A Lambda-Calculus Foundation for Universal Probabilistic Programming

Johannes Borgström¹, Ugo Dal Lago², Andrew D. Gordon³,⁴, and Marcin Szymczak⁴

¹ Uppsala University
² University of Bologna
³ Microsoft Research
⁴ University of Edinburgh

Abstract. We develop the operational semantics of a probabilistic λ-calculus with continuous distributions, as a foundation for universal probabilistic programming languages such as Church, Anglican, and Venture. Our first contribution is to adapt the classic operational semantics of λ-calculus to the continuous case, via creating a measure space on terms and defining step-indexed approximations. We prove equivalence of big-step and small-step formulations of this distributional semantics. Our second contribution is to formalize the implementation technique of trace MCMC for our calculus and to show correctness. A key step is defining a sampling semantics of a term as a function from a trace of random samples to a value, and showing that the distribution induced by integrating over all traces equals the distributional semantics. Another step is defining sufficient conditions for the distribution induced by trace MCMC to converge to the distributional semantics. To the best of our knowledge, this is the first rigorous correctness proof for trace MCMC for languages in the Church family.

1 Semantics for Universal Probabilistic Programming


The following is the general form of a probabilistic query in Church:

( query (define x₁ e₁) ... (define xₙ eₙ) e_q (condition e_c) )

The result of the query is a representation of the distribution given by the probabilistic expression e_q, given variables xᵢ defined by potentially probabilistic expressions eᵢ, conditioned by the binary predicate e_c.

For example, the following query defines a probability p at random, defines a function to flip a coin with bias p, conditions the model on single observations of 0 and 1, and returns a representation of the distribution of p. We use calls (rnd) to sample a probability from the uniform distribution on the unit interval.
The first problem we address in this work is to provide an operational semantics for universal probabilistic programming languages. Our example illustrates the common situation in machine learning that models are based on continuous distributions (such as \((\text{rnd})\)), but previous work on operational semantics for probabilistic \(\lambda\)-calculi are based on discrete distributions.

We introduce a call-by-value \(\lambda\)-calculus with primitives for random draws from various continuous distributions (such as \((\text{rnd})\)), and exceptions to represent conditioning. We describe a measure space of \(\lambda\)-terms and let \(D\) range over value distributions, that is, measures on values of the \(\lambda\)-calculus. We define step-indexed operational semantics, in both small-step \((M \rightarrow_n D)\) and big-step \((M \Downarrow_n D)\) styles, which we prove equivalent, and enable us to define the distributional semantics \(D = \llbracket M \rrbracket\) for each closed term \(M\).

2 Semantics and Correctness of Trace MCMC

The original implementation of Church introduced the implementation technique trace MCMC [4]. A closed \(\lambda\)-term \(M\) has a trace \(s\) if there is a run of \(M\) making a finite sequence of random choices \(s\), which yields a result \(V(s)\) functionally dependent on \(s\). In our example, each trace has form \(s = [p, q_1, q_2]\) where \(p\) is the bias of the coin, and each binary flip \(b_i\) is true if and only if \(q_i < p\).

We consider trace MCMC as an instance of the Metropolis-Hastings (MH) algorithm. Given a closed term \(M\), trace MCMC generates a Markov chain of traces, with a stationary distribution on traces that induces a distribution over values corresponding to the semantics of \(M\). The algorithm is parametric in a target distribution \(\mathcal{D}\) and a proposal kernel \(Q\), that takes any trace \(s\) of \(M\) to a probability distribution over traces, corresponding to a perturbation of \(s\).

We formalize the algorithm rigorously, defining the target distribution on program traces and the proposal kernel as Lebesgue integrals of corresponding density functions. We prove these functions measurable with respect to the \(\sigma\)-algebra on program traces—a step usually omitted in similar developments.

To define the target distribution, we give a deterministic sampling semantics for our calculus based on the explicit consumption of a program trace \(s\) of random draws and production of an explicit weight \(w\) for each outcome. We formulate the sampling semantics in big-step style \((M \Downarrow^*_w V)\) and small-step style \(((M, w, s) \rightarrow (M', w', s'))\) and prove equivalence. Moreover, we prove that the value distributions induced by the sampling and distributional semantics are indeed the same. We exploit this equivalence to show that, subject to sufficient aperiodicity and irreducibility conditions on the transition kernel induced by \(Q\) and the acceptance ratio, the distribution on values induced by trace MCMC on \(M\) converges to the distributional semantics \(\mathcal{D} = \llbracket M \rrbracket\).
3 Related Work (Partial)

To the best of our knowledge, the only previous theoretical justification for trace MCMC is the recent work by Hur and others [7], who show correctness of trace MCMC for the imperative probabilistic language R2 [10]. Their result does not apply to higher-order languages such as Church.

While giving a domain theory for probabilistic $\lambda$-calculi is known to be hard [8], there have been recent advances on, e.g., probabilistic coherent spaces [1, 3] and also game semantics [2], which in some cases are not only adequate, but also fully abstract. We don’t see strong obstacles in applying all these to our calculus, but this remains outside the scope of this contribution.

The full version of this paper, in preparation, will include a full bibliography.

References


