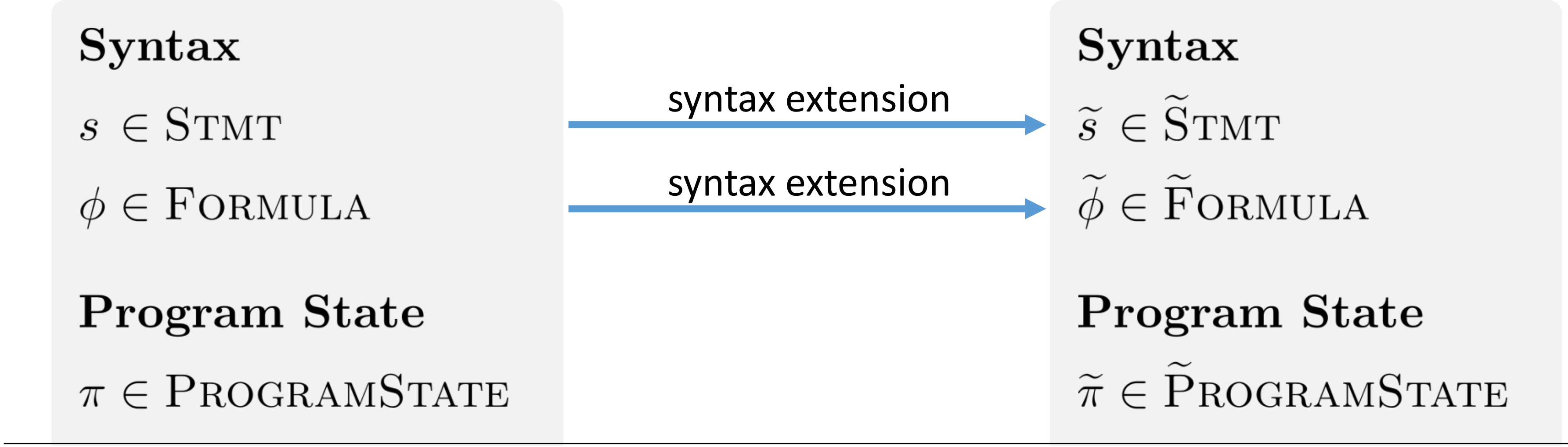
Rules

$\textsc{Stmt} \subseteq \widetilde{\textsc{Stmt}}$

Implementation

$$s \quad ::= \quad x \ \texttt{:=}\ e \ \mid\ \texttt{assert}\ \phi \ \mid\ s_1\,\texttt{;}\ s_2$$

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

syntax extension

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

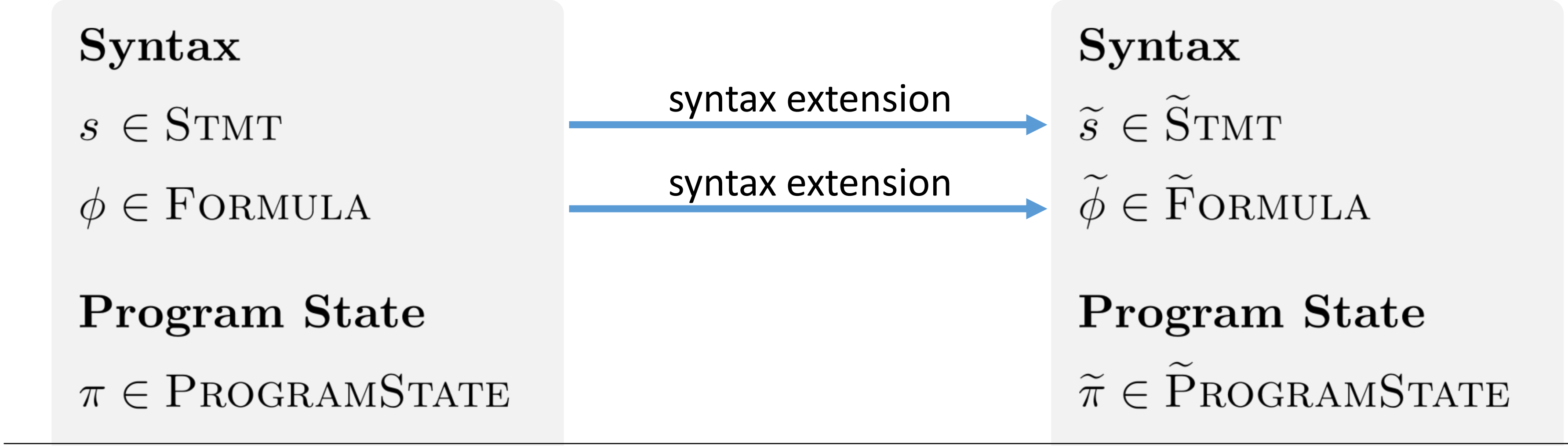
Rules

$\textsc{Stmt} \subseteq \widetilde{\textsc{Stmt}}$

Implementation

$$\widetilde{s} \quad ::= \quad x \ \texttt{:=}\ e \mid \texttt{assert}\ \widetilde{\phi} \mid \widetilde{s}_1\texttt{;}\ \widetilde{s}_2$$

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

syntax extension

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Rules

$\textsc{Stmt} \subseteq \widetilde{\textsc{Stmt}}$

Implementation

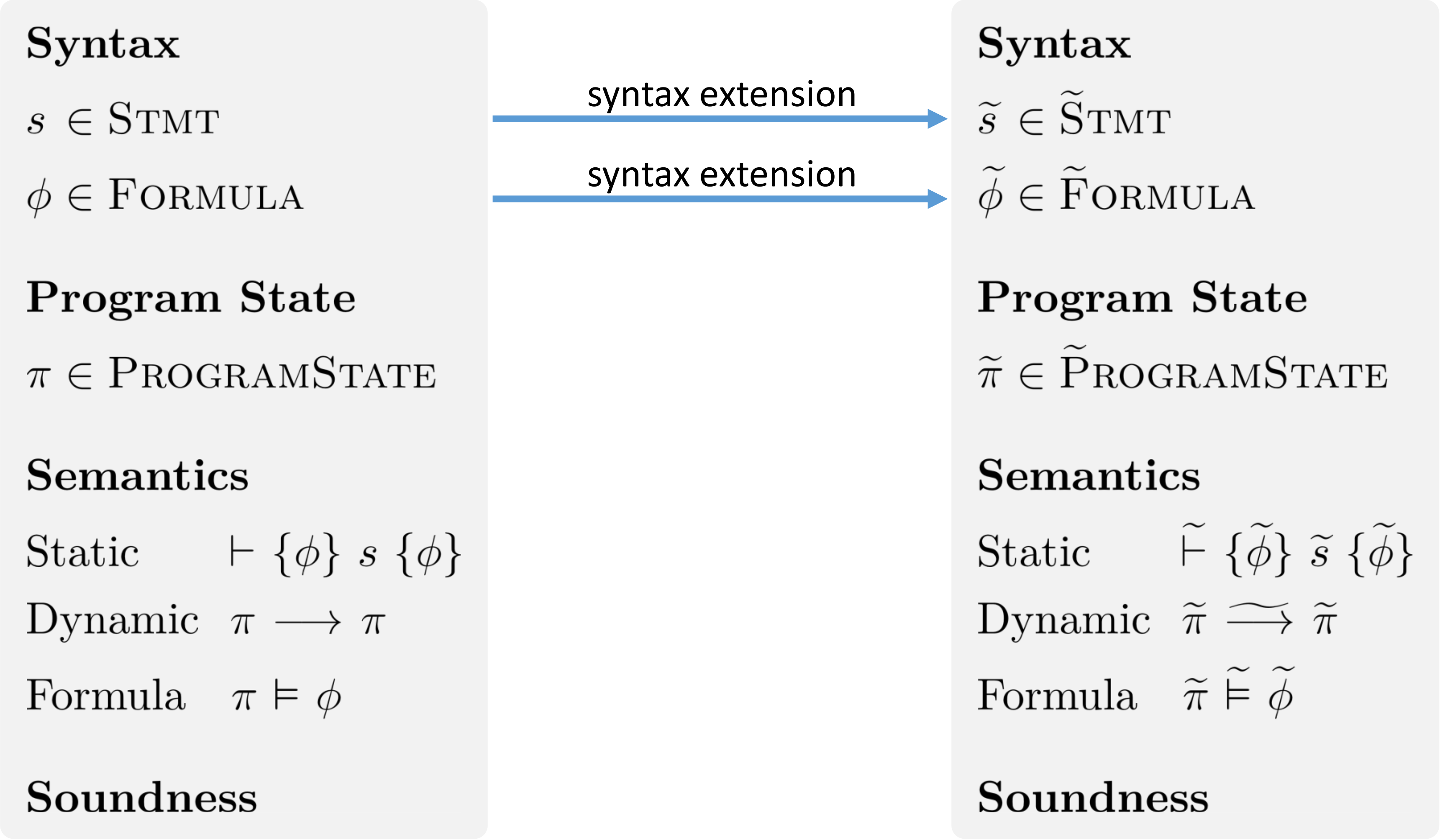
$$\widetilde{s} \quad ::= \quad x \ \texttt{:=}\ e \ \mid \ \texttt{assert}\ \widetilde{\phi} \ \mid \ \widetilde{s}_1\,\texttt{;}\ \widetilde{s}_2$$

$$\gamma : \widetilde{\textsc{Stmt}} \rightarrow \mathcal{P}^{\textsc{Stmt}}$$

$$\gamma(\texttt{assert}\ \widetilde{\phi}) = \{\ \texttt{assert}\ \phi \mid \phi \in \gamma(\widetilde{\phi})\ \}$$

$\dots$

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness

syntax extension

syntax extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Semantics

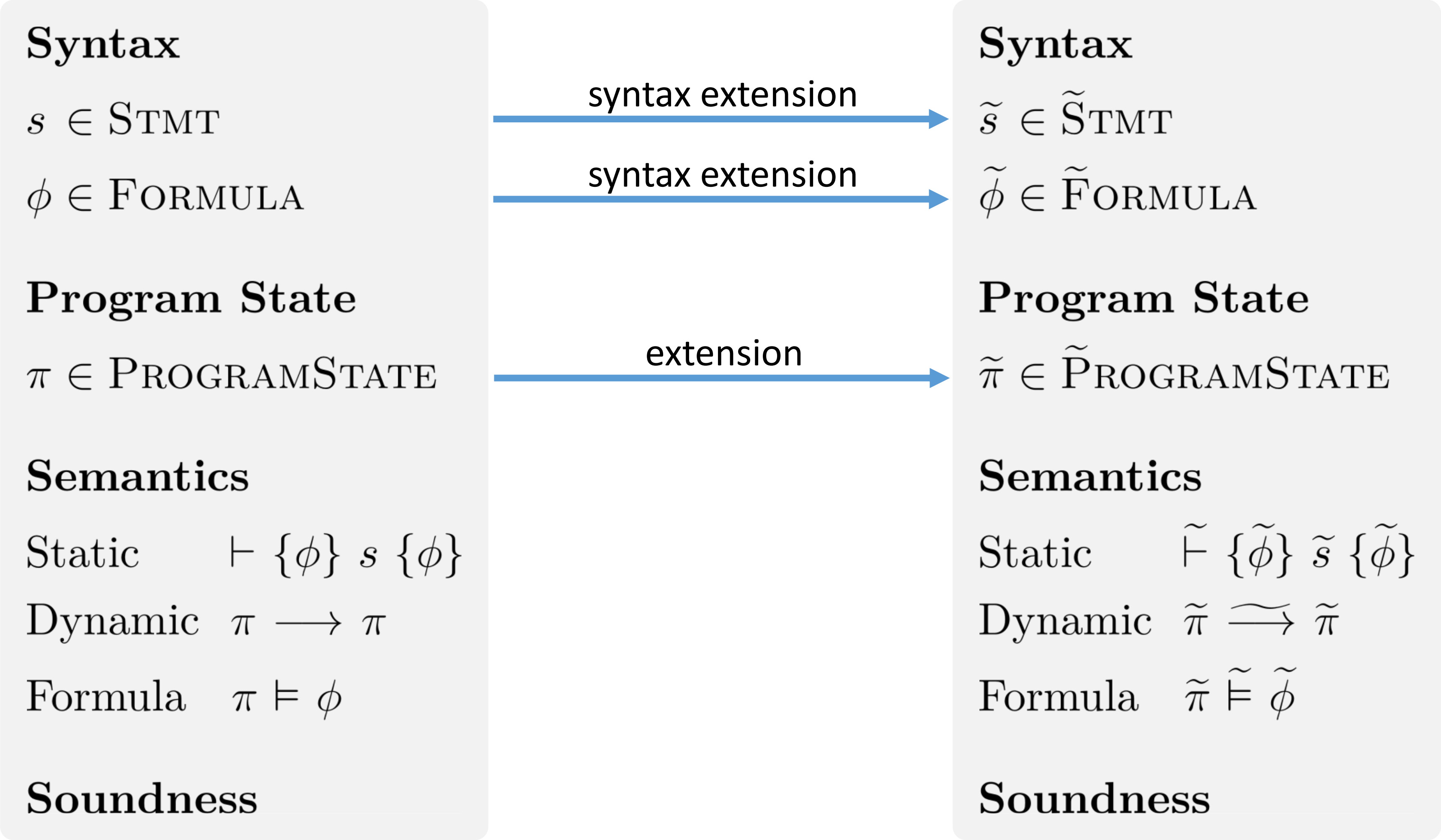
Static $\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$

Dynamic $\widetilde{\pi} \mathrel{\widetilde{\longrightarrow}} \widetilde{\pi}$

Formula $\widetilde{\pi} \mathrel{\widetilde{\vDash}} \widetilde{\phi}$

Soundness

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness

syntax extension

syntax extension

extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Semantics

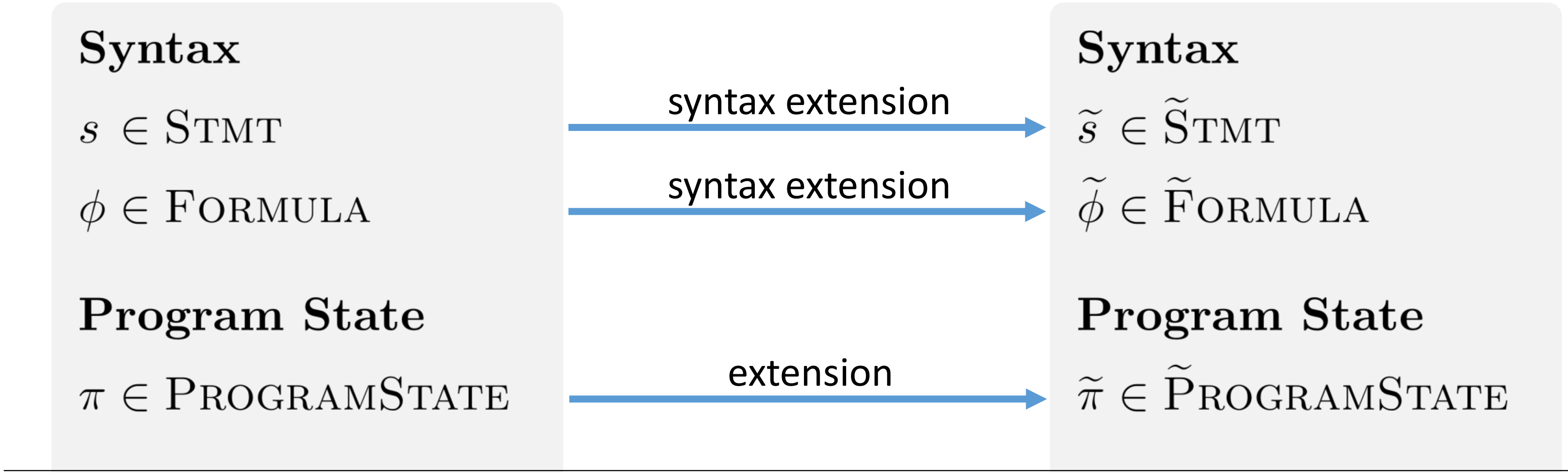
Static $\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$

Dynamic $\widetilde{\pi} \mathrel{\widetilde{\longrightarrow}} \widetilde{\pi}$

Formula $\widetilde{\pi} \mathrel{\widetilde{\vDash}} \widetilde{\phi}$

Soundness

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

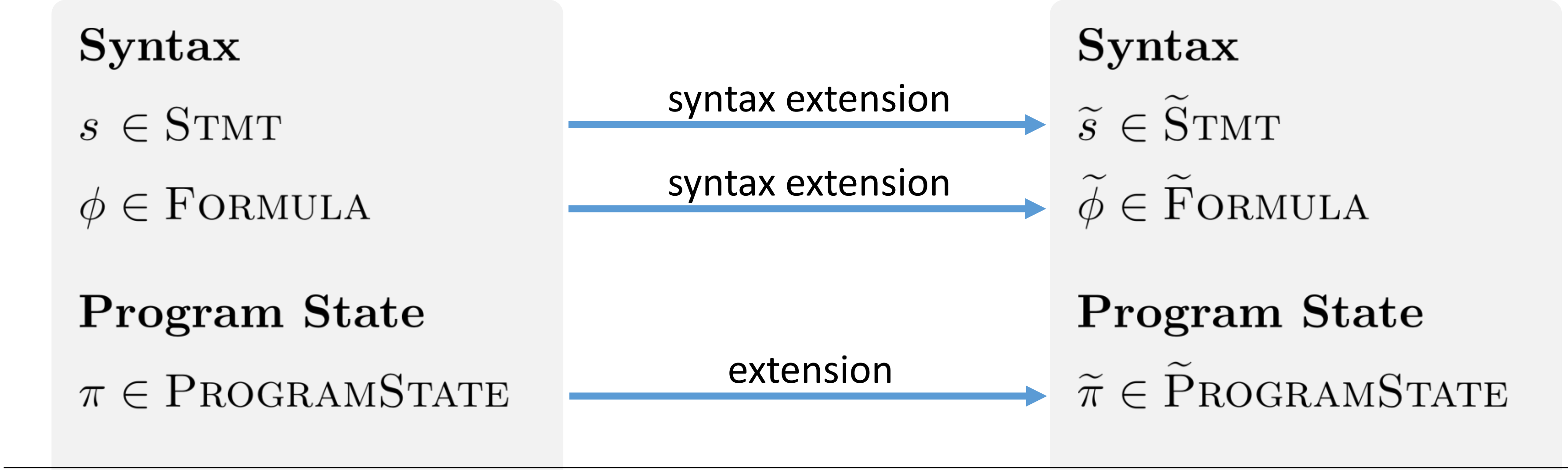
Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Rules

Implementation

Gradualization – Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

syntax extension

syntax extension

extension

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Rules

$$\textsc{ProgramState} \subseteq \widetilde{\textsc{ProgramState}}$$

Implementation

$$\textsc{ProgramState} = (\textsc{Var} \rightharpoonup \mathbb{N}_0) \times \textsc{Stmt}$$

$$\widetilde{\textsc{ProgramState}} = (\textsc{Var} \rightharpoonup \mathbb{N}_0) \times \widetilde{\textsc{Stmt}}$$

Gradualization – Approach

Syntax | **Syntax**

$$s \in \textsc{Stmt} \xrightarrow{\text{syntax extension}} \widetilde{s} \in \widetilde{\textsc{Stmt}}$$

$$\phi \in \textsc{Formula} \xrightarrow{\text{syntax extension}} \widetilde{\phi} \in \widetilde{\textsc{Formula}}$$

Program State | **Program State**

$$\pi \in \textsc{ProgramState} \xrightarrow{\text{extension}} \widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$$

Rules

$$\textsc{ProgramState} \subseteq \widetilde{\textsc{ProgramState}}$$

Implementation

$$\textsc{ProgramState} = (\textsc{Var} \rightharpoonup \mathbb{N}_0) \times \textsc{Stmt}$$

$$\widetilde{\textsc{ProgramState}} = (\textsc{Var} \rightharpoonup \mathbb{N}_0) \times \widetilde{\textsc{Stmt}}$$

$$\gamma(\langle \sigma, \widetilde{s} \rangle) = \{\sigma\} \times \gamma(\widetilde{s})$$

Gradual Lifting

Gradual System

Static System

$$\begin{tikzcd}
\widetilde{\phi_1} \arrow[r, bend left, "\widetilde{f}"'] \arrow[d, dashed, "\gamma"'] & \widetilde{\phi_2} \arrow[d, dashed, "\gamma"] \\
\overline{\phi_1} \arrow[r, dashed, bend left, "\overline{f}"'] & \overline{\phi_2} \subseteq \overline{\phi_2}'
\end{tikzcd}$$

Gradual Lifting

Gradual System

Static System

$$\begin{tikzcd}
\widetilde{\phi_1} \arrow[rr, bend left, "\widetilde{f}"'] \arrow[d, dotted, "\gamma"'] & & \widetilde{\phi_2} \arrow[drr, dotted, "\gamma"] & & \\
\overline{\phi_1} \arrow[rr, dotted, bend left, "\overline{f}"'] & & \overline{\phi_2} & \subseteq & \overline{\phi_2}'
\end{tikzcd}$$

Gradual Lifting

Gradual System

Static System

$$\begin{tikzcd}
\widetilde{\phi_1} \arrow[rr, bend left, "\widetilde{f}"'] \arrow[d, dashed, "\gamma"'] & & \widetilde{\phi_2} \arrow[d, dashed, "\gamma"] \\
\overline{\phi_1} \arrow[r, dashed, bend left, "\overline{f}"'] & \overline{\phi_2} \arrow[r, phantom, "\subseteq"] & \overline{\phi_2}'
\end{tikzcd}$$

Gradual Lifting

Gradual System

Static System

$$\begin{tikzcd}
\widetilde{\pi_1} \arrow[r, bend left, "\widetilde{\longrightarrow}"'] \arrow[d, dotted, "\gamma"'] & \widetilde{\pi_2} \arrow[d, dotted, "\gamma"] \\
\overline{\pi_1} \arrow[r, dotted, bend left, "\overline{\Longrightarrow}"'] & \overline{\pi_2} \subseteq \overline{\pi_2}'
\end{tikzcd}$$

Gradual Lifting

$$\begin{tikzcd}
\text{Gradual System} & \widetilde{\pi_1} \arrow[rr, bend left, "\widetilde{\longrightarrow}"'] \arrow[d, dotted, "\gamma"'] & & \widetilde{\pi_2} \arrow[d, dotted, "\gamma"] \\
\text{Static System} & \overline{\pi_1} \arrow[rr, dotted, bend left, "\Longrightarrow"'] & & \overline{\pi_2} \subseteq \overline{\pi_2}'
\end{tikzcd}$$

Gradual Lifting

Gradual System

Static System

$$\begin{tikzcd}
\widetilde{\pi_1} \arrow[r, bend left, "\widetilde{\longrightarrow}"'] \arrow[d, dotted, "\gamma"'] & \widetilde{\pi_2} \arrow[d, dotted, "\gamma"] \\
\overline{\pi_1} \arrow[r, dotted, bend left, "\Longrightarrow"'] & \overline{\pi_2} \subseteq \overline{\pi_2}'
\end{tikzcd}$$

Gradual Lifting

Gradual System

Static System

$$\begin{tikzcd}
\widetilde{\pi_1} \arrow[r, bend left, "\widetilde{\longrightarrow}"'] \arrow[d, dotted, "\gamma"'] & \widetilde{\pi_2} \arrow[d, dotted, "\gamma"] \\
\overline{\pi_1} \arrow[r, dotted, bend left, "\overline{\longrightarrow}"'] & \overline{\pi_2} \subseteq \overline{\pi_2}'
\end{tikzcd}$$

$$\frac{\langle \sigma, \texttt{assert}\ \ \phi_a \rangle \vDash \phi_a}{\langle \sigma, \texttt{assert}\ \ \phi_a \rangle \longrightarrow \langle \sigma, \texttt{skip} \rangle}\ \textsc{SsAssert}$$

Gradual Lifting

$$\frac{\langle \sigma, \texttt{assert}\ \ \phi_a\rangle \vDash \phi_a}{\langle \sigma, \texttt{assert}\ \ \phi_a\rangle \mathrel{\widetilde{\longrightarrow}} \langle \sigma, \texttt{skip}\rangle}\ \widetilde{\mathrm{S}}\textsc{sAssert}1
\qquad
\frac{}{\langle \sigma, \texttt{assert}\ \ ?\rangle \mathrel{\widetilde{\longrightarrow}} \langle \sigma, \texttt{skip}\rangle}\ \widetilde{\mathrm{S}}\textsc{sAssert}2$$

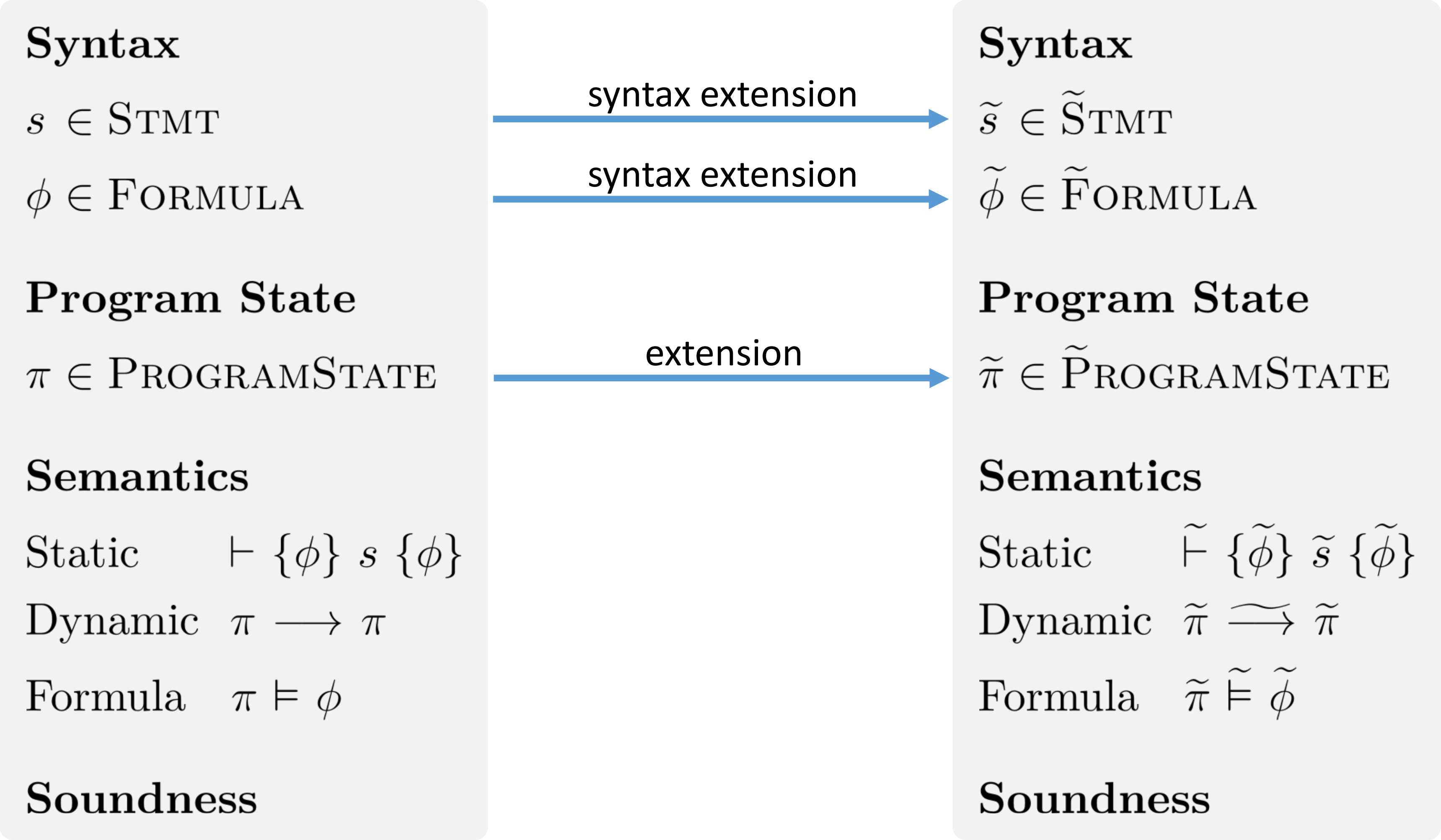
Gradual System

$$\begin{tikzcd}
\widetilde{\pi_1} \arrow[r, bend left, "\widetilde{\longrightarrow}"] \arrow[d, dotted, "\gamma"'] & \widetilde{\pi_2} \arrow[d, dotted, "\gamma"] \\
\overline{\pi_1} \arrow[r, dotted, bend left, "\Longrightarrow"'] & \overline{\pi_2} \subseteq \overline{\pi_2}'
\end{tikzcd}$$

Static System

$$\frac{\langle \sigma, \texttt{assert}\ \ \phi_a\rangle \vDash \phi_a}{\langle \sigma, \texttt{assert}\ \ \phi_a\rangle \longrightarrow \langle \sigma, \texttt{skip}\rangle}\ \textsc{SsAssert}$$

Gradual Verification - Approach



Syntax $\longrightarrow$ **Syntax**

$s \in \textsc{Stmt} \xrightarrow{\text{syntax extension}} \widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\phi \in \textsc{Formula} \xrightarrow{\text{syntax extension}} \widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

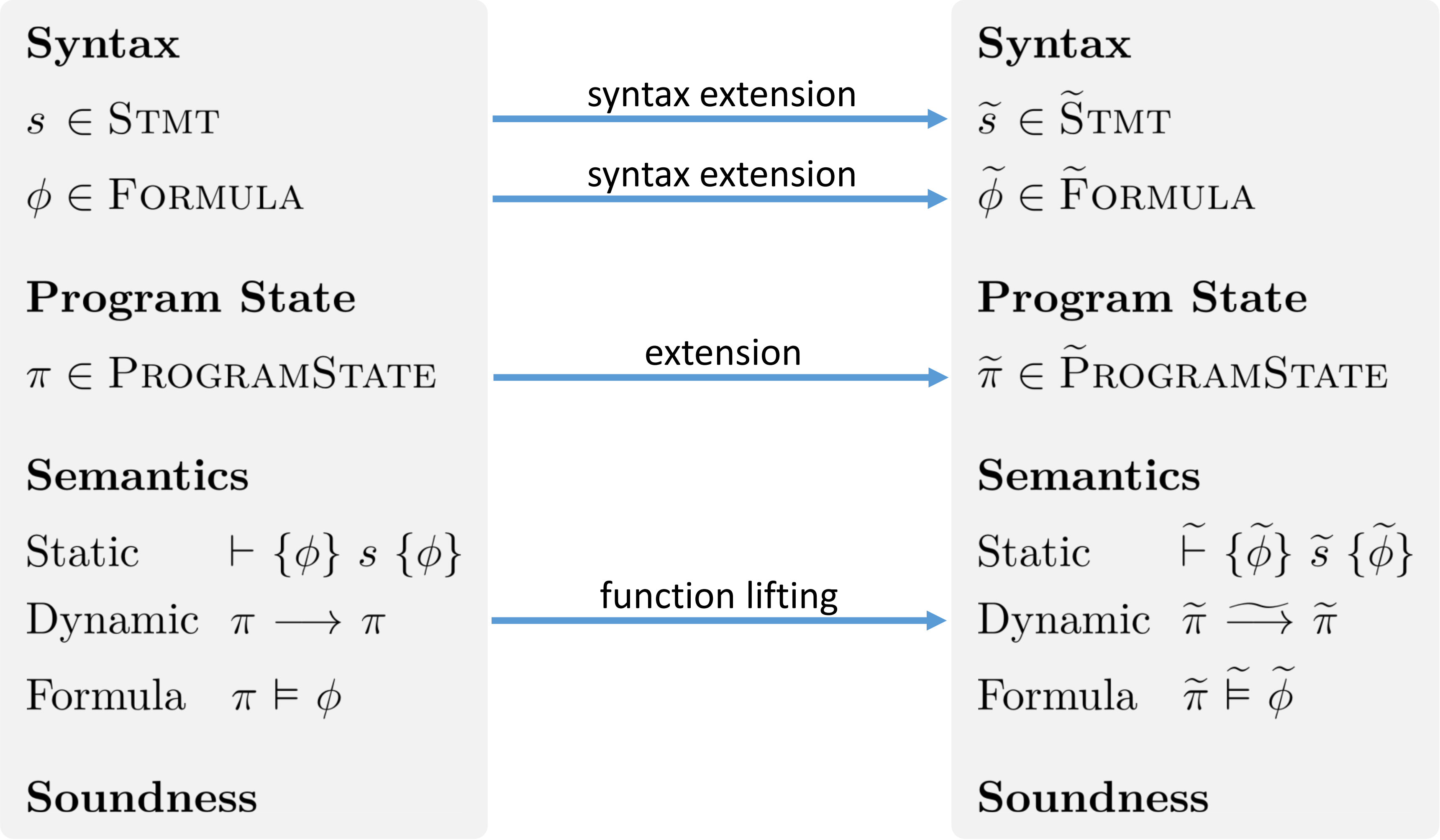
$\pi \in \textsc{ProgramState} \xrightarrow{\text{extension}} \widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Semantics

Static	$\vdash \{\phi\}\ s\ \{\phi\}$	$\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$
Dynamic	$\pi \longrightarrow \pi$	$\widetilde{\pi} \mathrel{\widetilde{\longrightarrow}} \widetilde{\pi}$
Formula	$\pi \models \phi$	$\widetilde{\pi} \mathrel{\widetilde{\models}} \widetilde{\phi}$

Soundness

Gradual Verification - Approach



Syntax

$s \in \textsc{Stmt}$

$\phi \in \textsc{Formula}$

Program State

$\pi \in \textsc{ProgramState}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$

Dynamic $\pi \longrightarrow \pi$

Formula $\pi \vDash \phi$

Soundness

syntax extension

syntax extension

extension

function lifting

Syntax

$\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Semantics

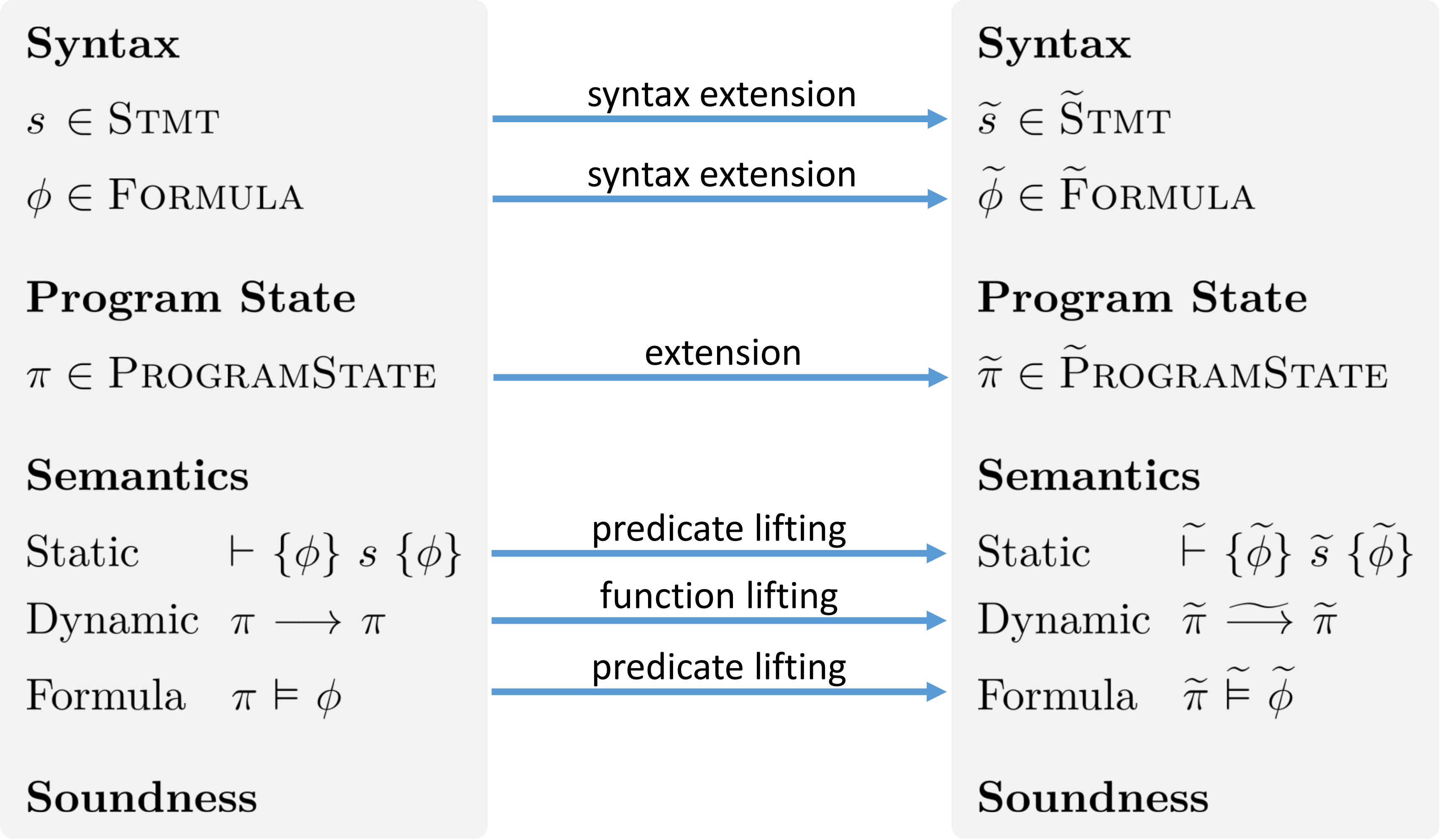
Static $\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$

Dynamic $\widetilde{\pi} \ \widetilde{\longrightarrow}\ \widetilde{\pi}$

Formula $\widetilde{\pi} \ \widetilde{\vDash}\ \widetilde{\phi}$

Soundness

Gradual Verification - Approach



Syntax | **Syntax**

$$s \in \textsc{Stmt} \xrightarrow{\text{syntax extension}} \widetilde{s} \in \widetilde{\textsc{Stmt}}$$

$$\phi \in \textsc{Formula} \xrightarrow{\text{syntax extension}} \widetilde{\phi} \in \widetilde{\textsc{Formula}}$$

Program State | **Program State**

$$\pi \in \textsc{ProgramState} \xrightarrow{\text{extension}} \widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$$

Semantics | **Semantics**

$$\text{Static} \quad \vdash \{\phi\}\ s\ \{\phi\} \xrightarrow{\text{predicate lifting}} \text{Static} \quad \widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$$

$$\text{Dynamic} \quad \pi \longrightarrow \pi \xrightarrow{\text{function lifting}} \text{Dynamic} \quad \widetilde{\pi}\ \widetilde{\longrightarrow}\ \widetilde{\pi}$$

$$\text{Formula} \quad \pi \vDash \phi \xrightarrow{\text{predicate lifting}} \text{Formula} \quad \widetilde{\pi}\ \widetilde{\vDash}\ \widetilde{\phi}$$

Soundness | **Soundness**

Predicate Lifting in a Nutshell

$$\begin{tikzcd}
\widetilde{P} \subseteq \widetilde{\textsc{Formula}} \times \widetilde{\textsc{Stmt}} \times \widetilde{\textsc{Formula}} \\
P \subseteq \textsc{Formula} \times \textsc{Stmt} \times \textsc{Formula} \arrow[u, "\text{lifting}"]
\end{tikzcd}$$

Predicate Lifting in a Nutshell

all gradually lifted predicates satisfy

$$\frac{\phi_1 \in \gamma(\widetilde{\phi_1}) \qquad \phi_2 \in \gamma(\widetilde{\phi_2}) \qquad \phi_3 \in \gamma(\widetilde{\phi_3}) \qquad P(\phi_1, \phi_2, \phi_3)}{\widetilde{P}(\widetilde{\phi_1}, \widetilde{\phi_2}, \widetilde{\phi_3})}$$

Predicate Lifting in a Nutshell

all gradually lifted predicates satisfy

$$\frac{\phi_1 \in \gamma(\widetilde{\phi_1}) \qquad \phi_2 \in \gamma(\widetilde{\phi_2}) \qquad \phi_3 \in \gamma(\widetilde{\phi_3}) \qquad P(\phi_1, \phi_2, \phi_3)}{\widetilde{P}(\widetilde{\phi_1}, \widetilde{\phi_2}, \widetilde{\phi_3})}$$

$$\frac{\phi \Rightarrow \phi_a}{\vdash \{\phi\}\ \texttt{assert}\ \phi_a\ \{\phi\}}\ \textsc{HAssert}$$

Predicate Lifting in a Nutshell

all gradually lifted predicates satisfy

$$\frac{\phi_1 \in \gamma(\widetilde{\phi_1}) \qquad \phi_2 \in \gamma(\widetilde{\phi_2}) \qquad \phi_3 \in \gamma(\widetilde{\phi_3}) \qquad P(\phi_1, \phi_2, \phi_3)}{\widetilde{P}(\widetilde{\phi_1}, \widetilde{\phi_2}, \widetilde{\phi_3})}$$

$$\frac{\phi \Rightarrow \phi_a}{\vdash \{\phi\}\ \texttt{assert}\ \ \phi_a\ \{\phi\}}\ \textsc{HAssert} \qquad P(\phi_1, \phi_a, \phi_2) = \phi_1 = \phi_2 \land \phi_1 \Rightarrow \phi_a$$

Predicate Lifting in a Nutshell

all gradually lifted predicates satisfy

$$\frac{\phi_1 \in \gamma(\widetilde{\phi_1}) \qquad \phi_2 \in \gamma(\widetilde{\phi_2}) \qquad \phi_3 \in \gamma(\widetilde{\phi_3}) \qquad P(\phi_1, \phi_2, \phi_3)}{\widetilde{P}(\widetilde{\phi_1}, \widetilde{\phi_2}, \widetilde{\phi_3})}$$

$$\frac{\phi \Rightarrow \phi_a}{\vdash \{\phi\}\ \texttt{assert}\ \ \phi_a\ \{\phi\}}\ \textsc{HAssert} \qquad P(\phi_1, \phi_a, \phi_2) = \phi_1 = \phi_2 \wedge \phi_1 \Rightarrow \phi_a$$

$$\vdash \{\texttt{(x = 3)} \wedge \texttt{(y = 4)}\}\ \texttt{assert (x = 3)}\ \{\texttt{(x = 3)} \wedge \texttt{(y = 4)}\}$$

Predicate Lifting in a Nutshell

all gradually lifted predicates satisfy

$$\frac{\phi_1 \in \gamma(\widetilde{\phi_1}) \qquad \phi_2 \in \gamma(\widetilde{\phi_2}) \qquad \phi_3 \in \gamma(\widetilde{\phi_3}) \qquad P(\phi_1, \phi_2, \phi_3)}{\widetilde{P}(\widetilde{\phi_1}, \widetilde{\phi_2}, \widetilde{\phi_3})}$$

$$\frac{\phi \Rightarrow \phi_a}{\vdash \{\phi\}\ \texttt{assert}\ \ \phi_a\ \{\phi\}}\ \textsc{HAssert} \qquad P(\phi_1, \phi_a, \phi_2) = \phi_1 = \phi_2 \land \phi_1 \Rightarrow \phi_a$$

$$\vdash \{\texttt{(x = 3)} \land \texttt{(y = 4)}\}\ \texttt{assert (x = 3)}\ \{\texttt{(x = 3)} \land \texttt{(y = 4)}\}$$

$$\widetilde{\vdash} \{\texttt{(x = 3)} \land \texttt{(y = 4)}\}\ \texttt{assert (x = 3)}\ \{\texttt{(x = 3)} \land \texttt{(y = 4)}\}$$

Predicate Lifting in a Nutshell

all gradually lifted predicates satisfy

$$\frac{\phi_1 \in \gamma(\widetilde{\phi_1}) \qquad \phi_2 \in \gamma(\widetilde{\phi_2}) \qquad \phi_3 \in \gamma(\widetilde{\phi_3}) \qquad P(\phi_1, \phi_2, \phi_3)}{\widetilde{P}(\widetilde{\phi_1}, \widetilde{\phi_2}, \widetilde{\phi_3})}$$

$$\frac{\phi \Rightarrow \phi_a}{\vdash \{\phi\}\ \texttt{assert}\ \ \phi_a\ \{\phi\}}\ \textsc{HAssert} \qquad P(\phi_1, \phi_a, \phi_2) = \phi_1 = \phi_2 \land \phi_1 \Rightarrow \phi_a$$

$$\vdash \{(\texttt{x = 3}) \land (\texttt{y = 4})\}\ \texttt{assert (x = 3)}\ \{(\texttt{x = 3}) \land (\texttt{y = 4})\}$$

$$\widetilde{\vdash} \{(\texttt{x = 3}) \land (\texttt{y = 4})\}\ \texttt{assert (x = 3)}\ \{(\texttt{x = 3}) \land (\texttt{y = 4})\}$$

$$\widetilde{\vdash} \{?\}\ \texttt{assert (x = 3)}\ \{(\texttt{x = 3}) \land (\texttt{y = 4})\}$$

Predicate Lifting in a Nutshell

all gradually lifted predicates satisfy

$$\dfrac{\phi_1 \in \gamma(\widetilde{\phi_1}) \qquad \phi_2 \in \gamma(\widetilde{\phi_2}) \qquad \phi_3 \in \gamma(\widetilde{\phi_3}) \qquad P(\phi_1, \phi_2, \phi_3)}{\widetilde{P}(\widetilde{\phi_1}, \widetilde{\phi_2}, \widetilde{\phi_3})}$$

$$\dfrac{\phi \Rightarrow \phi_a}{\vdash \{\phi\}\ \texttt{assert}\ \ \phi_a\ \{\phi\}}\ \textsc{HAssert} \qquad P(\phi_1, \phi_a, \phi_2) = \phi_1 = \phi_2 \land \phi_1 \Rightarrow \phi_a$$

$$\vdash \{\texttt{(x = 3)} \land \texttt{(y = 4)}\}\ \texttt{assert (x = 3)}\ \{\texttt{(x = 3)} \land \texttt{(y = 4)}\}$$

$$\widetilde{\vdash} \{\texttt{(x = 3)} \land \texttt{(y = 4)}\}\ \texttt{assert (x = 3)}\ \{\texttt{(x = 3)} \land \texttt{(y = 4)}\}$$

$$\widetilde{\vdash} \{\texttt{?}\}\ \texttt{assert (x = 3)}\ \{\texttt{(x = 3)} \land \texttt{(y = 4)}\}$$

$$\widetilde{\vdash} \{\texttt{(x = 3)} \land \texttt{(y = 4)}\}\ \texttt{assert ?}\ \{\texttt{(x = 3)} \land \texttt{(y = 4)}\}$$

Predicate Lifting in a Nutshell

all gradually lifted predicates satisfy

$$\frac{\phi_1 \in \gamma(\widetilde{\phi_1}) \qquad \phi_2 \in \gamma(\widetilde{\phi_2}) \qquad \phi_3 \in \gamma(\widetilde{\phi_3}) \qquad P(\phi_1, \phi_2, \phi_3)}{\widetilde{P}(\widetilde{\phi_1}, \widetilde{\phi_2}, \widetilde{\phi_3})}$$

$$\frac{\phi \Rightarrow \phi_a}{\vdash \{\phi\}\ \texttt{assert}\ \ \phi_a\ \{\phi\}}\ \textsc{HAssert} \qquad P(\phi_1, \phi_a, \phi_2) = \phi_1 = \phi_2 \wedge \phi_1 \Rightarrow \phi_a$$

$$\vdash \{\texttt{(x = 3)} \wedge \texttt{(y = 4)}\}\ \texttt{assert (x = 3)}\ \{\texttt{(x = 3)} \wedge \texttt{(y = 4)}\}$$

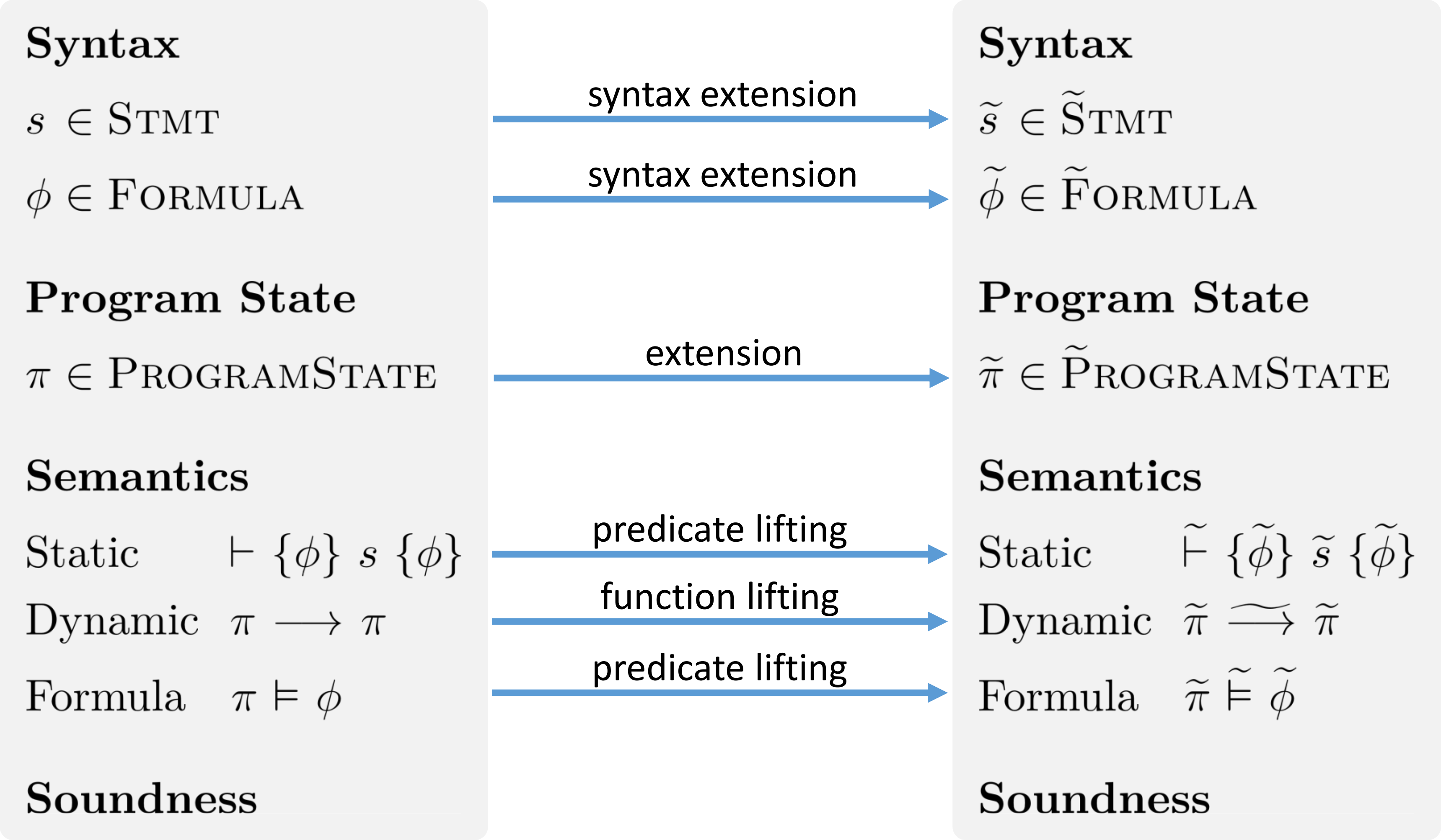
$$\widetilde{\vdash} \{\texttt{(x = 3)} \wedge \texttt{(y = 4)}\}\ \texttt{assert (x = 3)}\ \{\texttt{(x = 3)} \wedge \texttt{(y = 4)}\}$$

$$\widetilde{\vdash} \{\texttt{?}\}\ \texttt{assert (x = 3)}\ \{\texttt{(x = 3)} \wedge \texttt{(y = 4)}\}$$

$$\widetilde{\vdash} \{\texttt{(x = 3)} \wedge \texttt{(y = 4)}\}\ \texttt{assert ?}\ \{\texttt{(x = 3)} \wedge \texttt{(y = 4)}\}$$

$$\widetilde{\vdash} \{\texttt{(x = 3)} \wedge \texttt{(y = 4)}\}\ \texttt{assert (x = 3)}\ \{\texttt{?}\}$$

Gradual Verification - Approach



Syntax		**Syntax**
$s \in \textsc{Stmt}$	syntax extension →	$\widetilde{s} \in \widetilde{\textsc{Stmt}}$
$\phi \in \textsc{Formula}$	syntax extension →	$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$
Program State		**Program State**
$\pi \in \textsc{ProgramState}$	extension →	$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$
Semantics		**Semantics**
Static $\vdash \{\phi\}\ s\ \{\phi\}$	predicate lifting →	Static $\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$
Dynamic $\pi \longrightarrow \pi$	function lifting →	Dynamic $\widetilde{\pi}\ \widetilde{\longrightarrow}\ \widetilde{\pi}$
Formula $\pi \models \phi$	predicate lifting →	Formula $\widetilde{\pi}\ \widetilde{\models}\ \widetilde{\phi}$
Soundness		**Soundness**

Gradual Soundness

$$\frac{\vdash \{\phi\}\ s\ \{\phi'\}}{\vDash \{\phi\}\ s\ \{\phi'\}}\ \textsc{Soundness}$$

$$\vDash \{\phi\}\ s\ \{\phi'\}$$

$$\overset{\mathsf{def}}{\iff}$$

$$\forall \pi, \pi'.\ \pi \xrightarrow{s} \pi' \land \pi \vDash \phi \implies \pi' \vDash \phi'$$

Gradual Soundness

$$\frac{\vdash \{\phi\}\ s\ \{\phi'\}}{\vDash \{\phi\}\ s\ \{\phi'\}}\ \textsc{Soundness}$$

$$\vDash \{\phi\}\ s\ \{\phi'\}$$

$$\overset{\textsf{def}}{\iff}$$

$$\forall \pi, \pi'.\ \pi \xrightarrow{s} \pi' \land \pi \vDash \phi \implies \pi' \vDash \phi'$$

$$\frac{\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}{\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}\ \widetilde{\textsc{S}}\textsc{oundness}$$

$$\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}$$

$$\overset{\textsf{def}}{\iff}$$

$$\forall \widetilde{\pi}, \widetilde{\pi}'.\ \widetilde{\pi} \overset{\widetilde{s}}{\widetilde{\longrightarrow}} \widetilde{\pi}' \land \widetilde{\pi}\ \widetilde{\vDash}\ \widetilde{\phi} \implies \widetilde{\pi}'\ \widetilde{\vDash}\ \widetilde{\phi}'$$

Gradual Soundness

$$\dfrac{\vdash \{\phi\}\ s\ \{\phi'\}}{\vDash \{\phi\}\ s\ \{\phi'\}}\ \textsc{Soundness}$$

$$\vDash \{\phi\}\ s\ \{\phi'\}$$

$$\overset{\textsf{def}}{\iff}$$

$$\forall \pi, \pi'.\ \pi \xrightarrow{s} \pi' \land \pi \vDash \phi \implies \pi' \vDash \phi'$$

$$\dfrac{\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}{\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}\ \widetilde{\textsc{S}}\textsc{oundness}$$

$$\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}$$

$$\overset{\textsf{def}}{\iff}$$

$$\forall \widetilde{\pi}, \widetilde{\pi'}.\ \widetilde{\pi} \overset{\widetilde{s}}{\widetilde{\longrightarrow}} \widetilde{\pi'} \land \widetilde{\pi}\ \widetilde{\vDash}\ \widetilde{\phi} \implies \widetilde{\pi'}\ \widetilde{\vDash}\ \widetilde{\phi'}$$

$$\vdash \big\{(\texttt{x = 2})\big\}\ \texttt{y := 4}\ \big\{(\texttt{x = 2}) \land (\texttt{y = 4})\big\}$$

$$\vDash \big\{(\texttt{x = 2})\big\}\ \texttt{y := 4}\ \big\{(\texttt{x = 2}) \land (\texttt{y = 4})\big\}$$

Gradual Soundness

$$\dfrac{\vdash \{\phi\}\ s\ \{\phi'\}}{\vDash \{\phi\}\ s\ \{\phi'\}}\ \textsc{Soundness}$$

$$\vDash \{\phi\}\ s\ \{\phi'\}$$

$$\overset{\text{def}}{\iff}$$

$$\forall \pi, \pi'.\ \pi \xrightarrow{s} \pi' \wedge \pi \vDash \phi \implies \pi' \vDash \phi'$$

$$\dfrac{\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}{\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}\ \widetilde{\textsc{Soundness}}$$

$$\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}$$

$$\overset{\text{def}}{\iff}$$

$$\forall \widetilde{\pi}, \widetilde{\pi'}.\ \widetilde{\pi} \xrightarrow{\widetilde{s}} \widetilde{\pi'} \wedge \widetilde{\pi}\ \widetilde{\vDash}\ \widetilde{\phi} \implies \widetilde{\pi'}\ \widetilde{\vDash}\ \widetilde{\phi'}$$

$$\widetilde{\vdash} \{\texttt{(x = 2)}\}\ \texttt{y := 4}\ \{\texttt{(x = 2)} \wedge \texttt{(y = 4)}\}$$

$$\widetilde{\vDash} \{\texttt{(x = 2)}\}\ \texttt{y := 4}\ \{\texttt{(x = 2)} \wedge \texttt{(y = 4)}\}$$

Gradual Soundness

$$\frac{\vdash \{\phi\}\ s\ \{\phi'\}}{\vDash \{\phi\}\ s\ \{\phi'\}}\ \textsc{Soundness}$$

$$\vDash \{\phi\}\ s\ \{\phi'\}$$

$$\overset{\text{def}}{\iff}$$

$$\forall \pi, \pi'.\ \pi \xrightarrow{s} \pi' \land \pi \vDash \phi \implies \pi' \vDash \phi'$$

$$\frac{\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}{\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}\ \widetilde{\text{S}}\textsc{oundness}$$

$$\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}$$

$$\overset{\text{def}}{\iff}$$

$$\forall \widetilde{\pi}, \widetilde{\pi'}.\ \widetilde{\pi} \overset{\widetilde{s}}{\widetilde{\longrightarrow}} \widetilde{\pi'} \land \widetilde{\pi}\ \widetilde{\vDash}\ \widetilde{\phi} \implies \widetilde{\pi'}\ \widetilde{\vDash}\ \widetilde{\phi'}$$

$$\widetilde{\vdash} \{?\}\ \texttt{y := 4}\ \{(\texttt{x = 2}) \land (\texttt{y = 4})\}$$

$$\widetilde{\vDash} \{?\}\ \texttt{y := 4}\ \{(\texttt{x = 2}) \land (\texttt{y = 4})\}$$

Gradual Soundness

$$\dfrac{\vdash \{\phi\}\ s\ \{\phi'\}}{\vDash \{\phi\}\ s\ \{\phi'\}}\ \textsc{Soundness}$$

$$\vDash \{\phi\}\ s\ \{\phi'\}$$

$$\overset{\text{def}}{\iff}$$

$$\forall \pi, \pi'.\ \pi \xrightarrow{s} \pi' \wedge \pi \vDash \phi \implies \pi' \vDash \phi'$$

$$\dfrac{\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}{\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}\ \widetilde{\textsc{S}}\textsc{oundness}$$

$$\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}$$

$$\overset{\text{def}}{\iff}$$

$$\forall \widetilde{\pi}, \widetilde{\pi}'.\ \widetilde{\pi} \overset{\widetilde{s}}{\widetilde{\longrightarrow}} \widetilde{\pi}' \wedge \widetilde{\pi}\ \widetilde{\vDash}\ \widetilde{\phi} \implies \widetilde{\pi}'\ \widetilde{\vDash}\ \widetilde{\phi}'$$

$$\widetilde{\vdash} \{?\}\ \texttt{y := 4}\ \{(\texttt{x = 2}) \wedge (\texttt{y = 4})\}$$

$$\neg\ \widetilde{\vDash} \{?\}\ \texttt{y := 4}\ \{(\texttt{x = 2}) \wedge (\texttt{y = 4})\}$$

Gradual Soundness

$$\frac{\vdash \{\phi\}\ s\ \{\phi'\}}{\models \{\phi\}\ s\ \{\phi'\}}\ \textsc{Soundness}$$

$$\models \{\phi\}\ s\ \{\phi'\}$$

$$\overset{\mathsf{def}}{\iff}$$

$$\forall \pi, \pi'.\ \pi \xrightarrow{s} \pi' \land \pi \models \phi \implies \pi' \models \phi'$$

$$\frac{\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}'\}}{\widetilde{\models} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}'\}}\ \widetilde{\textsc{Soundness}}$$

$$\widetilde{\models} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}'\}$$

$$\overset{\mathsf{def}}{\iff}$$

$$\forall \widetilde{\pi}, \widetilde{\pi}'.\ \widetilde{\pi} \xrightarrow{\widetilde{s}} \widetilde{\pi}' \land \widetilde{\pi}\ \widetilde{\models}\ \widetilde{\phi} \implies \widetilde{\pi}'\ \widetilde{\models}\ \widetilde{\phi}'$$

$$\widetilde{\vdash} \{?\}\ \mathtt{y\ :=\ 4}\ \{\mathtt{(x\ =\ 2)} \land \mathtt{(y\ =\ 4)}\}$$

$$\neg\ \widetilde{\models} \{?\}\ \mathtt{y\ :=\ 4}\ \{\mathtt{(x\ =\ 2)} \land \mathtt{(y\ =\ 4)}\}$$

Gradual Soundness

$$\frac{\vdash \{\phi\}\ s\ \{\phi'\}}{\vDash \{\phi\}\ s\ \{\phi'\}}\ \textsc{Soundness}$$

$$\vDash \{\phi\}\ s\ \{\phi'\}$$

$$\overset{\text{def}}{\iff}$$

$$\forall \pi, \pi'.\ \pi \xrightarrow{s} \pi' \land \pi \vDash \phi \implies \pi' \vDash \phi'$$

$$\frac{\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}'\}}{\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}'\}}\ \widetilde{\textsc{Soundness}}$$

if it were true, we would have a gradual system without runtime checks

$$\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}'\}$$

$$\overset{\text{def}}{\iff}$$

$$\forall \widetilde{\pi}, \widetilde{\pi}'.\ \widetilde{\pi} \overset{\widetilde{s}}{\widetilde{\longrightarrow}} \widetilde{\pi}' \land \widetilde{\pi}\ \widetilde{\vDash}\ \widetilde{\phi} \implies \widetilde{\pi}'\ \widetilde{\vDash}\ \widetilde{\phi}'$$

$$\widetilde{\vdash} \{?\}\ \texttt{y := 4}\ \{(\texttt{x = 2}) \land (\texttt{y = 4})\}$$

$$\neg\ \widetilde{\vDash} \{?\}\ \texttt{y := 4}\ \{(\texttt{x = 2}) \land (\texttt{y = 4})\}$$

Gradual Soundness

$$\frac{\vdash \{\phi\}\ s\ \{\phi'\}}{\vDash \{\phi\}\ s\ \{\phi'\}}\ \textsc{Soundness}$$

$$\vDash \{\phi\}\ s\ \{\phi'\}$$

$$\overset{\text{def}}{\iff}$$

$$\forall \pi, \pi'.\ \pi \xrightarrow{s} \pi' \land \pi \vDash \phi \implies \pi' \vDash \phi'$$

$$\frac{\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}'\}}{\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}'\}}\ \widetilde{\textsc{Soundness}}$$

$$\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}'\}$$

$$\overset{\text{def}}{\iff}$$

$$\forall \widetilde{\pi}, \widetilde{\pi}'.\ \widetilde{\pi} \xrightarrow{\widetilde{s}} \widetilde{\pi}' \land \widetilde{\pi}\ \widetilde{\vDash}\ \widetilde{\phi} \implies \widetilde{\pi}'\ \widetilde{\vDash}\ \widetilde{\phi}'$$

$$\widetilde{\vdash} \{?\}\ \texttt{y := 4}\ \{(\texttt{x = 2}) \land (\texttt{y = 4})\}$$

$$\widetilde{\vDash} \{?\}\ \texttt{y := 4; assert (x = 2)}\ \{(\texttt{x = 2}) \land (\texttt{y = 4})\}$$

Gradual Soundness

$$\frac{\vdash \{\phi\}\ s\ \{\phi'\}}{\vDash \{\phi\}\ s\ \{\phi'\}}\ \textsc{Soundness}$$

$$\vDash \{\phi\}\ s\ \{\phi'\}$$

$$\overset{\text{def}}{\iff}$$

$$\forall \pi, \pi'.\ \pi \xrightarrow{s} \pi' \land \pi \vDash \phi \implies \pi' \vDash \phi'$$

$$\frac{\widetilde{\vdash}\ \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}{\widetilde{\vDash}\ \{\widetilde{\phi}\}\ \widetilde{s};\ \texttt{assert}\ \widetilde{\phi}'\ \{\widetilde{\phi'}\}}\ \widetilde{\text{S}}\textsc{oundness}$$

$$\widetilde{\vDash}\ \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}$$

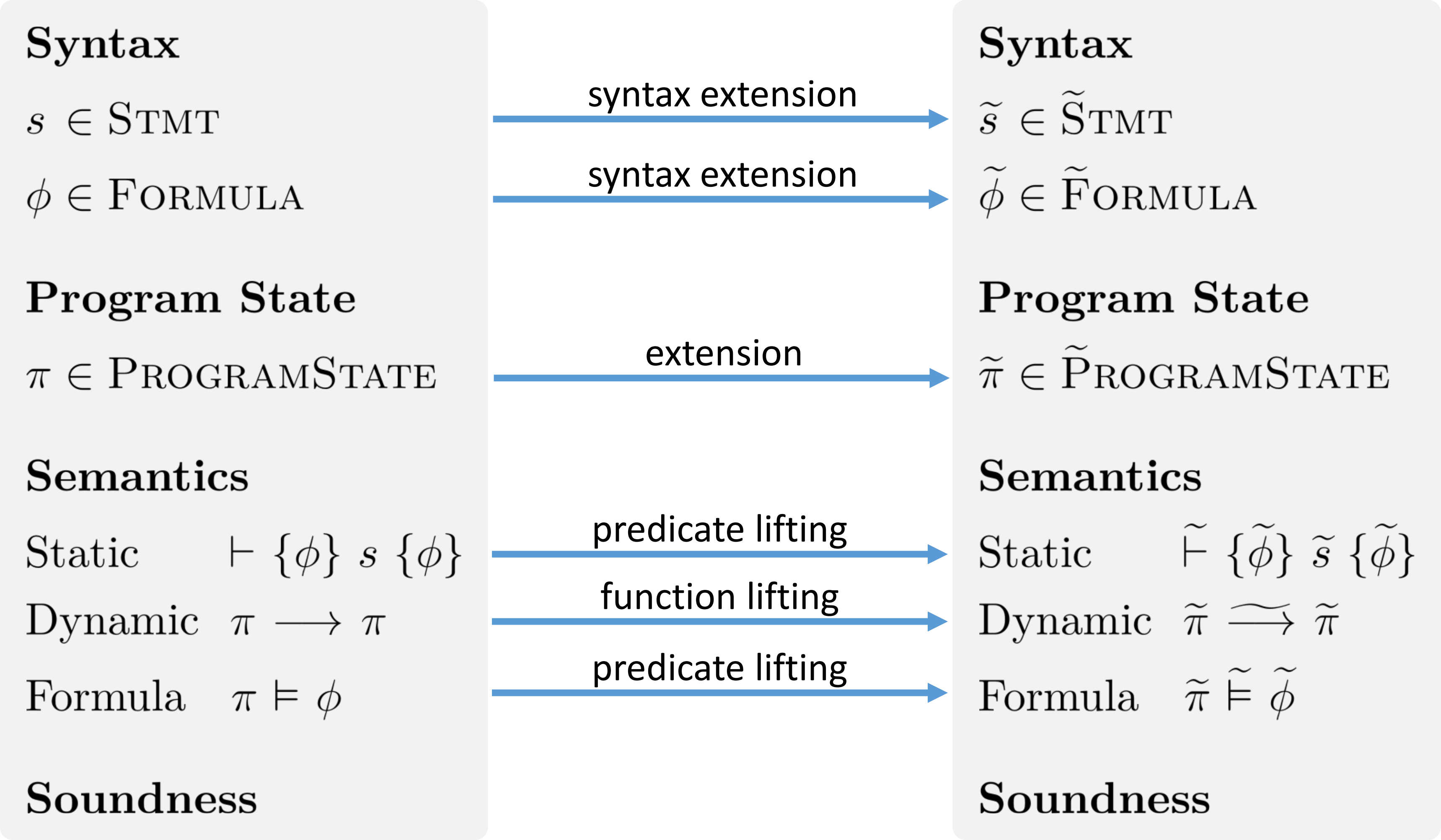
$$\overset{\text{def}}{\iff}$$

$$\forall \widetilde{\pi}, \widetilde{\pi}'.\ \widetilde{\pi} \overset{\widetilde{s}}{\widetilde{\longrightarrow}} \widetilde{\pi}' \land \widetilde{\pi}\ \widetilde{\vDash}\ \widetilde{\phi} \implies \widetilde{\pi}'\ \widetilde{\vDash}\ \widetilde{\phi}'$$

$$\widetilde{\vdash}\ \{?\}\ \texttt{y := 4}\ \{\texttt{(x = 2)} \land \texttt{(y = 4)}\}$$

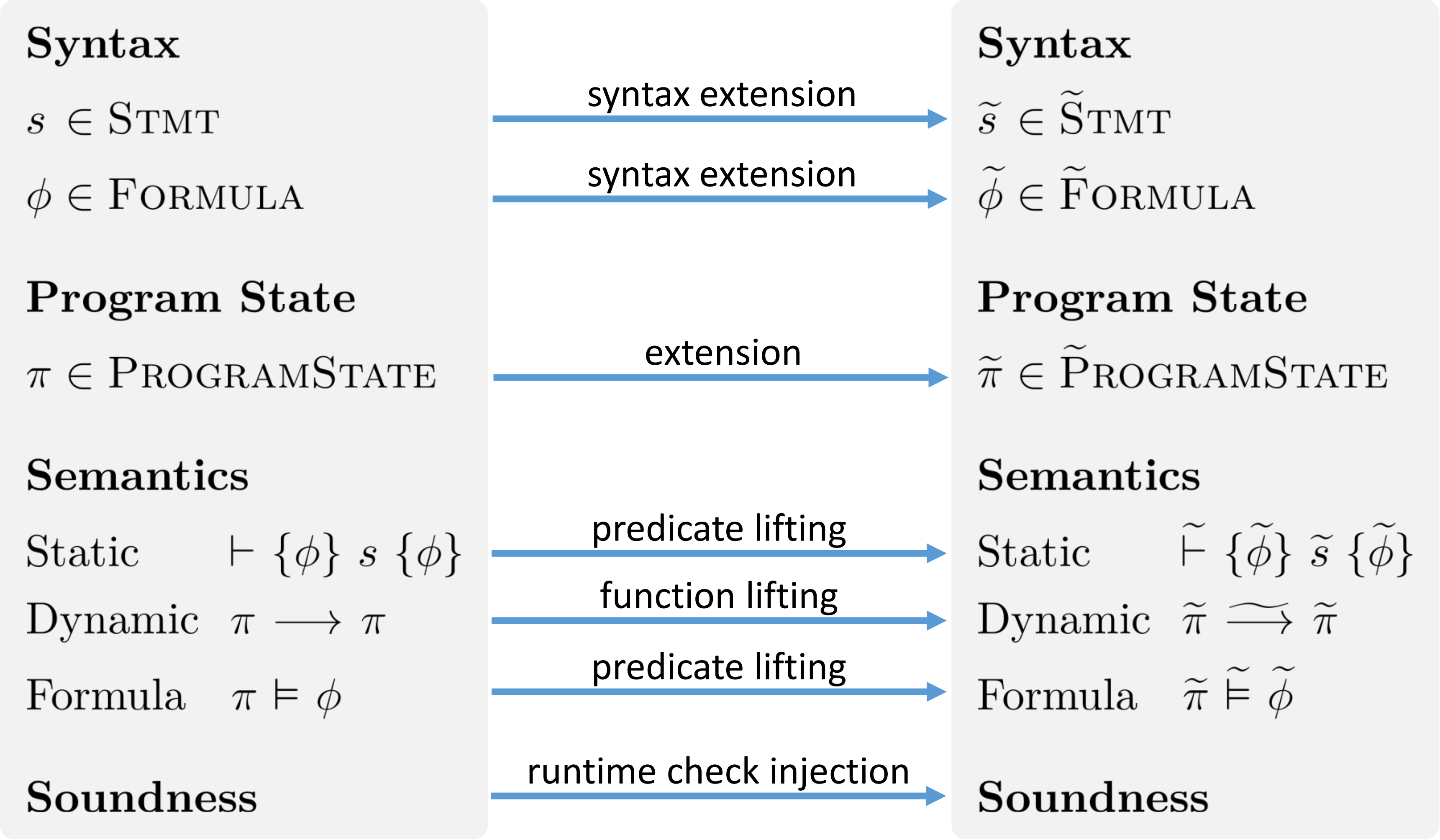
$$\widetilde{\vDash}\ \{?\}\ \texttt{y := 4; assert (x = 2)}\ \{\texttt{(x = 2)} \land \texttt{(y = 4)}\}$$

Gradual Verification - Approach



Syntax		**Syntax**
$s \in \textsc{Stmt}$	$\xrightarrow{\text{syntax extension}}$	$\widetilde{s} \in \widetilde{\textsc{Stmt}}$
$\phi \in \textsc{Formula}$	$\xrightarrow{\text{syntax extension}}$	$\widetilde{\phi} \in \widetilde{\textsc{Formula}}$
Program State		**Program State**
$\pi \in \textsc{ProgramState}$	$\xrightarrow{\text{extension}}$	$\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$
Semantics		**Semantics**
Static $\vdash \{\phi\}\ s\ \{\phi\}$	$\xrightarrow{\text{predicate lifting}}$	Static $\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$
Dynamic $\pi \longrightarrow \pi$	$\xrightarrow{\text{function lifting}}$	Dynamic $\widetilde{\pi}\ \widetilde{\longrightarrow}\ \widetilde{\pi}$
Formula $\pi \vDash \phi$	$\xrightarrow{\text{predicate lifting}}$	Formula $\widetilde{\pi}\ \widetilde{\vDash}\ \widetilde{\phi}$
Soundness		**Soundness**

Gradual Verification - Approach



Syntax

$s \in \textsc{Stmt}$ — syntax extension → $\widetilde{s} \in \widetilde{\textsc{Stmt}}$

$\phi \in \textsc{Formula}$ — syntax extension → $\widetilde{\phi} \in \widetilde{\textsc{Formula}}$

Program State

$\pi \in \textsc{ProgramState}$ — extension → $\widetilde{\pi} \in \widetilde{\textsc{ProgramState}}$

Semantics

Static $\vdash \{\phi\}\ s\ \{\phi\}$ — predicate lifting → Static $\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}\}$

Dynamic $\pi \longrightarrow \pi$ — function lifting → Dynamic $\widetilde{\pi} \mathrel{\widetilde{\longrightarrow}} \widetilde{\pi}$

Formula $\pi \vDash \phi$ — predicate lifting → Formula $\widetilde{\pi} \mathrel{\widetilde{\vDash}} \widetilde{\phi}$

Soundness — runtime check injection → **Soundness**

Gradual Verification – Put to the Test

$$\widetilde{\vdash} \{?\}\ \texttt{y := 2; x := 3}\ \{(\texttt{x = 3}) \wedge (\texttt{y = 2})\}$$

Gradual Verification – Put to the Test

$$\dfrac{\begin{array}{c}\widetilde{\phi_1} \mathrel{\widetilde{\Rightarrow}} \widetilde{\phi_2} \\ \widetilde{\vdash} \{?\}\ \texttt{y := 2}\ \{\widetilde{\phi_1}\} \qquad \widetilde{\vdash} \{\widetilde{\phi_2}\}\ \texttt{x := 3}\ \{(\texttt{x = 3}) \land (\texttt{y = 2})\}\end{array}}{\widetilde{\vdash} \{?\}\ \texttt{y := 2; x := 3}\ \{(\texttt{x = 3}) \land (\texttt{y = 2})\}}\ \widetilde{\text{HS}}\textsc{eq}$$

Gradual Verification – Put to the Test

$$\frac{\begin{array}{c} \widetilde{\phi_1} \mathrel{\widetilde{\Rightarrow}} \widetilde{\phi_2} \\ \widetilde{\vdash} \{?\}\ \texttt{y := 2}\ \{\widetilde{\phi_1}\} \qquad \widetilde{\vdash} \{\widetilde{\phi_2}\}\ \texttt{x := 3}\ \{(\texttt{x = 3}) \wedge (\texttt{y = 2})\} \end{array}}{\widetilde{\vdash} \{?\}\ \texttt{y := 2; x := 3}\ \{(\texttt{x = 3}) \wedge (\texttt{y = 2})\}}\ \widetilde{\mathrm{H}}\mathrm{S}\small{\mathrm{EQ}}$$

$$\widetilde{\phi_1} = (\texttt{y = 2})$$

$$\widetilde{\phi_2} = (\texttt{y = 2})$$

Gradual Verification – Put to the Test

$$\frac{\widetilde{\vdash} \{?\}\ \mathtt{y\ :=\ 2}\ \{\widetilde{\phi_1}\} \qquad \overset{\widetilde{\phi_1} \mathrel{\widetilde{\Rightarrow}} \widetilde{\phi_2}}{} \qquad \widetilde{\vdash} \{\widetilde{\phi_2}\}\ \mathtt{x\ :=\ 3}\ \{(\mathtt{x\ =\ 3}) \wedge (\mathtt{y\ =\ 2})\}}{\widetilde{\vdash} \{?\}\ \mathtt{y\ :=\ 2;\ x\ :=\ 3}\ \{(\mathtt{x\ =\ 3}) \wedge (\mathtt{y\ =\ 2})\}}\ \widetilde{\text{HS}}\text{EQ}$$

a) $\widetilde{\phi_1} = (\mathtt{y\ =\ 2})$

$\widetilde{\phi_2} = (\mathtt{y\ =\ 2})$

b) $\widetilde{\phi_1} = ?$

$\widetilde{\phi_2} = ?$

Gradual Verification – Put to the Test

$$\frac{\widetilde{\vdash} \{?\}\ \texttt{y := 2}\ \{\widetilde{\phi_1}\} \qquad \widetilde{\phi_1} \mathrel{\widetilde{\Rightarrow}} \widetilde{\phi_2} \qquad \widetilde{\vdash} \{\widetilde{\phi_2}\}\ \texttt{x := 3}\ \{(\texttt{x = 3}) \land (\texttt{y = 2})\}}{\widetilde{\vdash} \{?\}\ \texttt{y := 2; x := 3}\ \{(\texttt{x = 3}) \land (\texttt{y = 2})\}}\ \widetilde{\text{HSEQ}}$$

a) $\widetilde{\phi_1} = (\texttt{y = 2})$

$\widetilde{\phi_2} = (\texttt{y = 2})$

"good" (information carried over)

b) $\widetilde{\phi_1} = ?$

$\widetilde{\phi_2} = ?$

Gradual Verification – Put to the Test

$$\dfrac{\begin{array}{c}\widetilde{\phi_1} \mathrel{\widetilde{\Rightarrow}} \widetilde{\phi_2} \\ \widetilde{\vdash} \{?\}\ \texttt{y := 2}\ \{\widetilde{\phi_1}\} \qquad \widetilde{\vdash} \{\widetilde{\phi_2}\}\ \texttt{x := 3}\ \{(\texttt{x = 3}) \land (\texttt{y = 2})\}\end{array}}{\widetilde{\vdash} \{?\}\ \texttt{y := 2; x := 3}\ \{(\texttt{x = 3}) \land (\texttt{y = 2})\}}\ \widetilde{\text{H}}\text{S}\small{\text{EQ}}$$

a) $\widetilde{\phi_1} = (\texttt{y = 2})$

$\widetilde{\phi_2} = (\texttt{y = 2})$

“good” (information carried over)

b) $\widetilde{\phi_1} = ?$

$\widetilde{\phi_2} = ?$

“too weak” (could prove invalid triples)
idea: try to be as precise as possible

Gradual Verification – Put to the Test

$$\frac{\begin{array}{c} \widetilde{\phi_1} \mathrel{\widetilde{\Rightarrow}} \widetilde{\phi_2} \\ \widetilde{\vdash} \{?\}\ \texttt{y := 2}\ \{\widetilde{\phi_1}\} \qquad \widetilde{\vdash} \{\widetilde{\phi_2}\}\ \texttt{x := 3}\ \{\texttt{(x = 3)} \wedge \texttt{(y = 2)}\} \end{array}}{\widetilde{\vdash} \{?\}\ \texttt{y := 2; x := 3}\ \{\texttt{(x = 3)} \wedge \texttt{(y = 2)}\}}\ \widetilde{\text{HSEQ}}$$

a) $\widetilde{\phi_1} = \texttt{(y = 2)}$

$\widetilde{\phi_2} = \texttt{(y = 2)}$

"good" (information carried over)

b) $\widetilde{\phi_1} = ?$

$\widetilde{\phi_2} = ?$

"too weak" (could prove invalid triples)
idea: try to be as precise as possible

c) $\widetilde{\phi_1} = \texttt{(y = 2)} \wedge \texttt{(x = 4)}$

$\widetilde{\phi_2} = \texttt{(y = 2)}$

Gradual Verification – Put to the Test

$$\frac{\begin{array}{c} \widetilde{\phi_1} \mathrel{\widetilde{\Rightarrow}} \widetilde{\phi_2} \\ \widetilde{\vdash} \{?\}\ \texttt{y := 2}\ \{\widetilde{\phi_1}\} \qquad \widetilde{\vdash} \{\widetilde{\phi_2}\}\ \texttt{x := 3}\ \{(\texttt{x = 3}) \land (\texttt{y = 2})\} \end{array}}{\widetilde{\vdash} \{?\}\ \texttt{y := 2; x := 3}\ \{(\texttt{x = 3}) \land (\texttt{y = 2})\}}\ \widetilde{\textsc{HSeq}}$$

a) $\widetilde{\phi_1} = (\texttt{y = 2})$

$\widetilde{\phi_2} = (\texttt{y = 2})$

“good” (information carried over)

b) $\widetilde{\phi_1} = ?$

$\widetilde{\phi_2} = ?$

“too weak” (could prove invalid triples)
idea: try to be as precise as possible

c) $\widetilde{\phi_1} = (\texttt{y = 2}) \land (\texttt{x = 4})$

$\widetilde{\phi_2} = (\texttt{y = 2})$

“too strict” (Hoare triple invalid)
idea: try to produce valid Hoare triple

Gradual Verification – Put to the Test

$$\frac{\begin{array}{c} \widetilde{\phi_1} \mathrel{\widetilde{\Rightarrow}} \widetilde{\phi_2} \\ \widetilde{\vdash} \{?\}\ \texttt{y := 2}\ \{\widetilde{\phi_1}\} \qquad \widetilde{\vdash} \{\widetilde{\phi_2}\}\ \texttt{x := 3}\ \{(\texttt{x = 3}) \wedge (\texttt{y = 2})\} \end{array}}{\widetilde{\vdash} \{?\}\ \texttt{y := 2; x := 3}\ \{(\texttt{x = 3}) \wedge (\texttt{y = 2})\}}\ \widetilde{\text{HS}}\text{EQ}$$

a) $\widetilde{\phi_1} = (\texttt{y = 2})$

$\widetilde{\phi_2} = (\texttt{y = 2})$

“good” (information carried over)

b) $\widetilde{\phi_1} = ?$

$\widetilde{\phi_2} = ?$

“too weak” (could prove invalid triples)
idea: try to be as precise as possible

c) $\widetilde{\phi_1} = (\texttt{y = 2}) \wedge (\texttt{x = 4})$

$\widetilde{\phi_2} = (\texttt{y = 2})$

“too strict” (Hoare triple invalid)
~~idea: try to produce valid Hoare triple~~
if we could decide that we would not be here

Deterministic Lifting

don’t even present the gradual verifier with choices in the first place

Deterministic Lifting

don’t even present the gradual verifier with choices in the first place

- treat static Hoare logic as (multivalued) function

$$\begin{tikzcd}
\vdash \{\cdot\} \cdot \{\cdot\} \subseteq \textsc{Formula} \times \textsc{Stmt} \times \textsc{Formula} \arrow[d, bend right, leftrightarrow] \\
\vdash \{\cdot\} \cdot \{\cdot\} \ : \ \textsc{Formula} \times \textsc{Stmt} \to \mathcal{P}^{\textsc{Formula}}
\end{tikzcd}$$

Deterministic Lifting

don’t even present the gradual verifier with choices in the first place

- treat static Hoare logic as (multivalued) function

$$\begin{tikzcd} \vdash \{\cdot\} \cdot \{\cdot\} \subseteq \textsc{Formula} \times \textsc{Stmt} \times \textsc{Formula} \arrow[d, leftrightarrow, bend right] \\ \vdash \{\cdot\} \cdot \{\cdot\} \ : \ \textsc{Formula} \times \textsc{Stmt} \to \mathcal{P}^{\textsc{Formula}} \end{tikzcd}$$

- …and then lift that function

$$\vec{\vdash} \{\cdot\} \cdot \{\cdot\} \ : \ \tilde{\mathrm{F}}\textsc{ormula} \times \tilde{\mathrm{S}}\textsc{tmt} \to \tilde{\mathrm{F}}\textsc{ormula}$$

Deterministic Lifting

don’t even present the gradual verifier with choices in the first place

- treat static Hoare logic as (multivalued) function

$$\begin{tikzcd} \vdash \{\cdot\} \cdot \{\cdot\} \subseteq \textsc{Formula} \times \textsc{Stmt} \times \textsc{Formula} \arrow[d, bend right, leftrightarrow] \\ \vdash \{\cdot\} \cdot \{\cdot\} \;:\; \textsc{Formula} \times \textsc{Stmt} \rightarrow \mathcal{P}^{\textsc{Formula}} \end{tikzcd}$$

- …and then lift that function

$$\overset{\rightarrow}{\vdash} \{\cdot\} \cdot \{\cdot\} \;:\; \widetilde{\textsc{Formula}} \times \widetilde{\textsc{Stmt}} \rightarrow \widetilde{\textsc{Formula}}$$

- Properties
  - can derive gradual lifting (that’s still what the verifier needs)
  - deterministic verifier
  - stronger, assertion-free notion of soundness

Deterministic Lifting – $\textsc{HSeq}$

$$\frac{\phi_{q1} \Rightarrow \phi_{q2} \qquad \vdash \{\phi_p\}\ s_1\ \{\phi_{q1}\} \qquad \vdash \{\phi_{q2}\}\ s_2\ \{\phi_r\}}{\vdash \{\phi_p\}\ s_1;\ \ s_2\ \{\phi_r\}}\ \textsc{HSeq}$$

Deterministic Lifting – $\mathrm{H}\textsc{Seq}$

$$\frac{\begin{array}{c}\phi_{q1} \Rightarrow \phi_{q2} \\ \vdash \{\phi_p\}\ s_1\ \{\phi_{q1}\} \qquad \vdash \{\phi_{q2}\}\ s_2\ \{\phi_r\}\end{array}}{\vdash \{\phi_p\}\ s_1;\ \ s_2\ \{\phi_r\}}\ \mathrm{H}\textsc{Seq}$$

$$\frac{\begin{array}{c}\widetilde{\phi_{q1}} \vec{\Rightarrow} \widetilde{\phi_{q2}} \\ \vec{\vdash} \{\widetilde{\phi_p}\}\ \widetilde{s_1}\ \{\widetilde{\phi_{q1}}\} \qquad \vec{\vdash} \{\widetilde{\phi_{q2}}\}\ \widetilde{s_2}\ \{\widetilde{\phi_r}\}\end{array}}{\vec{\vdash} \{\widetilde{\phi_p}\}\ \widetilde{s_1};\ \ \widetilde{s_2}\ \{\widetilde{\phi_r}\}}\ \vec{\mathrm{H}}\textsc{Seq}$$

Deterministic Lifting – $\mathrm{H}\textsc{Seq}$

$$\frac{\begin{array}{c}\phi_{q1} \Rightarrow \phi_{q2} \\ \vdash \{\phi_p\}\ s_1\ \{\phi_{q1}\} \qquad \vdash \{\phi_{q2}\}\ s_2\ \{\phi_r\}\end{array}}{\vdash \{\phi_p\}\ s_1;\ s_2\ \{\phi_r\}}\ \mathrm{H}\textsc{Seq}$$

$$\frac{\vec{\vdash}\ \{\widetilde{\phi_p}\}\ \widetilde{s_1}\ \{\widetilde{\phi_q}\} \qquad \vec{\vdash}\ \{\widetilde{\phi_q}\}\ \widetilde{s_2}\ \{\widetilde{\phi_r}\}}{\vec{\vdash}\ \{\widetilde{\phi_p}\}\ \widetilde{s_1};\ \widetilde{s_2}\ \{\widetilde{\phi_r}\}}\ \vec{\mathrm{H}}\textsc{Seq}$$

Deterministic Lifting – $\mathrm{H}\textsc{Seq}$

$$\frac{\begin{array}{c}\phi_{q1} \Rightarrow \phi_{q2} \\ \vdash \{\phi_p\}\ s_1\ \{\phi_{q1}\} \qquad \vdash \{\phi_{q2}\}\ s_2\ \{\phi_r\}\end{array}}{\vdash \{\phi_p\}\ s_1;\ \ s_2\ \{\phi_r\}}\ \mathrm{H}\textsc{Seq}$$

$$\frac{\vec{\vdash}\ \{\widetilde{\phi_p}\}\ \widetilde{s_1}\ \{\widetilde{\phi_q}\} \qquad \vec{\vdash}\ \{\widetilde{\phi_q}\}\ \widetilde{s_2}\ \{\widetilde{\phi_r}\}}{\vec{\vdash}\ \{\widetilde{\phi_p}\}\ \widetilde{s_1};\ \ \widetilde{s_2}\ \{\widetilde{\phi_r}\}}\ \vec{\mathrm{H}}\textsc{Seq}$$

$$\frac{\begin{array}{c}\vec{\vdash}\ \{\texttt{true}\}\ \texttt{y := 2}\ \{(\texttt{y = 2})\} \\ \vec{\vdash}\ \{(\texttt{y = 2})\}\ \texttt{x := 3}\ \{(\texttt{x = 3})\ \land\ (\texttt{y = 2})\}\end{array}}{\vec{\vdash}\ \{\texttt{true}\}\ \texttt{y := 2; x := 3}\ \{(\texttt{x = 3})\ \land\ (\texttt{y = 2})\}}\ \vec{\mathrm{H}}\textsc{Seq}$$

Deterministic Lifting – $\textsc{HSeq}$

$$\dfrac{\begin{array}{c}\phi_{q1} \Rightarrow \phi_{q2} \\ \vdash \{\phi_p\}\ s_1\ \{\phi_{q1}\} \qquad \vdash \{\phi_{q2}\}\ s_2\ \{\phi_r\}\end{array}}{\vdash \{\phi_p\}\ s_1;\ s_2\ \{\phi_r\}}\ \textsc{HSeq}$$

$$\dfrac{\vec{\vdash} \{\widetilde{\phi_p}\}\ \widetilde{s}_1\ \{\widetilde{\phi_q}\} \qquad \vec{\vdash} \{\widetilde{\phi_q}\}\ \widetilde{s}_2\ \{\widetilde{\phi_r}\}}{\vec{\vdash} \{\widetilde{\phi_p}\}\ \widetilde{s}_1;\ \widetilde{s}_2\ \{\widetilde{\phi_r}\}}\ \vec{\text{H}}\textsc{Seq}$$

$$\neg \vec{\vdash} \{\texttt{true}\}\ \texttt{y := 4; x := 3}\ \{(\texttt{x = 3}) \land (\texttt{y = 2})\}$$

Deterministic Lifting

Obtaining a Gradual Lifting

$$\begin{tikzcd}
\vdash \{\cdot\} \cdot \{\cdot\} \subseteq \textsc{Formula} \times \textsc{Stmt} \times \textsc{Formula} \arrow[d, "\text{1. obtain deterministic lifting}"] \\
\vec{\vdash} \{\cdot\} \cdot \{\cdot\} \ : \ \widetilde{\textsc{Formula}} \times \widetilde{\textsc{Stmt}} \to \widetilde{\textsc{Formula}} \arrow[d, "\text{2. derive gradual lifting}"] \\
\widetilde{\vdash} \{\cdot\} \cdot \{\cdot\} \ : \ \widetilde{\textsc{Formula}} \times \widetilde{\textsc{Stmt}} \times \widetilde{\textsc{Formula}}
\end{tikzcd}$$

$$\widetilde{\vdash} \{\widetilde{\phi_1}\} \ \widetilde{s} \ \{\widetilde{\phi_2}\} \quad \overset{\text{def}}{\Longleftrightarrow} \quad \exists \widetilde{\phi_2'}. \ \vec{\vdash} \{\widetilde{\phi_1}\} \ \widetilde{s} \ \{\widetilde{\phi_2'}\} \land \widetilde{\phi_2'} \mathrel{\widetilde{\Rightarrow}} \widetilde{\phi_2}$$

Lemma: $\widetilde{\vdash} \{\cdot\} \cdot \{\cdot\}$ is a gradual lifting of $\vdash \{\cdot\} \cdot \{\cdot\}$

Deterministic Lifting

Soundness

$$\frac{\widetilde{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi}'\}}{\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s};\ \texttt{assert}\ \widetilde{\phi}'\ \{\widetilde{\phi}'\}}\ \widetilde{\textsc{Soundness}}$$

Deterministic Lifting

Soundness

$$\frac{\vec{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}{\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}\ \vec{\mathrm{S}}\text{OUNDNESS}$$

Deterministic Lifting

Soundness

$$\frac{\vec{\vdash} \; \{\widetilde{\phi}\} \; \widetilde{s} \; \{\widetilde{\phi'}\}}{\widetilde{\vDash} \; \{\widetilde{\phi}\} \; \widetilde{s} \; \{\widetilde{\phi'}\}} \; \vec{\mathrm{S}}\textsc{oundness}$$

- very helpful for optimizations

Deterministic Lifting

Soundness

$$\frac{\vec{\vdash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}{\widetilde{\vDash} \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}\ \vec{\text{S}}\text{OUNDNESS}$$

- very helpful for optimizations

```
y := 2
x := 3
assert (y = 2) ∧ (a = 5);
```

Deterministic Lifting

Soundness

$$\frac{\vec{\vdash}\ \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}{\widetilde{\vDash}\ \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}\ \vec{\text{S}}\text{OUNDNESS}$$

- very helpful for optimizations

```
y := 2
x := 3
assert (y = 2) ∧ (a = 5);
```

$$\vec{\vdash}\ \{\texttt{true}\}\ \texttt{y := 2; x := 3}\ \{(\texttt{x = 3}) \land (\texttt{y = 2})\}$$

Deterministic Lifting

Soundness

$$\frac{\vec{\vdash}\ \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}{\widetilde{\vDash}\ \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}\ \vec{\text{S}}\text{OUNDNESS}$$

- very helpful for optimizations

```
y := 2
x := 3
assert ~~(y = 2) ∧~~ (a = 5);
```

$\vec{\vdash}\ \{\texttt{true}\}\ \texttt{y := 2; x := 3}\ \{(\texttt{x = 3}) \land (\texttt{y = 2})\}$

Deterministic Lifting

Soundness

$$\dfrac{\vec{\vdash} \; \{\widetilde{\phi}\} \; \widetilde{s} \; \{\widetilde{\phi'}\}}{\widetilde{\vDash} \; \{\widetilde{\phi}\} \; \widetilde{s} \; \{\widetilde{\phi'}\}} \; \vec{\mathrm{S}}\textsc{oundness}$$

- very helpful for optimizations

Deterministic Lifting

Soundness

$$\frac{\vec{\vdash}\ \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}{\widetilde{\vDash}\ \{\widetilde{\phi}\}\ \widetilde{s}\ \{\widetilde{\phi'}\}}\ \vec{\text{S}}\textsc{oundness}$$

- very helpful for optimizations
- in the thesis:
    - criteria for gradual system to satisfy $\vec{\text{S}}\textsc{oundness}$
    - criteria for a system with zero runtime overhead when all annotations are static

Implicit Dynamic Frames

$\{(\texttt{p1.age = 19}) \land (\texttt{p2.age = 19})\}$

```
p1.age++
```

$\{(\texttt{p1.age = 20}) \land (\texttt{p2.age = 19})\}$

Implicit Dynamic Frames

$\{(\texttt{p1.age = 19}) \land (\texttt{p2.age = 19})\}$

```
p1.age++
```

$\{(\texttt{p1.age = 20}) \land (\texttt{p2.age = 19})\}$

$\{\texttt{acc(p1.age)} * \texttt{acc(p2.age)} * (\texttt{p1.age = 19}) * (\texttt{p2.age = 19})\}$

```
p1.age++
```

$\{\texttt{acc(p1.age)} * \texttt{acc(p2.age)} * (\texttt{p1.age = 20}) * (\texttt{p2.age = 19})\}$

Implicit Dynamic Frames

$$\{(\mathtt{p1.age} = 19) \land (\mathtt{p2.age} = 19)\}$$

$$\mathtt{p1.age{+}{+}}$$

$$\{(\mathtt{p1.age} = 20) \land (\mathtt{p2.age} = 19)\}$$

$$\{\mathtt{acc}(\mathtt{p1.age}) * \mathtt{acc}(\mathtt{p2.age}) * (\mathtt{p1.age} = 19) * (\mathtt{p2.age} = 19)\}$$

$$\mathtt{p1.age{+}{+}}$$

$$\{\mathtt{acc}(\mathtt{p1.age}) * \mathtt{acc}(\mathtt{p2.age}) * (\mathtt{p1.age} = 20) * (\mathtt{p2.age} = 19)\}$$

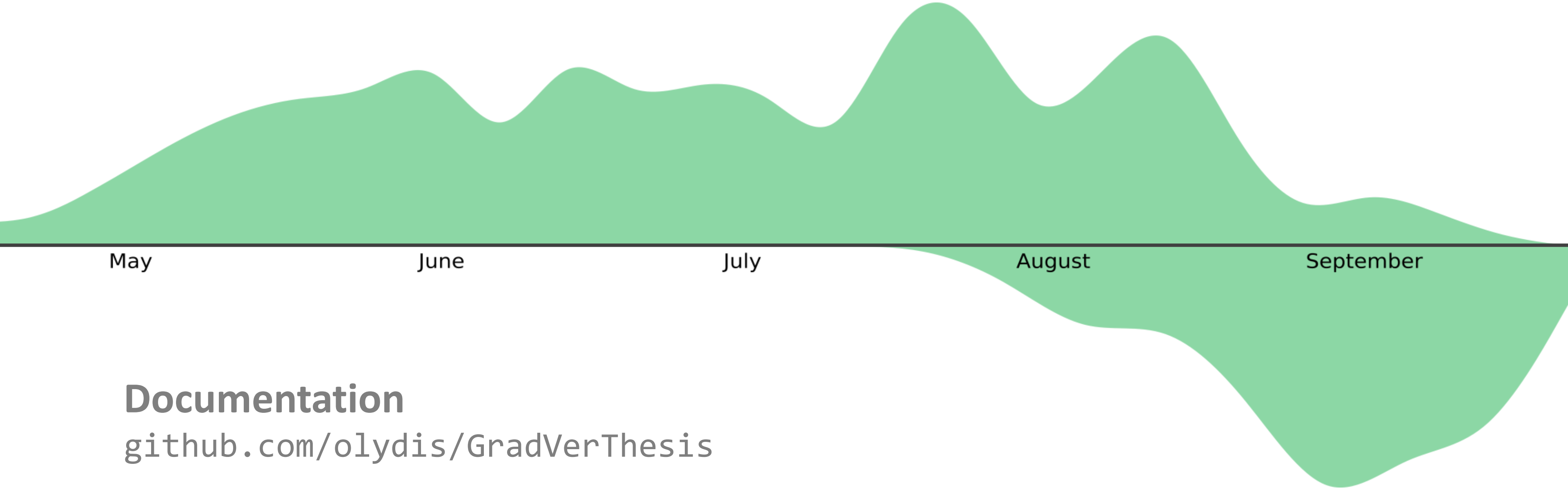
$$\{? * (\mathtt{p1.age} = 19) * (\mathtt{p2.age} = 19)\}$$

$$\mathtt{p1.age{+}{+}}$$

$$\{? * (\mathtt{p1.age} = 20) * (\mathtt{p2.age} = 19)\}$$

Timeline

Design & Implementation
github.com/olydis/GradVer

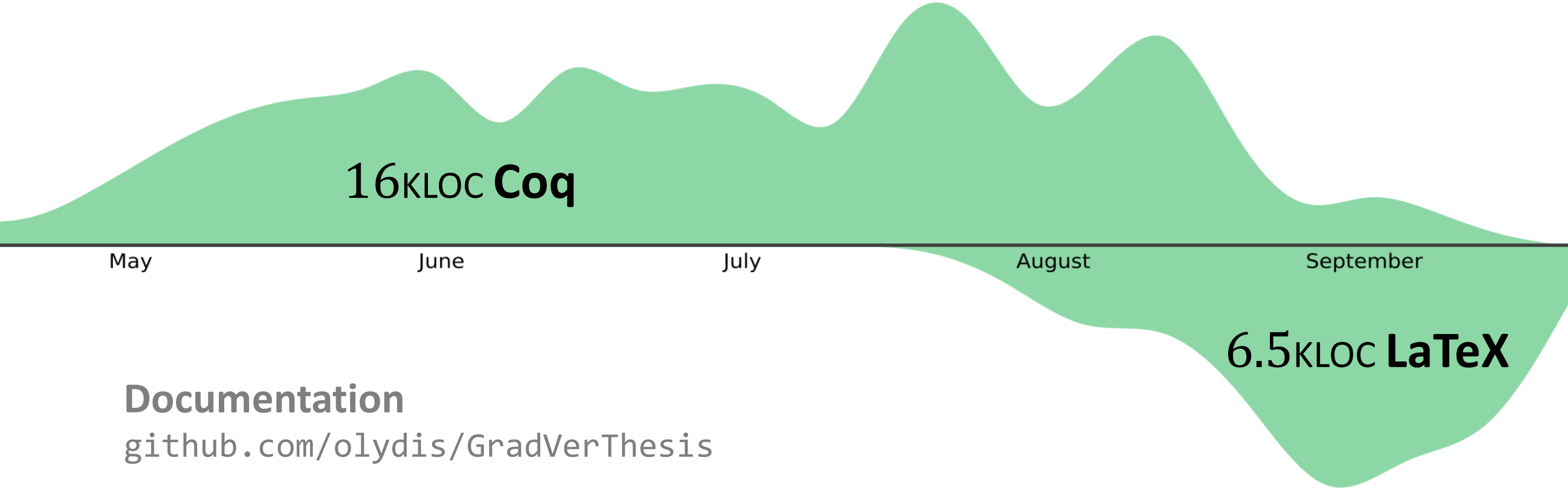


Documentation
github.com/olydis/GradVerThesis

Timeline

Design & Implementation

`github.com/olydis/GradVer`



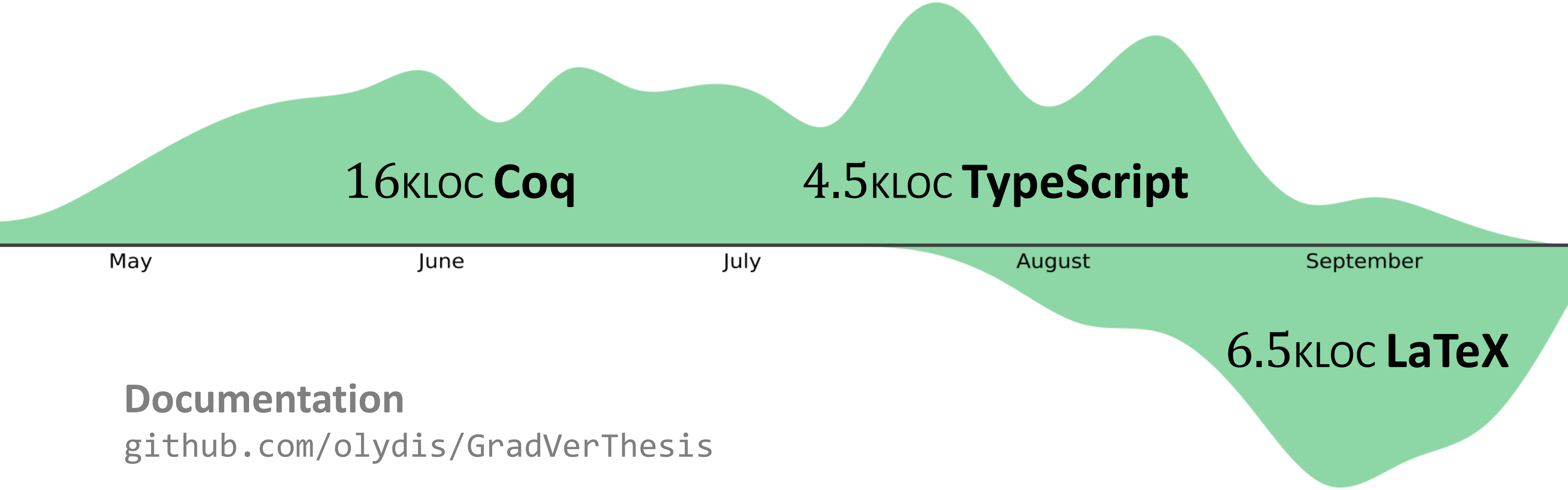
Documentation

`github.com/olydis/GradVerThesis`

Timeline

Design & Implementation
`github.com/olydis/GradVer`

olydis.github.io/GradVer/impl/HTML5



Documentation
`github.com/olydis/GradVerThesis`