A Weakest Pre–Expectation Semantics for Mixed–Sign Expectations

Benjamin Lucien Kaminski  Joost-Pieter Katoen

32nd Annual Symposium on Logic in Computer Science 2017

June 22, 2017, Reykjavík, Iceland
### Example of a Probabilistic Program

\[
\begin{align*}
\{c := 0\} & \left[\frac{1}{2}\right] \{c := 1\}; \\
\text{if} (c = 1) & \{x := 1\} \text{ else } \{x := 2x + 1\}; \\
\text{skip}
\end{align*}
\]
Example of a Probabilistic Program

```cpp
{c := 0} \ [1/2] \ {c := 1};  // coin flip

if (c = 1) \{x := 1\} else \{x := 2x + 1\};

skip
```

What does a probabilistic program \( C \) do?
Run \( C \) on initial state \( \sigma \in \Sigma \)
Obtain probability distribution \( J_{C K} \sigma \) over final states

Formal verification of probabilistic programs!
Example of a Probabilistic Program

\[
\begin{align*}
\{ c := 0 \} & \text{[1/2]} \{ c := 1 \}; \\
\text{if } (c = 1) & \{ x := 1 \} \text{ else } \{ x := 2x + 1 \}; \\
\text{skip}
\end{align*}
\]

// coin flip

What does a probabilistic program \( C \) do?
Example of a Probabilistic Program

\{ c := 0 \} [1/2] \{ c := 1 \};

// coin flip

if (c = 1) \{ x := 1 \} else \{ x := 2x + 1 \};

skip

What does a probabilistic program $C$ do?

- Run $C$ on initial state $\sigma \in \Sigma$
Example of a Probabilistic Program

\[
\{c := 0\} \frac{1}{2} \{c := 1\}; \quad // \text{coin flip}
\]

\[
\text{if} \ (c = 1) \ \{x := 1\} \ \text{else} \ \{x := 2x + 1\};
\]

\text{skip}

What does a probabilistic program $C$ do?

- Run $C$ on initial state $\sigma \in \Sigma$
- Obtain probability distribution $[C]_\sigma$ over final states
Example of a Probabilistic Program

```plaintext
{c := 0} [1/2] {c := 1}; // coin flip
if (c = 1) {x := 1} else {x := 2x + 1};
skip
```

What does a probabilistic program $C$ do?

- Run $C$ on initial state $\sigma \in \Sigma$
- Obtain probability distribution $[C]_\sigma$ over final states

**Formal verification of probabilistic programs!**
Classical Weakest Pre–Expectations

The Non–Negative Case
Classical Weakest Pre–Expectations

Weakest Pre–Expectations

Expectation is a non–negative random variable $f : \Sigma \rightarrow \mathbb{R}_{\geq 0}$, where $\mathbb{E}[f] \neq$ expected value.

What we are interested in:

Given a post–expectation $f$ to be evaluated in the final states after termination of a probabilistic program $C$ on input $\sigma$.

Expected value of $f$ after termination of $C$ on $\sigma$:

$\lambda_{\sigma} \cdot EV_{J}^{C}(f)$
Classical Weakest Pre–Expectations

Expectations:
Classical Weakest Pre–Expectations

**Expectations:**

- **Expectation is a non-negative random variable** $f : \Sigma \to \mathbb{R}_{\geq 0}$
Classical Weakest Pre–Expectations

Expectations:

- **Expectation** is a non-negative random variable \( f : \Sigma \to \mathbb{R}_{\geq 0} \)
- expectation \( \neq \) expected value
Classical Weakest Pre–Expectations

Expectations:

- Expectation is a non–negative random variable $f: \Sigma \rightarrow \mathbb{R}_{\geq 0}$
- expectation $\neq$ expected value

What we are interested in:
Classical Weakest Pre–Expectations

**Expectations:**

- Expectation is a non-negative random variable $f : \Sigma \rightarrow \mathbb{R}_{\geq 0}$
- expectation $\neq$ expected value

**What we are interested in:**

- Given an expectation $f$ to be evaluated in the final states after termination of a probabilistic program $C$ on input $\sigma$
Classical Weakest Pre–Expectations

Expectations:

- **Expectation** is a non-negative random variable $f : \Sigma \rightarrow \mathbb{R}_{\geq 0}$
- **expectation** $\neq$ **expected value**

What we are interested in:

- Given a post–expectation $f$ to be evaluated in the final states after termination of a probabilistic program $C$ on input $\sigma$
Classical Weakest Pre–Expectations

**Expectations:**

- **Expectation** is a non–negative random variable $f : \Sigma \rightarrow \mathbb{R}_{\geq 0}$
- expectation $\neq$ expected value

**What we are interested in:**

- Given a post–expectation $f$ to be evaluated in the final states after termination of a probabilistic program $C$ on input $\sigma$
- Expected value of $f$ after termination of $C$ on $\sigma$:
  
  \[ \text{EV} (f) \]
Classical Weakest Pre–Expectations

Expectations:

- **Expectation** is a non–negative random variable \( f : \Sigma \rightarrow \mathbb{R}_{\geq 0} \)
- expectation \( \neq \) expected value

What we are interested in:

- Given a post–expectation \( f \) to be evaluated in the final states after termination of a probabilistic program \( C \) on input \( \sigma \)
- Expected value of \( f \) after termination of \( C \) on \( \sigma \):

\[
\text{EV}_{[C]_{\sigma}}(f)
\]
Classical Weakest Pre–Expectations

Expectations:

- Expectation is a non–negative random variable $f : \Sigma \rightarrow \mathbb{R}_{\geq 0}$
- expectation $\neq$ expected value

What we are interested in:

- Given a post–expectation $f$ to be evaluated in the final states after termination of a probabilistic program $C$ on input $\sigma$
- Expected value of $f$ after termination of $C$ on $\sigma$:
  \[ \lambda\sigma \cdot \text{EV}_{[C]}(f) \]
Classical Weakest Pre–Expectations

The Standard wp Transformer [Kozen, McIver & Morgan]
Classical Weakest Preexpectations

The Standard wp Transformer [Kozen, McIver & Morgan]

Use a **backward moving** expectation transformer \( \text{wp}[C] : \mathbb{E} \rightarrow \mathbb{E} \).
The Standard wp Transformer [Kozen, McIver & Morgan]

Use a backward moving expectation transformer wp[C]: \( E \rightarrow E \).
The Standard wp Transformer [Kozen, McIver & Morgan]

Use a **backward moving** expectation transformer \( \text{wp}[C] : \mathbb{E} \rightarrow \mathbb{E} \).

\[
\begin{align*}
C & \quad f \\
\uparrow & \quad \uparrow \\
\text{post–expectation } f & \quad \text{evaluated in final states} \\
& \quad \text{after termination of } C
\end{align*}
\]
Classical Weakest Preexpectations

The Standard \( wp \) Transformer [Kozen, Mclver & Morgan]

Use a **backward moving** expectation transformer \( wp[C]: E \rightarrow E \).

\[
C \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad f
\]

post–expectation \( f \)
evaluated in final states
after termination of \( C \)
Classical Weakest Preexpectations

The Standard \( wp \) Transformer [Kozen, McIver & Morgan]

Use a **backward moving** expectation transformer \( wp[C] : \mathbb{E} \to \mathbb{E} \).

\[
wp[C](f) \quad C \quad f
\]

post–expectation \( f \)
evaluated in final states
after termination of \( C \)
Classical Weakest Pre–Expectations

The Standard wp Transformer [Kozen, McIver & Morgan]

Use a backward moving expectation transformer $wp[C] : \mathbb{E} \rightarrow \mathbb{E}$.

$$\lambda \sigma. \text{EV}_{[C]}(f) \triangleright wp[C](f)$$

post–expectation $f$ evaluated in final states after termination of $C$

evaluated in initial states before executing $C$
Classical Weakest Pre–Expectations

The Standard wp Transformer [Kozen, McIver & Morgan]

Use a **backward moving** expectation transformer \( wp[C] : E \rightarrow E \).

\[
\lambda \sigma . \ EV_{[C]}_{\sigma} (f) \overset{!}{=} wp [C] (f)
\]

- weakest pre–expectation of \( C \) with respect to \( f \) evaluated in initial states before executing \( C \)
- post–expectation \( f \) evaluated in final states after termination of \( C \)
Weakest Pre–Expectation Reasoning

Example of wp Reasoning

\[
\{ c : = 0 \} \left\lceil \frac{1}{2} \right\rceil \{ c : = 1 \} ;
\]

\[
\text{if} \ (c = 1) \{ x : = 1 \} \text{ else } \{ x : = 2x + 1 \};
\]

\text{skip}
Weakest Pre–Expectation Reasoning

Example of wp Reasoning

\[
\{ c := 0 \} [1/2] \{ c := 1 \};
\]

if \((c = 1)\) \{ \(x := 1\) \} else \{ \(x := 2x + 1\) \};

skip

\(x\)
Weakest Pre–Expectation Reasoning

Example of wp Reasoning

\[
\{c := 0\} \left[ \frac{1}{2} \right] \{c := 1\} ;
\]

\[
\text{if} (c = 1) \{x := 1\} \text{ else } \{x := 2x + 1\};
\]

\[
\text{wp}[\text{skip}](x)
\]

\[
\text{skip}
\]

\[
x
\]
Weakest Pre–Expectation Reasoning

Example of wp Reasoning

\[
\{ c := 0 \} \frac{1}{2} \{ c := 1 \} ;
\]

\[
\text{if} \ (c = 1) \{ x := 1 \} \text{ else } \{ x := 2x + 1 \} ;
\]

\[
\begin{align*}
&x \\
&\text{skip} \\
&x
\end{align*}
\]
Weakest Pre–Expectation Reasoning

Example of wp Reasoning

\[
\{ c := 0 \} \frac{1}{2} \{ c := 1 \} ; \\
wp [ \text{if} (c = 1) \ldots ] (x) \\
\text{if} (c = 1) \{ x := 1 \} \text{else} \{ x := 2x + 1 \}; \\
x \\
\text{skip} \\
x
\]
Weakest Pre–Expectation Reasoning

Example of wp Reasoning

\[
\{c := 0\} \frac{1}{2} \{c := 1\};
\]

\[
[c = 1] \cdot 1 + [c = 0] \cdot (2x + 1)
\]

\[
\text{if } (c = 1) \{x := 1\} \text{ else } \{x := 2x + 1\};
\]

\[
x
\]

\[
\text{skip}
\]

\[
x
\]
Weakest Pre–Expectation Reasoning

Example of wp Reasoning

\[
\begin{align*}
\{ c := 0 \} & \quad [1/2] \quad \{ c := 1 \} ; \\
1 + [c = 0] \cdot 2x & \\
\text{if} \ (c = 1) \{ x := 1 \} \quad \text{else} \quad \{ x := 2x + 1 \}; \\
x & \\
\text{skip} & \\
x &
\end{align*}
\]
Weakest Pre–Expectation Reasoning

Example of wp Reasoning

\[
\text{wp} \{\ldots\} \left[\frac{1}{2}\right] \{\ldots\} (1 + \llbracket c = 0 \rrbracket \cdot 2x)
\]

\[
\{ c := 0 \} \left[\frac{1}{2}\right] \{ c := 1 \};
\]

\[
1 + \llbracket c = 0 \rrbracket \cdot 2x
\]

\[
\text{if } (c = 1) \{ x := 1 \} \text{ else } \{ x := 2x + 1 \};
\]

\[
x
\]

skip

\[
x
\]
Weakest Pre–Expectation Reasoning

Example of wp Reasoning

\[
\frac{1}{2} \cdot (1 + [0 = 0] \cdot 2x) + \frac{1}{2} \cdot (1 + [1 = 0] \cdot 2x)
\]

\[
\{ c := 0 \} [1/2] \{ c := 1 \};
\]

\[
1 + [c = 0] \cdot 2x
\]

\[
\text{if } (c = 1) \{ x := 1 \} \text{ else } \{ x := 2x + 1 \};
\]

\[
x
\]

\[
\text{skip}
\]

\[
x
\]
Weakest Pre–Expectation Reasoning

Example of wp Reasoning

\[
1 + x \\
\{c := 0\} [1/2] \{c := 1\} ; \\
1 + [c = 0] \cdot 2x \\
\text{if} (c = 1) \{x := 1\} \text{ else } \{x := 2x + 1\} ; \\
x \\
\text{skip} \\
x
\]
The \(\text{wp}\) Transformer for While Loops

Use least fixed point construct:
The wp Transformer for While Loops

Use least fixed point construct:

$$\text{wp[while } (\xi) \{C\} ](f) = \text{lfp } F_f(X)$$
The wp Transformer for While Loops

Use least fixed point construct:

$$\text{wp}[\text{while } (\xi) \{C\}](f) = \text{lfp } F_f(X)$$

$$\begin{align*}
\text{[\neg \xi]} \cdot f & + \text{[\xi]} \cdot \text{wp}[C](X)
\end{align*}$$
The wp Transformer for While Loops

Use least fixed point construct:

\[
\text{wp}[\text{while } (\xi) \{ C \}](f) = \text{lfp } F_f(X) = \sup_n F^n_f(0)
\]

\[
\left( [\neg \xi] \cdot f + [\xi] \cdot \text{wp}[C](X) \right)
\]
The wp Transformer for While Loops

Use least fixed point construct:

\[
\text{wp}[\text{while } (\xi) \{ C \}] (f) = \text{lfp } F_f(X) = \sup_n F^n_f(0)
\]

\[
= [\neg \xi] \cdot f + [\xi] \cdot \text{wp}[C](X)
\]
The wp Transformer for While Loops

Use least fixed point construct:

\[
\text{wp} \left[ \text{while} (\xi) \{ C \} \right] (f) = \text{lfp} F_f(X) = \sup_n F_f^n(0)
\]

\[
\text{Kleene Fixed Point Theorem}
\]

\[
\lbrack \neg \xi \rbrack \cdot f + \lbrack \xi \rbrack \cdot \text{wp}[C](X)
\]

Complete partial order on expectations:

\[
f_1 \preceq f_2 \iff \forall \sigma : f_1(\sigma) \leq f_2(\sigma)
\]
The Motivation

Example of wp Reasoning

\[
\begin{align*}
1 + x \\
\text{if } (1/2) \{ c := 0 \} \text{ else } \{ c := 1 \}; \\
1 + [c = 0] \cdot 2x \\
\text{if } (c = 1) \{ x := 1 \} \text{ else } \{ x := 2x + 1 \}; \\
x \\
\text{skip} \\
x
\end{align*}
\]
The Motivation

Example of wp Reasoning

\[ 1 + x \notin E \]
\[ \text{if } (\frac{1}{2}) \{ c := 0 \} \text{ else } \{ c := 1 \}; \]
\[ 1 + [c = 0] \cdot 2x \notin E \]
\[ \text{if } (c = 1) \{ x := 1 \} \text{ else } \{ x := 2x + 1 \}; \]
\[ x \notin E \]
\[ \text{skip} \]
\[ x \notin E \]
The Motivation

Example of wp Reasoning

\[ 1 + x \not\in E \]
if \((1/2) \{ c := 0 \} \) else \( \{ c := 1 \} \);

\[ 1 + [c = 0] \cdot 2x \not\in E \]
if \((c = 1) \{ x := 1 \} \) else \( \{ x := 2x + 1 \} \);

\[ x \not\in E \]

skip

\[ x \not\in E \]

Neither post–expectation \( x \) nor any of the pre–expectations are proper expectations!
Mixed–Sign Weakest Pre–Expectations

The Non–Non–Negative Case
Our Solution: Integrability–Witnessing Expectations
Our Solution: Integrability–Witnessing Expectations

Define set of mixed–sign expectations $\mathbb{E}^* = \{ f \mid f : \Sigma \to \mathbb{R} \}$
Our Solution: Integrability–Witnessing Expectations

- Define set of mixed-sign expectations \( E^* = \{ f \mid f : \Sigma \to \mathbb{R} \} \)

- \( \text{EV}(f) \) is well-defined if and only if \( \text{EV}(|f|) < \infty \)
Our Solution: Integrability–Witnessing Expectations

- Define set of mixed–sign expectations $\mathbb{E}^* = \{ f | f : \Sigma \rightarrow \mathbb{R} \}$
- $\text{EV}(f)$ is well–defined if and only if $\text{EV}(|f|) < \infty$
  - In probability theory terms: $f$ should be integrable
Our Solution: Integrability–Witnessing Expectations

- Define set of mixed–sign expectations $\mathbb{E}^* = \{ f \mid f : \Sigma \rightarrow \mathbb{R} \}$

- $EV(f)$ is well–defined if and only if $EV(|f|) < \infty$
  - In probability theory terms: $f$ should be integrable

- Integrability–witnessing pairs:

  $$(f, g)$$

  such that $f \in \mathbb{E}^*$, $g \in \mathbb{E}$, and $|f| \leq g$
Our Solution: Integrability–Witnessing Expectations

\[ \mathbb{R}^{\pm \infty} \]

\[ \sum \]
Our Solution: Integrability–Witnessing Expectations
Our Solution: Integrability–Witnessing Expectations

\[ \mathbb{R}^{\pm \infty} \]

\[ \Sigma \]

\[ f, g, -g \]
Our Solution: Integrability–Witnessing Expectations
Our Solution: Integrability–Witnessing Expectations

\[ \mathbb{R}^{\pm \infty} \]

\[ (f, g) \]

\[ \Sigma \]

\[ f \]

\[ g \]

\[ -g \]
Our Solution: Integrability–Witnessing Expectations

\[ \mathbb{R}^{\pm \infty} \]

\[ (f, g) \]

\[ g \]

\[ f \]

\[ \Sigma \]

\[ -g \]
wp with Integrability–Witnessing Pairs

Example of Integrability–Witnessing Pair Reasoning

\[
\text{if } (1/2) \{ c := 0 \} \text{ else } \{ c := 1 \}; \\
\text{if } (c = 1) \{ x := 1 \} \text{ else } \{ x := 2x + 1 \}; \\
\text{skip}
\]
wp with Integrability–Witnessing Pairs

Example of Integrability–Witnessing Pair Reasoning

\[
\begin{align*}
\text{if } (1/2) \{ c := 0 \} \text{ else } \{ c := 1 \}; \\
\text{if } (c = 1) \{ x := 1 \} \text{ else } \{ x := 2x + 1 \}; \\
\text{skip} \quad (x, \, |x|)
\end{align*}
\]
wp with Integrability–Witnessing Pairs

Example of Integrability–Witnessing Pair Reasoning

\[
\text{if } (1/2) \{ c := 0 \} \text{ else } \{ c := 1 \};
\]
\[
\text{if } (c = 1) \{ x := 1 \} \text{ else } \{ x := 2x + 1 \};
\]
\[
\text{(wp [skip]} (x), \text{wp [skip]} (|x|))
\]
\[
\text{skip}
\]
\[
(x, |x|)
\]
wp with Integrability–Witnessing Pairs

Example of Integrability–Witnessing Pair Reasoning

\[ \text{if } (1/2) \{ c := 0 \} \text{ else } \{ c := 1 \}; \]

\[ \text{if } (c = 1) \{ x := 1 \} \text{ else } \{ x := 2x + 1 \}; \]

\[ (x, |x|) \]

\textbf{skip}

\[ (x, |x|) \]
wp with Integrability–Witnessing Pairs

Example of Integrability–Witnessing Pair Reasoning

\[
\text{if } (1/2) \{ c := 0 \} \text{ else } \{ c := 1 \}; \\
(wp[\text{if}(c = 1) \ldots](x), wp[\text{if}(c = 1) \ldots](|x|))
\]

\[
\text{if } (c = 1) \{ x := 1 \} \text{ else } \{ x := 2x + 1 \}; \\
(x, |x|)
\]

skip

\[
(x, |x|)
\]
wp with Integrability–Witnessing Pairs

Example of Integrability–Witnessing Pair Reasoning

\[
\begin{align*}
\text{if } (1/2) \{ c := 0 \} \text{ else } \{ c := 1 \}; \\
(1 + [c = 0] \cdot 2x, [c = 1] + [c = 0] \cdot |2x + 1|) \\
\text{if } (c = 1) \{ x := 1 \} \text{ else } \{ x := 2x + 1 \}; \\
(x, |x|) \\
\text{skip} \\
(x, |x|)
\end{align*}
\]
wp with Integrability–Witnessing Pairs

Example of Integrability–Witnessing Pair Reasoning

\[
\left( 1 + x, \frac{1}{2} + \left| x + \frac{1}{2} \right| \right)
\]

if \((1/2) \{ c := 0 \} \text{ else } \{ c := 1 \} \);

\((1 + [c = 0] \cdot 2x, [c = 1] + [c = 0] \cdot |2x + 1|)\)

if \(c = 1\) \{ \(x := 1\) \} else \{ \(x := 2x + 1\) \};

\((x, |x|)\)

skip

\((x, |x|)\)
What about Loops?

- Consider $\texttt{while} (\xi) \{ C \}$
What about Loops?

- Consider while $(\xi)\{C\}$
- Recall $wp[\text{while } (\xi) \{C\}] (f) = \sup_n F^n_f (0)$ (KFPT)
What about Loops?

- Consider \( \text{while} (\xi) \{C\} \)
- Recall \( \text{wp} [\text{while} (\xi) \{C\}](f) = \sup_n F^n_f(0) \) (KFPT)
- Can we do KFPT–style approximations of loop semantics using integrability–witnessing pairs?
What about Loops?

- Consider \( \text{while} (\xi) \{ C \} \)

- Recall \( \text{wp} [ \text{while} (\xi) \{ C \}] (f) = \sup_n F^n_f(0) \) (KFPT)

- Can we do KFPT–style approximations of loop semantics using integrability–witnessing pairs?

\[
\lim_{n \to \omega} \left( F^n_f(0), F^n_{|f|}(0) \right)
\]
What about Loops?

- Consider \texttt{while} $\langle \xi \rangle \{ C \}$
- Recall $\text{wp} \left[ \texttt{while} \langle \xi \rangle \{ C \} \right] (f) = \sup_{n} F_{f}^{n}(0)$ (KFPT)
- Can we do KFPT–style approximations of loop semantics using integrability–witnessing pairs?

$$\lim_{n \to \omega} \left( F_{f}^{n}(0), F_{\|f\|}^{n}(0) \right)$$

- Consider the program

\[
C_{\text{geo2}} \triangleright \texttt{while} \left( \frac{1}{2} \right) \{ \\
\quad x := -2 \cdot x \}
\]
What about Loops?

- Consider \( \text{while} (\xi) \{ C \} \)
- Recall \( \text{wp} [\text{while} (\xi) \{ C \}] (f) = \sup_n F^n_f (0) \) (KFPT)

- Can we do KFPT–style approximations of loop semantics using integrability–witnessing pairs?

\[
\lim_{n \to \omega} \left( F^n_f (0), F^n_{|f|} (0) \right)
\]

- Consider the program

\[
C_{geo2} \triangleright \text{while } (1/2) \{ \\
\quad x := -2 \cdot x \\
\}
\]

- According sequence:
What about Loops?

- Consider while ($\xi$) $\{C\}$
- Recall $wp[\text{while } (\xi) \{C\}] (f) = \sup_n F^n_f(0)$ (KFPT)
- Can we do KFPT–style approximations of loop semantics using integrability–witnessing pairs?

$$\lim_{n \to \omega} \left( F^n_f(0), F^n_{|f|}(0) \right)$$

- Consider the program

$$C_{geo2} \triangleright \text{while } (1/2) \{ \text{ } \}
\text{ } \text{ } \text{ } x := -2 \cdot x$$

- According sequence:

$$\left( \frac{x}{2}, \frac{|x|}{2} \right)$$
What about Loops?

- Consider while \((\xi) \{ C \}\)

- Recall \(\text{wp}[\text{while} (\xi) \{ C \}](f) = \sup_n F^n_f(0)\) (KFPT)

- Can we do KFPT–style approximations of loop semantics using integrability–witnessing pairs?

\[
\lim_{n \to \infty} \left( F^n_f(0), F^n_{|f|}(0) \right)
\]

- Consider the program

\[
C_{geo2} \triangleright \text{while } (1/2) \{ \text{x := } -2 \cdot \text{x} \}\]

- According sequence:

\[
\left( \frac{x}{2}, \frac{|x|}{2} \right) \quad \left( 0, |x| \right)
\]
What about Loops?

- Consider \( \text{while } (\xi) \{ C \} \)
- Recall \( \text{wp } [\text{while } (\xi) \{ C \}] (f) = \sup_n F^n_f(0) \) (KFPT)
- Can we do KFPT–style approximations of loop semantics using integrability–witnessing pairs?

\[
\lim_{n \to \omega} \left( F^n_f(0), F^n_{|f|}(0) \right)
\]

- Consider the program

\[
C_{geo2} \triangleright \text{ while } (1/2) \{ \\
\quad x := -2 \cdot x \}
\]

- According sequence:

\[
\left( \frac{x}{2}, \frac{|x|}{2} \right) \quad \left( 0, |x| \right) \quad \left( \frac{x}{2}, \frac{3|x|}{2} \right)
\]
What about Loops?

- Consider `while (ξ) {C}`
- Recall `wp [while (ξ) {C}] (f) = sup_n F^n_f (0)` (KFPT)
- Can we do KFPT–style approximations of loop semantics using integrability–witnessing pairs?

\[
\lim_{n \to \omega} \left( F^n_f (0), F^n_{|f|} (0) \right)
\]

- Consider the program

\[
C_{geo2} \triangleright \text{ while } (1/2) \{ \text{ } x := -2 \cdot x \text{ } \}
\]

- According sequence:

\[
\left( \frac{x}{2}, \frac{|x|}{2} \right) \quad \left( 0, |x| \right) \quad \left( \frac{x}{2}, \frac{3|x|}{2} \right) \quad \left( 0, 2|x| \right)
\]
What about Loops?

- Consider $\text{while } (\xi) \{ C \}$
- Recall $\text{wp [while } (\xi) \{ C \} ] (f) = \sup_n F^n_f (0)$ (KFPT)
- Can we do KFPT–style approximations of loop semantics using integrability–witnessing pairs?

$$\lim_{n \to \omega} \left( F^n_f (0), F^n_{|f|} (0) \right)$$

- Consider the program

$$C_{geo2} \triangleright \text{while } (1/2) \{ \begin{align*} x &:= -2 \cdot x \end{align*} \}$$

- According sequence:

$$\left( \frac{x}{2}, \frac{|x|}{2} \right) \quad \left( 0, |x| \right) \quad \left( \frac{x}{2}, \frac{3|x|}{2} \right) \quad \left( 0, 2|x| \right) \quad \cdots$$
What about Loops?

- Consider \( \text{while}(\xi) \{ C \} \)
- Recall \( \text{wp}[\text{while}(\xi) \{ C \}](f) = \sup_n F^n_f(0) \) (KFPT)
- Can we do KFPT–style approximations of loop semantics using integrability–witnessing pairs?

\[
\lim_{n \to \omega} \left( F^n_f(0), F^n_{|f|}(0) \right)
\]

- Consider the program

\[
C_{geo2} \triangleright \text{while}(1/2) \{ \quad x := -2 \cdot x \}
\]

- According sequence:

\[
\left( \frac{x}{2}, \frac{|x|}{2} \right) \quad \left( 0, |x| \right) \quad \left( \frac{x}{2}, \frac{3|x|}{2} \right) \quad \left( 0, 2|x| \right) \quad \ldots
\]
What about Loops?

- Consider while \((\xi) \{C\}\)

- Recall \(wp[while(\xi)\{C\}](f) = \sup_n F^n_f(0)\) (KFPT)

- Can we do KFPT–style approximations of loop semantics using integrability–witnessing pairs?

\[
\lim_{n \to \omega} \left( F^n_f(0), F^n_{|f|}(0) \right)
\]

- Consider the program

\[
C_{geo2} \triangleright \text{while } (1/2) \{ x := -2 \cdot x \}
\]

- According sequence:

\[
\left(\frac{x}{2}, \frac{|x|}{2}\right) \quad \left(0, |x|\right) \quad \left(\frac{x}{2}, \frac{3|x|}{2}\right) \quad \left(0, 2|x|\right) \quad \cdots \quad \ddots
\]
Our Solution: Integrability–Witnessing Expectations
Our Solution: Integrability–Witnessing Expectations
Our Solution: Integrability–Witnessing Expectations
Our Solution: Integrability–Witnessing Expectations
Our Solution: Integrability–Witnessing Expectations

\[ \mathbb{R}^{\pm \infty} \]

\[ f \]

\[ g \]

\[ -g \]
Our Solution: Integrability–Witnessing Expectations

equivalence relation $\approx$ on pairs: $(f_1, g_1) \approx (f_2, g_2)$ iff $\forall \sigma: g(\sigma) \neq \infty$ implies $f_1(\sigma) = f_2(\sigma)$

Integrability–witnessing expectation (IWE): $\approx$–equivalence class $H_{f, g}$ of $(f, g)$

partial order on IWEs: $H_{f, g} \sqsubseteq H_{f', g'}$ iff $\forall \sigma: g'(\sigma) \neq \infty$ implies $f(\sigma) \leq f'(\sigma)$ and $g(\sigma) \leq g'(\sigma)$

$\sqsubseteq$ is not a cpo. No least element!

In particular: $H_{0, 0} \not\sqsubseteq H_{-1, 1}$

Still we can define a wp transformer acting on IWEs
Our Solution: Integrability–Witnessing Expectations

- equivalence relation $\approx$ on pairs: $(f_1, g) \approx (f_2, g)$ iff
  \[ \forall \sigma: g(\sigma) \neq \infty \text{ implies } f_1(\sigma) = f_2(\sigma) \]
Our Solution: Integrability–Witnessing Expectations

- equivalence relation \( \approx \) on pairs: \((f_1, g) \approx (f_2, g)\) iff

\[ \forall \sigma : g(\sigma) \neq \infty \text{ implies } f_1(\sigma) = f_2(\sigma) \]

- Integrability–witnessing expectation (IWE):

\( \approx \)-equivalence class \( \langle f, g \rangle \) of \((f, g)\)
Our Solution: Integrability–Witnessing Expectations

- equivalence relation $\sim$ on pairs: $(f_1, g) \sim (f_2, g)$ iff

  $$\forall \sigma : g(\sigma) \neq \infty \text{ implies } f_1(\sigma) = f_2(\sigma)$$

- Integrability–witnessing expectation (IWE):

  $\sim$–equivalence class $\ll f, g \rr$ of $(f, g)$

- partial order on IWEs: $\ll f, g \rr \sqsubseteq \ll f', g' \rr$ iff

  $$\forall \sigma : g'(\sigma) \neq \infty \text{ implies } f(\sigma) \leq f'(\sigma) \text{ and } g(\sigma) \leq g'(\sigma)$$
Our Solution: Integrability–Witnessing Expectations

- equivalence relation \(\approx\) on pairs: \((f_1, g) \approx (f_2, g)\) iff
  \[\forall \sigma: \ g(\sigma) \neq \infty \implies f_1(\sigma) = f_2(\sigma)\]

- Integrability–witnessing expectation (IWE): \(\approx\)-equivalence class \(\langle f, g \rangle\) of \((f, g)\)

- partial order on IWEs: \(\langle f, g \rangle \sqsubseteq \langle f', g' \rangle\) iff
  \[\forall \sigma: \ g'(\sigma) \neq \infty \implies f(\sigma) \leq f'(\sigma) \text{ and } g(\sigma) \leq g'(\sigma)\]

- \(\sqsubseteq\) is not a cpo. No least element! In particular:
  \(\langle 0, 0 \rangle \not\sqsubseteq \langle -1, 1 \rangle\)
Our Solution: Integrability–Witnessing Expectations

- equivalence relation \( \approx \) on pairs: \((f_1, g) \approx (f_2, g)\) iff
  \[
  \forall \sigma : g(\sigma) \neq \infty \text{ implies } f_1(\sigma) = f_2(\sigma)
  \]

- Integrability–witnessing expectation (IWE):
  \(\approx\)–equivalence class \(\langle f, g \rangle_I\) of \((f, g)\)

- partial order on IWEs: \(\langle f, g \rangle_I \sqsubseteq \langle f', g' \rangle_I\) iff
  \[
  \forall \sigma : g'(\sigma) \neq \infty \text{ implies } f(\sigma) \leq f'(\sigma) \text{ and } g(\sigma) \leq g'(\sigma)
  \]
  \(\sqsubseteq\) is not a cpo. No least element! In particular:
  \[
  \langle 0, 0 \rangle \not\sqsubseteq \langle -1, 1 \rangle
  \]

- Still we can define a wp transformer acting on IWEs
IWEs and Loops

\[
wp[\text{while } (\xi) \{C\}]_{f, g} = \lim_{n \to \omega} F^n_{f, g} \langle 0, 0 \rangle
\]

\[
F_{f, g} \langle X, Y \rangle = [\neg \xi] \cdot \langle f, g \rangle + [\xi] \cdot wp[C] \langle X, Y \rangle
\]
IWEs and Loops

\[
\text{wp[while } (\xi) \{ C \}] f, g = \lim_{n \to \omega} F^n_{f, g} [0, 0]
\]

\[
F_{f, g} [X, Y] = [\neg \xi] \cdot f, g + [\xi] \cdot \text{wp}[C] [X, Y]
\]

- Reconsider the program

\[
C_{geo2} \triangleright \text{ while } (1/2) \{ \\
\hspace{1cm} x := -2 \cdot x \}
\]
IWEs and Loops

\[ \text{wp[while } (\xi) \{ C \}] f, g J = \lim_{n \to \omega} F^n_{f, g} J 0, 0 J \]

\[ F_{f, g} J X, Y J = [\neg \xi] \cdot \ll f, g J + [\xi] \cdot \text{wp}[C] J X, Y J \]

- Reconsider the program

\[ C_{geo2} \triangleright \text{while } (1/2) \{ \]
\[ x := -2 \cdot x \}

- According sequence \( F^n_{f, g} J 0, 0 J \):
IWEs and Loops

\[
\text{wp[while } (\xi) \{ C \} ] \ll f, g \rr = \lim_{n \to \omega} F^n_{\ll f, g \rr} \ll 0, 0 \rr
\]

\[
F_{\ll f, g \rr} \ll X, Y \rr = [\neg \xi] \cdot \ll f, g \rr + [\xi] \cdot \text{wp}[C] \ll X, Y \rr
\]

- Reconsider the program

\[
C_{\text{geo2}} \triangleright \text{while } (1/2) \{ \\
\quad x := -2 \cdot x \}
\]

- According sequence \( F^n_{\ll f, g \rr} \ll 0, 0 \rr: \)

\[
\ll \frac{x}{2}, \frac{|x|}{2} \rr
\]
IWEs and Loops

\[
wp[\text{while} (\xi) \{C\}] \ll f, g \rr = \lim_{n \to \omega} \mathcal{F}_{\ll f, g \rr}^n \ll 0, 0 \rr
\]

\[
\mathcal{F}_{\ll f, g \rr}^n \ll X, Y \rr = \ll \neg \xi \rr \cdot \ll f, g \rr + \ll \xi \rr \cdot wp[C] \ll X, Y \rr
\]

- Reconsider the program

\[
C_{\text{geo2}} \triangleright \text{while } (1/2) \{ \\
\quad x := -2 \cdot x \}
\]

- According sequence \( \mathcal{F}_{\ll f, g \rr}^n \ll 0, 0 \rr \):

\[
\ll x/2, |x|/2 \rr \ll 0, |x| \rr
\]
IWEs and Loops

\[
\text{wp[while}(\xi)\{C\}]\{f, g\} = \lim_{n \to \omega} F^n_{\xi, f, g} \{0, 0\}
\]

\[
F_{\xi, f, g} \{X, Y\} = [\neg \xi] \cdot \{f, g\} + [\xi] \cdot \text{wp}[C] \{X, Y\}
\]

- Reconsider the program

\[
\text{while } (1/2) \{ \quad x := -2 \cdot x \quad \}\]

- According sequence \( F^n_{\xi, f, g} \{0, 0\} \):

\[
\left\{ \frac{x}{2}, \frac{|x|}{2} \right\} \quad \left\{ 0, |x| \right\} \quad \left\{ \frac{x}{2}, \frac{3|x|}{2} \right\}
\]
IWEs and Loops

\[ \text{wp[while } (\xi) \{ C \}] \langle f, g \rangle = \lim_{n \to \omega} F^n_{\langle f, g \rangle} \langle 0, 0 \rangle \]

\[ F_{\langle f, g \rangle} \langle X, Y \rangle = \lbrack -\xi \rbrack \cdot \langle f, g \rangle + \lbrack \xi \rbrack \cdot \text{wp}[C] \langle X, Y \rangle \]

■ Reconsider the program

\[ C_{geo2} \triangleright \text{ while } (1/2) \{ \]

\[ x := -2 \cdot x \}

■ According sequence \( F^n_{\langle f, g \rangle} \langle 0, 0 \rangle \):

\[ \langle x, \frac{|x|}{2} \rangle \langle 0, |x| \rangle \langle x, \frac{3|x|}{2} \rangle \langle 0, 2|x| \rangle \]
IWEs and Loops

\[ \text{wp[while } (\xi) \{ C \} ] f, g \cap \xi = \lim_{n \to \omega} F^n_{\xi} f, g \cap 0, 0 \cap \xi \]

\[ F_{\xi} f, g \cap X, Y \cap \xi = [\neg \xi] \cdot f, g \cap \xi + [\xi] \cdot \text{wp}[C] \cap X, Y \cap \xi \]

- Reconsider the program

\[ C_{\text{geo2}} \triangleright \text{while } (1/2) \{ \]
\[ \quad x := -2 \cdot x \}

- According sequence \( F^n_{\xi} f, g \cap 0, 0 \cap \xi \):
IWEs and Loops

\[ wp[\text{while } (\xi) \{C\}] \prec f, g \prec = \lim_{n \to \omega} F^n_{\prec f, g} \prec 0, 0 \prec \]

\[ F_{\prec f, g} \prec X, Y \prec = [\neg \xi] \cdot \prec f, g \prec + [\xi] \cdot wp[C] \prec X, Y \prec \]

Reconsider the program

\[ C_{\text{geo2}} \triangleright \text{while } (1/2) \{ \]
\[ x := -2 \cdot x \} \]

According sequence \( F^n_{\prec f, g} \prec 0, 0 \prec \):

\[ \left\lfloor \frac{x}{2}, \left| \frac{x}{2} \right| \right\rfloor \left\lfloor 0, |x| \right\rfloor \left\lfloor \frac{x}{2}, \left| \frac{3x}{2} \right| \right\rfloor \left\lfloor 0, 2|x| \right\rfloor \cdots \xrightarrow{\omega} \left\lfloor 0, \infty \cdot |x| \right\rfloor \]
**IWEs and Loops**

\[ wp[\text{while } (\xi) \{ C \}]\{ f, g \} = \lim_{{n \to \omega}} F_n^{\{ f, g \}} \{ 0, 0 \} \]

\[ F^{\{ f, g \}} \{ X, Y \} = [\neg \xi] \cdot \{ f, g \} + [\xi] \cdot wp[C] \{ X, Y \} \]

- Reconsider the program

  \[ C_{geo2} \triangleright \text{while } (1/2) \{ \]

  \[ x := -2 \cdot x \} \]

- According sequence \[ F_n^{\{ f, g \}} \{ 0, 0 \}: \]

  \[ \{ x/2, |x|/2 \} \{ 0, |x| \} \{ x/2, 3|x|/2 \} \{ 0, 2|x| \} \ldots \omega \rightarrow \{ x/2, \infty \cdot |x| \} \]
IWEs and Loops

\[ \text{wp[while}(\xi) \{C\}]f, g \succeq \lim_{n \to \omega} F^n_{f, g} \langle 0, 0 \rangle \]

\[ F_{f, g} \langle X, Y \rangle = [\neg \xi] \cdot f, g \succeq + [\xi] \cdot \text{wp}[C] \langle X, Y \rangle \]

- Reconsider the program

\[ C_{geo2} \triangleright \text{while } (1/2) \{ \]
\[ x := -2 \cdot x \}

- According sequence \( F^n_{f, g} \langle 0, 0 \rangle : \)

\[
\begin{align*}
\langle x, \frac{|x|}{2} \rangle & \langle 0, |x| \rangle \\
\langle x, \frac{3|x|}{2} \rangle & \langle 0, 2|x| \rangle \\
& \cdots \xrightarrow{\omega} \langle 17 \cdot x, \infty \cdot |x| \rangle
\end{align*}
\]
Well-definedness of while-loop semantics:
\[
\lim_{n \to \omega} F_n H f, g I_{H 0, 0} \]
always exists and is unique.

Soundness:
If \( \text{wp}[C_{H f, |f| I}] = H f, g' I_{\sigma} \neq \infty \),
then \( f'(\sigma) = \text{EV}_{J_{C_K \sigma}}(f) \).

Monotonicity:
If \( H f, g I \sqsubseteq H f', g' I_{\sigma} \),
then \( \text{wp}[C_{H f, g I}] \sqsubseteq \text{wp}[C_{H f', g' I}] \).
Properties of wp Transformer Acting on IWEs

Well-definedness of while-loop semantics:
\[ \lim_{n \to \omega} F_n H_f, g I H_0, 0 I \] always exists and is unique.

Soundness:
If \( \text{wp}[C] H_f, |f| I = H_f', g' I \), \( \sigma \neq \infty \), then
\[ f'(\sigma) = \text{EV}_J C K \sigma (f) \].

Monotonicity:
If \( H_f, g I \sqsubseteq H_f', g' I \), then \( \text{wp}[C] H_f, g I \sqsubseteq \text{wp}[C] H_f', g' I \).
Properties of wp Transformer Acting on IWEs

- **Well-definedness of while-loop semantics:**

  \[
  \lim_{n \to \omega} F_n^{f, g} \langle 0, 0 \rangle \text{ always exists and is unique.}
  \]
Properties of wp Transformer Acting on IWEs

- **Well-definedness of while-loop semantics:**
  \[ \lim_{n \to \omega} F^n \{ f, g \} \{ 0, 0 \} \text{ always exists and is unique.} \]

- **Soundness:**
  If \( \text{wp}[C]\{f, |f|\} = \{ f', g' \}, g'(\sigma) \neq \infty \),
Properties of wp Transformer Acting on IWEs

- **Well-definedness of while-loop semantics:**
  \[
  \lim_{n \to \omega} F_n f, g \downarrow 0, 0 \]
  always exists and is unique.

- **Soundness:**
  If \( wp[C] f, |f| \downarrow f', g' \downarrow \), \( g'(\sigma) \neq \infty \), then
  \[
  f'(\sigma) = EV_{[C] \sigma}(f).
  \]
Properties of \( \text{wp} \) Transformer Acting on IWEs

- **Well–definedness of while–loop semantics:**
  \[
  \lim_{n \to \omega} F^n_{f, g} \langle 0, 0 \rangle \text{ always exists and is unique.}
  \]

- **Soundness:**
  If \( \text{wp}[C] \langle f, |f| \rangle = \langle f', g' \rangle, \ g'(\sigma) \neq \infty \), then
  \[
  f'(\sigma) = \text{EV}_{[C]}(f).
  \]

- **Monotonicity:**
  If \( \langle f, g \rangle \sqsubseteq \langle f', g' \rangle \)
Properties of wp Transformer Acting on IWEs

- **Well-definedness of while-loop semantics:**
  \[
  \lim_{n \to \omega} F^n_{f,g}[0,0] \text{ always exists and is unique.}
  \]

- **Soundness:**
  If \(\text{wp}[C][f,|f|] = [f',g'], g'(|\sigma|) \neq \infty\), then
  \[
  f'(\sigma) = \text{EV}_{[C]|\sigma}(f).
  \]

- **Monotonicity:**
  If \([f,g] \subseteq [f',g']\), then \(\text{wp}[C][f,g] \subseteq \text{wp}[C][f',g']\).
Reasoning about While–Loops
### Reasoning about While–Loops

**Invariant Rule for While–Loops**

Let $I, G \in \mathbb{E}$ and $\{H_n\}_{n \in \mathbb{N}} \subseteq \mathbb{E}$. If

\[
F_{|f|+f}(I) \leq I, \quad F_g(G) \leq G, \\
H_0 \leq F_{|f|}(0), \quad \text{and} \quad H_{n+1} \leq F_{|f|}(H_n),
\]

then

\[
\text{wp}[\text{while} (\xi) \{C'\}] f, g \trianglelefteq \bigcup_{n} I - \sup_{H_n, 2 \cdot G}. 
\]
Example

Geometric distribution with alternating sign:

\[ C_{\text{altgeo}} : \quad \text{while } (1/2) \{ x := -x - \text{sign}(x) \} \]
Example

Geometric distribution with alternating sign:

\[ C_{altgeo} : \quad \text{while } (1/2) \{ x := -x - \text{sign}(x) \} \]

- Expected value of \( x \) after execution of \( C_{altgeo} \) is \( \frac{x}{3} - \frac{\text{sign}(x)}{9} \)
Example

Geometric distribution with alternating sign:

\[ C_{\text{altgeo}} : \text{while } (1/2) \{ x := -x - \text{sign}(x) \} \]

- Expected value of \( x \) after execution of \( C_{\text{altgeo}} \) is \( \frac{x}{3} - \frac{\text{sign}(x)}{9} \)

- Our technique yields \( \left\lfloor \frac{x}{3} - \frac{\text{sign}(x)}{9}, |x| + 1 \right\rfloor \)
Example

Geometric distribution with alternating sign:

\[ C_{altgeo} : \quad \text{while } (1/2) \{ x := -x - \text{sign}(x) \} \]

- Expected value of \( x \) after execution of \( C_{altgeo} \) is \( \frac{x}{3} - \frac{\text{sign}(x)}{9} \)
- Our technique yields \( \left\lfloor \frac{x}{3} - \frac{\text{sign}(x)}{9}, |x| + 1 \right\rfloor \)
- Traditional wp does not allow mixed–sign post expectation \( x \)
Example

Geometric distribution with alternating sign:

\[ C_{altgeo} : \text{ while } (1/2) \{ x := -x - \text{sign}(x) \} \]

- Expected value of \( x \) after execution of \( C_{altgeo} \) is \( \frac{x}{3} - \frac{\text{sign}(x)}{9} \)

- Our technique yields \( \left\lfloor \frac{x}{3} - \frac{\text{sign}(x)}{9} \right\rfloor, |x| + 1 \)

- Traditional wp does not allow mixed–sign post expectation \( x \)
  - Workaround: Jordan decomposition of \( x \) into \( +x \) and \( -x \)
Example

Geometric distribution with alternating sign:

\[ C_{altgeo} : \text{while } (1/2) \{ x := -x - \text{sign}(x) \} \]

- Expected value of \( x \) after execution of \( C_{altgeo} \) is \( \frac{x}{3} - \frac{\text{sign}(x)}{9} \)
- Our technique yields \( \left\{ \frac{x}{3} - \frac{\text{sign}(x)}{9}, |x| + 1 \right\} \)
- Traditional wp does not allow mixed–sign post expectation \( x \)
  - Workaround: Jordan decomposition of \( x \) into \( +x \) and \( -x \)
  - Takes about twice the effort for \( C_{altgeo} \) with standard wp
Example

Geometric distribution with alternating sign:

\[ C_{\text{altgeo}} : \quad \text{while} \ (1/2) \ \{x := -x - \text{sign}(x)\} \]

- Expected value of \( x \) after execution of \( C_{\text{altgeo}} \) is \( \frac{x}{3} - \frac{\text{sign}(x)}{9} \)

- Our technique yields \( \left\{ \frac{x}{3} - \frac{\text{sign}(x)}{9}, \ |x| + 1 \right\} \)

- Traditional wp does not allow mixed–sign post expectation \( x \)
  - Workaround: Jordan decomposition of \( x \) into \( +x \) and \( -x \)
  - Takes (me) about twice the effort for \( C_{\text{altgeo}} \) with standard wp
Summary
Our transformer allows for reasoning about mixed–sign expectations at source code level.
Summary

- Our transformer allows for reasoning about mixed–sign expectations at source code level
- Future work: Connection to an operational semantics
Summary

- Our transformer allows for reasoning about mixed–sign expectations at source code level
- Future work: Connection to an operational semantics
- Future work: Come up with nicer proof rules for loops!
Summary

- Our transformer allows for reasoning about mixed-sign expectations at source code level
- Future work: Connection to an operational semantics
- Future work: Come up with nicer proof rules for loops!
- What could all this be useful for (i.e. the actual motivation)?
Summary

- Our transformer allows for reasoning about mixed-sign expectations at source code level
- Future work: Connection to an operational semantics
- Future work: Come up with nicer proof rules for loops!
- What could all this be useful for (ie. the actual motivation)?
  - Expected values of signed program variables
Summary

- Our transformer allows for reasoning about mixed–sign expectations at source code level
- Future work: Connection to an operational semantics
- Future work: Come up with nicer proof rules for loops!
- What could all this be useful for (ie. the actual motivation)?
  - Expected values of signed program variables
  - Calculus for amortized expected run–times
Summary

- Our transformer allows for reasoning about mixed–sign expectations at source code level
- Future work: Connection to an operational semantics
- Future work: Come up with nicer proof rules for loops!
- What could all this be useful for (ie. the actual motivation)?
  - Expected values of signed program variables
  - Calculus for amortized expected run–times

Thank you for your kind attention!
## Backup Slides: Rules for wp Acting on IWEs

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C )</td>
<td>( \text{wp}[C] \langle f, g \rangle )</td>
</tr>
<tr>
<td>\text{skip}</td>
<td>( \langle f, g \rangle )</td>
</tr>
</tbody>
</table>
Rules for the `wp` Transformer Acting on $\mathbb{P}/\approx$

<table>
<thead>
<tr>
<th>$C$</th>
<th>$\text{wp}[C] \downarrow f, g$</th>
</tr>
</thead>
<tbody>
<tr>
<td>skip</td>
<td>$\downarrow f, g$</td>
</tr>
<tr>
<td>$x := E$</td>
<td>$\downarrow f[x/E], g[x/E]$</td>
</tr>
</tbody>
</table>
## Backup Slides: Rules for wp Acting on IWEs

### Rules for the wp Transformer Acting on $\mathbb{P}/\sim$

<table>
<thead>
<tr>
<th>$C$</th>
<th>$\text{wp} [C] \langle f, g \rangle$</th>
</tr>
</thead>
<tbody>
<tr>
<td>skip</td>
<td>$\langle f, g \rangle$</td>
</tr>
<tr>
<td>$x := E$</td>
<td>$\langle f [x/E], g [x/E] \rangle$</td>
</tr>
<tr>
<td>$C_1; C_2$</td>
<td>$\text{wp} [C_1] (\text{wp}[C_2]\langle f, g \rangle)$</td>
</tr>
</tbody>
</table>
Backup Slides: Rules for wp Acting on IWEs

Rules for the wp Transformer Acting on $\mathbb{P}/\approx$

<table>
<thead>
<tr>
<th>$C$</th>
<th>$\text{wp }[C] \langle f, g \rangle$</th>
</tr>
</thead>
<tbody>
<tr>
<td>skip</td>
<td>$\langle f, g \rangle$</td>
</tr>
<tr>
<td>$x := E$</td>
<td>$\langle f[x/E], g[x/E] \rangle$</td>
</tr>
<tr>
<td>$C_1; C_2$</td>
<td>$\text{wp }[C_1] (\text{wp}[C_2] \langle f, g \rangle)$</td>
</tr>
<tr>
<td>if $(\xi) {C_1} \text{ else } {C_2}$</td>
<td>$[[\xi]] \cdot \text{wp}[C_1] \langle f, g \rangle + [[\neg \xi]] \cdot \text{wp}[C_2] \langle f, g \rangle$</td>
</tr>
</tbody>
</table>
Backup Slides: Rules for \( \text{wp} \) Acting on IWEs

Rules for the \( \text{wp} \) Transformer Acting on \( \mathbb{P}/\approx \)

<table>
<thead>
<tr>
<th>( C )</th>
<th>( \text{wp}[C] \ll f, g )</th>
</tr>
</thead>
<tbody>
<tr>
<td>skip</td>
<td>( \ll f, g )</td>
</tr>
<tr>
<td>( x := E )</td>
<td>( \ll f \left[ x/E \right], g \left[ x/E \right] )</td>
</tr>
<tr>
<td>( C_1; C_2 )</td>
<td>( \text{wp}[C_1] \left( \text{wp}[C_2] \ll f, g \right) )</td>
</tr>
<tr>
<td>if (( \xi )) { ( C_1 ) } else { ( C_2 ) }</td>
<td>( \ll \xi \cdot \text{wp}[C_1] \ll f, g + \ll \neg \xi \cdot \text{wp}[C_2] \ll f, g )</td>
</tr>
<tr>
<td>while (( \xi )) { ( C' ) }</td>
<td>( \lim_{n \to \omega} F_{\ll f, g}^n \ll 0, 0 )</td>
</tr>
</tbody>
</table>

\[
F_{\ll f, g} \ll X, Y \\ll = \ll \neg \xi \cdot \ll f, g + \ll \xi \cdot \text{wp}[C'] \ll X, Y
\]
Basic idea:

If \( \sum a_i \) absolutely convergent, then

\[
\sum a_i = \sum (|a_i| + a_i) - \sum |a_i|
\]

Apply this principle to \( f, g \):

\[
F_n H f, g I H 0, 0 I = H F_n |f| + f(0) - F_n |f|(0), F_n g(0) I
\]

So how to over-approximate \( \lim_{n \to \omega} F_n H f, g I H 0, 0 I \) w.r.t. \( \preceq \)?
Backup Slides: Reasoning about While–Loops

- Basic idea: \( \sum a_i \) absolutely convergent if \( \sum |a_i| \) convergent
Backup Slides: Reasoning about While–Loops

- Basic idea: $\sum a_i$ absolutely convergent if $\sum |a_i|$ convergent
- If $\sum a_i$ abs. conv., then $\sum a_i = \sum (|a_i| + a_i) - \sum |a_i|$
Backup Slides: Reasoning about While–Loops

- Basic idea: $\sum a_i$ absolutely convergent if $\sum |a_i|$ convergent
- If $\sum a_i$ abs. conv., then $\sum a_i = \sum (|a_i| + a_i) - \sum |a_i|$
- Apply this principle to $F_{\ell f, g}^{n}$:

$$F_{\ell f, g}^{n}(0, 0) = \bigcup F_{|f|+f}^{n}(0) - F_{|f|}^{n}(0), \quad F_{g}^{n}(0)$$
Backup Slides: Reasoning about While–Loops

- Basic idea: $\sum a_i$ absolutely convergent if $\sum |a_i|$ convergent
- If $\sum a_i$ abs. conv., then $\sum a_i = \sum (|a_i| + a_i) - \sum |a_i|
- Apply this principle to $F_{f,g}^n$: 

$$F_{f,g}^n(0, 0) = \{ F_{|f|+f(0)}^n(0), F_g^n(0) \}$$

- So how to over–approximate $\lim_{n \to \omega} F_{f,g}^n(0, 0)$ w.r.t. $\sqsubseteq$?
Backup Slides: Reasoning about While–Loops

- Basic idea: $\sum a_i$ absolutely convergent if $\sum |a_i|$ convergent
- If $\sum a_i$ abs. conv., then $\sum a_i = \sum (|a_i| + a_i) - \sum |a_i|
- Apply this principle to $F_{f,g}^n$: $F_{f,g}^n(0,0) = \mathcal{L} F_{|f|+f}^n(0) - F_{|f|}^n(0), F_g^n(0) \mathcal{L}$
- So how to over–approximate $\lim_{n \to \omega} F_{f,g}^n(0,0) \mathcal{L}$ w.r.t. $\sqsubseteq$?
- Over–approximate $\sup_n F_{|f|+f}^n(0)$
Backup Slides: Reasoning about While–Loops

- Basic idea: $\sum a_i$ absolutely convergent if $\sum |a_i|$ convergent.
- If $\sum a_i$ abs. conv., then $\sum a_i = \sum (|a_i| + a_i) - \sum |a_i|$
- Apply this principle to $F_{f,g}^\ell$:

$$F_{f,g}^\ell(0, 0) = \bigcap F_{|f| + f}(0) - F_{|f|}(0), \quad F_g(0)$$

- So how to over–approximate $\lim_{n \to \omega} F_{f,g}^\ell(0, 0)$ w.r.t. $\sqsubseteq$?
  - Over–approximate $\sup_n F_{|f| + f}(0)$
  - Under–approximate $\sup_n F_{|f|}(0)$
Backup Slides: Reasoning about While–Loops

- Basic idea: \( \sum a_i \) absolutely convergent if \( \sum |a_i| \) convergent
- If \( \sum a_i \) abs. conv., then \( \sum a_i = \sum (|a_i| + a_i) - \sum |a_i| \)
- Apply this principle to \( F_{\ell f, g}^n I H 0, 0 I \): 

\[
F_{\ell f,g}^n \lhd 0, 0 \rhd = \bigcup \left[ F_{|f| + f(0)}^n - F_{|f|}^n(0) , F_g^n(0) \right]
\]

- So how to over–approximate \( \lim_{n \to \omega} F_{\ell f,g}^n \lhd 0, 0 \rhd \) w.r.t. \( \sqsubseteq \)?
  - Over–approximate \( \sup_n F_{|f| + f(0)}^n \)
  - Under–approximate \( \sup_n F_{|f|}^n(0) \)
  - Over–approximate \( \sup_n F_g^n(0) \)
Backup Slides: Reasoning about While–Loops

Reminder: \( F^n_{\xi f, g}(0, 0) = \left\{ F^n_{|f|+f}(0) - F^n_{|f|}(0), F^n_g(0) \right\} \)
Reminder: \( F^n_{l_f, g} (\{0, 0\}) = \bigcap F^n_{|f|+f} (0) - F^n_{|f|} (0), F^n_{g} (0) \bigcap \)

**Invariant Rule for While–Loops**

Let \( I, G \in \mathbb{E} \) and \( \{H_n\}_{n \in \mathbb{N}} \subseteq \mathbb{E} \). If

\[
F_{|f|+f} (I) \leq I, \quad F_{g} (G) \leq G,
\]

\[
H_0 \leq F_{|f|} (0), \quad \text{and} \quad H_{n+1} \leq F_{|f|} (H_n),
\]

then

\[
\wp[\text{while} (\xi) \{C'\}] \{ f, g \} \subseteq \bigcap I - \sup_n H_n, 2 \cdot G \bigcap .
\]
Recall the definition of \( \text{wp} \):

\[
\text{wp}\left[ C \right] (X) = \text{lfp} (X \cdot J \neg \xi K \cdot f + J \xi K \cdot \text{wp}\left[ C \right] (X))
\]

**Theorem: Upper Bounds from Upper Invariants**

Let \( I \in \mathcal{E} \).

\[
F f(I) \leq I \implies \text{wp}\left[ \text{while} (\xi) \{ C \} \right] (f) \leq I.
\]
Backup Slides: Upper Bounds for \( \text{wp} \) of While–Loops

Recall the definition of \( \text{wp} [\text{while} (\xi) \{ C \}] (f) \):

\[
\text{lfp} \ X \cdot [\neg \xi] \cdot f + [\xi] \cdot \text{wp} [C] (X)
\]
Backup Slides: Upper Bounds for wp of While–Loops

Recall the definition of \( \text{wp} \left[ \text{while} (\xi) \{ C \} \right] (f) \):

\[
\text{lfp } X \cdot \left[ \neg \xi \right] \cdot f + \left[ \xi \right] \cdot \text{wp} [C] (X) =: F_f (X)
\]
Backup Slides: Upper Bounds for wp of While–Loops

Recall the definition of \( \text{wp} [\text{while} (\xi) \{C\}] (f) \):

\[
\text{lfp } X \cdot \underbrace{\lnot \xi \cdot f + \llbracket \xi \rrbracket \cdot \text{wp}[C](X)}_{=: F_f(X)}
\]

Theorem: Upper Bounds from Upper Invariants

Let \( I \in \mathbb{E} \).
Recall the definition of $\text{wp} \left[ \text{while} \left( \xi \right) \{C\} \right] (f)$:

$$\text{lfp} X \cdot \lfloor \neg \xi \rfloor \cdot f + \lfloor \xi \rfloor \cdot \text{wp} \left[ C \right] (X) =: F_f(X)$$

**Theorem: Upper Bounds from Upper Invariants**

Let $I \in \mathbb{E}$. Then

$$F_f(I) \leq I$$
Recall the definition of \( \text{wp} [\text{while} (\xi) \{ C \}] (f) \):

\[
\text{lfp } X \cdot \lbrack \neg \xi \rbrack \cdot f + \lbrack \xi \rbrack \cdot \text{wp} [C] (X) =: F_f(X)
\]

**Theorem: Upper Bounds from Upper Invariants**

Let \( I \in \mathbb{E} \). Then

\[
F_f(I) \leq I \implies \text{wp} [\text{while} (\xi) \{ C \}] (f) \leq I.
\]
Backup Slides: Lower Bounds for \( wp \) of While–Loops

Reasoning on lower bounds is more involved:
Find an argument for being below a least fixed point!

**Theorem:** Lower Bounds from Lower \( \omega \)–Invariants

Let 
\[
\{ I_n \}_{n \in \mathbb{N}} \subseteq E
\]

Then 
\[
I_0 \leq F(f)(0) \quad \text{and} \quad I_{n+1} \leq F(f)(I_n)
\]

implies 
\[
\sup_{n \in \mathbb{N}} I_n \leq wp[while(\xi)\{C\}](f).
\]
Backup Slides: Lower Bounds for wp of While–Loops

Reasoning on lower bounds is more involved:

Find an argument for being below a least fixed point!
Reasoning on lower bounds is more involved:

Find an argument for being **below a least fixed point**!

**Theorem: Lower Bounds from Lower $\omega$–Invariants**

Let $\{I_n\}_{n \in \mathbb{N}} \subseteq \mathbb{E}$. 
Backup Slides: Lower Bounds for wp of While–Loops

Reasoning on lower bounds is more involved:

Find an argument for being below a least fixed point!

**Theorem: Lower Bounds from Lower \( \omega \)–Invariants**

Let \( \{I_n\}_{n \in \mathbb{N}} \subseteq \mathbb{E} \). Then

\[
I_0 \leq F_f(0) \quad \text{and} \quad I_{n+1} \leq F_f(I_n)
\]
Backup Slides: Lower Bounds for wp of While–Loops

Reasoning on lower bounds is more involved:

Find an argument for being below a least fixed point!

Theorem: Lower Bounds from Lower $\omega$–Invariants

Let $\{I_n\}_{n \in \mathbb{N}} \subseteq \mathcal{E}$. Then

$$I_0 \leq F_f(0) \quad \text{and} \quad I_{n+1} \leq F_f(I_n)$$

implies

$$\sup_{n \in \mathbb{N}} I_n \leq \text{wp}[\text{while}(\xi)\{C\}](f).$$
Backup Slides: Park’s Lemma
Backup Slides: Park’s Lemma

\[ lfp F \leq I \]

\[ F(I) \leq I \]

\[ gfp F \]

\[ \infty \]

\[ 0 \]

\[ I \]
Backup Slides: Park’s Lemma

The diagram illustrates the concept of fixed points in the context of Park’s Lemma. The notation "\( F(I) \)" and "\( I \)" are used to denote specific values, and the fixed points are represented as "lfp \( F \)" and "gfp \( F \)". The diagram visually propagates these values through a series of mapped points, indicating the flow of information or state changes in the context of the lemma.
Backup Slides: Park’s Lemma

\[ F(I) \leq I \quad \text{implies} \quad \text{lfp } F \leq I \]
Backup Slides: Park’s Lemma

\[ F(I) \leq I \quad \text{implies} \quad \text{lfp } F \leq I \]
Backup Slides: Park’s Lemma

\[ F(I) \leq I \quad \text{implies} \quad \text{lfp } F \leq I \]
PPDL’s While Rule [Kozen ’85, p. 168]

\[ \neg B \cdot f + B \cdot \langle C \rangle I \preceq I \text{ implies } \langle \text{while}(B)\{C\}\rangle f \preceq I , \]

for \( f \geq 0 \).