Probabilistic Programming

Lecture #10: Liberal Expectations and Syntax of Expectations

Joost-Pieter Katoen

RWTH Lecture Series on Probabilistic Programming 2022-23

Overview

1. Relation to operational semantics
2. Motivation
3. Weakest liberal expectations
4. A syntax for weakest expectations

Overview

1. Relation to operational semantics
2. Motivation
3. Weakest liberal expectations
4. A syntax for weakest expectations

Recall: operational semantics of pGCL

Rewards

To reason about resource usage in MCs: use rewards.

A reward MC is a pair (D, r) with D an MC with state space Σ and $r : \Sigma \to \mathbb{R}$ a function assigning a real reward to each state.

The reward $r(\sigma)$ stands for the reward earned on leaving state σ.

Let $\pi = \sigma_0 \ldots \sigma_n$ be a finite path in (D, r) and $G \subseteq \Sigma$ a set of target states with $\pi \in \Diamond G$. The cumulative reward along π until reaching G is:

$$r_G(\pi) = r(\sigma_0) + \ldots + r(\sigma_{k-1}) \text{ where } \sigma_i \notin G \text{ for all } i < k \text{ and } \sigma_k \in G.$$

If $\pi \notin \Diamond G$, then $r_G(\pi) = \infty$.

Expected reward for reachability

Let σ be such that $Pr(\sigma \models \Diamond G) = 1$.

Then: the expected reward until reaching $G \subseteq \Sigma$ from $\sigma \in \Sigma$ is:

$$\mathrm{ER}(\sigma, \Diamond G) \;=\; \sum_{\widehat{\pi}} Pr(\widehat{\pi}) \cdot r_G(\widehat{\pi})$$

where $\widehat{\pi} = \sigma_0 \ldots \sigma_k$ is such that $\sigma_k \in G$, $\sigma_0 = \sigma$ and $\sigma_i \notin G$ for all $i < k$.

If $Pr(\sigma \models \Diamond G) < 1$, then let $\mathrm{ER}(\sigma, \Diamond G) = \infty$.

On computing expected rewards

Expected rewards in finite Markov chains can be computed in polynomial time by solving a system of linear equations.
(details on the black board.)

Equation system for expected rewards

Weakest pre-expectations = expected rewards

Compatibility theorem

For every pGCL program P, input s and expectation f:

$$\underbrace{wp[\![P]\!](f)(s)}_{\text{wp-semantics}} \quad = \quad \underbrace{ER^{[\![P]\!]}(s, \Diamond sink)}_{\text{operational semantics}}$$

In words: the $wp[\![P]\!](f)$ for input s equals the expected reward to reach final state *sink* in MC $[\![P]\!]$ where reward function r in $[\![P]\!]$ is defined by:

$$r(\langle\downarrow, s'\rangle) = f(s') \quad \text{and} \quad r(\cdot) = 0 \text{ otherwise.}$$

For finite-state programs, weakest pre-expectations can be computed by solving a system of linear equations, cf. a previous lecture.

Overview

1 Relation to operational semantics
2 Motivation
3 Weakest liberal expectations
4 A syntax for weakest expectations

Motivation

The expectation $wp[\![P]\!](f)$ is the expected value of f on termination of probabilistic program P.

Weakest pre-expectations thus consider f on termination

What if we want to also reason about possible divergence, i.e., non-termination?

This is exactly what weakest **liberal** pre-expectations do

Overview

1 Relation to operational semantics
2 Motivation
3 Weakest liberal expectations
4 A syntax for weakest expectations

Bounded expectations

Bounded expectations

The set of (one-)bounded expectations, denoted $\mathbb{E}_{\leqslant 1}$ is defined as:

$$\mathbb{E}_{\leqslant 1} \;=\; \{\, f \in \mathbb{E} \mid f \sqsubseteq \mathbf{1} \,\}$$

$(\mathbb{E}_{\leqslant 1}, \sqsubseteq)$ is a complete lattice.

Proof.

Left as an exercise. The least element is $\lambda s.0$; the greatest element is $\lambda s.1$ and suprema are defined as for $\mathbb{E}$. □

Weakest liberal pre-expectations

Weakest liberal pre-expectation

For pGCL program P and $e, f \in \mathbb{E}_{\leqslant 1}$, the function $wlp[\![P]\!](\cdot) : \mathbb{E}_{\leqslant 1} \to \mathbb{E}_{\leqslant 1}$ is defined by $wlp[\![P]\!](f) = e$ such that e equals the expected value of f after executing P on s plus the probability that P may diverge on s.

The characterising equation—à la Kozen's duality theorem—is:

$$wlp[\![P]\!](f) \;=\; \lambda s. \int_{\mathbb{S}} f \, d\mu_P^s \;+\; \left(1 - \int_{\mathbb{S}} 1 \, d\mu_P^s\right)$$

where μ_P^s is the distribution over the final states when executing P (reached on termination) on the initial state s.

Colloquially stated: $wlp[\![P]\!](f) \;=\; wp[\![P]\!](f) + \underbrace{Pr[P \text{ diverges}]}_{1 - wp[\![P]\!](\mathbf{1})}$.

Bounded expectation transformer semantics

Syntax probabilistic program P	Semantics $wlp[\![P]\!](f)$
`skip`	f
$x := E$	$f[x := E]$
$x :\approx \mu$	$\lambda s. \int_{\mathbb{Q}} (\lambda v. f(s[x := v]))\, d\mu_s$
$P; Q$	$wlp[\![P]\!]\,(wlp[\![Q]\!](f))$
`if` (φ) P `else` Q	$[\varphi] \cdot wlp[\![P]\!](f) + [\neg\varphi] \cdot wlp[\![Q]\!](f)$
$P\,[p]\,Q$	$p \cdot wlp[\![P]\!](f) + (1-p) \cdot wlp[\![Q]\!](f)$
`while` (φ) $\{P\}$	$\mathrm{gfp}\, X.\, (([\varphi] \cdot wlp[\![P]\!](X)) + [\neg\varphi] \cdot f)$

where gfp is the greatest fixed point wrt. the ordering $\sqsubseteq$ on $\mathbb{E}_{\leqslant 1}$.

Loops

$$wlp[\![\text{while } (G)\,\{P\}]\!](f) \;=\; \mathrm{gfp}\, X.\, \underbrace{([G] \cdot wlp[\![P]\!](X) + [\neg G] \cdot f)}_{\Psi(X)}$$

- Function $\Psi : \mathbb{E}_{\leqslant 1} \to \mathbb{E}_{\leqslant 1}$ (defined above) is continuous on $(\mathbb{E}_{\leqslant 1}, \sqsubseteq)$
- By Kleene's fixed point theorem, it follows: $\mathrm{gfp}\, \Psi = \inf_{n \in \mathbb{N}} \Psi^n(\mathbf{1})$
- $\Psi^n(\mathbf{1})$ denotes the expected value over the final states of running the loop n times for the constant expectation $\mathbf{1}$

Properties of weakest liberal pre-expectations

For all pGCL programs P and bounded expectations f, g it holds:

- Continuity: $wlp[\![P]\!](\cdot)$ is continuous on $(\mathbb{E}_{\leqslant 1}, \sqsubseteq)$
- Monotonicity: $f \sqsubseteq g$ implies $wlp[\![P]\!](f) \sqsubseteq wlp[\![P]\!](g)$
- Superlinearity: for any $r \in \mathbb{R}_{\geqslant 0}$:

$$wlp[\![P]\!](r \cdot f + g) \sqsupseteq r \cdot wlp[\![P]\!](f) + wlp[\![P]\!](g)$$

- Co-strictness: $wlp[\![P]\!](\mathbf{1}) = \mathbf{1}$

Relating wp and wlp

Duality of wp and wlp:

For all pGCL programs P and bounded expectation f it holds:

$$wlp[\![P]\!](f) \;=\; wp[\![P]\!](f) + \underbrace{(1 - wp[\![P]\!](\mathbf{1}))}_{\text{probability to diverge}}$$

Thus if $wp[\![P]\!](\mathbf{1}) = \mathbf{1}$, then $wlp[\![P]\!](f) \;=\; wp[\![P]\!](f)$

Sandwiching wlp:

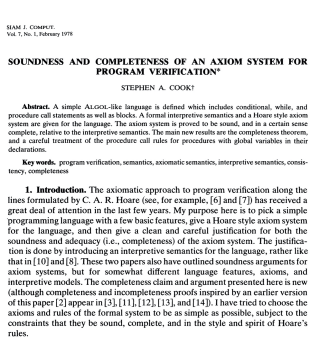
For all pGCL programs P, bounded expectation f, and predicate G such that $[G] \sqsubseteq wp[\![P]\!](\mathbf{1})$ it holds:

$$[G] \cdot wlp[\![P]\!](f) \sqsubseteq wp[\![P]\!](f) \sqsubseteq wlp[\![P]\!](f)$$

Overview

1. Relation to operational semantics
2. Motivation
3. Weakest liberal expectations
4. A syntax for weakest expectations

RELATIVE COMPLETENESS



SIAM J. COMPUT.
Vol. 7, No. 1, February 1978

SOUNDNESS AND COMPLETENESS OF AN AXIOM SYSTEM FOR PROGRAM VERIFICATION*

STEPHEN A. COOK†

Abstract. A simple ALGOL-like language is defined which includes conditional, while, and procedure call statements as well as blocks. A formal interpretive semantics and a Hoare style axiom system are given for the language. The axiom system is proved to be sound, and in a certain sense complete, relative to the interpretive semantics. The main new results are the completeness theorem, and a careful treatment of the procedure call rules for procedures with global variables in their declarations.

Key words. program verification, semantics, axiomatic semantics, interpretive semantics, consistency, completeness

1. Introduction. The axiomatic approach to program verification along the lines formulated by C. A. R. Hoare (see, for example, [6] and [7]) has received a great deal of attention in the last few years. My purpose here is to pick a simple programming language with a few basic features, give a Hoare style axiom system for the language, and then give a clean and careful justification for both the soundness and adequacy (i.e., completeness) of the axiom system. The justification is done by introducing an interpretive semantics for the language, rather like that in [10] and [8]. These two papers also have outlined soundness arguments for axiom systems, but for somewhat different language features, axioms, and interpretive models. The completeness claim and argument presented here is new (although completeness and incompleteness proofs inspired by an earlier version of this paper [2] appear in [3], [11], [12], [13], and [14]). I have tried to choose the axioms and rules of the formal system to be as simple as possible, subject to the constraints that they be sound, complete, and in the style and spirit of Hoare's rules.

SIAM J. on Computing, 1978

Stephen Cook

Two verification perspectives

- Extensional: use mathematical formulas as assertions
- Intensional: provide a syntax for assertions
- Why bother about providing a syntax?
  - Enable automation (e.g., Dafny, Boogie, Viper, ...)
  - Often used (e.g., invariant templates, super martingales)
  - Enables guided search for specialised fragments
  - .....

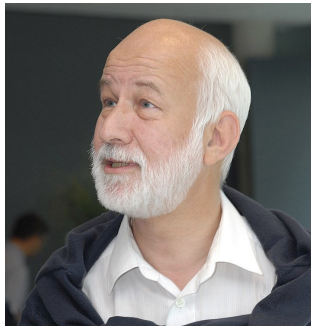
Relative complete verification

Ordinary Programs	Probabilistic Programs
$F \in$ FO-Arithmetic	$f \in$ SomeSyntax
implies	implies
$wp[\![P]\!](F) \in$ FO-Arithmetic	$wp[\![P]\!](f) \in$ SomeSyntax
$G \Longrightarrow wp[\![P]\!](F)$	$g \sqsubseteq wp[\![P]\!](f)$
is effectively decidable	is effectively decidable
modulo an oracle for deciding $\Rightarrow$	modulo an oracle for deciding $\sqsubseteq$ between two syntactic expectations.

Q: How does the SomeSyntax look like?

50 years of Hoare logic

"Completeness is a subtle manner and requires a careful analysis"



Krzysztof R. Apt

Ernst-Rüdiger Olderog

Requirements on a syntax

$\frac{\sqrt{5}-1}{2}$ (reciprocal of Golden ratio)

```
x := 1;
while (x > 0) {
 x +:= 2 [1/2] x -:= 1
}
```

1

$\frac{1}{\pi}$

```
x := geometric(1/4);
y := geometric(1/4);
t := x+y;
t := t+1 [5/9] skip;
r := 1;
for i in 1..3 {
 s := iid(bernouilli(1/2),2t);
 if (s != t) { r := 0 }
}
```

$[\mathbf{r} = \mathbf{1}]$

rational numbers, algebraic numbers, transcedental numbers, etc.

Syntax: expressions

- Arithmetic expressions

$$
\begin{array}{lcll}
a & \longrightarrow & r \in \mathbb{Q}_{\geqslant 0} & \text{non-negative rational} \\
 & \mid & x \in \mathit{Vars} & \mathbb{Q}_{\geqslant 0}\text{-valued variable} \\
 & \mid & a + a & \text{addition} \\
 & \mid & a \cdot a & \text{multiplication} \\
 & \mid & a \dot{-} a & \text{subtraction truncated at zero}
\end{array}
$$

- Boolean expressions

$$
\begin{array}{lcll}
\varphi & \longrightarrow & a < a & \text{comparing arithmetic expressions} \\
 & \mid & \varphi \wedge \varphi & \text{conjunction} \\
 & \mid & \neg \varphi & \text{negation}
\end{array}
$$

Syntax: expectations

- Expectations

$$
\begin{array}{lcll}
f & \longrightarrow & a & \text{arithmetic expressions} \\
 & \mid & [\varphi] \cdot f & \text{guarding} \\
 & \mid & f + f & \text{addition} \\
 & \mid & a \cdot f & \text{scaling by arithmetic expressions} \\
 & \mid & \mathfrak{S} x \colon f & \text{supremum over variable } x \\
 & \mid & \mathfrak{z} x \colon f & \text{infimum over variable } x
\end{array}
$$

- Examples:

$$\mathfrak{S} x \colon [x \cdot x < y] \cdot x \;\equiv\; \sqrt{y} \qquad \mathfrak{S} z \colon [z \cdot (x+1) = 1] \cdot z \;\equiv\; \frac{1}{x+1}$$

- $f \in \mathbb{E}$ is syntactic, if f is expressible in this syntax, i.e., if $f \in \mathrm{Exp}$

Semantics of syntactic expressions

Recall that state $s : \mathit{Vars} \to \mathbb{Q}_{\geqslant 0}$.

$$
\begin{array}{rcl}
[\![a]\!]^s & = & [\![a]\!]^s \\
[\![[\varphi] \cdot f]\!]^s & = & \begin{cases} [\![f]\!]^s & \text{if } [\![\varphi]\!]^s = \mathit{true} \\ 0 & \text{otherwise} \end{cases} \\
[\![f + g]\!]^s & = & [\![f]\!]^s + [\![g]\!]^s \\
[\![a \cdot f]\!]^s & = & [\![a]\!]^s \cdot [\![f]\!]^s \\
[\![\mathfrak{S} x \colon f]\!]^s & = & \sup \left\{ [\![f]\!]^{s[x \mapsto r]} \mid r \in \mathbb{Q}_{\geqslant 0} \right\} \\
[\![\mathfrak{z} x \colon f]\!]^s & = & \inf \left\{ [\![f]\!]^{s[x \mapsto r]} \mid r \in \mathbb{Q}_{\geqslant 0} \right\}
\end{array}
$$

Examples of expressible expectations

Starting from only rational-valued variable one can express:

- polynomials $y + x^3 + 2x^2 + x - 7$ — widely used as templates
- rational functions $\dfrac{x^2 - 3x + 4}{y^2 \cdot x - 3y + 1}$
- square roots $\sqrt{x}$
- irrational, algebraic and transcendental numbers e, π, Ω
- Harmonic numbers $H_k = \sum_{k=1}^{x} \frac{1}{k}$ — used in run-time/termination analysis

$$H_x \;=\; \mathrm{Sum}\left[v_{sum}, \tfrac{1}{v_{sum}}, x\right] \text{ with } [\![\mathrm{Sum}[v_{sum}, f, x]]\!]^s \;=\; \sum_{j=0}^{s(x)} [\![f[v_{sum}/j]]\!]^s$$

Expressiveness

The set Exp of syntactic expectations is expressive.

That is, for all pGCL programs P and $f \in \text{Exp}$ it holds:

$$wp[\![P]\!]([\![f]\!]) \quad = \quad [\![g]\!]$$

for some syntactic expectation $g \in \text{Exp}$.

Relevance

- Relative completeness à la Cook:

  bounds like $[\![g]\!] \sqsubseteq wp[\![P]\!](f)$ are effectively decidable

- Termination probabilities $wp[\![P]\!](\mathbf{1})$ on any input are expressible

- Probability to terminate in postcondition φ as $wp[\![P]\!]([\varphi])$

- Distribution over final states where $t(x_i) = v_i$:

$$\mu^s_P(t) \quad = \quad wp[\![P]\!]([x_1 = v_1 \wedge \cdots \wedge x_k = v_k])$$

Thus Kozen's measure transformers can be syntactically expressed

Take-home messages

- Expectations are the quantitative analogue of predicates
- Probabilistic weakest preconditions relate to expected rewards
- Liberal preconditions take possible divergence into account
- Computing expectations for straight-line programs is simple
- Calculating expectations for loopy programs requires fixed points
  As fixed points are incomputable, loop invariants are used
- A syntax to express all weakest preconditions for pGCL programs

Next lecture: how to treat loops?

Next lecture

Tuesday Nov 22, 16:30