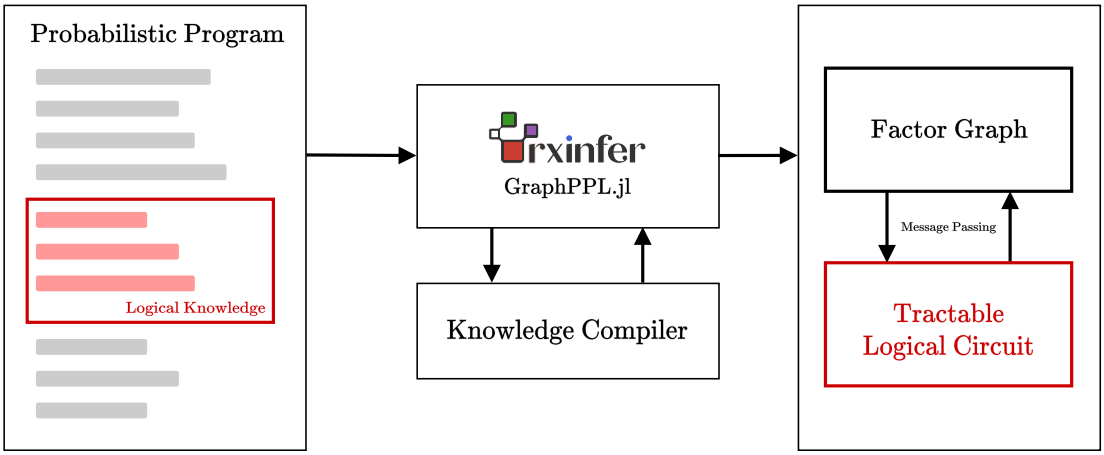
Leveraging Logical Circuits in Probabilistic Programming Languages

Joint project with TU Eindhoven



Probabilistic Programming Languages (PPLs) allow users to specify probabilistic models in a declarative fashion while the PPL framework performs probabilistic inference under the hood. For example, the open source Julia-package `RxInfer` converts a given probabilistic program to a (loopy) factor graph and uses message passing algorithms to perform (approximate) Bayesian inference. Sometimes, the user wants to include *logical knowledge* (i.e., boolean functions) in their program. While these logical constraints can also be converted to a (possibly loopy) factor graph, we wish to *leverage structure* in the boolean formula to construct so-called *tractable circuits* which allow us to answer inference queries *exactly and efficiently*. This conversion process is called *Knowledge Compilation*. In this project, we wish to implement this idea on the system level of `RxInfer`, giving rise to a framework that combines message passing with exact inference using tractable logical circuits.

Goals & Tasks

- Review of the state-of-the-art on Knowledge Compilation.
- Extending `RxInfer` to handle logical knowledge in probabilistic programs by compiling it to tractable circuits.
- Exploring fallback strategies in case of intractable compilation.

Qualifications

- Interested in probabilistic inference and tractable circuits.
- Experience with a scripting language (e.g., Python). Experience in software engineering is beneficial.
- Note: Prior knowledge of Julia or Probabilistic Programming Languages is **not** required.
- Registered to one of the following:
  - ✓ **Seminar Project**
  - ✓ **Master Thesis**

Contact

Robert Peharz & Thomas Wedenig
robert.peharz@tugraz.at
thomas.wedenig@tugraz.at