Weighted Programming

A Programming Paradigm for Specifying Mathematical Models

Emma Ahrens

April 4, 2024

Probabilistic Programming pGCL

Symmetric random walk

```
while (x>0) {
  {x++} [0.5] {x--}
}
```

Geometric distribution

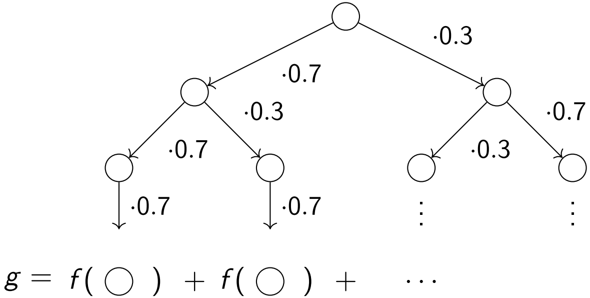
```
bool c := true;
int i := 0;
while (c) {
  i++;
  (c := false [p] c := true)
}
```

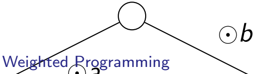
- express probability distributions via probabilistic programs
- prove correctness (e.g. probability distribution)
- prove termination with certain likelihood (e.g. almost-surely terminating)
- calculate the expected runtime

Generalize Probabilistic Programming? [BGK+22]

- describe probability distribution via *probabilistic program*
- over set $[0, 1]$
- $f : \mathbb{S} \to \mathbb{R}_{\geq 0}^{\infty}$ is a postexpectation
- e.g. $f = [x = 0]$ or $f = x$

- describe mathematical model via *weighted program*
- over an arbitrary semiring $(\mathcal{S}, \oplus, \odot, \mathbf{0}, \mathbf{1})$
- $f : \mathbb{S} \to \mathcal{S}$ is a postexpectation





Semirings

Monoid $\mathcal{W} = (W, \odot, \mathbf{1})$

with a carrier set W, an associative operation $\odot$, and neutral element $\mathbf{1}$.

The monoid $\mathcal{W}$ might additionally be commutative.

Semiring $\mathcal{S} = (\mathcal{S}, \oplus, \odot, \mathbf{0}, \mathbf{1})$

- $(\mathcal{S}, \oplus, \mathbf{0})$ is a commutative monoid,
- $(\mathcal{S}, \odot, \mathbf{1})$ is a monoid,
- distribution of multiplication over addition, hence for all $a, b, c \in S$

$$a \odot (b \oplus c) = a \odot b \oplus a \odot c \text{ and } (a \oplus b) \odot c = a \odot c \oplus b \odot c$$

- $\mathbf{0} \odot a = a \odot \mathbf{0} = \mathbf{0}$ for all $a \in S$.

$\Rightarrow$ A generalization are $\mathcal{W}$-modules $\mathcal{M}$.

Weighted Programming wGCL

Syntax

$$
\begin{aligned}
C \quad \rightarrow \quad & x := E \quad \textit{(assignment)} && \mid \quad \odot \boldsymbol{a} \quad \textit{(weighting)} \\
\mid \quad & C_1; C_2 \quad \textit{(sequential composition)} && \mid \quad \text{if } (\varphi)\ C_1 \text{ else } C_2 \quad \textit{(conditional choice)} \\
\mid \quad & \boldsymbol{C_1 \oplus C_2} \quad \textit{(branching)} && \mid \quad \text{while } (\varphi)\ C_1 \quad \textit{(loop)}
\end{aligned}
$$

$$
\begin{array}{l}
\{ \\
\quad x--; \quad \odot\texttt{true} \\
\} \oplus \{ \\
\quad \odot\, \texttt{false} \\
\}
\end{array}
$$

$$
\begin{array}{l}
\texttt{while}\,(n > 0)\,\{ \\
\quad n := n - 1; \\
\quad \{\, \odot 1 \,\} \oplus \{\, \odot y;\ n := 0 \,\} \\
\}
\end{array}
$$

$\{$

$\texttt{while}\,(n > 0)\,\{$

Weighted Programming wGCL

Semantics

- states $\mathbb{S} := \{\sigma : \mathsf{Vars} \to \mathbb{N} \mid \{x \in \mathsf{Vars} \mid \sigma(x) \neq 0\} \text{ is finite}\}$
- $Q = (\mathsf{wGCL} \cup \{\downarrow\}) \times \mathbb{S} \times \mathbb{N} \times \{L, R\}^*)$ called configurations
- $\Delta \subseteq Q \times \mathcal{S} \times Q$ called transitions

```
{
  x--;   ⊙true
} ⊕ {
   ⊙ false
}
```

$$\frac{\sigma' = \sigma[x \mapsto [\![E]\!](\sigma)]}{\langle x := E, \sigma, n, \beta\rangle \vdash_{\mathbf{1}} \langle \downarrow, \sigma', n+1, \beta\rangle}\ (\text{assign})$$

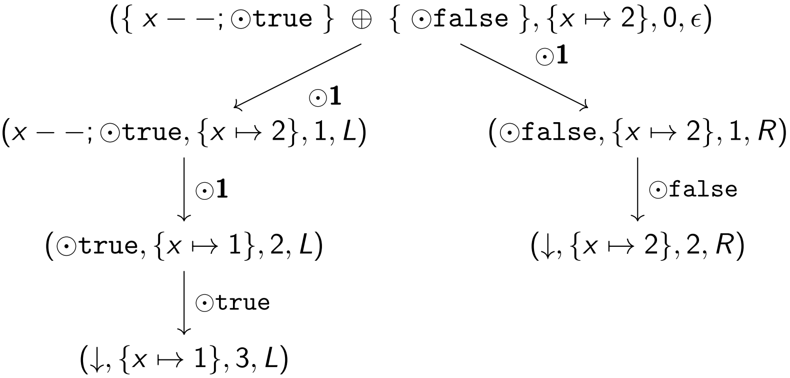
$$\frac{}{\langle \odot a, \sigma, n, \beta\rangle \vdash_{a} \langle \downarrow, \sigma, n+1, \beta\rangle}\ (\text{weight})$$

$$\frac{}{\langle \{\ C_1\ \}\ \oplus\ \{\ C_2\ \}, \sigma, n, \beta\rangle \vdash_{\mathbf{1}} \langle C_1, \sigma, n+1, \beta L\rangle}\ (\text{l. branch})$$

$$\frac{\sigma \models \varphi}{\langle\ \mathtt{while}\ (\varphi)\ C, \sigma, n, \beta\rangle \vdash_{\mathbf{1}} \langle C;\ \mathtt{while}\ (\varphi)\ C, \sigma, n+1, \beta\rangle}\ (\text{while})$$

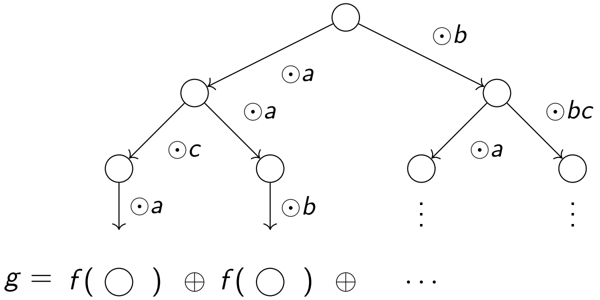
Weighted Programming wGCL

Semantics



$\texttt{true} \vee \texttt{false}$

Weakest Preweightings wp



Weakest Preweightings wp

- inspired by Dijkstra's weakest preconditions
- using semiring $(\mathcal{S}, \oplus, \odot, \mathbf{0}, \mathbf{1})$: define weightings $f, g : \mathbb{S} \rightarrow \mathcal{S}$
- weakest preweighting transformer $\mathsf{wp} : \mathsf{wGCL} \rightarrow ((\mathbb{S} \rightarrow \mathcal{S}) \rightarrow (\mathbb{S} \rightarrow \mathcal{S}))$:

wGCL-program P	$\mathsf{wp}[\![P]\!](f)$
$x := E$	$f[x/E]$
$\odot a$	$a \odot f$
$C_1; C_2$	$\mathsf{wp}[\![C_1]\!](\ \mathsf{wp}[\![C_2]\!](f)\ )$
$C_1 \oplus C_2$	$\mathsf{wp}[\![C_1]\!](f) \oplus \mathsf{wp}[\![C_2]\!](f)$
$\mathtt{if}\ (\varphi)\ \{\ C_1\ \}\ \mathtt{else}\ \{\ C_2\ \}$	$[\varphi]\, \mathsf{wp}[\![C_1]\!](f) \oplus [\neg\varphi]\, \mathsf{wp}[\![C_2]\!](f)$
$\mathtt{while}\ (\varphi)\ \{\ C\ \}$	$\mathsf{lfp}\, X. [\neg\varphi] f \oplus [\varphi]\, \mathsf{wp}[\![C]\!](X)$

A Random Example

```
// true ⊕ false = true
{
  // true
  x−−;
  // true ⊙ true = true
  ⊙ true
  // true
} ⊕ {
  // false ⊙ true = false
  ⊙ false
  // true
}
// true
```

P	$\mathrm{wp}[\![P]\!](f)$
$x := E$	$f[x/E]$
$\odot a$	$a \odot f$
$C_1; C_2$	$\mathrm{wp}[\![C_1]\!](\ \mathrm{wp}[\![C_2]\!](f)\ )$
$C_1 \oplus C_2$	$\mathrm{wp}[\![C_1]\!](f) \oplus \mathrm{wp}[\![C_2]\!](f)$
if else	$[\varphi]\,\mathrm{wp}[\![C_1]\!](f) \oplus [\neg\varphi]\,\mathrm{wp}[\![C_2]\!](f)$
while	$\mathrm{lfp}\, X. [\neg\varphi] f \oplus [\varphi]\,\mathrm{wp}[\![C]\!](X)$

Loop Invariants

- loop $\texttt{while}\ (\varphi)\ C$ is equivalent to
$$\texttt{if}\ (\varphi)\ \{C;\ \ \texttt{if}\ (\varphi)\ \{C;\ \ldots\}\ \texttt{else}\ \{\ \texttt{skip}\ \}\ \texttt{else}\ \{\ \texttt{skip}\ \}\}$$
- for postweighting f define wp-characteristic function and weighting transformer
$$\Phi_f : (\mathbb{S} \to \mathcal{S}) \to (\mathbb{S} \to \mathcal{S}), X \mapsto [\neg\varphi] f \oplus [\varphi]\, \mathsf{wp}[\![C]\!](X)$$
$$\mathsf{wp}[\![\ \texttt{while}\ (\varphi)\ C]\!](f) = \mathsf{lfp}\, \Phi_f$$
- well-defined if $(\mathcal{S}, \oplus, \odot, \mathbf{0}, \mathbf{1})$ is *ω-complete partially ordered*, $\oplus$ and $\odot$ are *ω-continuous* due to Kleene's fixed point theorem
- for weighting $I \in (\mathbb{S} \to \mathcal{S})$
$$\Phi_f(I) \leq I \qquad \text{implies} \qquad \mathsf{wp}[\![\texttt{while}\ (\varphi)\ \{\ C\ \}]\!](f) \leq I$$
- if $\texttt{while}\ (\varphi)\ \{\ C\ \}$ and C are *universally certainly terminating* then
$$I \leq \Phi_f(I) \qquad \text{implies} \qquad I \leq \mathsf{wp}[\![\texttt{while}\ (\varphi)\ \{\ C\ \}]\!](f)$$
- if $\texttt{while}\ (\varphi)\ \{\ C\ \}$ and C are *universally certainly terminating* then
$$I = \Phi_f(I) \qquad \text{implies} \qquad I = \mathsf{wp}[\![\texttt{while}\ (\varphi)\ \{\ C\ \}]\!](f)$$

Ski Rental Problem

$$\Phi_f : (\mathbb{S} \to \mathcal{S}) \to (\mathbb{S} \to \mathcal{S}), X \mapsto [\neg\varphi] f \oplus [\varphi]\, \mathrm{wp}[\![C]\!](X), \qquad \mathrm{wp}[\![\, \texttt{while}\ (\varphi)\ C]\!](f) = \mathrm{lfp}\, \Phi_f$$

$$I = \Phi_f(I) \qquad \text{implies} \qquad I = \mathrm{wp}[\![\texttt{while}\ (\varphi)\ \{\ C\ \}]\!](f)$$

$// \ [\neg\varphi] \cdot f \ \min\ [\varphi] \cdot I' = [n = 0] \cdot 0 \ \min\ [n > 0] \cdot (n \ \min\ y) = n \ \min\ y = I$

$\texttt{while}\ (n > 0)\ \{$

$\quad // \ (n - 1 + 1) \ \min\ y = n \ \min\ y = I'$

$\quad n := n - 1;$

$\quad // \ (n + 1) \ \min\ (y + 1) \ \min\ y = (n + 1) \ \min\ y$

$\quad \{ \ // \ (n + 1) \ \min\ (y + 1)$

$\qquad + 1; \ // \ I$

$\quad \} \ \min\ \{ \ // \ y$

$\qquad + y; \ // \ 0 \ \min\ y = 0$

$\qquad n := 0 \ // \ I$

$\quad \} \ // \ I = n \ \min\ y$

$\} \ // \ 0$

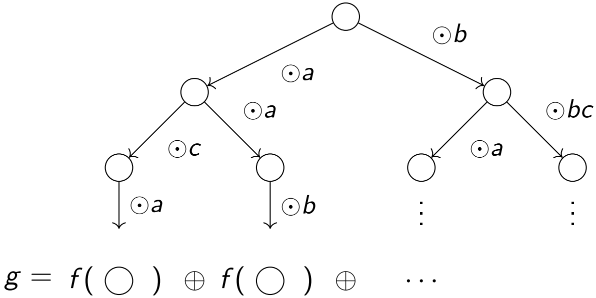
Outlook

What have we seen?

- weighted programs describe general mathematical models via wGCL
- arbitrary ω-continuous semirings for weighting
- wp allows general reasoning
- analysis of ski rental problem

What do we want to do now?

- find and study further possible applications, e.g. online algorithms (paging algorithm) and their competitive analysis
- analyse automation
- find further proof rules



References I

Kevin Batz, Adrian Gallus, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Tobias Winkler.
Weighted programming: a programming paradigm for specifying mathematical models.
Proc. ACM Program. Lang., 6(OOPSLA1):1–30, 2022.