Automatic Verification of Probabilistic Programs

Joost-Pieter Katoen & **Kevin Batz**

Lecture on Probabilistic Programming

Automatic Verification of Probabilistic Loops

How and **in which cases** can we automate the wp-calculus?

Automatic Verification of Probabilistic Loops

How and **in which cases** can we automate the wp-calculus?

What about the ert-calculus for expected runtimes?

Automatic Verification of Probabilistic Loops

How and **in which cases** can we automate the wp-calculus?

What about the ert-calculus for expected runtimes?

How: Quantitative Loop Invariants.

Automatic Verification of Probabilistic Loops

How and **in which cases** can we automate the wp-calculus?

What about the ert-calculus for expected runtimes?

How: Quantitative Loop Invariants.

In which cases: Restrict to a certain class of probabilistic loops and expectations.

Automatic Verification of Probabilistic Loops

How and **in which cases** can we automate the wp-calculus?

What about the ert-calculus for expected runtimes?

How: Quantitative Loop Invariants.

In which cases: Restrict to a certain class of probabilistic loops and expectations.

Motivation:

Verification of **Randomized Algorithms** and **Stochastic Processes**

The Bounded Retransmission Protocol [Helmink *et al.* '93, D'Argenio *et al.* '97]

Goal: Sent file of N packets via lossy channel.

The Bounded Retransmission Protocol **[Helmink *et al.* '93, D'Argenio *et al.* '97]**

Goal: Sent file of N packets via lossy channel.

Transmitting a single packet fails with some probability, say 0.01.

The Bounded Retransmission Protocol [Helmink *et al.* '93, D'Argenio *et al.* '97]

Goal: Sent file of N packets via lossy channel.

Transmitting a single packet fails with some probability, say 0.01.

Allow for at most F retransmission tries per packet; otherwise fail.

The Bounded Retransmission Protocol [Helmink *et al.* '93, D'Argenio *et al.* '97]

Goal: Sent file of N packets via lossy channel.

Transmitting a single packet fails with some probability, say 0.01.

Allow for at most F retransmission tries per packet; otherwise fail.

$sent := 0;\ fail := 0;\ totalFail := 0;$

The Bounded Retransmission Protocol [Helmink *et al.* '93, D'Argenio *et al.* '97]

Goal: Sent file of N packets via lossy channel.

Transmitting a single packet fails with some probability, say 0.01.

Allow for at most F retransmission tries per packet; otherwise fail.

$sent := 0;\ fail := 0;\ totalFail := 0;$

$\texttt{while}\,(\,sent < N \wedge fail < F\,)\,\{$

The Bounded Retransmission Protocol [Helmink *et al.* '93, D'Argenio *et al.* '97]

Goal: Sent file of N packets via lossy channel.

Transmitting a single packet fails with some probability, say 0.01.

Allow for at most F retransmission tries per packet; otherwise fail.

$$sent := 0;\ fail := 0;\ totalFail := 0;$$

$$\texttt{while}\,(\,sent < N \wedge fail < F\,)\,\{$$

$$\{\ \underbrace{fail := fail + 1\,;\,totalFail := totalFail + 1}_{\text{failed transmission}}\ \}\,[0.01]$$

The Bounded Retransmission Protocol [Helmink *et al.* '93, D'Argenio *et al.* '97]

Goal: Sent file of N packets via lossy channel.

Transmitting a single packet fails with some probability, say 0.01.

Allow for at most F retransmission tries per packet; otherwise fail.

$$sent := 0;\ fail := 0;\ totalFail := 0;$$

$$\texttt{while}\,(\,sent < N \wedge fail < F\,)\,\{$$

$$\{\ \underbrace{fail := fail + 1\,;\, totalFail := totalFail + 1}_{\text{failed transmission}}\ \}[0.01]\{\ \underbrace{fail := 0\,;\, sent := sent + 1}_{\text{successful transmission}}\ \}\}$$

The Bounded Retransmission Protocol [Helmink *et al.* '93, D'Argenio *et al.* '97]

Goal: Sent file of N packets via lossy channel.

Transmitting a single packet fails with some probability, say 0.01.

Allow for at most F retransmission tries per packet; otherwise fail.

$$sent := 0;\ fail := 0;\ totalFail := 0;$$

$$\texttt{while}\,(\,sent < N \wedge fail < F\,)\,\{$$

$$\{\ \underbrace{fail := fail + 1\,;\, totalFail := totalFail + 1}_{\text{failed transmission}}\ \}[0.01]\{\ \underbrace{fail := 0\,;\, sent := sent + 1}_{\text{successful transmission}}\ \}\}$$

Is the probability of failing to transmit file (reaching $fail = F$) at most 0.0001?

The Bounded Retransmission Protocol [Helmink *et al.* '93, D'Argenio *et al.* '97]

Goal: Sent file of N packets via lossy channel.

Transmitting a single packet fails with some probability, say 0.01.

Allow for at most F retransmission tries per packet; otherwise fail.

$$sent := 0;\ fail := 0;\ totalFail := 0;$$

$$\texttt{while}\,(\,sent < N \wedge fail < F\,)\,\{$$

$$\{\ \underbrace{fail := fail + 1\,;\, totalFail := totalFail + 1}_{\text{failed transmission}}\ \}[0.01]\{\ \underbrace{fail := 0\,;\, sent := sent + 1}_{\text{successful transmission}}\ \}\}$$

Is the probability of failing to transmit file (reaching $fail = F$) at most 0.0001?

Is the expected number of retransmission tries ($totalFail$) at most 10?

Optimal Discrete Uniform Generation from Coin Flips [Lumbroso '13]

Sample uniformly from $\{0, \ldots, n-1\}$ using fair coin flips only.

Optimal Discrete Uniform Generation from Coin Flips [Lumbroso '13]

Sample uniformly from $\{0, \ldots, n-1\}$ using fair coin flips only.

$v \coloneqq 1\,;\, c \coloneqq 0\,;\, \mathit{term} \coloneqq 0\,;$

`while` $(\,\mathit{term} = 0\,)\,\{$

$\quad v \coloneqq 2 \cdot v\,;$

$\quad \{\, c \coloneqq 2 \cdot c \,\}\ [1/2]\ \{\, c \coloneqq 2 \cdot c + 1 \,\}\,;$

$\quad$ `if` $(\,v \geq n\,)\,\{$

$\quad\quad$ `if` $(\,c < n\,)\,\{\, \mathit{term} \coloneqq 1 \,\}$

$\quad\quad$ `else` $\{\, v \coloneqq v - n\,;\, c \coloneqq c - n \,\}$

$\quad \}$

$\}$

Optimal Discrete Uniform Generation from Coin Flips [Lumbroso '13]

Sample uniformly from $\{0, \ldots, n-1\}$ using fair coin flips only.

$v \coloneqq 1\,;\, c \coloneqq 0\,;\, term \coloneqq 0\,;$

`while` $(\, term = 0 \,)\, \{$

$\quad v \coloneqq 2 \cdot v\,;$

$\quad \{\, c \coloneqq 2 \cdot c \,\}\ [1/2]\ \{\, c \coloneqq 2 \cdot c + 1 \,\}\,;$

$\quad$ `if` $(\, v \geq n \,)\, \{$

$\quad\quad$ `if` $(\, c < n \,)\, \{\, term \coloneqq 1 \,\}$

$\quad\quad$ `else` $\{\, v \coloneqq v - n\,;\, c \coloneqq c - n \,\}$

$\quad \}$

$\}$

Is $\Pr(c = i) = 1/n$ for all $i \in \{0, \ldots, n-1\}$?

Weakest Preexpectation Reasoning

Weakest Preexpectation Reasoning

What is the expected value of program variable x after termination?

Weakest Preexpectation Reasoning

What is the expected value of program variable x after termination?

What is the probability of reaching a state $fail = F$?

Weakest Preexpectation Reasoning

What is the expected value of program variable x after termination?

What is the probability of reaching a state $fail = F$?

What is the termination probability?

Recap: Weakest Preexpectations & Loop Invariants

Consider the complete lattice $(\mathbb{E}, \sqsubseteq)$ of *expectations*:

$$\mathbb{E} = \{f \mid f \colon \mathsf{States} \to \mathbb{R}^{\infty}_{\geq 0}\}$$

Recap: Weakest Preexpectations & Loop Invariants

Consider the complete lattice $(\mathbb{E}, \sqsubseteq)$ of *expectations*:

$$\mathbb{E} = \{f \mid f\colon \mathsf{States} \to \mathbb{R}^{\infty}_{\geq 0}\} \qquad \text{with} \qquad f \sqsubseteq g \quad \text{iff} \quad \forall \sigma \in \mathsf{States}\colon f(\sigma) \leq g(\sigma)$$

Recap: Weakest Preexpectations & Loop Invariants

Consider the complete lattice $(\mathbb{E}, \sqsubseteq)$ of *expectations*:

$$\mathbb{E} = \{f \mid f\colon \mathsf{States} \to \mathbb{R}^{\infty}_{\geq 0}\} \qquad \text{with} \qquad f \sqsubseteq g \quad \text{iff} \quad \forall \sigma \in \mathsf{States}\colon f(\sigma) \leq g(\sigma)$$

Weakest preexpectation transformer:

$\mathsf{wp}[\![C]\!]\colon \mathbb{E} \to \mathbb{E}$

Recap: Weakest Preexpectations & Loop Invariants

Consider the complete lattice $(\mathbb{E}, \sqsubseteq)$ of *expectations*:

$$\mathbb{E} = \{f \mid f\colon \mathsf{States} \to \mathbb{R}^{\infty}_{\geq 0}\} \qquad \text{with} \qquad f \sqsubseteq g \quad \text{iff} \quad \forall \sigma \in \mathsf{States}\colon f(\sigma) \leq g(\sigma)$$

Weakest preexpectation transformer:

$$\mathsf{wp}[\![C]\!]\colon \mathbb{E} \to \mathbb{E} \qquad \mathsf{wp}[\![C]\!]\,(f)$$

Recap: Weakest Preexpectations & Loop Invariants

Consider the complete lattice $(\mathbb{E}, \sqsubseteq)$ of *expectations*:

$$\mathbb{E} = \{f \mid f\colon \mathsf{States} \to \mathbb{R}^{\infty}_{\geq 0}\} \qquad \text{with} \qquad f \sqsubseteq g \quad \text{iff} \quad \forall \sigma \in \mathsf{States}\colon f(\sigma) \leq g(\sigma)$$

Weakest preexpectation transformer:

$$\mathsf{wp}[\![C]\!]\colon \mathbb{E} \to \mathbb{E} \qquad \mathsf{wp}[\![C]\!]\,(f)\,(\sigma)$$

Recap: Weakest Preexpectations & Loop Invariants

Consider the complete lattice $(\mathbb{E}, \sqsubseteq)$ of *expectations*:

$$\mathbb{E} = \{f \mid f\colon \mathsf{States} \to \mathbb{R}^{\infty}_{\geq 0}\} \qquad \text{with} \qquad f \sqsubseteq g \quad \text{iff} \quad \forall \sigma \in \mathsf{States}\colon f(\sigma) \leq g(\sigma)$$

Weakest preexpectation transformer:

$$\mathsf{wp}[\![C]\!]\colon \mathbb{E} \to \mathbb{E} \qquad \mathsf{wp}[\![C]\!]\,(f)\,(\sigma) \;=\; \text{expected value of } f \text{ w.r.t. distribution of final states reached after executing } C \text{ on } \sigma$$

Recap: Weakest Preexpectations & Loop Invariants

Consider the complete lattice $(\mathbb{E}, \sqsubseteq)$ of *expectations*:

$$\mathbb{E} = \{f \mid f\colon \mathsf{States} \to \mathbb{R}^{\infty}_{\geq 0}\} \qquad \text{with} \qquad f \sqsubseteq g \quad \text{iff} \quad \forall \sigma \in \mathsf{States}\colon f(\sigma) \leq g(\sigma)$$

Weakest preexpectation transformer:

$$\mathsf{wp}[\![C]\!]\colon \mathbb{E} \to \mathbb{E} \qquad \mathsf{wp}[\![C]\!]\,(f)\,(\sigma) \;=\; \begin{array}{c}\text{expected value of } f \text{ w.r.t. distribution of final states}\\ \text{reached after executing } C \text{ on } \sigma\end{array}$$

In this lecture, we restrict to **computable** postexpectations $f\colon \mathsf{States} \to \mathbb{Q}^{\infty}_{\geq 0}$.

Recap: Weakest Preexpectations & Loop Invariants

Consider the complete lattice $(\mathbb{E}, \sqsubseteq)$ of *expectations*:

$$\mathbb{E} = \{f \mid f\colon \mathsf{States} \to \mathbb{R}^{\infty}_{\geq 0}\} \qquad \text{with} \qquad f \sqsubseteq g \quad \text{iff} \quad \forall \sigma \in \mathsf{States}\colon f(\sigma) \leq g(\sigma)$$

Weakest preexpectation transformer:

$$\mathsf{wp}[\![C]\!]\colon \mathbb{E} \to \mathbb{E} \qquad \mathsf{wp}[\![C]\!]\,(f)\,(\sigma) \;=\; \begin{array}{c}\text{expected value of } f \text{ w.r.t. distribution of final states}\\ \text{reached after executing } C \text{ on } \sigma\end{array}$$

In this lecture, we restrict to **computable** postexpectations $f\colon \mathsf{States} \to \mathbb{Q}^{\infty}_{\geq 0}$.

$$\mathsf{wp}[\![\{\,\texttt{skip}\,\}\ [\,1/2\,]\ \{\,x \coloneqq x+2\,\}]\!]\,(x)$$

Recap: Weakest Preexpectations & Loop Invariants

Consider the complete lattice $(\mathbb{E}, \sqsubseteq)$ of *expectations*:

$$\mathbb{E} = \{f \mid f\colon \mathsf{States} \to \mathbb{R}^{\infty}_{\geq 0}\} \qquad \text{with} \qquad f \sqsubseteq g \quad \text{iff} \quad \forall \sigma \in \mathsf{States}\colon f(\sigma) \leq g(\sigma)$$

Weakest preexpectation transformer:

$$\mathsf{wp}[\![C]\!]\colon \mathbb{E} \to \mathbb{E} \qquad \mathsf{wp}[\![C]\!]\,(f)\,(\sigma) \;=\; \text{expected value of } f \text{ w.r.t. distribution of final states reached after executing } C \text{ on } \sigma$$

In this lecture, we restrict to **computable** postexpectations $f\colon \mathsf{States} \to \mathbb{Q}^{\infty}_{\geq 0}$.

$$\mathsf{wp}[\![\{\,\texttt{skip}\,\}\;[\,1/2\,]\;\{\,x \coloneqq x+2\,\}]\!]\,(x) \;=\; 1/2 \cdot x + 1/2 \cdot (x+2)$$

Recap: Weakest Preexpectations & Loop Invariants

Consider the complete lattice $(\mathbb{E}, \sqsubseteq)$ of *expectations*:

$$\mathbb{E} = \{f \mid f\colon \mathsf{States} \to \mathbb{R}^{\infty}_{\geq 0}\} \qquad \text{with} \qquad f \sqsubseteq g \quad \text{iff} \quad \forall \sigma \in \mathsf{States}\colon f(\sigma) \leq g(\sigma)$$

Weakest preexpectation transformer:

$$\mathsf{wp}[\![C]\!]\colon \mathbb{E} \to \mathbb{E} \qquad \mathsf{wp}[\![C]\!]\,(f)\,(\sigma) \;=\; \text{expected value of } f \text{ w.r.t. distribution of final states reached after executing } C \text{ on } \sigma$$

In this lecture, we restrict to **computable** postexpectations $f\colon \mathsf{States} \to \mathbb{Q}^{\infty}_{\geq 0}$.

$$\mathsf{wp}[\![\{\,\texttt{skip}\,\}\ [1/2]\ \{\,x \coloneqq x+2\,\}]\!]\,(x) \;=\; 1/2 \cdot x + 1/2 \cdot (x+2) \;=\; x+1$$

Recap: Weakest Preexpectations & Loop Invariants

Consider the complete lattice $(\mathbb{E}, \sqsubseteq)$ of *expectations*:

$$\mathbb{E} = \{f \mid f\colon \mathsf{States} \to \mathbb{R}^{\infty}_{\geq 0}\} \qquad \text{with} \qquad f \sqsubseteq g \quad \text{iff} \quad \forall \sigma \in \mathsf{States}\colon f(\sigma) \leq g(\sigma)$$

Weakest preexpectation transformer:

$$\mathsf{wp}[\![C]\!]\colon \mathbb{E} \to \mathbb{E} \qquad \mathsf{wp}[\![C]\!]\,(f)\,(\sigma) \;=\; \text{expected value of } f \text{ w.r.t. distribution of final states reached after executing } C \text{ on } \sigma$$

In this lecture, we restrict to **computable** postexpectations $f\colon \mathsf{States} \to \mathbb{Q}^{\infty}_{\geq 0}$.

$$\mathsf{wp}[\![\{\,\mathtt{skip}\,\}\ [1/2]\ \{\,x := x+2\,\}]\!]\,(x) \;=\; 1/2 \cdot x + 1/2 \cdot (x+2) \;=\; x+1$$

$$\mathsf{wp}[\![\{\,\mathtt{skip}\,\}\ [1/2]\ \{\,x := x+2\,\}]\!]\,([x=4])$$

Recap: Weakest Preexpectations & Loop Invariants

Consider the complete lattice $(\mathbb{E}, \sqsubseteq)$ of *expectations*:

$$\mathbb{E} = \{f \mid f\colon \mathsf{States} \to \mathbb{R}^{\infty}_{\geq 0}\} \qquad \text{with} \qquad f \sqsubseteq g \quad \text{iff} \quad \forall \sigma \in \mathsf{States}\colon f(\sigma) \leq g(\sigma)$$

Weakest preexpectation transformer:

$$\mathsf{wp}[\![C]\!]\colon \mathbb{E} \to \mathbb{E} \qquad \mathsf{wp}[\![C]\!]\,(f)\,(\sigma) \;=\; \text{expected value of } f \text{ w.r.t. distribution of final states reached after executing } C \text{ on } \sigma$$

In this lecture, we restrict to **computable** postexpectations $f\colon \mathsf{States} \to \mathbb{Q}^{\infty}_{\geq 0}$.

$$\mathsf{wp}[\![\{\,\mathtt{skip}\,\}\ [1/2]\ \{\,x := x+2\,\}]\!]\,(x) \;=\; 1/2 \cdot x + 1/2 \cdot (x+2) \;=\; x+1$$

$$\mathsf{wp}[\![\{\,\mathtt{skip}\,\}\ [1/2]\ \{\,x := x+2\,\}]\!]\,([x=4]) \;=\; 1/2 \cdot [x=4] + 1/2 \cdot [x=2]$$

Recap: Weakest Preexpectations & Loop Invariants

Consider the complete lattice $(\mathbb{E}, \sqsubseteq)$ of *expectations*:

$$\mathbb{E} = \{ f \mid f \colon \mathsf{States} \to \mathbb{R}^{\infty}_{\geq 0} \} \qquad \text{with} \qquad f \sqsubseteq g \quad \text{iff} \quad \forall \sigma \in \mathsf{States} \colon f(\sigma) \leq g(\sigma)$$

Weakest preexpectation transformer:

$$\mathsf{wp}[\![C]\!] \colon \mathbb{E} \to \mathbb{E} \qquad \mathsf{wp}[\![C]\!]\,(f)\,(\sigma) \;=\; \text{expected value of } f \text{ w.r.t. distribution of final states reached after executing } C \text{ on } \sigma$$

In this lecture, we restrict to **computable** postexpectations $f \colon \mathsf{States} \to \mathbb{Q}^{\infty}_{\geq 0}$.

$$\mathsf{wp}[\![\{\,\texttt{skip}\,\}\ [1/2]\ \{\,x \coloneqq x+2\,\}]\!]\,(x) \;=\; 1/2 \cdot x + 1/2 \cdot (x+2) \;=\; x+1$$

$$\mathsf{wp}[\![\{\,\texttt{skip}\,\}\ [1/2]\ \{\,x \coloneqq x+2\,\}]\!]\,([x=4]) \;=\; 1/2 \cdot [x=4] + 1/2 \cdot [x=2]$$

$$\mathsf{wp}[\![\texttt{while}\,(\,c=1\,)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x)$$

Recap: Weakest Preexpectations & Loop Invariants

Consider the complete lattice $(\mathbb{E}, \sqsubseteq)$ of *expectations*:

$$\mathbb{E} = \{ f \mid f \colon \mathsf{States} \to \mathbb{R}^{\infty}_{\geq 0} \} \qquad \text{with} \qquad f \sqsubseteq g \quad \text{iff} \quad \forall \sigma \in \mathsf{States} \colon f(\sigma) \leq g(\sigma)$$

Weakest preexpectation transformer:

$$\mathsf{wp}[\![C]\!] \colon \mathbb{E} \to \mathbb{E} \qquad \mathsf{wp}[\![C]\!]\,(f)\,(\sigma) \;=\; \text{expected value of } f \text{ w.r.t. distribution of final states reached after executing } C \text{ on } \sigma$$

In this lecture, we restrict to **computable** postexpectations $f \colon \mathsf{States} \to \mathbb{Q}^{\infty}_{\geq 0}$.

$$\begin{aligned}
\mathsf{wp}[\![\{\,\texttt{skip}\,\}\ [1/2]\ \{\,x \coloneqq x+2\,\}]\!]\,(x) &= 1/2 \cdot x + 1/2 \cdot (x+2) = x+1 \\
\mathsf{wp}[\![\{\,\texttt{skip}\,\}\ [1/2]\ \{\,x \coloneqq x+2\,\}]\!]\,([x=4]) &= 1/2 \cdot [x=4] + 1/2 \cdot [x=2] \\
\mathsf{wp}[\![\texttt{while}\,(\,c=1\,)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x) &= [c=1] \cdot (x+1) + [c \neq 1] \cdot x
\end{aligned}$$

C	$\mathsf{wp}\,[\![C]\!]\,(f)$
`skip`	f
$x \coloneqq a$	$f\,[x/a]$
$C_1\,;\,C_2$	$\mathsf{wp}[\![C_1]\!]\,(\mathsf{wp}[\![C_2]\!]\,(f))$
$\{\,C_1\,\}\,[\,p\,]\,\{\,C_2\,\}$	$p \cdot \mathsf{wp}[\![C_1]\!]\,(f) + (1-p) \cdot \mathsf{wp}[\![C_2]\!]\,(f)$
`if` $(\,\varphi\,)\,\{\,C_1\,\}$ `else` $\{\,C_2\,\}$	$[\varphi] \cdot \mathsf{wp}[\![C_1]\!]\,(f) + [\neg\varphi] \cdot \mathsf{wp}[\![C_2]\!]\,(f)$
`while` $(\,\varphi\,)\,\{\,body\,\}$	$\mathsf{lfp}\,h.\ [\neg\varphi] \cdot f + [\varphi] \cdot \mathsf{wp}[\![body]\!]\,(h)$

The lfp is taken w.r.t. $\sqsubseteq$ on expectations.

Recap: Weakest Preexpectations & Loop Invariants

C	$\mathsf{wp}\,[\![C]\!]\,(f)$
`skip`	f
$x \coloneqq a$	$f\,[x/a]$
$C_1\,;\,C_2$	$\mathsf{wp}[\![C_1]\!]\,(\mathsf{wp}[\![C_2]\!]\,(f))$
$\{\,C_1\,\}\,[\,p\,]\,\{\,C_2\,\}$	$p \cdot \mathsf{wp}[\![C_1]\!]\,(f) + (1-p) \cdot \mathsf{wp}[\![C_2]\!]\,(f)$
`if` $(\,\varphi\,)\,\{\,C_1\,\}$ `else` $\{\,C_2\,\}$	$[\varphi] \cdot \mathsf{wp}[\![C_1]\!]\,(f) + [\neg\varphi] \cdot \mathsf{wp}[\![C_2]\!]\,(f)$
`while` $(\,\varphi\,)\,\{\,body\,\}$	$\mathsf{lfp}\ h.\ [\neg\varphi] \cdot f + [\varphi] \cdot \mathsf{wp}[\![body]\!]\,(h)$

The lfp is taken w.r.t. $\sqsubseteq$ on expectations.

$$
\begin{array}{l}
\texttt{if}\ (y = 5)\ \{ \\
\qquad \{\,\texttt{skip}\,\}\ [1/2]\ \{\, x \coloneqq x + 2 \,\} \\
\}\ \texttt{else}\ \{ \\
\qquad x \coloneqq 5 \\
\}
\end{array}
$$

Recap: Weakest Preexpectations & Loop Invariants

$$
\begin{array}{l}
\texttt{if}\ (y = 5)\ \{ \\
\qquad \{\,\texttt{skip}\,\}\ [1/2]\ \{\, x \coloneqq x + 2 \,\} \\
\}\ \texttt{else}\ \{ \\
\qquad x \coloneqq 5 \\
\}
\end{array}
$$

$/\!/\!/\ x$ x is the postexpectation

Recap: Weakest Preexpectations & Loop Invariants

$$
\begin{array}{l}
\texttt{if}\ (y = 5)\ \{ \\
\qquad \{\,\texttt{skip}\,\}\ [1/2]\ \{\, x := x + 2 \,\} \\
\qquad /\!/\!/\ x \\
\}\ \texttt{else}\ \{ \\
\qquad x := 5 \\
\qquad /\!/\!/\ x \\
\} \\
/\!/\!/\ x
\end{array}
$$

x is the postexpectation

Recap: Weakest Preexpectations & Loop Invariants

$$
\begin{array}{l}
\texttt{if}\ (y = 5)\ \{ \\
\qquad \{\,\texttt{skip}\,\}\ [1/2]\ \{\,x := x + 2\,\} \\
\qquad /\!/\!/\ x \\
\}\,\texttt{else}\,\{ \\
\qquad /\!/\!/\ 5 \\
\qquad x := 5 \\
\qquad /\!/\!/\ x \\
\} \\
/\!/\!/\ x
\end{array}
$$

x is the postexpectation

Recap: Weakest Preexpectations & Loop Invariants

$$
\begin{array}{l}
\texttt{if}\ (y = 5)\ \{ \\
\quad /\!/\!/\ x + 1 \\
\quad \{\,\texttt{skip}\,\}\ [1/2]\ \{\,x := x + 2\,\} \\
\quad /\!/\!/\ x \\
\}\ \texttt{else}\ \{ \\
\quad /\!/\!/\ 5 \\
\quad x := 5 \\
\quad /\!/\!/\ x \\
\} \\
/\!/\!/\ x
\end{array}
$$

x is the postexpectation

Recap: Weakest Preexpectations & Loop Invariants

$/\!/\!/\ [y = 5] \cdot (x + 1) + [y \neq 5] \cdot 5$ this is $\mathsf{wp}[\![C]\!]\,(x)$

$\texttt{if}\ (\, y = 5\,)\ \{$

$\quad /\!/\!/\ x + 1$

$\quad \{\, \texttt{skip} \,\}\ [1/2]\ \{\, x \coloneqq x + 2 \,\}$

$\quad /\!/\!/\ x$

$\}\ \texttt{else}\ \{$

$\quad /\!/\!/\ 5$

$\quad x \coloneqq 5$

$\quad /\!/\!/\ x$

$\}$

$/\!/\!/\ x$ x is the postexpectation

Recap: Weakest Preexpectations & Loop Invariants

$/\!/\!/\ [y = 5] \cdot (x + 1) + [y \neq 5] \cdot 5$ this is $\mathsf{wp}[\![C]\!]\,(x)$

$\texttt{if}\ (\,y = 5\,)\ \{$

$\quad /\!/\!/\ x + 1$

$\quad \{\,\texttt{skip}\,\}\ [1/2]\ \{\,x := x + 2\,\}$

$\quad /\!/\!/\ x$

$\}\ \texttt{else}\ \{$

$\quad /\!/\!/\ 5$

$\quad x := 5$

$\quad /\!/\!/\ x$

$\}$

$/\!/\!/\ x$ x is the postexpectation

Observation: Determining wp for loop-free programs is automatable.

Recap: Weakest Preexpectations & Loop Invariants

C	$\mathsf{wp}\,[\![C]\!]\,(f)$
`skip`	f
$x \coloneqq a$	$f\,[x/a]$
$C_1\,;\,C_2$	$\mathsf{wp}[\![C_1]\!]\,(\mathsf{wp}[\![C_2]\!]\,(f))$
$\{\,C_1\,\}\,[\,p\,]\,\{\,C_2\,\}$	$p \cdot \mathsf{wp}[\![C_1]\!]\,(f) + (1-p) \cdot \mathsf{wp}[\![C_2]\!]\,(f)$
`if` $(\,\varphi\,)\,\{\,C_1\,\}$ `else` $\{\,C_2\,\}$	$[\varphi] \cdot \mathsf{wp}[\![C_1]\!]\,(f) + [\neg\varphi] \cdot \mathsf{wp}[\![C_2]\!]\,(f)$
`while` $(\,\varphi\,)\,\{\,body\,\}$	$\mathsf{lfp}\,h.\ [\neg\varphi] \cdot f + [\varphi] \cdot \mathsf{wp}[\![body]\!]\,(h)$

Observation: Determining wp for loop-free programs is automatable.

Recap: Weakest Preexpectations & Loop Invariants

C	$\mathsf{wp}\,[\![C]\!]\,(f)$
`skip`	f
$x \coloneqq a$	$f\,[x/a]$
$C_1\,;\,C_2$	$\mathsf{wp}[\![C_1]\!]\,(\mathsf{wp}[\![C_2]\!]\,(f))$
$\{\,C_1\,\}\,[\,p\,]\,\{\,C_2\,\}$	$p \cdot \mathsf{wp}[\![C_1]\!]\,(f) + (1-p) \cdot \mathsf{wp}[\![C_2]\!]\,(f)$
`if` $(\,\varphi\,)\,\{\,C_1\,\}$ `else` $\{\,C_2\,\}$	$[\varphi] \cdot \mathsf{wp}[\![C_1]\!]\,(f) + [\neg\varphi] \cdot \mathsf{wp}[\![C_2]\!]\,(f)$
`while` $(\,\varphi\,)\,\{\,body\,\}$	$\mathsf{lfp}\,h.\ [\neg\varphi] \cdot f + [\varphi] \cdot \mathsf{wp}[\![body]\!]\,(h)$

Observation: Determining wp for loop-free programs is automatable.

This does not hold for loops (Lec. 13). How to tackle the verification of loops?

Verifying Probabilistic Loops

... by means of quantitative loop invariants.

Recap: Weakest Preexpectations & Loop Invariants

Goal: Given computable expectations f, g, verify that $\mathtt{wp}[\![\mathtt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq g$.

Recap: Weakest Preexpectations & Loop Invariants

Goal: Given computable expectations f, g, verify that $\mathtt{wp}[\![\mathtt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq g$.

Example: $\mathtt{wp}[\![\text{bounded retransmission protocol}]\!]\,([fail = F]) \sqsubseteq 0.0001$

Goal: Given computable expectations f, g, verify that $\mathsf{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq g$.

Example: $\mathsf{wp}[\![\text{bounded retransmission protocol}]\!]\,([fail = F]) \sqsubseteq 0.0001$

Problem: for $\mathsf{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) = \mathsf{lfp}\,\Phi_f$, where

$$\Phi_f \colon \mathbb{E} \to \mathbb{E}, \quad \Phi_f(h) = [\neg\varphi] \cdot f + [\varphi] \cdot \mathsf{wp}[\![body]\!]\,(h)\ ,$$

Recap: Weakest Preexpectations & Loop Invariants

Goal: Given computable expectations f, g, verify that $\mathsf{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq g$.

Example: $\mathsf{wp}[\![\text{bounded retransmission protocol}]\!]\,([fail = F]) \sqsubseteq 0.0001$

Problem: for $\mathsf{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) = \mathsf{lfp}\,\Phi_f$, where

$$\Phi_f \colon \mathbb{E} \to \mathbb{E}, \quad \Phi_f(h) = [\neg\varphi] \cdot f + [\varphi] \cdot \mathsf{wp}[\![body]\!]\,(h)\ ,$$

$\mathsf{lfp}\,\Phi_f$ is uncomputable in general (Lec. 13).

Goal: Given computable expectations f, g, verify that $\mathsf{wp}[\![\texttt{while}\,(\varphi)\,\{\,body\,\}]\!]\,(f) \sqsubseteq g$.

Example: $\mathsf{wp}[\![\text{bounded retransmission protocol}]\!]\,([fail = F]) \sqsubseteq 0.0001$

Problem: for $\mathsf{wp}[\![\texttt{while}\,(\varphi)\,\{\,body\,\}]\!]\,(f) = \mathsf{lfp}\,\Phi_f$, where

$$\Phi_f \colon \mathbb{E} \to \mathbb{E}, \quad \Phi_f(h) = [\neg\varphi] \cdot f + [\varphi] \cdot \mathsf{wp}[\![body]\!]\,(h)\ ,$$

$\mathsf{lfp}\,\Phi_f$ is uncomputable in general (Lec. 13).

Solution: Quantitative Loop Invariants

Recap: Weakest Preexpectations & Loop Invariants

$\mathsf{wp}[\![\texttt{while}\,(\varphi)\,\{\,body\,\}]\!]\,(f) = \mathsf{lfp}\,\Phi_f$, where

$$\Phi_f \colon \mathbb{E} \to \mathbb{E}, \quad \Phi_f(h) = [\neg\varphi] \cdot f + [\varphi] \cdot \mathsf{wp}[\![body]\!]\,(h)$$

Theorem (Superinvariants for Loops)

Let $I \in \mathbb{E}$. We have

$$\underbrace{\Phi_f(I) \sqsubseteq I}_{I \text{ is wp-superinvariant}} \quad \textit{implies} \quad \mathsf{wp}[\![\texttt{while}\,(\varphi)\,\{\,body\,\}]\!]\,(f) \sqsubseteq I\,.$$

Recap: Weakest Preexpectations & Loop Invariants

$\mathsf{wp}[\![\texttt{while}\,(\varphi)\,\{\,body\,\}]\!]\,(f) = \mathsf{lfp}\,\Phi_f$, where

$$\Phi_f \colon \mathbb{E} \to \mathbb{E}, \quad \Phi_f(h) = [\neg\varphi] \cdot f + [\varphi] \cdot \mathsf{wp}[\![body]\!]\,(h)$$

Theorem (Superinvariants for Loops)

Let $I \in \mathbb{E}$. We have

$$\underbrace{\Phi_f(I) \sqsubseteq I}_{I \text{ is wp-superinvariant}} \quad \textit{implies} \quad \mathsf{wp}[\![\texttt{while}\,(\varphi)\,\{\,body\,\}]\!]\,(f) \sqsubseteq I\,.$$

Loop invariants reduce reasoning about loops to **reasoning about one loop iteration.**

Recap: Weakest Preexpectations & Loop Invariants

Goal: $\mathsf{wp}[\![\texttt{while}\,(c = 1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x + 1\,\}]\!]\,(x)$

Recap: Weakest Preexpectations & Loop Invariants

Goal: $\mathsf{wp}[\![\texttt{while}\,(c = 1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x + 1\,\}]\!]\,(x) \sqsubseteq [c \neq 1] \cdot x + [c = 1] \cdot (x + 1)$

Recap: Weakest Preexpectations & Loop Invariants

Goal: $\mathsf{wp}[\![\texttt{while}\,(c = 1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \sqsubseteq \underbrace{[c \neq 1]\cdot x + [c = 1]\cdot(x+1)}_{\text{choose as superinvariant } I}$

Recap: Weakest Preexpectations & Loop Invariants

Goal: $\mathsf{wp}[\![\texttt{while}\,(c = 1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x + 1\,\}]\!]\,(x) \sqsubseteq \underbrace{[c \neq 1] \cdot x + [c = 1] \cdot (x + 1)}_{\text{choose as superinvariant } I}$

$\Phi_x(I) = [c \neq 1] \cdot x + [c = 1] \cdot \mathsf{wp}[\![body]\!]\,(I)$

Recap: Weakest Preexpectations & Loop Invariants

Goal: $\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \sqsubseteq \underbrace{[c \neq 1]\cdot x + [c=1]\cdot(x+1)}_{\text{choose as superinvariant } I}$

$\texttt{while}\,(c=1)\,\{$

$\quad \{\,c \coloneqq 0\,\}\,[1/2]\,\{\,x \coloneqq x+1\,\}$

$\quad /\!/\!/\ [c=1]\cdot(x+1) + [c\neq 1]\cdot x$

$\}$

$\Phi_x(I) = [c \neq 1]\cdot x + [c=1]\cdot \mathsf{wp}[\![body]\!]\,(I)$

Recap: Weakest Preexpectations & Loop Invariants

Goal: $\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \sqsubseteq \underbrace{[c \neq 1]\cdot x + [c=1]\cdot(x+1)}_{\text{choose as superinvariant } I}$

$$
\begin{aligned}
&\texttt{while}\,(c=1)\,\{\\
&\quad /\!/\!/\ 1/2\cdot(x+[c=1]\cdot(x+2)+[c\neq 1]\cdot(x+1))\\
&\quad \{\,c \coloneqq 0\,\}\,[1/2]\,\{\,x \coloneqq x+1\,\}\\
&\quad /\!/\!/\ [c=1]\cdot(x+1)+[c\neq 1]\cdot x\\
&\}
\end{aligned}
$$

$\Phi_x(I) = [c \neq 1]\cdot x + [c=1]\cdot \mathsf{wp}[\![body]\!]\,(I)$

Recap: Weakest Preexpectations & Loop Invariants

Goal: $\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \sqsubseteq \underbrace{[c\neq 1]\cdot x + [c=1]\cdot(x+1)}_{\text{choose as superinvariant } I}$

$$
\begin{aligned}
&\texttt{while}\,(c=1)\,\{\\
&\quad /\!/\!/\ 1/2\cdot(x+[c=1]\cdot(x+2)+[c\neq 1]\cdot(x+1))\\
&\quad \{\,c\coloneqq 0\,\}\,[1/2]\,\{\,x\coloneqq x+1\,\}\\
&\quad /\!/\!/\ [c=1]\cdot(x+1)+[c\neq 1]\cdot x\\
&\}
\end{aligned}
$$

$$
\begin{aligned}
\Phi_x(I) &= [c\neq 1]\cdot x + [c=1]\cdot \mathsf{wp}[\![body]\!]\,(I)\\
&= [c\neq 1]\cdot x + [c=1]\cdot (1/2\cdot(x+[c=1]\cdot(x+2)+[c\neq 1]\cdot(x+1))) \sqsubseteq I
\end{aligned}
$$

Recap: Weakest Preexpectations & Loop Invariants

Goal: $\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \sqsubseteq \underbrace{[c\neq 1]\cdot x + [c=1]\cdot(x+1)}_{\text{choose as superinvariant } I}$

$$
\begin{aligned}
&\texttt{while}\,(c=1)\,\{\\
&\quad /\!/\!/\ 1/2\cdot(x+[c=1]\cdot(x+2)+[c\neq 1]\cdot(x+1))\\
&\quad \{\,c\coloneqq 0\,\}\,[1/2]\,\{\,x\coloneqq x+1\,\}\\
&\quad /\!/\!/\ [c=1]\cdot(x+1)+[c\neq 1]\cdot x\\
&\}
\end{aligned}
$$

$$
\begin{aligned}
\Phi_x(I) &= [c\neq 1]\cdot x + [c=1]\cdot \mathsf{wp}[\![body]\!]\,(I)\\
&= [c\neq 1]\cdot x + [c=1]\cdot (1/2\cdot(x+[c=1]\cdot(x+2)+[c\neq 1]\cdot(x+1))) \sqsubseteq I \quad \checkmark
\end{aligned}
$$

Recap: Weakest Preexpectations & Loop Invariants

Goal: $\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \sqsubseteq \underbrace{[c\neq 1]\cdot x + [c=1]\cdot(x+1)}_{\text{choose as superinvariant } I}$

$$
\begin{aligned}
&\texttt{while}\,(c=1)\,\{\\
&\quad /\!/\!/\ 1/2\cdot(x+[c=1]\cdot(x+2)+[c\neq 1]\cdot(x+1))\\
&\quad \{\,c\coloneqq 0\,\}\,[1/2]\,\{\,x\coloneqq x+1\,\}\\
&\quad /\!/\!/\ [c=1]\cdot(x+1)+[c\neq 1]\cdot x\\
&\}
\end{aligned}
$$

$$
\begin{aligned}
\Phi_x(I) &= [c\neq 1]\cdot x + [c=1]\cdot \mathsf{wp}[\![body]\!]\,(I)\\
&= \underbrace{[c\neq 1]\cdot x + [c=1]\cdot(1/2\cdot(x+[c=1]\cdot(x+2)+[c\neq 1]\cdot(x+1)))\ \sqsubseteq\ I}_{\textbf{When and how is this decidable?}}\ \checkmark
\end{aligned}
$$

Recap: Weakest Preexpectations & Loop Invariants

Goal: $\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \sqsubseteq \underbrace{[c\neq 1]\cdot x + [c=1]\cdot(x+1)}_{\text{choose as superinvariant } I}$

$$
\begin{aligned}
&\texttt{while}\,(c=1)\,\{\\
&\quad /\!/\!/\ 1/2\cdot(x+[c=1]\cdot(x+2)+[c\neq 1]\cdot(x+1))\\
&\quad \{\,c\coloneqq 0\,\}\,[1/2]\,\{\,x\coloneqq x+1\,\}\\
&\quad /\!/\!/\ [c=1]\cdot(x+1)+[c\neq 1]\cdot x\\
&\}
\end{aligned}
$$

$$
\begin{aligned}
\Phi_x(I) &= [c\neq 1]\cdot x + [c=1]\cdot \mathsf{wp}[\![body]\!]\,(I)\\
&= \underbrace{[c\neq 1]\cdot x + [c=1]\cdot(1/2\cdot(x+[c=1]\cdot(x+2)+[c\neq 1]\cdot(x+1)))\ \sqsubseteq\ I}_{\textbf{When and how is this decidable?}}\ \checkmark
\end{aligned}
$$

Being able to decide $\Phi_f(I) \sqsubseteq I$ enables automatic verification of loops.

Automatic Verification of Loops

Automatic Verification of Loops

... by restricting expectations f, I and loops $\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}$ such that

the wp-superinvariant condition $\Phi_f(I) \sqsubseteq I$ is decidable.

Definition

The set LinExp of *piecewise linear expectations* consists of all expectations f of the form

$$f \;=\; [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{, where}$$

Definition

The set LinExp of *piecewise linear expectations* consists of all expectations f of the form

$$f \;=\; [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{, where}$$

- $[c \neq 1] \cdot x + [c = 1] \cdot (x + 1) \in \mathsf{LinExp}$

Definition

The set LinExp of *piecewise linear expectations* consists of all expectations f of the form

$$f \;=\; [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{, where}$$

1. each φ_i is a Boolean combination of linear (in)equalities over Vars,

- $[c \neq 1] \cdot x + [c = 1] \cdot (x + 1) \in \mathsf{LinExp}$

Definition

The set LinExp of *piecewise linear expectations* consists of all expectations f of the form

$$f \;=\; [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{, where}$$

1. each φ_i is a Boolean combination of linear (in)equalities over Vars,
2. each e_i is a linear expression over Vars with $\forall \sigma \in \mathsf{States}\colon \sigma \models \varphi_i \implies e_i(\sigma) \geq 0$,

- $[c \neq 1] \cdot x + [c = 1] \cdot (x + 1) \in \mathsf{LinExp}$

Definition

The set LinExp of *piecewise linear expectations* consists of all expectations f of the form

$$f \;=\; [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{, where}$$

1. each φ_i is a Boolean combination of linear (in)equalities over Vars,
2. each e_i is a linear expression over Vars with $\forall \sigma \in \mathsf{States}\colon \sigma \models \varphi_i \implies e_i(\sigma) \geq 0$,
3. the φ_i partition the state space: for all $\sigma \in \mathsf{States}$ we have $\sigma \models \varphi_j$ for exactly one j.

- $[c \neq 1] \cdot x + [c = 1] \cdot (x + 1) \in \mathsf{LinExp}$

Definition

The set LinExp of *piecewise linear expectations* consists of all expectations f of the form

$$f \;=\; [\varphi_1]\cdot e_1 + \ldots + [\varphi_n]\cdot e_n \quad \text{, where}$$

1. each φ_i is a Boolean combination of linear (in)equalities over Vars,
2. each e_i is a linear expression over Vars with $\forall \sigma \in \text{States}\colon \sigma \models \varphi_i \implies e_i(\sigma) \geq 0$,
3. the φ_i partition the state space: for all $\sigma \in \text{States}$ we have $\sigma \models \varphi_j$ for exactly one j.

- $[c \neq 1]\cdot x + [c = 1]\cdot(x+1) \in \text{LinExp}$
- $[c \neq 1]\cdot (x \cdot y) + [c = 1]\cdot(x+1) \notin \text{LinExp}$

Definition

The set LinExp of *piecewise linear expectations* consists of all expectations f of the form

$$f \;=\; [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{, where}$$

1. each φ_i is a Boolean combination of linear (in)equalities over Vars,
2. each e_i is a linear expression over Vars with $\forall \sigma \in \mathsf{States}\colon \sigma \models \varphi_i \implies e_i(\sigma) \geq 0$,
3. the φ_i partition the state space: for all $\sigma \in \mathsf{States}$ we have $\sigma \models \varphi_j$ for exactly one j.

- $[c \neq 1] \cdot x + [c = 1] \cdot (x + 1) \in \mathsf{LinExp}$
- $[c \neq 1] \cdot (x \cdot y) + [c = 1] \cdot (x + 1) \notin \mathsf{LinExp}$
- $[x \geq 1] \cdot x + [x \geq 2] \cdot y \notin \mathsf{LinExp}$

Definition

The set LinExp of *piecewise linear expectations* consists of all expectations f of the form

$$f \;=\; [\varphi_1]\cdot e_1 + \ldots + [\varphi_n]\cdot e_n \quad \text{, where}$$

1. each φ_i is a Boolean combination of linear (in)equalities over Vars,
2. each e_i is a linear expression over Vars with $\forall \sigma \in \text{States}\colon \sigma \models \varphi_i \implies e_i(\sigma) \geq 0$,
3. the φ_i partition the state space: for all $\sigma \in \text{States}$ we have $\sigma \models \varphi_j$ for exactly one j.

- $[c \neq 1]\cdot x + [c = 1]\cdot(x+1) \in \text{LinExp}$
- $[c \neq 1]\cdot (x \cdot y) + [c = 1]\cdot(x+1) \notin \text{LinExp}$
- $[x \geq 1]\cdot x + [x \geq 2]\cdot y \notin \text{LinExp}$ but
  $[x \geq 1 \wedge x \geq 2]\cdot(x+y)$

Definition

The set LinExp of *piecewise linear expectations* consists of all expectations f of the form

$$f \;=\; [\varphi_1]\cdot e_1 + \ldots + [\varphi_n]\cdot e_n \quad \text{, where}$$

1. each φ_i is a Boolean combination of linear (in)equalities over Vars,
2. each e_i is a linear expression over Vars with $\forall \sigma \in \text{States}\colon \sigma \models \varphi_i \implies e_i(\sigma) \geq 0$,
3. the φ_i partition the state space: for all $\sigma \in \text{States}$ we have $\sigma \models \varphi_j$ for exactly one j.

- $[c \neq 1]\cdot x + [c = 1]\cdot(x+1) \in \text{LinExp}$
- $[c \neq 1]\cdot (x \cdot y) + [c = 1]\cdot(x+1) \notin \text{LinExp}$
- $[x \geq 1]\cdot x + [x \geq 2]\cdot y \notin \text{LinExp}$ but
  $[x \geq 1 \land x \geq 2]\cdot(x+y) + [x \geq 1 \land x \not\geq 2]\cdot x$

Definition

The set LinExp of *piecewise linear expectations* consists of all expectations f of the form

$$f \;=\; [\varphi_1]\cdot e_1 + \ldots + [\varphi_n]\cdot e_n \quad \text{, where}$$

1. each φ_i is a Boolean combination of linear (in)equalities over Vars,
2. each e_i is a linear expression over Vars with $\forall \sigma \in \mathsf{States}\colon \sigma \models \varphi_i \implies e_i(\sigma) \geq 0$,
3. the φ_i partition the state space: for all $\sigma \in \mathsf{States}$ we have $\sigma \models \varphi_j$ for exactly one j.

- $[c \neq 1]\cdot x + [c = 1]\cdot(x+1) \in \mathsf{LinExp}$
- $[c \neq 1]\cdot (x \cdot y) + [c = 1]\cdot(x+1) \notin \mathsf{LinExp}$
- $[x \geq 1]\cdot x + [x \geq 2]\cdot y \notin \mathsf{LinExp}$ but
  $[x \geq 1 \land x \geq 2]\cdot(x+y) + [x \geq 1 \land x \not\geq 2]\cdot x + [x \not\geq 1 \land x \not\geq 2]\cdot 0 \in \mathsf{LinExp}$

Definition

A loop `while` (φ) { *body* } is called *linear* if *body* is loop-free and all expressions occurring in φ and *body* are linear.

Definition

A loop `while` $(\varphi)\{\mathit{body}\}$ is called *linear* if *body* is loop-free and all expressions occurring in φ and *body* are linear.

The following loop is linear:

$$
\begin{array}{l}
\texttt{while}\,(\mathit{term} = 0)\,\{ \\
\quad v \coloneqq 2 \cdot v\,; \\
\quad \{\, c \coloneqq 2 \cdot c \,\}\,[1/2]\,\{\, c \coloneqq 2 \cdot c + 1 \,\}\,; \\
\quad \texttt{if}\,(v \geq n)\,\{ \\
\qquad \texttt{if}\,(c < n)\,\{\, \mathit{term} \coloneqq 1 \,\} \\
\qquad \texttt{else}\,\{\, v \coloneqq v - n\,;\, c \coloneqq c - n \,\} \\
\quad \}\ \}
\end{array}
$$

Definition

A loop `while` (φ) { *body* } is called *linear* if *body* is loop-free and all expressions occurring in φ and *body* are linear.

Definition

A loop `while` (φ) { *body* } is called *linear* if *body* is loop-free and all expressions occurring in φ and *body* are linear.

Lemma

Let $f, I \in \mathsf{LinExp}$ *and let* `while` (φ) { *body* } *be linear. Then* $\Phi_f(I) \in \mathsf{LinExp}$.

Definition

A loop `while` $(\varphi)\ \{\ body\ \}$ is called *linear* if *body* is loop-free and all expressions occurring in φ and *body* are linear.

Lemma

Let $f, I \in \mathsf{LinExp}$ *and let* `while` $(\varphi)\ \{\ body\ \}$ *be linear. Then* $\Phi_f(I) \in \mathsf{LinExp}$.

Lemma

For $f, g \in \mathsf{LinExp}$, *it is decidable whether* $f \sqsubseteq g$ *holds.*

Definition

A loop `while` $(\varphi)\ \{\ body\ \}$ is called *linear* if *body* is loop-free and all expressions occurring in φ and *body* are linear.

Lemma

Let $f, I \in \mathsf{LinExp}$ *and let* `while` $(\varphi)\ \{\ body\ \}$ *be linear. Then* $\Phi_f(I) \in \mathsf{LinExp}$.

Lemma

For $f, g \in \mathsf{LinExp}$, *it is decidable whether* $f \sqsubseteq g$ *holds.*

Theorem

Let $f, I \in \mathsf{LinExp}$ *and let* `while` $(\varphi)\ \{\ body\ \}$ *be linear. Then* $\Phi_f(I) \sqsubseteq I$ *is decidable.*

Automatic Verification of Probabilistic Loops

Given: $f, g \in \mathsf{LinExp}$ with

$$f = [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{and} \quad g = [\psi_1] \cdot a_1 + \ldots + [\psi_m] \cdot a_m$$

Goal: Decide whether $f \sqsubseteq g$ holds.

Automatic Verification of Probabilistic Loops

Given: $f, g \in \mathsf{LinExp}$ with

$$f = [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{and} \quad g = [\psi_1] \cdot a_1 + \ldots + [\psi_m] \cdot a_m$$

Goal: Decide whether $f \sqsubseteq g$ holds.

Idea: Compute logical formula Θ over Vars such that

$$f \sqsubseteq g \qquad \text{iff}$$

Given: $f, g \in \mathsf{LinExp}$ with

$$f = [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{and} \quad g = [\psi_1] \cdot a_1 + \ldots + [\psi_m] \cdot a_m$$

Goal: Decide whether $f \sqsubseteq g$ holds.

Idea: Compute logical formula Θ over Vars such that

$$f \sqsubseteq g \qquad \text{iff} \qquad \underbrace{\forall \sigma \in \mathsf{States}\colon \sigma \models \Theta}_{\text{decidable using existing tools}} \quad .$$

Given: $f, g \in \mathsf{LinExp}$ with

$$f = [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{and} \quad g = [\psi_1] \cdot a_1 + \ldots + [\psi_m] \cdot a_m$$

Goal: Decide whether $f \sqsubseteq g$ holds.

Automatic Verification of Probabilistic Loops

Given: $f, g \in \mathsf{LinExp}$ with

$$f = [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{and} \quad g = [\psi_1] \cdot a_1 + \ldots + [\psi_m] \cdot a_m$$

Goal: Decide whether $f \sqsubseteq g$ holds.

$f \sqsubseteq g$ iff for all $\sigma \in \mathsf{States}$: $f(\sigma) \leq g(\sigma)$

Automatic Verification of Probabilistic Loops

Given: $f, g \in \mathsf{LinExp}$ with

$$f = [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{and} \quad g = [\psi_1] \cdot a_1 + \ldots + [\psi_m] \cdot a_m$$

Goal: Decide whether $f \sqsubseteq g$ holds.

$f \sqsubseteq g$ iff for all $\sigma \in \mathsf{States}$: $f(\sigma) \leq g(\sigma)$

iff for all $\sigma \in \mathsf{States}$ and all φ_i, ψ_j:

Automatic Verification of Probabilistic Loops

Given: $f, g \in \mathsf{LinExp}$ with

$$f = [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{and} \quad g = [\psi_1] \cdot a_1 + \ldots + [\psi_m] \cdot a_m$$

Goal: Decide whether $f \sqsubseteq g$ holds.

$f \sqsubseteq g$ iff for all $\sigma \in \mathsf{States}$: $f(\sigma) \leq g(\sigma)$

iff for all $\sigma \in \mathsf{States}$ and all φ_i, ψ_j: $\big(\underbrace{\sigma \models \varphi_i}_{f(\sigma) = e_i(\sigma)}$

Given: $f, g \in \mathsf{LinExp}$ with

$$f = [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{and} \quad g = [\psi_1] \cdot a_1 + \ldots + [\psi_m] \cdot a_m$$

Goal: Decide whether $f \sqsubseteq g$ holds.

$f \sqsubseteq g$ iff for all $\sigma \in \mathsf{States}$: $f(\sigma) \leq g(\sigma)$

iff for all $\sigma \in \mathsf{States}$ and all φ_i, ψ_j: $\big(\underbrace{\sigma \models \varphi_i}_{f(\sigma)=e_i(\sigma)} \text{ and } \underbrace{\sigma \models \psi_j}_{g(\sigma)=a_j(\sigma)} \big)$

Given: $f, g \in \mathsf{LinExp}$ with

$$f = [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{and} \quad g = [\psi_1] \cdot a_1 + \ldots + [\psi_m] \cdot a_m$$

Goal: Decide whether $f \sqsubseteq g$ holds.

$f \sqsubseteq g$ iff for all $\sigma \in \mathsf{States}$: $f(\sigma) \leq g(\sigma)$

iff for all $\sigma \in \mathsf{States}$ and all φ_i, ψ_j: $\big(\underbrace{\sigma \models \varphi_i}_{f(\sigma)=e_i(\sigma)} \text{ and } \underbrace{\sigma \models \psi_j}_{g(\sigma)=a_j(\sigma)} \big)$ implies $e_i(\sigma) \leq a_j(\sigma)$

Given: $f, g \in \mathsf{LinExp}$ with

$$f = [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{and} \quad g = [\psi_1] \cdot a_1 + \ldots + [\psi_m] \cdot a_m$$

Goal: Decide whether $f \sqsubseteq g$ holds.

$f \sqsubseteq g$ iff for all $\sigma \in \mathsf{States}$: $f(\sigma) \leq g(\sigma)$

iff for all $\sigma \in \mathsf{States}$ and all φ_i, ψ_j: $\big(\underbrace{\sigma \models \varphi_i}_{f(\sigma)=e_i(\sigma)} \text{ and } \underbrace{\sigma \models \psi_j}_{g(\sigma)=a_j(\sigma)} \big)$ implies $e_i(\sigma) \leq a_j(\sigma)$

iff for all $\sigma \in \mathsf{States}$: $\underbrace{\sigma \models \bigwedge_{i=1}^{n} \bigwedge_{j=1}^{m} (\varphi_i \wedge \psi_j) \longrightarrow e_i \leq a_j}_{\text{formula of linear inequalities}}$

Given: $f, g \in \mathsf{LinExp}$ with

$$f = [\varphi_1] \cdot e_1 + \ldots + [\varphi_n] \cdot e_n \quad \text{and} \quad g = [\psi_1] \cdot a_1 + \ldots + [\psi_m] \cdot a_m$$

Goal: Decide whether $f \sqsubseteq g$ holds.

$f \sqsubseteq g$ iff for all $\sigma \in \mathsf{States}$: $f(\sigma) \leq g(\sigma)$

iff for all $\sigma \in \mathsf{States}$ and all φ_i, ψ_j: $\big(\underbrace{\sigma \models \varphi_i}_{f(\sigma)=e_i(\sigma)} \text{ and } \underbrace{\sigma \models \psi_j}_{g(\sigma)=a_j(\sigma)} \big)$ implies $e_i(\sigma) \leq a_j(\sigma)$

iff for all $\sigma \in \mathsf{States}$: $\underbrace{\sigma \models \bigwedge_{i=1}^{n} \bigwedge_{j=1}^{m} (\varphi_i \wedge \psi_j) \longrightarrow e_i \leq a_j}_{\text{formula of linear inequalities}}$

The latter problem is decidable using **Satisfiability Modulo Theories** techniques.[a]

[a] `https://www.microsoft.com/en-us/research/project/z3-3/`

Hence, our example

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; [c \neq 1]\cdot x + [c=1]\cdot(x+1)$$

can be verified **automatically.**

Hence, our example

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; [c \neq 1]\cdot x + [c=1]\cdot(x+1)$$

can be verified **automatically.**

Given $f, I \in \mathsf{LinExp}$, and linear $\texttt{while}\,(\varphi)\,\{\,body\,\}$:

Hence, our example

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; [c \neq 1]\cdot x + [c=1]\cdot(x+1)$$

can be verified **automatically.**

Given $f, I \in \mathsf{LinExp}$, and linear $\texttt{while}\,(\varphi)\,\{\,body\,\}$:

1. Compute $\Phi_f(I) \in \mathsf{LinExp}$.

Hence, our example

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; [c \neq 1]\cdot x + [c=1]\cdot(x+1)$$

can be verified **automatically.**

Given $f, I \in \mathsf{LinExp}$, and linear $\texttt{while}\,(\varphi)\,\{\,body\,\}$:

1. Compute $\Phi_f(I) \in \mathsf{LinExp}$.
2. If $\Phi_f(I) \sqsubseteq I$, then return verify;

Hence, our example

$$\mathsf{wp}[\![\texttt{while}\,(\,c = 1\,)\,\{\,c \coloneqq 0\,[\,1/2\,]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; [c \neq 1]\cdot x + [c=1]\cdot(x+1)$$

can be verified **automatically.**

Given $f, I \in \mathsf{LinExp}$, and linear $\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}$:

1. Compute $\Phi_f(I) \in \mathsf{LinExp}$.
2. If $\Phi_f(I) \sqsubseteq I$, then return verify; otherwise return inconclusive.

Hence, our example

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; [c \neq 1]\cdot x + [c=1]\cdot(x+1)$$

can be verified **automatically.**

Given $f, I \in \mathsf{LinExp}$, and linear $\texttt{while}\,(\varphi)\,\{\,body\,\}$:

1. Compute $\Phi_f(I) \in \mathsf{LinExp}$.
2. If $\Phi_f(I) \sqsubseteq I$, then return verify; otherwise return inconclusive.

Question: What to do if $\Phi_f(I) \sqsubseteq I$ does **not** hold?

Hence, our example

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; [c \neq 1]\cdot x + [c=1]\cdot(x+1)$$

can be verified **automatically.**

Hence, our example

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; [c \neq 1]\cdot x + [c=1]\cdot(x+1)$$

can be verified **automatically.** Problem: Not every upper bound is a superinvariant:

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; x+1$$

Hence, our example

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; [c \neq 1]\cdot x + [c=1]\cdot(x+1)$$

can be verified **automatically.** Problem: Not every upper bound is a superinvariant:

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; x+1$$

We have

$$\Phi_x(x+1) \;=\; [c \neq 1]\cdot x + [c=1]\cdot(x+1.5)$$

Hence, our example

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; [c\neq 1]\cdot x + [c=1]\cdot(x+1)$$

can be verified **automatically.** Problem: Not every upper bound is a superinvariant:

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; x+1$$

We have

$$\Phi_x(x+1) \;=\; [c\neq 1]\cdot x + [c=1]\cdot(x+1.5) \;\not\sqsubseteq\; x+1\,.$$

Hence, our example

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; [c\neq 1]\cdot x + [c=1]\cdot(x+1)$$

can be verified **automatically.** Problem: Not every upper bound is a superinvariant:

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; x+1$$

We have

$$\Phi_x(x+1) \;=\; [c\neq 1]\cdot x + [c=1]\cdot(x+1.5) \;\not\sqsubseteq\; x+1\,.$$

Even though $\mathsf{lfp}\,\Phi_x \sqsubseteq x+1$ we have $\Phi_x(x+1) \not\sqsubseteq x+1$!

Hence, our example

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; [c\neq 1]\cdot x + [c=1]\cdot(x+1)$$

can be verified **automatically.** Problem: Not every upper bound is a superinvariant:

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; x+1$$

We have

$$\Phi_x(x+1) \;=\; [c\neq 1]\cdot x + [c=1]\cdot(x+1.5) \;\not\sqsubseteq\; x+1\,.$$

Even though $\mathsf{lfp}\,\Phi_x \sqsubseteq x+1$ we have $\Phi_x(x+1) \not\sqsubseteq x+1$!

Mitigate this problem using a more powerful proof rule: k-induction.

Improved Automatic Verification of Loops

... by means of k-induction.

Improved Automatic Verification of Probabilistic Loops

Given $h, h' \in \mathbb{E}$, define their *pointwise minimum* by $h \sqcap h' = \lambda\sigma.\, \min\{h(\sigma), h'(\sigma)\}$.

Improved Automatic Verification of Probabilistic Loops

Given $h, h' \in \mathbb{E}$, define their *pointwise minimum* by $h \sqcap h' = \lambda\sigma.\, \min\{h(\sigma), h'(\sigma)\}$.

Example:

$$[c \neq 1] \cdot x + [c = 1] \cdot (x + 1.5) \;\sqcap\; x + 1$$

Improved Automatic Verification of Probabilistic Loops

Given $h, h' \in \mathbb{E}$, define their *pointwise minimum* by $h \sqcap h' = \lambda\sigma.\, \min\{h(\sigma), h'(\sigma)\}$.

Example:

$$[c \neq 1] \cdot x + [c = 1] \cdot (x + 1.5) \;\sqcap\; x + 1 \quad = \quad [c \neq 1] \cdot x$$

Given $h, h' \in \mathbb{E}$, define their *pointwise minimum* by $h \sqcap h' = \lambda\sigma.\, \min\{h(\sigma), h'(\sigma)\}$.

Example:

$$[c \neq 1] \cdot x + [c = 1] \cdot (x + 1.5) \sqcap x + 1 \quad = \quad [c \neq 1] \cdot x + [c = 1] \cdot (x + 1)$$

Given $h, h' \in \mathbb{E}$, define their *pointwise minimum* by $h \sqcap h' = \lambda\sigma.\, \min\{h(\sigma), h'(\sigma)\}$.

Theorem

For $f, I \in \mathbb{E}$ and loop $C = \texttt{while}\,(\,\varphi\,)\,\{\,body\,\}$, it holds:

- 1-*induction:* $\Phi_f(I) \sqsubseteq I$ *implies* $\texttt{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq I$

Improved Automatic Verification of Probabilistic Loops

Given $h, h' \in \mathbb{E}$, define their *pointwise minimum* by $h \sqcap h' = \lambda\sigma.\, \min\{h(\sigma), h'(\sigma)\}$.

Theorem

For $f, I \in \mathbb{E}$ and loop $C = \texttt{while}\,(\,\varphi\,)\,\{\,body\,\}$, it holds:

- 1-*induction:* $\Phi_f(I) \sqsubseteq I$ *implies* $\texttt{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq I$
- 2-*induction:* $\Phi_f(\Phi_f(I) \sqcap I) \sqsubseteq I$ *implies* $\texttt{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq I$

Given $h, h' \in \mathbb{E}$, define their *pointwise minimum* by $h \sqcap h' = \lambda\sigma.\, \min\{h(\sigma), h'(\sigma)\}$.

Theorem

For $f, I \in \mathbb{E}$ and loop $C = \texttt{while}\,(\,\varphi\,)\,\{\,body\,\}$, it holds:

- 1-*induction:* $\Phi_f(I) \sqsubseteq I$ *implies* $\mathsf{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq I$
- 2-*induction:* $\Phi_f(\Phi_f(I) \sqcap I) \sqsubseteq I$ *implies* $\mathsf{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq I$
- 3-*induction:* $\Phi_f(\Phi_f(\Phi_f(I) \sqcap I) \sqcap I) \sqsubseteq I$ *implies* $\mathsf{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq I$

Given $h, h' \in \mathbb{E}$, define their *pointwise minimum* by $h \sqcap h' = \lambda\sigma.\, \min\{h(\sigma), h'(\sigma)\}$.

Theorem

For $f, I \in \mathbb{E}$ and loop $C = \texttt{while}\,(\,\varphi\,)\,\{\,body\,\}$, it holds:

- 1-*induction:* $\Phi_f(I) \sqsubseteq I$ *implies* $\mathsf{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq I$
- 2-*induction:* $\Phi_f(\Phi_f(I) \sqcap I) \sqsubseteq I$ *implies* $\mathsf{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq I$
- 3-*induction:* $\Phi_f(\Phi_f(\Phi_f(I) \sqcap I) \sqcap I) \sqsubseteq I$ *implies* $\mathsf{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq I$
- $\ldots$

Given $h, h' \in \mathbb{E}$, define their *pointwise minimum* by $h \sqcap h' = \lambda\sigma.\, \min\{h(\sigma), h'(\sigma)\}$.

Theorem

For $f, I \in \mathbb{E}$ and loop $C = \texttt{while}\,(\,\varphi\,)\,\{\,body\,\}$, it holds:

- 1-*induction:* $\Phi_f(I) \sqsubseteq I$ *implies* $\mathsf{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq I$
- 2-*induction:* $\Phi_f(\Phi_f(I) \sqcap I) \sqsubseteq I$ *implies* $\mathsf{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq I$
- 3-*induction:* $\Phi_f(\Phi_f(\Phi_f(I) \sqcap I) \sqcap I) \sqsubseteq I$ *implies* $\mathsf{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \sqsubseteq I$
- $\ldots$

Theorem

For $f, I \in \mathsf{LinExp}$, linear C, and $k \geq 1$, the condition for k-induction is decidable.

Follows from the fact that for $h, h' \in \mathsf{LinExp}$, one can compute $h \sqcap h' \in \mathsf{LinExp}$.

Back to our example

$$\mathtt{wp}[\![\,\mathtt{while}\,(\,c=1\,)\,\{\,c \coloneqq 0\;[\,1/2\,]\;x \coloneqq x+1\,\}\,]\!]\,(x) \;\sqsubseteq\; x+1\,,$$

1-induction fails:

$$\Phi_x(x+1) = [c \neq 1]\cdot x + [c \neq 1]\cdot(x+1.5) \not\sqsubseteq x+1$$

Back to our example

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; x+1\,,$$

1-induction fails:

$$\Phi_x(x+1) = [c \neq 1]\cdot x + [c \neq 1]\cdot(x+1.5) \not\sqsubseteq x+1$$

Now consider 2-induction:

$$\Phi_x(\Phi_x(x+1) \sqcap x+1)$$

Back to our example

$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\;[1/2]\;x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; x+1\,,$$

1-induction fails:

$$\Phi_x(x+1) = [c \neq 1]\cdot x + [c \neq 1]\cdot(x+1.5) \not\sqsubseteq x+1$$

Now consider 2-induction:

$$\Phi_x(\Phi_x(x+1) \sqcap x+1) \;=\; \Phi_x([c \neq 1]\cdot x + [c=1]\cdot(x+1.5) \sqcap x+1)$$

Back to our example

$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; x+1\,,$$

1-induction fails:

$$\Phi_x(x+1) = [c \neq 1]\cdot x + [c \neq 1]\cdot(x+1.5) \not\sqsubseteq x+1$$

Now consider 2-induction:

$$\begin{aligned}\Phi_x(\Phi_x(x+1) \sqcap x+1) &= \Phi_x([c \neq 1]\cdot x + [c=1]\cdot(x+1.5) \sqcap x+1)\\ &= \Phi_x([c \neq 1]\cdot x + [c=1]\cdot(x+1))\end{aligned}$$

Back to our example

$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; x+1\,,$$

1-induction fails:

$$\Phi_x(x+1) = [c \neq 1]\cdot x + [c \neq 1]\cdot(x+1.5) \not\sqsubseteq x+1$$

Now consider 2-induction:

$$\begin{aligned}\Phi_x(\Phi_x(x+1) \sqcap x+1) &= \Phi_x([c \neq 1]\cdot x + [c=1]\cdot(x+1.5) \sqcap x+1)\\ &= \Phi_x([c \neq 1]\cdot x + [c=1]\cdot(x+1))\\ &= [c \neq 1]\cdot x + [c=1]\cdot(x+1) \;\sqsubseteq\; x+1\end{aligned}$$

Back to our example

$$\mathtt{wp}[\![\,\mathtt{while}\,(\,c=1\,)\,\{\,c \coloneqq 0\,[\,1/2\,]\,x \coloneqq x+1\,\}\,]\!]\,(x) \;\sqsubseteq\; x+1\,,$$

1-induction fails:

$$\Phi_x(x+1) = [c \neq 1]\cdot x + [c \neq 1]\cdot(x+1.5) \not\sqsubseteq x+1$$

Now consider 2-induction:

$$\begin{aligned}
\Phi_x(\Phi_x(x+1) \sqcap x+1) &= \Phi_x([c \neq 1]\cdot x + [c=1]\cdot(x+1.5) \sqcap x+1) \\
&= \Phi_x([c \neq 1]\cdot x + [c=1]\cdot(x+1)) \\
&= [c \neq 1]\cdot x + [c=1]\cdot(x+1) \sqsubseteq x+1 \;\checkmark
\end{aligned}$$

Back to our example

$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\;[1/2]\;x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; x+1\,,$$

1-induction fails:

$$\Phi_x(x+1) = [c \neq 1]\cdot x + [c \neq 1]\cdot(x+1.5) \not\sqsubseteq x+1$$

Now consider 2-induction:

$$\begin{aligned}\Phi_x(\Phi_x(x+1) \sqcap x+1) &= \Phi_x([c\neq 1]\cdot x + [c=1]\cdot(x+1.5) \sqcap x+1)\\ &= \Phi_x([c\neq 1]\cdot x + [c=1]\cdot(x+1))\\ &= [c\neq 1]\cdot x + [c=1]\cdot(x+1) \sqsubseteq x+1\ \checkmark\end{aligned}$$

k-induction is more powerful and decidable in the linear setting.

We have implemented k-induction for probabilistic loops:

KIPRO2: k-Induction for PRObabilistic PROgrams[1]

[1] `https://github.com/moves-rwth/kipro2`

We have implemented k-induction for probabilistic loops:

KIPRO2: k-Induction for PRObabilistic PROgrams[1]

Given piecewise linear f, I and linear $\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}$, kipro2 aims to verify

$$\texttt{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \;\sqsubseteq\; I$$

by checking:

[1] `https://github.com/moves-rwth/kipro2`

We have implemented k-induction for probabilistic loops:

KIPRO2: k-Induction for PRObabilistic PROgrams[1]

Given piecewise linear f, I and linear $\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}$, kipro2 aims to verify

$$\texttt{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \;\sqsubseteq\; I$$

by checking:

1-induction: If $\Phi_f(I) \sqsubseteq I$, then return verify; otherwhise

[1] `https://github.com/moves-rwth/kipro2`

We have implemented k-induction for probabilistic loops:

KIPRO2: k-Induction for PRObabilistic PROgrams[1]

Given piecewise linear f, I and linear $\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}$, kipro2 aims to verify

$$\texttt{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \;\sqsubseteq\; I$$

by checking:

1-induction: If $\Phi_f(I) \sqsubseteq I$, then return verify; otherwhise

2-induction: If $\Phi_f(\Phi_f(I) \sqcap I) \sqsubseteq I$, then return verify; otherwhise

[1] `https://github.com/moves-rwth/kipro2`

We have implemented k-induction for probabilistic loops:

KIPRO2: k-Induction for PRObabilistic PROgrams[1]

Given piecewise linear f, I and linear $\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}$, kipro2 aims to verify

$$\texttt{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \;\sqsubseteq\; I$$

by checking:

1-induction: If $\Phi_f(I) \sqsubseteq I$, then return verify; otherwhise

2-induction: If $\Phi_f(\Phi_f(I) \sqcap I) \sqsubseteq I$, then return verify; otherwhise

3-induction: If $\Phi_f(\Phi_f(\Phi_f(I) \sqcap I) \sqcap I) \sqsubseteq I$, then return verify; otherwhise

[1] `https://github.com/moves-rwth/kipro2`

We have implemented k-induction for probabilistic loops:

KIPRO2: k-Induction for PRObabilistic PROgrams[1]

Given piecewise linear f, I and linear $\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}$, kipro2 aims to verify

$$\texttt{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \;\sqsubseteq\; I$$

by checking:

1-induction: If $\Phi_f(I) \sqsubseteq I$, then return verify; otherwhise

2-induction: If $\Phi_f(\Phi_f(I) \sqcap I) \sqsubseteq I$, then return verify; otherwhise

3-induction: If $\Phi_f(\Phi_f(\Phi_f(I) \sqcap I) \sqcap I) \sqsubseteq I$, then return verify; otherwhise

...

[1] `https://github.com/moves-rwth/kipro2`

For C_{brp} given by

$$\texttt{while}\,(\,sent < N \wedge fail < F\,)\,\{$$
$$\{\underbrace{fail := fail + 1\,;\, totalFail := totalFail + 1}_{\text{failed transmission}}\}[0.1]\{\underbrace{fail := 0\,;\, sent := sent + 1}_{\text{successful transmission}}\}\}$$

For C_{brp} given by

$$\texttt{while}\,(\,sent < N \wedge fail < F\,)\,\{$$
$$\{\underbrace{fail := fail + 1\,;\, totalFail := totalFail + 1}_{\text{failed transmission}}\}[0.1]\{\underbrace{fail := 0\,;\, sent := sent + 1}_{\text{successful transmission}}\}\}$$

We fully automatically verify that

$$\mathsf{wp}[\![C_{\text{brp}}]\!]\,(totalFail) \sqsubseteq [N \leq 3] \cdot (totalFail + 1) + [N > 3] \cdot \infty$$

by means of is 4-induction.

Optimal Discrete Uniform Generation from Coin Flips [Lumbroso '13]

Sample uniformly from $\{0, \ldots, n-1\}$ using fair coin flips only.

$v \coloneqq 1\,;\, c \coloneqq 0\,;\, term \coloneqq 0\,;$

`while` $(\, term = 0 \,)\, \{$

$\quad v \coloneqq 2 \cdot v\,;$

$\quad \{\, c \coloneqq 2 \cdot c \,\}\ [1/2]\ \{\, c \coloneqq 2 \cdot c + 1 \,\}\,;$

$\quad$ `if` $(\, v \geq n \,)\, \{$

$\quad\quad$ `if` $(\, c < n \,)\, \{\, term \coloneqq 1 \,\}$

$\quad\quad$ `else` $\{\, v \coloneqq v - n\,;\, c \coloneqq c - n \,\}$

$\quad \}$

$\}$

Optimal Discrete Uniform Generation from Coin Flips [Lumbroso '13]

Sample uniformly from $\{0, \ldots, n-1\}$ using fair coin flips only.

$v \coloneqq 1\,;\, c \coloneqq 0\,;\, term \coloneqq 0\,;$

`while` $(\, term = 0 \,)\, \{$

$\quad v \coloneqq 2 \cdot v\,;$

$\quad \{\, c \coloneqq 2 \cdot c \,\}\ [1/2]\ \{\, c \coloneqq 2 \cdot c + 1 \,\}\,;$

$\quad$ `if` $(\, v \geq n \,)\, \{$

$\quad\quad$ `if` $(\, c < n \,)\, \{\, term \coloneqq 1 \,\}$

$\quad\quad$ `else` $\{\, v \coloneqq v - n\,;\, c \coloneqq c - n \,\}$

$\quad \}$

$\}$

For $n \in \{2,3,4,5\}$, we automatically verify

$\mathbf{Pr}(c = i) \leq 1/n$ **for all** $i \in \{0, \ldots, n-1\}$

by means of 2-,3-, and 5-induction.

k-induction is more powerful and decidable in the linear setting.

The property

$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; x+1\,,$$

is provable by 2-induction.

k-induction is more powerful and decidable in the linear setting.

The property

$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; x+1\,,$$

is provable by 2-induction. But: Not every upper bound is provable by k-induction:

$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

k-induction is more powerful and decidable in the linear setting.

The property

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; x+1\,,$$

is provable by 2-induction. But: Not every upper bound is provable by k-induction:

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

We have $\Phi_x(2x+1) \not\sqsubseteq 2x+1$, $\Phi_x((\Phi_x(2x+1) \sqcap 2x+1)) \not\sqsubseteq 2x+1, \ldots$

k-induction is more powerful and decidable in the linear setting.

The property

$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; x+1\,,$$

is provable by 2-induction. But: Not every upper bound is provable by k-induction:

$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

We have $\Phi_x(2x+1) \not\sqsubseteq 2x+1$, $\Phi_x((\Phi_x(2x+1) \sqcap 2x+1)) \not\sqsubseteq 2x+1, \ldots$

Solution: **Invariant Synthesis**

Invariant Synthesis

... i.e., computing superinvariants.

Invariant Synthesis

Idea: To verify

$$\mathsf{wp}[\![\texttt{while}\,(c = 1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x + 1\,\}]\!]\,(x) \;\sqsubseteq\; 2x + 1$$

Invariant Synthesis

Idea: To verify

$$\mathsf{wp}[\![\texttt{while}\,(c = 1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

we compute an expectation I such that

1. I is a superinvariant: $\Phi_x(I) \sqsubseteq I$

Invariant Synthesis

Idea: To verify

$$\mathsf{wp}[\![\texttt{while}\,(c = 1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

we compute an expectation I such that

1. I is a superinvariant: $\Phi_x(I) \sqsubseteq I$
2. I is not too large: $I \sqsubseteq 2x+1$.

Invariant Synthesis

Idea: To verify

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

we compute an expectation I such that

1. I is a superinvariant: $\Phi_x(I) \sqsubseteq I$
2. I is not too large: $I \sqsubseteq 2x+1$.

We then have

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x) \underbrace{\sqsubseteq}_{I \text{ is superinvariant}} I$$

Invariant Synthesis

Idea: To verify

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

we compute an expectation I such that

1. I is a superinvariant: $\Phi_x(I) \sqsubseteq I$
2. I is not too large: $I \sqsubseteq 2x+1$.

We then have

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x) \quad \underbrace{\sqsubseteq}_{I \text{ is superinvariant}} \quad I \quad \underbrace{\sqsubseteq}_{I \text{ is not too large}} \quad 2x+1\ \checkmark$$

Invariant Synthesis

Idea: To verify

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

we compute an expectation I such that

1. I is a superinvariant: $\Phi_x(I) \sqsubseteq I$
2. I is not too large: $I \sqsubseteq 2x+1$.

We then have

$$\mathsf{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \quad \underbrace{\sqsubseteq}_{I \text{ is superinvariant}} \quad I \quad \underbrace{\sqsubseteq}_{I \text{ is not too large}} \quad 2x+1 \;\checkmark$$

Question: **How to compute I?**

$$\mathtt{wp}[\![\mathtt{while}\,(c = 1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x + 1\,\}]\!]\,(x) \;\sqsubseteq\; 2x + 1$$

Goal: Compute $I \in \mathsf{LinExp}$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x + 1$.

Invariant Synthesis

$$\texttt{wp}[\![\texttt{while}\,(c = 1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x + 1\,\}]\!]\,(x) \;\sqsubseteq\; 2x + 1$$

Goal: Compute $I \in \mathsf{LinExp}$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x + 1$.

Idea: Set up a *template*

$$T \;=\; [c \neq 1] \cdot x + [c = 1] \cdot (\alpha \cdot c + \beta \cdot x + \gamma)$$

$$\mathtt{wp}[\![\mathtt{while}\,(c = 1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x + 1\,\}]\!]\,(x) \ \sqsubseteq\ 2x + 1$$

Goal: Compute $I \in \mathsf{LinExp}$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x + 1$.

Idea: Set up a *template*

$$T \ = \ [c \neq 1] \cdot x + [c = 1] \cdot (\alpha \cdot c + \beta \cdot x + \gamma)$$

New goal: Find appropriate $\alpha, \beta, \gamma \in \mathbb{Q}$.

$$\mathtt{wp}[\![\mathtt{while}\,(c = 1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x + 1\,\}]\!]\,(x) \;\sqsubseteq\; 2x + 1$$

Goal: Compute $I \in \mathsf{LinExp}$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x + 1$.

Idea: Set up a *template*

$$T \;=\; [c \neq 1] \cdot x + [c = 1] \cdot (\alpha \cdot c + \beta \cdot x + \gamma)$$

New goal: Find appropriate $\alpha, \beta, \gamma \in \mathbb{Q}$.

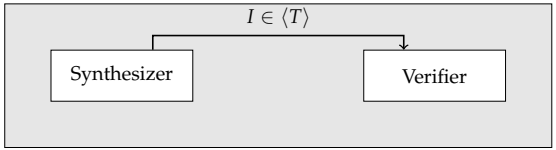
$$\mathtt{wp}[\![\mathtt{while}\,(c = 1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x + 1\,\}]\!]\,(x) \;\sqsubseteq\; 2x + 1$$

Goal: Compute $I \in \mathsf{LinExp}$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x + 1$.

Idea: Set up a *template*

$$T \;=\; [c \neq 1] \cdot x + [c = 1] \cdot (\alpha \cdot c + \beta \cdot x + \gamma)$$

New goal: Find appropriate $\alpha, \beta, \gamma \in \mathbb{Q}$.



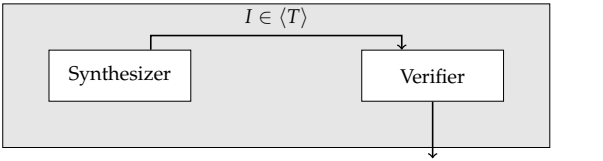
$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

Goal: Compute $I \in \mathsf{LinExp}$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Idea: Set up a *template*

$$T \;=\; [c \neq 1] \cdot x + [c = 1] \cdot (\alpha \cdot c + \beta \cdot x + \gamma)$$

New goal: Find appropriate $\alpha, \beta, \gamma \in \mathbb{Q}$.



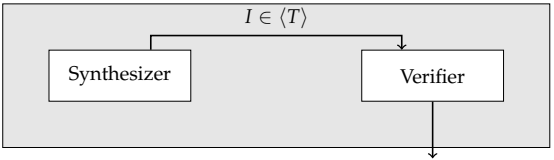
$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!](x) \;\sqsubseteq\; 2x+1$$

Goal: Compute $I \in \mathsf{LinExp}$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Idea: Set up a *template*

$$T \;=\; [c \neq 1] \cdot x + [c=1] \cdot (\alpha \cdot c + \beta \cdot x + \gamma)$$

New goal: Find appropriate $\alpha, \beta, \gamma \in \mathbb{Q}$.



Remember:

$$\Phi_x(I) \sqsubseteq I \text{ iff } \forall \sigma\colon \Phi_x(I)(\sigma) \leq I(\sigma)$$

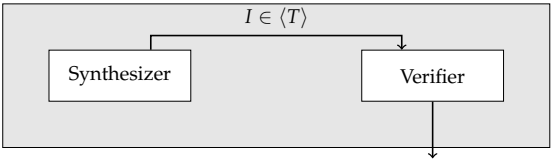
$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!](x) \;\sqsubseteq\; 2x+1$$

Goal: Compute $I \in \mathsf{LinExp}$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Idea: Set up a *template*

$$T \;=\; [c \neq 1] \cdot x + [c=1] \cdot (\alpha \cdot c + \beta \cdot x + \gamma)$$

New goal: Find appropriate $\alpha, \beta, \gamma \in \mathbb{Q}$.



I is superinvariant and not too large ✓

Remember:

$$\Phi_x(I) \sqsubseteq I \text{ iff } \forall \sigma\colon \Phi_x(I)(\sigma) \leq I(\sigma)$$

$$I \sqsubseteq 2x+1 \text{ iff } \forall \sigma\colon I(\sigma) \leq 2 \cdot \sigma(x) + 1$$

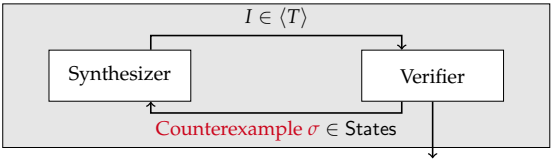
$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

Goal: Compute $I \in \mathsf{LinExp}$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Idea: Set up a *template*

$$T \;=\; [c \neq 1] \cdot x + [c=1] \cdot (\alpha \cdot c + \beta \cdot x + \gamma)$$

New goal: Find appropriate $\alpha, \beta, \gamma \in \mathbb{Q}$.



I is superinvariant and not too large ✓

Remember:

$$\Phi_x(I) \sqsubseteq I \text{ iff } \forall \sigma\colon \Phi_x(I)(\sigma) \leq I(\sigma)$$

$$I \sqsubseteq 2x+1 \text{ iff } \forall \sigma\colon I(\sigma) \leq 2 \cdot \sigma(x) + 1$$

Invariant Synthesis

$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Invariant Synthesis

$$\texttt{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!](x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 0\cdot x + 0) + [c\neq 1]\cdot x$

Invariant Synthesis

$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 0\cdot x + 0) + [c\neq 1]\cdot x$
Verifier: cex. $\sigma(c)=1, \sigma(x)=1$ with $\Phi_x(I)(\sigma) > I(\sigma)$

Invariant Synthesis

$$\texttt{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 0\cdot x + 0) + [c\neq 1]\cdot x$
Verifier: cex. $\sigma(c)=1, \sigma(x)=1$ with $\Phi_x(I)(\sigma) > I(\sigma)$

Rule out the cex. σ in all future iterations.

$$\texttt{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 0\cdot x + 0) + [c\neq 1]\cdot x$
Verifier: cex. $\sigma(c)=1, \sigma(x)=1$ with $\Phi_x(I)(\sigma) > I(\sigma)$

$$\Phi_x(T) \quad = [c\neq 1]\cdot x + [c=1]\cdot\frac{1}{2}\cdot(\alpha\cdot c + \beta\cdot(x+1) + \gamma + x)$$

$$\texttt{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 0\cdot x + 0) + [c\neq 1]\cdot x$
Verifier: cex. $\sigma(c)=1, \sigma(x)=1$ with $\Phi_x(I)(\sigma) > I(\sigma)$

$$\begin{aligned}\Phi_x(T) &= [c\neq 1]\cdot x + [c=1]\cdot\frac{1}{2}\cdot(\alpha\cdot c + \beta\cdot(x+1) + \gamma + x)\\ T &= [c\neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)\end{aligned}$$

Invariant Synthesis

$$\texttt{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!](x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 0\cdot x + 0) + [c\neq 1]\cdot x$
Verifier: cex. $\sigma(c)=1, \sigma(x)=1$ with $\Phi_x(I)(\sigma) > I(\sigma)$

$$\Phi_x(T)(\sigma) = \frac{1}{2}\cdot(\alpha\cdot 1 + \beta\cdot(1+1) + \gamma + 1)$$

$$T(\sigma) = \alpha\cdot 1 + \beta\cdot 1 + \gamma$$

$$\texttt{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 0\cdot x + 0) + [c\neq 1]\cdot x$
Verifier: cex. $\sigma(c)=1, \sigma(x)=1$ with $\Phi_x(I)(\sigma) > I(\sigma)$

$$\Phi_x(T)(\sigma) = \quad \frac{1}{2}\cdot(\alpha\cdot 1 + \beta\cdot(1+1) + \gamma + 1)$$

$$T(\sigma) = \quad \alpha\cdot 1 + \beta\cdot 1 + \gamma$$

Learn constraint $\frac{1}{2}\cdot(\alpha\cdot 1 + \beta\cdot 2 + \gamma + 1) \leq \alpha\cdot 1 + \beta\cdot 1 + \gamma$

Invariant Synthesis

$$\texttt{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!](x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Learned constraints:
$\frac{1}{2}(\alpha + 2\beta + \gamma + 1) \leq \alpha + \beta + \gamma$

Invariant Synthesis

$$\texttt{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 0\cdot x + 1) + [c\neq 1]\cdot x$

Learned constraints:
$\frac{1}{2}(\alpha + 2\beta + \gamma + 1) \leq \alpha + \beta + \gamma$

Invariant Synthesis

$$\texttt{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!](x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 0\cdot x + 1) + [c\neq 1]\cdot x$

Verifier: cex. $\sigma(c)=1, \sigma(x)=2$ with $\Phi_x(I)(\sigma) > I(\sigma)$

Learned constraints:

$\frac{1}{2}(\alpha + 2\beta + \gamma + 1) \leq \alpha + \beta + \gamma$

Invariant Synthesis

$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 0\cdot x + 1) + [c\neq 1]\cdot x$
Verifier: cex. $\sigma(c)=1, \sigma(x)=2$ with $\Phi_x(I)(\sigma) > I(\sigma)$

Learned constraints:
$\frac{1}{2}(\alpha + 2\beta + \gamma + 1) \leq \alpha + \beta + \gamma$

$$\Phi_x(T) \quad = [c\neq 1]\cdot x + [c=1]\cdot\frac{1}{2}\cdot(\alpha\cdot c + \beta\cdot(x+1) + \gamma + x)$$

$$\texttt{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!](x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 0\cdot x + 1) + [c\neq 1]\cdot x$
Verifier: cex. $\sigma(c)=1, \sigma(x)=2$ with $\Phi_x(I)(\sigma) > I(\sigma)$

Learned constraints:
$\frac{1}{2}(\alpha + 2\beta + \gamma + 1) \leq \alpha + \beta + \gamma$

$$\begin{aligned}\Phi_x(T) &= [c\neq 1]\cdot x + [c=1]\cdot\frac{1}{2}\cdot(\alpha\cdot c + \beta\cdot(x+1) + \gamma + x)\\ T &= [c\neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)\end{aligned}$$

Invariant Synthesis

$$\texttt{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T\rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 0\cdot x + 1) + [c\neq 1]\cdot x$
Verifier: cex. $\sigma(c)=1, \sigma(x)=2$ with $\Phi_x(I)(\sigma) > I(\sigma)$

Learned constraints:
$\frac{1}{2}(\alpha+2\beta+\gamma+1) \leq \alpha+\beta+\gamma$

$$\Phi_x(T)(\sigma) = \frac{1}{2}\cdot(\alpha\cdot 1 + \beta\cdot(2+1) + \gamma + 1)$$

$$T(\sigma) = \alpha\cdot 1 + \beta\cdot 2 + \gamma$$

$$\texttt{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T\rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 0\cdot x + 1) + [c\neq 1]\cdot x$
Verifier: cex. $\sigma(c)=1, \sigma(x)=2$ with $\Phi_x(I)(\sigma) > I(\sigma)$

Learned constraints:
$\frac{1}{2}(\alpha+2\beta+\gamma+1) \leq \alpha+\beta+\gamma$

$$\Phi_x(T)(\sigma) = \frac{1}{2}\cdot(\alpha\cdot 1+\beta\cdot(2+1)+\gamma+1)$$

$$T(\sigma) = \alpha\cdot 1+\beta\cdot 2+\gamma$$

Learn constraint $\frac{1}{2}\cdot(\alpha\cdot 1+\beta\cdot 3+\gamma+1) \leq \alpha\cdot 1+\beta\cdot 2+\gamma$

Invariant Synthesis

$$\mathtt{wp}[\![\mathtt{while}\,(c=1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Learned constraints:
$\frac{1}{2}(\alpha + 2\beta + \gamma + 1) \leq \alpha + \beta + \gamma$
$\frac{1}{2}(\alpha + 3\beta + \gamma + 1) \leq \alpha + 2\beta + \gamma$

Invariant Synthesis

$$\texttt{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\,[1/2]\,x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 1\cdot x + 1) + [c\neq 1]\cdot x$

Learned constraints:
$\frac{1}{2}(\alpha + 2\beta + \gamma + 1) \leq \alpha + \beta + \gamma$
$\frac{1}{2}(\alpha + 3\beta + \gamma + 1) \leq \alpha + 2\beta + \gamma$

Invariant Synthesis

$$\texttt{wp}[\![\texttt{while}\,(c=1)\,\{\,c \coloneqq 0\ [1/2]\ x \coloneqq x+1\,\}]\!]\,(x) \;\sqsubseteq\; 2x+1$$

$$T \;=\; [c \neq 1]\cdot x + [c=1]\cdot(\alpha\cdot c + \beta\cdot x + \gamma)$$

Goal: Find $I \in \langle T \rangle$ such that $\Phi_x(I) \sqsubseteq I$ and $I \sqsubseteq 2x+1$.

Synthesizer: $I = [c=1]\cdot(0\cdot c + 1\cdot x + 1) + [c\neq 1]\cdot x$

Verifier: verified! ✓

Learned constraints:

$\frac{1}{2}(\alpha + 2\beta + \gamma + 1) \leq \alpha + \beta + \gamma$

$\frac{1}{2}(\alpha + 3\beta + \gamma + 1) \leq \alpha + 2\beta + \gamma$

Invariant Synthesis

We have implemented invariant synthesis for probabilistic loops:

CEGISRO2: **CounterExample Guided Synthesis for PRObabilistic PROgrams**

Invariant Synthesis

We have implemented invariant synthesis for probabilistic loops:

CEGISRO2: CounterExample Guided Synthesis for PRObabilistic PROgrams

Given piecewise linear f, g and linear $\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}$, cegispro2 aims to verify

$$\texttt{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \;\sqsubseteq\; g$$

by computing a piecewise linear expectation I with $\Phi_f(I) \sqsubseteq I$ and $I \sqsubseteq g$.

Invariant Synthesis

We have implemented invariant synthesis for probabilistic loops:

CEGISRO2: CounterExample Guided Synthesis for PRObabilistic PROgrams

Given piecewise linear f, g and linear $\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}$, cegispro2 aims to verify

$$\mathsf{wp}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(f) \;\sqsubseteq\; g$$

by computing a piecewise linear expectation I with $\Phi_f(I) \sqsubseteq I$ and $I \sqsubseteq g$.
Using the same technique cegispro2 can also compute upper bounds on

$$\mathsf{ert}[\![\texttt{while}\,(\,\varphi\,)\,\{\,body\,\}]\!]\,(0)$$

by computing an ert-superinvariant.

Invariant synthesis enables verifying programs with gigantic state spaces.

For C_{brp} given by

$$\texttt{while}\left(sent < 8 \cdot 10^9 \wedge fail < 10\right)\{$$

$$\{\underbrace{fail := fail + 1\,;\, totalFail := totalFail + 1}_{\text{failed transmission}}\}[0.01]\{\underbrace{fail := 0\,;\, sent := sent + 1}_{\text{successful transmission}}\}\}$$

Invariant synthesis enables verifying programs with gigantic state spaces.

For C_{brp} given by

$$\texttt{while}\left(\mathit{sent} < 8 \cdot 10^9 \land \mathit{fail} < 10\right)\{$$
$$\{\underbrace{\mathit{fail} := \mathit{fail}+1\,;\mathit{totalFail} := \mathit{totalFail}+1}_{\text{failed transmission}}\}[0.01]\{\underbrace{\mathit{fail} := 0\,;\mathit{sent} := \mathit{sent}+1}_{\text{successful transmission}}\}\}$$

We fully automatically verify that

$$\mathsf{wp}[\![C_{\text{brp}}]\!]\left([\mathit{fail}=10]\right) \;\sqsubseteq\; [\mathit{sent}=0 \land \mathit{fail}=0]\cdot 0.0001 + [\neg(N=0 \land \mathit{fail}=0)]\cdot\infty\,.$$

in 58 seconds using 106 counterexamples.

Invariant Synthesis

Invariant synthesis enables upper bounding **expected runtimes**.

Consider the program C modeling a race between a tortoise and a hare:

$$\texttt{while}\,(\,h \leq t\,)\,\{$$
$$\quad t := t + 1;$$
$$\quad \{\,\texttt{skip}\,\}\,[\,1/2\,]\,\{\,h := 1/11 \cdot [h+0\rangle + \ldots + 1/11 \cdot [h+10\rangle\,\}\,\}$$

[2] counting only the number of loop iterations

Invariant synthesis enables upper bounding **expected runtimes**.

Consider the program C modeling a race between a tortoise and a hare:

$$
\begin{aligned}
&\texttt{while}\,(\,h \leq t\,)\,\{\\
&\quad t := t + 1;\\
&\quad \{\,\texttt{skip}\,\}\,[\,1/2\,]\,\{\,h := 1/11 \cdot [h+0\rangle + \ldots + 1/11 \cdot [h+10\rangle\,\}\,\}
\end{aligned}
$$

We fully automatically compute an ert-superinvariant[2] and obtain

$$\mathsf{ert}[\![C]\!]\,(0) \;\sqsubseteq\; [h \leq t] \cdot (2/3 \cdot t - 2/3 \cdot h + 13/3) + [h > t] \cdot 1$$

in 1.3 seconds using 32 counterexamples.

[2] counting only the number of loop iterations

Conclusion

The concepts of this lecture series provide the foundations for the
automatic verification of probabilistic programs.

Conclusion

The concepts of this lecture series provide the foundations for the
automatic verification of probabilistic programs.

Are you interested in writing a Bachelor/Master Thesis? Contact us!

The concepts of this lecture series provide the foundations for the
automatic verification of probabilistic programs.

Are you interested in writing a Bachelor/Master Thesis? Contact us!

- Lutz Klinkenberg (Algebraic Techniques): `lutz.klinkenberg@cs.rwth-aachen.de`
- Tobias Winkler (Automata Theoretic Techniques): `tobias.winkler@cs.rwth-aachen.de`
- Ira Fesefeldth (Concurrency): `fesefeldt@cs.rwth-aachen.de`
- Philipp Schröer (Deductive Verification): `phisch@cs.rwth-aachen.de`
- Bahar Salmani (Bayesian Networks): `salmani@cs.rwth-aachen.de`
- Kevin Batz (Automatic Verificaton): `kevin.batz@cs.rwth-aachen.de`

`thesis@i2.informatik.rwth-aachen.de`