Probabilistic Programming Languages

M2 MPRI 2021-2022

Guillaume Baudart (guillaume.baudart@ens.fr)
Christine Tasson (christine.tasson@lip6.fr)

Semantics of Probabilistic Programming

Discrete probability (Part II)

Probabilistic PCF - Denotational Semantics

A sound model

Pcoh is a cartesian closed category, CPO-enriched, hence a model of PCF, which is also a model of probabilistic PCF.

Interpretation of terms

$$\left[\!\left[\frac{}{G \vdash \mathtt{bernoulli}(p) : \mathtt{bool\ dist}}\right]\!\right] : \begin{cases}(\gamma, \mathtt{true}) \mapsto p \\ (\gamma, \mathtt{false}) \mapsto 1 - p\end{cases}$$

$$\left[\!\left[\frac{G \vdash d \ : \ t_1 \ \mathtt{dist} \qquad G, x : t_1 \vdash e : t_2}{G \vdash \mathtt{let}\, x = d \,\mathtt{in}\, e \ : \ t_2}\right]\!\right] : (\gamma, a_2) \mapsto$$

$$\sum_a [\![d]\!](\gamma, a)[\![e]\!](\gamma, a, a_2)$$

$$\left[\!\left[\frac{}{G \vdash \mathit{true} : \mathit{bool}}\right]\!\right] : \begin{cases}(\gamma, \mathtt{true}) \mapsto 1 \\ (\gamma, \mathtt{false}) \mapsto 0\end{cases}$$

$$[\![\mathtt{if}\, M \,\mathtt{then}\, N \,\mathtt{else}\, P]\!](\gamma, a) = [\![M]\!](\gamma, \mathtt{true})[\![N]\!](\gamma, a) + [\![M]\!](\gamma, \mathtt{false})[\![P]\!](\gamma, a)$$

Compute the denotational semantics

$$\dfrac{\vdash \text{bern } 0{,}5 : \text{bool dist} \qquad a,b,c : \text{bool} \vdash a+b+c : \text{nat}}{\vdash \underbrace{\text{let } a = \text{sample bern in let } b = \dots \text{ in let } c = \dots \text{ in } a+b+c}_{M} : \text{nat}}$$

bern 0,5	
t	1/2
f	1/2

a	b	c	(a+b+c)	$\mathbb{R}^+$
t	t	t	3	1
t	t	f	2	1
t	t	t	2	0
...				

Bernoulli :

```
let a = sample(bernoulli 0.5) in
let b = sample(bernoulli 0.5) in
let c = sample(bernoulli 0.5) in
a+b+c
```

M : nat	$\mathbb{R}_+$
n	$\sum_{x \in \lvert \text{bool} \rvert} \mathbb{P}(a=x) \sum_{y \in \{t,f\}} \mathbb{P}(a=y) \sum_{z \in \{t,f\}} \mathbb{P}(c=z)\ \mathbb{1}_{a+b+c=n}$
0	$\mathbb{P}(a=f)\,\mathbb{P}(b=f)\,\mathbb{P}(c=f) = \frac{1}{8}$
1	$\mathbb{P}(a=f)\,\mathbb{P}(b=f)\,\mathbb{P}(c=t) + \mathbb{P}(a=f)\,\mathbb{P}(b=t)\,\mathbb{P}(c=f) + \mathbb{P}(a=t)\,\mathbb{P}(b=f)\,\mathbb{P}(c=f) = 3/8$
...	

Examples - Compute the semantics

(2) $a\ b\ c : bool \vdash a=0 \text{ or } b=0 : bool$

(3) $f : nat\ \ a,b,c : bool \vdash a+b+c : nat$

(4) $f : nat, a,b,c : bool \vdash f : nat$

(1) $\vdash$ bern 0.5 : dist bool

(5) $f : nat, a,b,c : bool \vdash$ if $a=0$ or $b=0$ then $a+b+c$ else f

(6)

(7) $f : nat \vdash$ let $a =$ sample bern 0,5 in let $b = \dots$ let $c = \dots$ in if $a=0$ or $b=0$ then $a+b+c$ else f

$\vdash$ fix $\lambda f. \dots$

Funny Bernoulli by rejection sampling :

```
fix lambda f:bool
   let a = sample(bernoulli 0.5) in
   let b = sample(bernoulli 0.5) in
   let c = sample(bernoulli 0.5) in
   if (a==0 or b==0) then a+b+c
                          else f
```

(3) $\cup$ (4) $\Rightarrow$ (5)

(1)

$\lvert bool \rvert$	$\mathbb{R}^+$
t	0,5
f	0,5

Proba Table

Factor

Σ

(2) Relation 0/1.

a b c	a=0 or b=0	$\mathbb{R}^+$
[t][t][ ]	f	1
[t][t][ ]	t	0
[t][f][ ]	t	1
...		...

(3)

a	b	c	(a+b+c)	$\mathbb{R}^+$ (0,1)
[t]	[t]	[t]	3	1
[t]	[t]	[f]	2	1
[t]	[t]	[t]	2	0

(4)

, a b c f	$\mathbb{R}^+$ (0,1)
[n] [ ] [ ] [ ] n	1
otherwise	0

Examples - Compute the semantics

⑥

f	π_6	$\mathbb{R}^+$
$[\,]$	k	$\sum_{x,y,z} \mathbb{P}(a=x)\mathbb{P}(b=x)\mathbb{P}(c=x)\, \mathbb{1}_{a=0 \text{ or } b=0}(x,y)\, \mathbb{1}_{a+b+c=k}(x,y,z)$
$[n]$	k	$\sum_{x,y,z} \mathbb{P}(a=x)\mathbb{P}(b=x)\mathbb{P}(c=x)\, \mathbb{1}_{a=1 \text{ and } b=1}(x,y)\, \mathbb{1}_{k=n} = \frac{1}{4}\mathbb{1}_{k=n}$

Funny Bernoulli by rejection sampling :

```
fix lambda f:bool
  let a = sample(bernoulli 0.5) in
  let b = sample(bernoulli 0.5) in
  let c = sample(bernoulli 0.5) in
  if (a==0 or b==0) then a+b+c
                    else f
```

⑦ F^0: does not use f : $0 \mapsto 1/8$

F^1: uses f at most 1 : $0 \mapsto 1/8 + \frac{1}{4}\cdot F^0 = 1/8\,(1+1/4)$

F_n: uses f at most n times : $0 \mapsto 1/8\,(1+1/4+\dots+1/4^n)$

$\boxed{F = \bigvee F_n : 0 \mapsto \frac{1}{8}\,\frac{1}{3/4} = \frac{2}{6}}$

PCF - Denotational semantics

CPO-enriched : CPO, REL, MREL

$\mathcal{C}(A, B)$ is a complete partial order and $\circ$ is continuous.

$$\left[\!\!\left[\frac{G \vdash e : t \to t}{G \vdash \mathit{fixe} : t} \right]\!\!\right] = \sup F^n \text{ with } \begin{cases} F^0 = \bot \in \mathcal{C}(G, t) \\ F^{n+1} = ev \circ \langle [\![G \vdash e : t \to t]\!], F^n \rangle \end{cases}$$

Probabilistic Language - Type and Semantics

$$[\![G \vdash e \ : \ t]\!] : [\![G]\!] \times [\![t]\!] \to \mathbb{R}^+ \text{ and } [\![G \vdash e \ : \ t \ \texttt{dist}]\!](\gamma, \mathtt{a}) \in \mathrm{P}\,(\mathtt{G} \to \mathtt{t})$$

$$\left[\!\!\left[\frac{}{G \vdash \texttt{bernoulli}(p) : \texttt{bool dist}}\right]\!\!\right] : \begin{cases} (\gamma, \texttt{true}) \mapsto p \\ (\gamma, \texttt{false}) \mapsto 1 - p \end{cases}$$

$$\left[\!\!\left[\frac{G \vdash d \ : \ t_1 \ \texttt{dist} \qquad G, x : t_1 \vdash e : t_2}{G \vdash \texttt{let}\, x = d \,\texttt{in}\, e \ : \ t_2}\right]\!\!\right] : (\gamma, a_2) \mapsto \sum_a [\![e]\!](\gamma, a, a_2)$$

$$\left[\!\!\left[\frac{G \vdash e_1 \ : \ \texttt{bool} \qquad G \vdash e_2 \ : \ t}{G \vdash \texttt{assume}(e_1) \ ; \ e_2 \ : \ t}\right]\!\!\right] (\gamma, a) = \begin{cases} [\![e_2]\!](\gamma, a) & \text{if } [\![e_1]\!](g, true) = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\left[\!\!\left[\frac{G \vdash e \ : \ t}{G \vdash \texttt{infer}(e) \ : \ t \ \texttt{dist}}\right]\!\!\right] (\gamma, a) = \frac{[\![G \vdash e \ : \ t]\!](\gamma, a)}{\sum_{a \in |t|} [\![G \vdash e \ : \ t]\!](\gamma, a)}$$

Examples

nat	$\mathbb{R}^+$
$x \in \{1,\dots,6\}$	$1/6$

a	b	(assume $\lvert a-b\rvert=1$)	$\mathbb{R}^+$
x	$x+1$	*	1
x	$x-1$	*	1
otherwise			0

	a	b	Ω	$\mathbb{R}^+$
$x \in \{1 \cdots 5\}$	x	$x+1$	$2x+1$	1
$x \in \{2 \cdots 6\}$	x	$x-1$	$2x-1$	1
	otherwise			0

Dice :

$\vdash \mathrm{Unif}(1,6)$ $a, b \vdash \overbrace{\text{assume } |a-b|=1 \,;\, a+b}^{\Omega} : \text{nat}$

$\vdash \text{Dice} \;;\; \text{nat}$

```
let Dice =
  let a = sample(unif(1,6)) in
  let b = sample(unif(1,6)) in
  assume(|a - b| = 1) ;
  a+b
```

Dice	$\mathbb{R}^+$
k	$\sum_x \mathbb{P}(a=x) \sum_y \mathbb{P}(a=y)\, \mathbb{1}_{\lvert a-b\rvert=1}(x,y)\, \mathbb{1}_{a+b=k}(x,y)$

Cannabis :

```
let Cannabis =
    let b = sample(bernoulli 0.5) in
    if b then let c = sample(bernoulli 0.6) in
              assume c; c
         else true
```

Examples - Compute the semantics

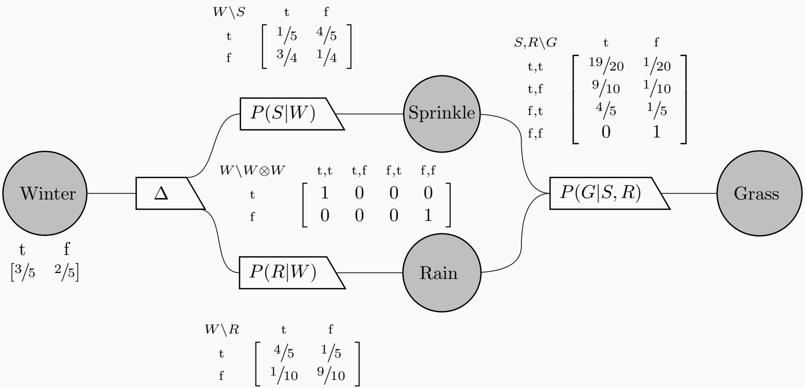
Funny Bernoulli :

```
let FB =
  let a = sample(bernoulli 0.5) in
  let b = sample(bernoulli 0.5) in
  let c = sample(bernoulli 0.5) in
  assume(a==0 or b==0) ;
  a+b+c
```

Semantics of Probabilistic Programming

Graphical Models

Bayesian Network - Example



Compute $\mathbb{P}(G)$ using :

Joint dist. $\mathbb{P}(G,R,S,W) = \mathbb{P}(G|S,R)\ \mathbb{P}(S|W)\ \mathbb{P}(R|W)\ \mathbb{P}(W)$

Marginal $\mathbb{P}(G) = \sum_{(r,s,w)\in|R|\times|S|\times|W|} \mathbb{P}(G,R,S,W)$

Bayesian Network

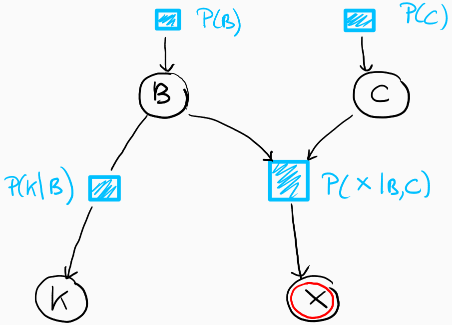
Definition

Directed Acyclic Graph with Conditional Probability Tables
Factors are functions from states to $\mathbb{R}^+$

Computing

Exact inference by variable elimination
Implemented by message passing

Message Passing - Example



Joint distribution

$$P(X, K, B, C) = P(X|B,C)\,P(C)\,P(K|B)\,P(B)$$

Message Passing - Example

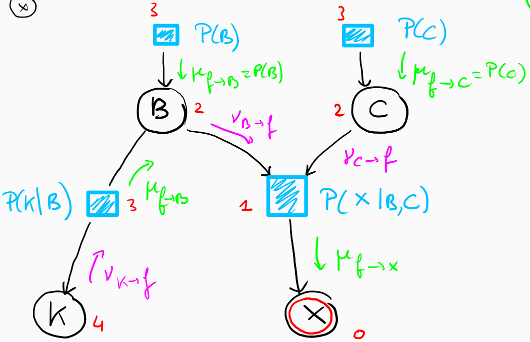
① distance from root X

② Messages from factors

③ Messages from variables

Factors ▨

Variables ⊗



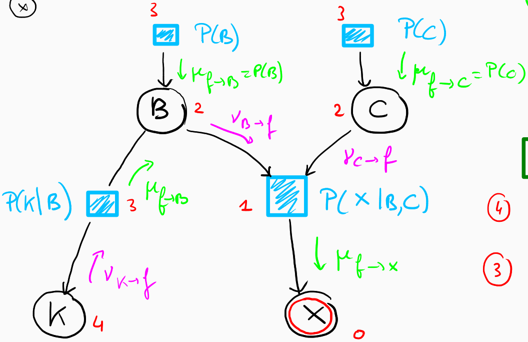
Joint distribution

$$(X, K, B, C) = P(X|B,C)\,P(C)\,P(K|B)\,P(B)$$

Message Passing - Example

Factors $\boxed{}$

Variables $\otimes$



① distance from root X

② Messages from factors

③ Message from variables

Compute From Leaves

④ $\nu_{K\to f} = 1$ (leaf)

③ $\varphi_B = \sum_K P(B)\cdot P(K|B)$

$\varphi_C = P(C)$

② $\varphi = \varphi_B \cdot \varphi_C$

① $\underbrace{\varphi_X}_{\mathbb{P}(X)} = \sum_{B,C} \varphi = \sum_{B,C} \mathbb{P}(C)\cdot \sum_K \mathbb{P}(B)\,\mathbb{P}(K|B)$

Joint distribution & Marginalisation

$$\mathbb{P}(X,K,B,C) = \mathbb{P}(X|B,C)\,\mathbb{P}(C)\,\mathbb{P}(K|B)\,\mathbb{P}(B)$$

$$\mathbb{P}(X) = \sum_{K,B,C} \mathbb{P}(X,K,B,C)$$

Message Passing - Algorithm

Initalisation of leaf nodes :

Messages from leaf node variable are set to 1

Messages from leaf node factors are set to the factor

Propagation

Starting from the further nodes

Variable to factor message : $\nu_{x \to f} = \prod_{g \in \mathtt{ne}(x) \setminus f} \mu_{g \to x}$

Factor to variable message : $\mu_{f \to x} = \sum_{X_f \setminus x} \prod_{y \in \mathtt{ne}(f) \setminus x} \nu_{y \to f}$

Implementation

Dag as an adjacent matrix

Message passing by memoisation

Numerical stability needs logs

Semantics of Probabilistic Programming

Verification of probabilistic Programs

Mc Iver, Morgan, Katoen

Hoare Logic - Principles

Language

$$
\begin{array}{rcl}
e & ::= & \texttt{true} \mid \texttt{false} \mid n \mid x \mid e\ op\ e \qquad \text{states } \Sigma : e \to val \\
op & ::= & + \mid - \mid * \mid = \mid \leq \mid and \mid not \\
s & ::= & \texttt{skip} \mid x := e \mid s; s \mid \texttt{if}\ \ e\ \texttt{then}\ \ s\ \texttt{else}\ \ s \mid \texttt{while}\ \ e\ \texttt{do}\ \ s
\end{array}
$$

Operational semantics

$$
\frac{}{\Sigma, x := e \rightsquigarrow \Sigma\{x \leftarrow \mathrm{ev}_\Sigma(e)\}, \texttt{skip}}
$$

$$
\frac{}{\Sigma, (\texttt{skip}; s) \rightsquigarrow \Sigma, s} \qquad \frac{\Sigma, s_1 \rightsquigarrow \Sigma', s_1'}{\Sigma, (s_1; s_2) \rightsquigarrow \Sigma', (s_1'; s_2)}
$$

$$
\frac{\mathrm{ev}_\Sigma(e) = \texttt{false}}{\Sigma, \texttt{if}\ \ e\ \texttt{then}\ \ s_1 \texttt{else}\ \ s_2 \rightsquigarrow \Sigma, s_1} \qquad \frac{\mathrm{ev}_\Sigma(e) = \texttt{true}}{\Sigma, \texttt{if}\ \ e\ \texttt{then}\ \ s\ \texttt{else}\ \ s \rightsquigarrow \Sigma, s_2}
$$

$$
\frac{\mathrm{ev}_\Sigma(e) = \texttt{false}}{\Sigma, \texttt{while}\ \ e\ \texttt{do}\ \ s \rightsquigarrow \Sigma, (s; \texttt{while}\ \ e\ \texttt{do}\ \ s)} \qquad \frac{\mathrm{ev}_\Sigma(e) = \texttt{true}}{\Sigma, \texttt{while}\ \ e\ \texttt{do}\ \ s \rightsquigarrow \Sigma, \texttt{skip}}
$$

Hoare Logic - Rules

Hoare triple $\{P\}s\{Q\}$ is valid if for any state Σ and Σ' such that $\Sigma, s \rightsquigarrow \Sigma', \texttt{skip}$, $[P]_\Sigma$ valid implies $[Q]_{\Sigma'}$ valid.

Rules

$$\frac{}{\{P[x := E]\}\ x\ :=\ E\ \{P\}} \qquad \frac{\{P\}\ C\ \{Q\} \qquad \{Q\}\ D\ \{R\}}{\{P\}\ C; D\ \{R\}}$$

$$\frac{\{E = \texttt{true}\ \land\ P\}\ C\ \{Q\} \qquad \{E = \texttt{false}\ \land\ P\}\ D\ \{Q\}}{\{P\}\ \texttt{if}\ E\ \texttt{then}\ C\ \texttt{else}\ D\ \{Q\}}$$

$$\frac{\{E = \texttt{true}\ \land\ I\ \}\ C\ \{I\}}{\{I\}\ \texttt{while}\ E\ \texttt{do}\ C\ \texttt{done}\ \{E = \texttt{false}\ \land\ I\}}$$

$$\frac{P' \ \Rightarrow\ P \qquad \{P\}\ C\ \{Q\} \qquad Q\ \Rightarrow\ Q'}{\{P'\}\ C\ \{Q'\}}$$

Soundness

If there is a proof tree with a hoare triple as root, then the program is correct with respect to pre and post conditions.

Weakest Precondition

Definition

Let Q be a predicate on the memory state and C be a program
$\mathrm{WP}(C, \cdot)$ is a predicate transformer, that is
$\mathrm{WP}(C, Q)$ **is the weakset precondition** P **such that** $\langle P \rangle \ \mathtt{C} \ \langle Q \rangle$

Inductive definition

$$\mathrm{WP}(\mathtt{x:=E}, Q(\mathtt{x})) = Q(\mathtt{E})$$
$$\mathrm{WP}(\mathtt{S_1;S_2}, Q) = \mathrm{WP}(\mathtt{S_1}, \mathrm{WP}(\mathtt{S_2}, Q))$$
$$\mathrm{WP}(\mathtt{if\ E\ then\ C\ else\ D}, Q) = (\mathtt{E} \wedge \mathrm{WP}(\mathtt{C}, Q)) \vee (\neg E \wedge \mathrm{WP}(\mathtt{D}, Q))$$
$$\mathrm{WP}(\mathtt{while\ E\ do\ C\ done}\{\mathrm{inv} I, \mathrm{var} V\}, Q) = I$$
$$\text{with} \begin{cases} \mathtt{E{=}true} \wedge I \wedge V = z \Rightarrow \mathrm{WP}(\mathtt{C}, I \wedge V < z) \\ I \Rightarrow V \geq 0 \\ (\mathtt{E{=}false} \wedge I) \Rightarrow Q \end{cases}$$

Examples

$\mathrm{WP}(\texttt{x:=y+3}, x > 3)$ = $x>3\,[x:=y+3]$ = $y+3>3$ = $y>0$.

$\mathrm{WP}(\texttt{n:=n+1}, n > 4)$

$\mathrm{WP}(\texttt{y:=x+2;y:=y-2}, y > 5)$

$\mathrm{WP}(\texttt{if x>2 then y:=1 else y:=-1}, y > 0)$

$\mathrm{WP}(\texttt{while n<>0 do n:=n-1 done } \{I = n \geq 0, V = n\}, n = 0)$

Verifying Probabilistic Programs

Language

$$\begin{array}{rcl} e & ::= & \texttt{true} \mid \texttt{false} \mid n \mid x \mid e\;op\;e \qquad \text{states } \Sigma : e \to val \\ op & ::= & + \mid - \mid * \mid = \mid \leq \mid and \mid not \\ s & ::= & \texttt{skip} \mid x := e \mid s; s \mid \texttt{if } e \texttt{ then } s \texttt{ else } s \mid \texttt{while } e \texttt{ do } s \mid \texttt{bernoul} \end{array}$$

Quantitative predicates : factors associate weights to states

$$\{f | f : |X| \to \mathbb{R}^+ \cup \{\infty\}\}$$

Weakest weighted precondition : $\mathrm{WP}[P]$ transform factors to factors.

Weakest Pre-expectation

Weakest weighted precondition

if f associates scores to states, then $g = \mathrm{WP}[P](f)$ maps a state s to the expected value of f after executing P on s. transform factors to factors.

Inductive definition

$$\mathrm{WP}[\mathtt{x:=E}](f) = f(\mathtt{x:=E}))$$
$$\mathrm{WP}[\mathtt{S}_1;\mathtt{S}_2] = \mathrm{WP}[\mathtt{S}_1](\mathrm{WP}[\mathtt{S}_2](f))$$
$$\mathrm{WP}[\mathtt{if\ E\ then\ C\ else\ D}](f) = [\mathtt{E}]\cdot\mathrm{WP}[\mathtt{C}](f)) + [\neg E]\cdot\mathrm{WP}[\mathtt{D}](f)$$
$$\mathrm{WP}(\mathtt{let\ x\ =\ sample\ bernoulli\ p\ in\ C})(f) = p\cdot\mathrm{WP}[\mathtt{C}](f[x := \mathit{true}])) + (1-p)\cdot\mathrm{WP}[\mathtt{C}](f[x := \mathit{false}])$$
$$\mathrm{WP}[\mathtt{while\ E\ do\ C\ done}](f) = \mathtt{lfp}\ X\cdot\ [\mathtt{E}]\cdot\mathrm{WP}[\mathtt{C}](X)) + [\neg E]\cdot f$$

Weakest Pre-expectation - Example

a=1, b=1

WP[while (a=1 and b=1) do let a = bern 0,5 in let b = bern 0,5 in a+b+c] (res = 0)

WP[while (y=1) { let a = sample bern 0,3 in if a then y := 0 else c := c+1 }] (f = 1)

WP[let a = bern 4/5 in if a then x := 5 else x := 10] (x)

WP[let a = bern 4/5 in if a then x := x+5 else x := 10] (x)

WP[let a = bern 4/5 in if a then x := x+5 else x := 10] (x = 10)

Take home

Semantics for (discrete) probabilistic programs

Probabilistic coherent spaces
Bayesian Networks and message passing
Weakest Predicate Tranformer for probabilistic programs

References

Semantics of probabilistic programs by Dexter Kozen
An Introduction to Probabilistic Programming. Jan-Willem van de Meent, Brooks Paige, Hongseok Yang and Frank Wood
Probabilistic Coherent Spaces by Vincent Danos and Thomas Ehrhard
Bayesian Reasoning and Machine Learning by David Barber
Deriving probabilistic semantics via the 'weakest completion' by Mc Iver and Morgan
Probabilistic Weakest Precondition by Katoen & al.