Nested Inference for Reactive Probabilistic Programming

Guillaume Baudart

March-August 2024

Abstract

This internship will contribute to the development of ProbZelus — a reactive probabilistic programming language — by adding support for nested inference to implement advanced controllers using, e.g., planning, or reinforcement learning.

Over the last few years, Probabilistic Programming Languages (PPL) have been introduced to describe probabilistic models and automatically infer distributions of parameters from observed data. Building on recent developments on probabilistic programming, we have been working on ProbZelus [1]: a synchronous language extended with probabilistic constructs. Dataflow synchronous languages are stream languages that are routinely used in industry for the design of critical embedded systems. In ProbZelus, a program is a stream processor which never stops, and a probabilistic model operates on infinite streams. A key feature of ProbZelus the ability to access partial results during inference. During execution, the inferred distribution can be used by deterministic components to compute new inputs that can, in turn, influence the inference.

As an example, a simplified controller for a self-driving car can be programmed as follows in ProbZelus. The driver relies on an estimation of its position to compute a command (e.g., the steering angle) which, in turn, influences the tracker model for the estimation of the next position.

```prob
proba tracker (u, obs) = x where
rec mu = x0 -> motion(pre x, u)
and x = sample (gaussian (mu, noise))
and () = observe (gaussian (x, noise), obs)

node driver (obs) = u where
rec x_dist = infer tracker (u, obs)
and u = u0 -> controller (mean (pre x_dist))
```

Using data gathered from observing the environment (e.g., obs in driver), a probabilistic program automatically infers the distribution of unobserved parameters (e.g., the position x) from prior beliefs (e.g., a simple Gaussian distribution) using Bayesian inference.
Nested inference [3] is the ability to nest probabilistic programming queries. This ability is critical to model agents that can reason about other agents [2], and can be used to implement advanced controllers using, e.g., planning, or reinforcement learning. For instance, a Markov Decision Process continuously simulates possible future states to reach a decision [4].

The goal of this internship is to add support for nested inference to the ProbZelus language. Compared to previous work [3, 4] the reactive context where programs never stops is a key challenge which needs to be addressed during this internship.

There are several possible research directions:

**Language Design** Add the programming language constructs for nested inference and define their semantics.

**Runtime** Identify and implement probabilistic inference algorithms that can be used for nested inference in a reactive context.

**Semantics** Define a framework to reason about program equivalence for models with nested queries.

The ideal candidate should have a strong interest in programming languages (semantics and implementation), and be curious about probabilistic programming and possible applications.

Contact Guillaume Baudart guillaume.baudart@inria.fr if you have any questions, or if you would like to apply for this internship.

**References**


