Probabilistic Program

February 21, 2015

Intuition

Probabilistic programs are regular programs extended by two constructs [GHNR14]:

- The ability to draw random values from probability distributions.
- The ability to condition values computed in the programs on probability distributions.

A probabilistic program implicitly defines a probability distribution over program output.

Definition

A probabilistic program is a stateful deterministic computation $\mathcal{P}(\theta)$ with the following properties:

- Initially, $\mathcal{P}$ expects a value of θ as the argument.
- On every invocation, $\mathcal{P}$ returns either a distribution F, a distribution and a value (G, y), a value z, or $\perp$.
- Upon returning F, $\mathcal{P}$ expects a value x drawn from F as the argument to continue.
- Upon returning (G, y) or z, $\mathcal{P}$ is invoked again without arguments.
- Upon returning $\perp$, $\mathcal{P}$ terminates.

A program is run by calling $\mathcal{P}$ repeatedly until termination. Every run of the program implicitly produces a sequence of pairs (F_i, x_i) of distributions and values drawn from them. We call this sequence a *trace* and denote it by $\boldsymbol{x}$. A trace induces a sequence of pairs (G_j, y_j) of observed random variables and their values. We call this sequence an *image* and denote it by $\boldsymbol{y}$. We call a sequence of values z_k an *output* of the program and denote it by $\boldsymbol{z}$. Program output is deterministic given the trace.

By definition, the probability of a trace is proportional to the product of the probability of all random choices $\boldsymbol{x}$ and the likelihood of all observations $\boldsymbol{y}$:

$$p_{\mathcal{P}}(\boldsymbol{x}|\theta) \propto \prod_{i=1}^{|\boldsymbol{x}|} p_{F_i}(x_i) \prod_{j=1}^{|\boldsymbol{y}|} p_{G_j}(y_j) \tag{1}$$

$p_{\mathcal{P}}(\boldsymbol{x}|\theta)$ has the interpreration of the posterior probability $p(\boldsymbol{x}|\boldsymbol{y})$ given the prior probability $p(\boldsymbol{x}) = \prod_{i=1}^{|\boldsymbol{x}|} p_{F_i}(x_i)$ and the likelihood $p(\boldsymbol{y}|\boldsymbol{x}) = \prod_{j=1}^{|\boldsymbol{y}|} p_{G_j}(y_j)$.

The objective of inference in probabilistic program $\mathcal{P}$ is to discover the distribution $p_{\mathcal{P}}(\boldsymbol{z}|\theta)$ of program output $\boldsymbol{z}$.

Implementation

Several implementations of general probabilistic programming languages are available [GMR+08, MSP14, WvdMM14] and the list is growing: `http://probabilistic-programming.org/wiki/Home`. Inference is usually performed using Monte Carlo sampling algorithms for probabilistic programs [WSG11, WvdMM14, PWDT14]. While some algorithms are better suited for certain inference types, most can be used with any valid probabilistic program.

References

[GHNR14] Andrew D. Gordon, Thomas A. Henzinger, Aditya V. Nori, and Sriram K. Rajamani. Probabilistic programming. In *International Conference on Software Engineering (ICSE, FOSE track)*, 2014.

[GMR+08] Noah D. Goodman, Vikash K. Mansinghka, Daniel M. Roy, Keith Bonawitz, and Joshua B. Tenenbaum. Church: a language for generative models. In *Proc. of Uncertainty in Artificial Intelligence*, 2008.

[MSP14] Vikash K. Mansinghka, Daniel Selsam, and Yura N. Perov. Venture: a higher-order probabilistic programming platform with programmable inference. *CoRR*, abs/1404.0099, 2014.

[PWDT14] B. Paige, F. Wood, A. Doucet, and Y.W. Teh. Asynchronous anytime sequential Monte Carlo. In *Advances in Neural Information Processing Systems*, 2014.

[WSG11] David Wingate, Andreas Stuhlmüller, and Noah D. Goodman. Lightweight implementations of probabilistic programming languages via transformational compilation. In *Proc. of the 14th Artificial Intelligence and Statistics*, 2011.

[WvdMM14] Frank Wood, Jan Willem van de Meent, and Vikash Mansinghka. A new approach to probabilistic programming inference. In *Artificial Intelligence and Statistics*, 2014.