GNA

Data Analysis Framework for Neutrino Experiments

A. Fatkina M. Gonchar D. Naumov K. Treskov

JINR DLNP

The XXII International Scientific Conference of Young Scientists and Specialists (AYSS-2018)

Outline

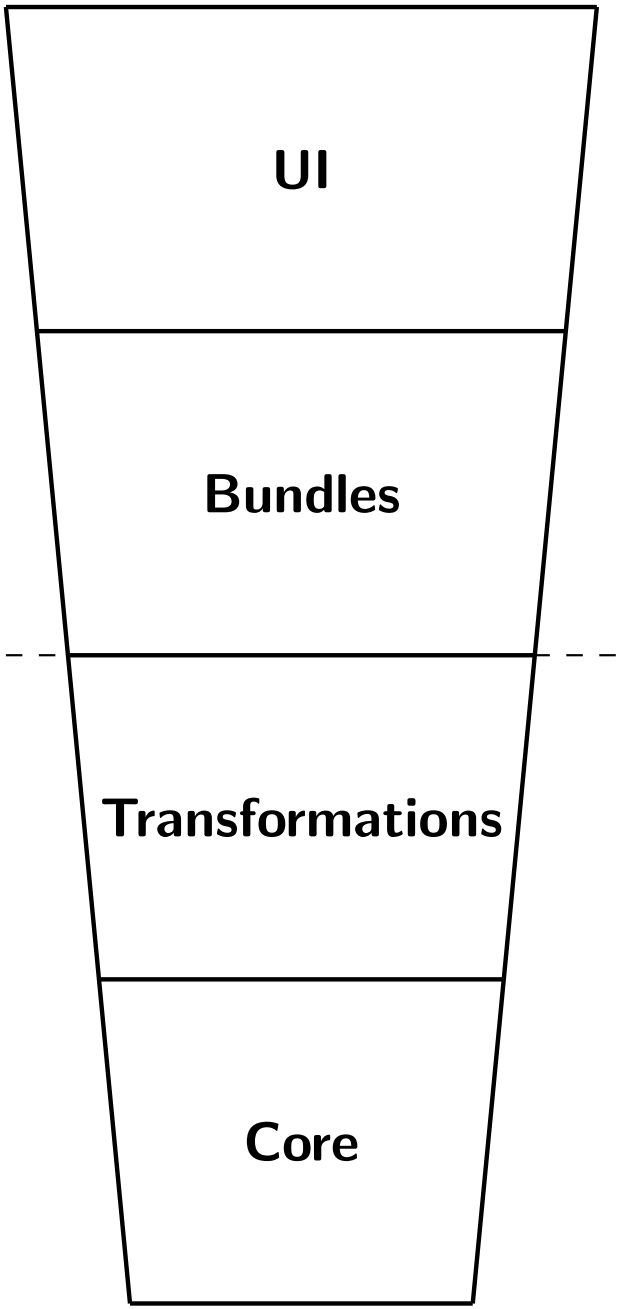
Introduction

GNA (Glabal Neutrino Analysis) — flexible, extensible framework for the data analysis of neutrino experiments.

GNA goals

- Data analysis for JUNO and Daya Bay experiments.
- Global analysis of neutrino data (experiments: Daya Bay, JUNO, NOνA, T2K, etc.).

GNA Structure



- Comprehensive command line chain.
- Computational graphs.
- Analysis.

- Read configuration.
- Variables.
- Small computational graph.

Python
(flexibility)

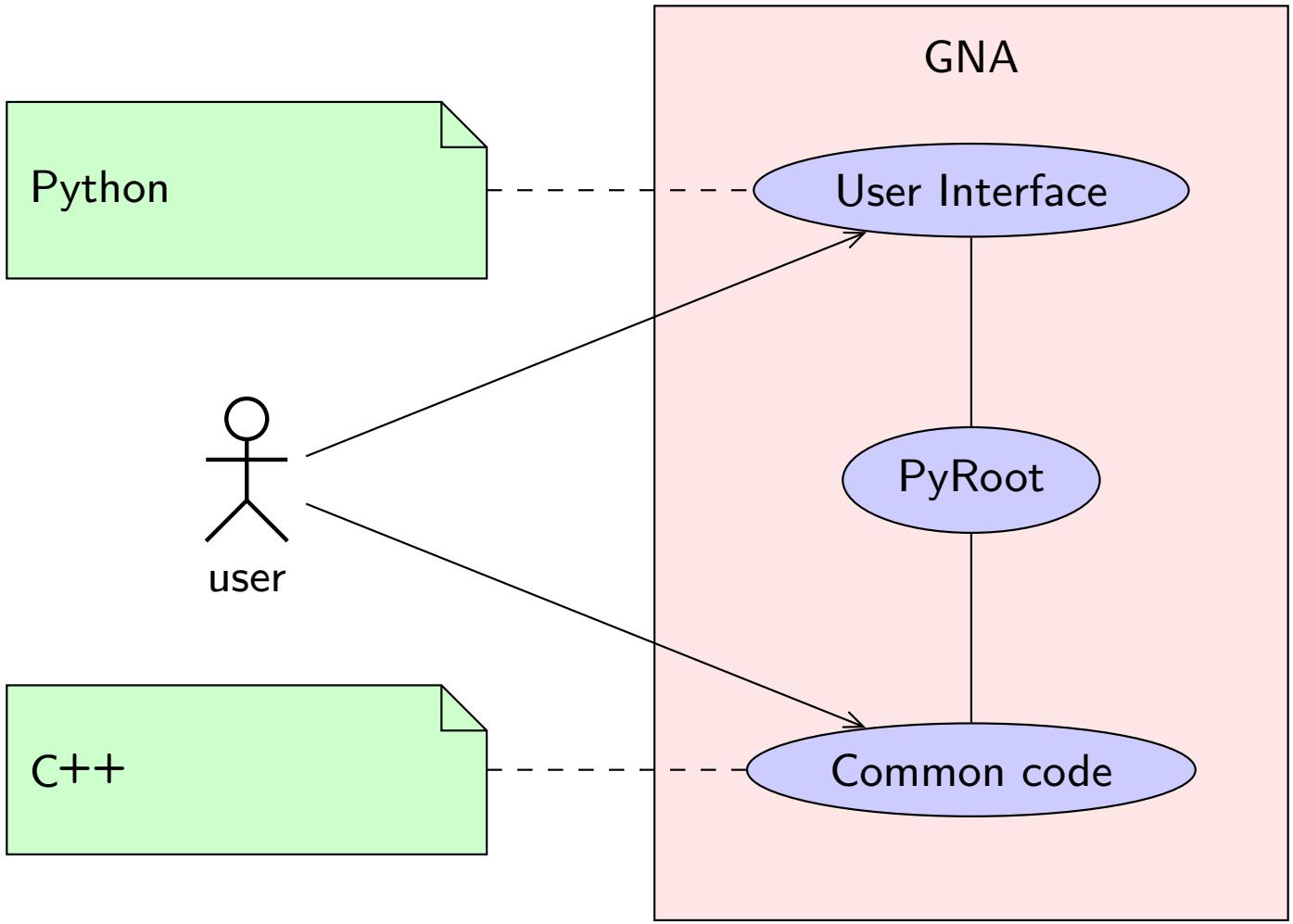
C++
(efficiency)

- Linear algebra
- Integration

- Statistics
- Physics

- Data
- Variable
- Transformation

General schema

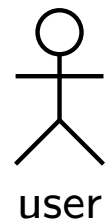


GNA schema.

User categories

Two categories of GNA user are expected:

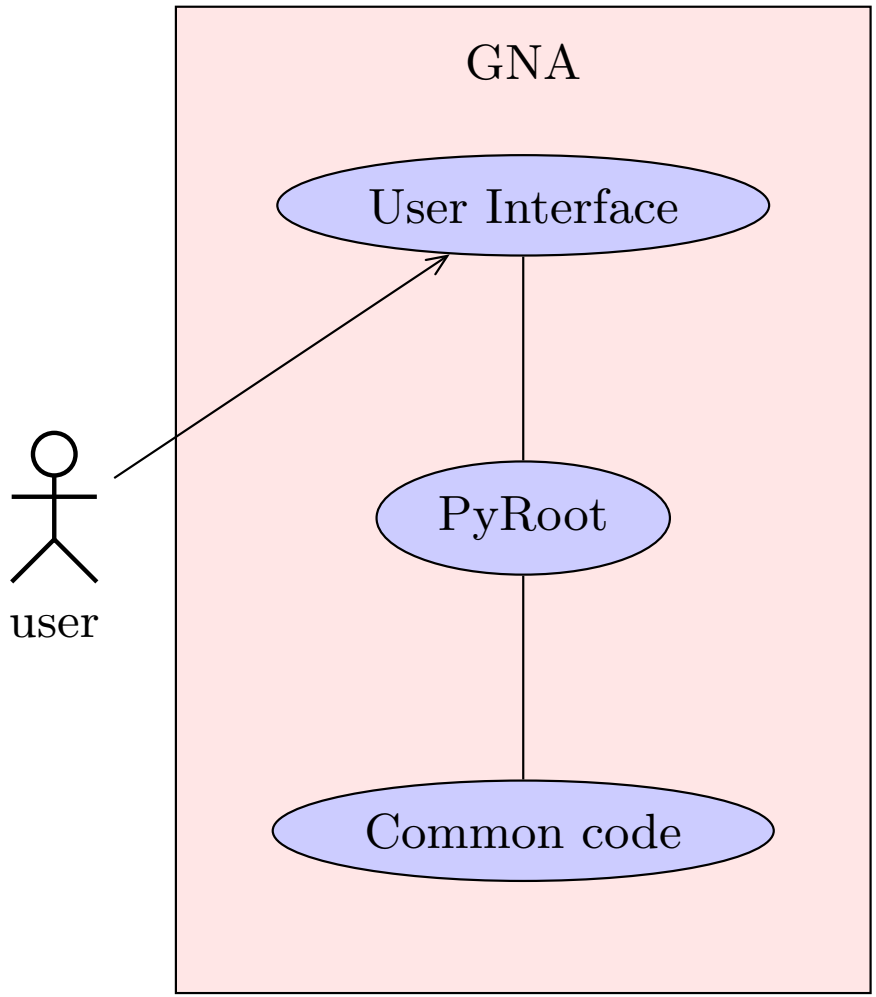
- End-user.
- Third-party transformation writer.



User categories

End-user

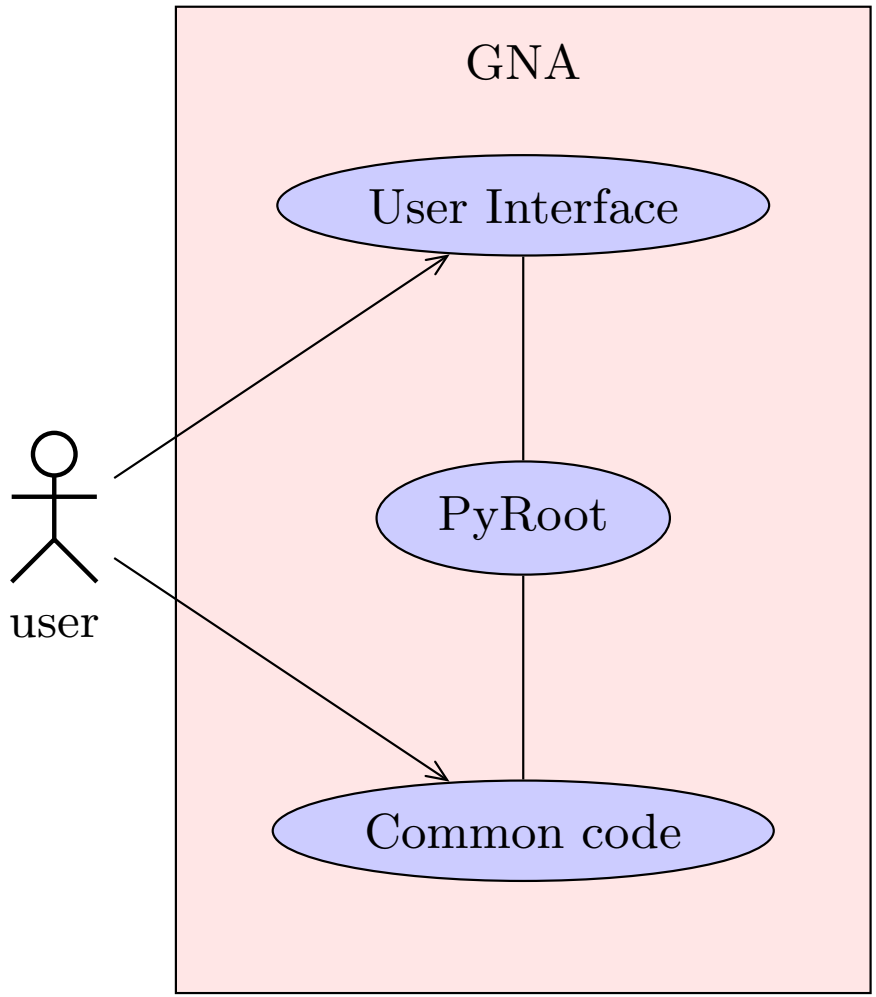
- Assembles computational chain by binding blocks (transformations) via Python UI.
- Doesn't need to think how it works on a low level.



User categories

Third-party developer

- Implements algorithms in C++.
- Implements interface in Python, integrates it into GNA environment. Linking is provided on the framework level.
- Assembles computational chain by binding blocks (transformations) via Python UI.
- Other programming issues.



Transformation

Transformation is an encapsulated function, basic component of computations.

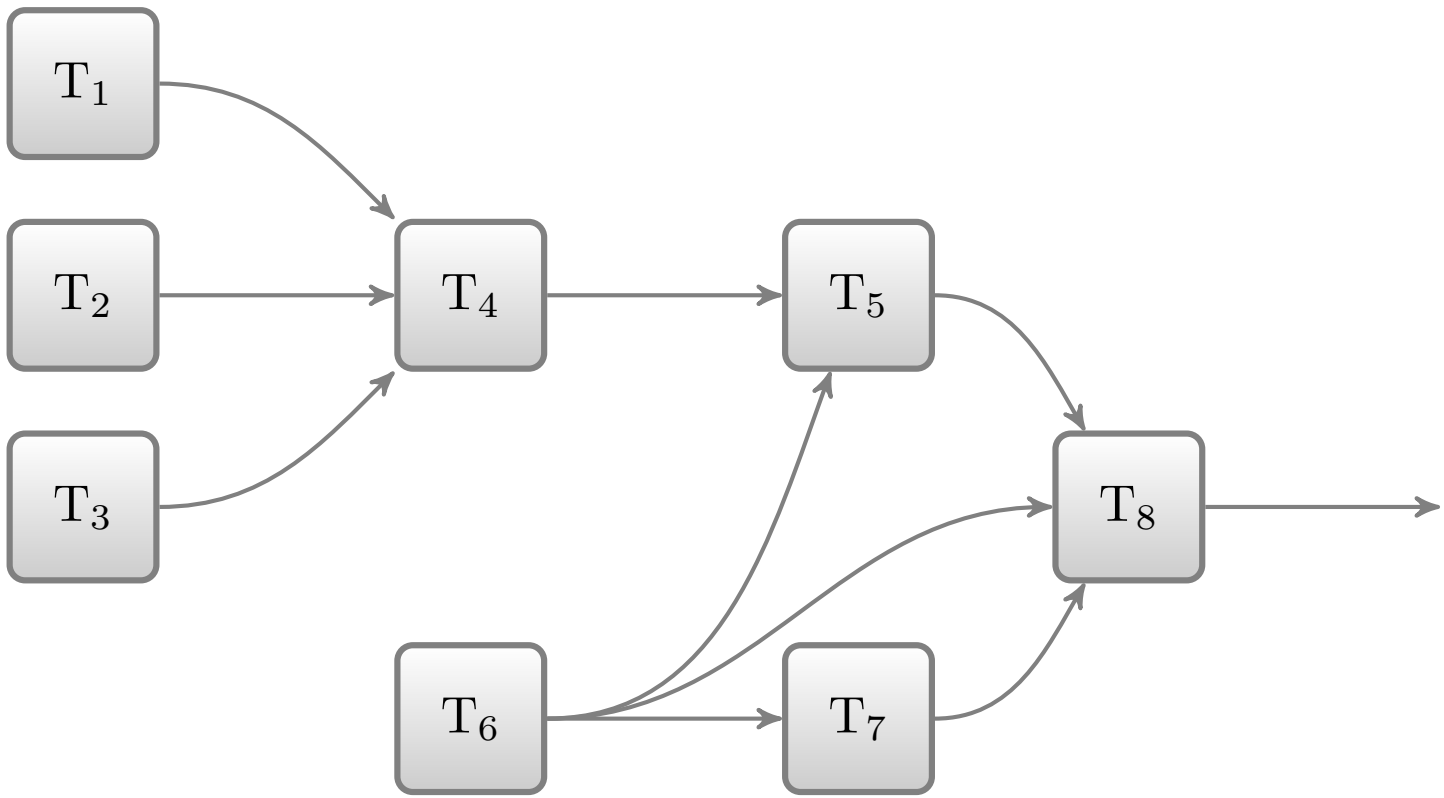
Highlights

- May have zero or more inputs and has at least one output (arrays).
- May depend on parameters (variables).
- Data container is associated with each transformation output.
- Transformation has taint flag. It is recomputed in case of it was tainted only. Taint flag is true when data is not up-to-date.

Computational graph

Schematic view

Transformations may be bound in arbitrary way.



An example with transformation of different types.

Computational graph

Highlights

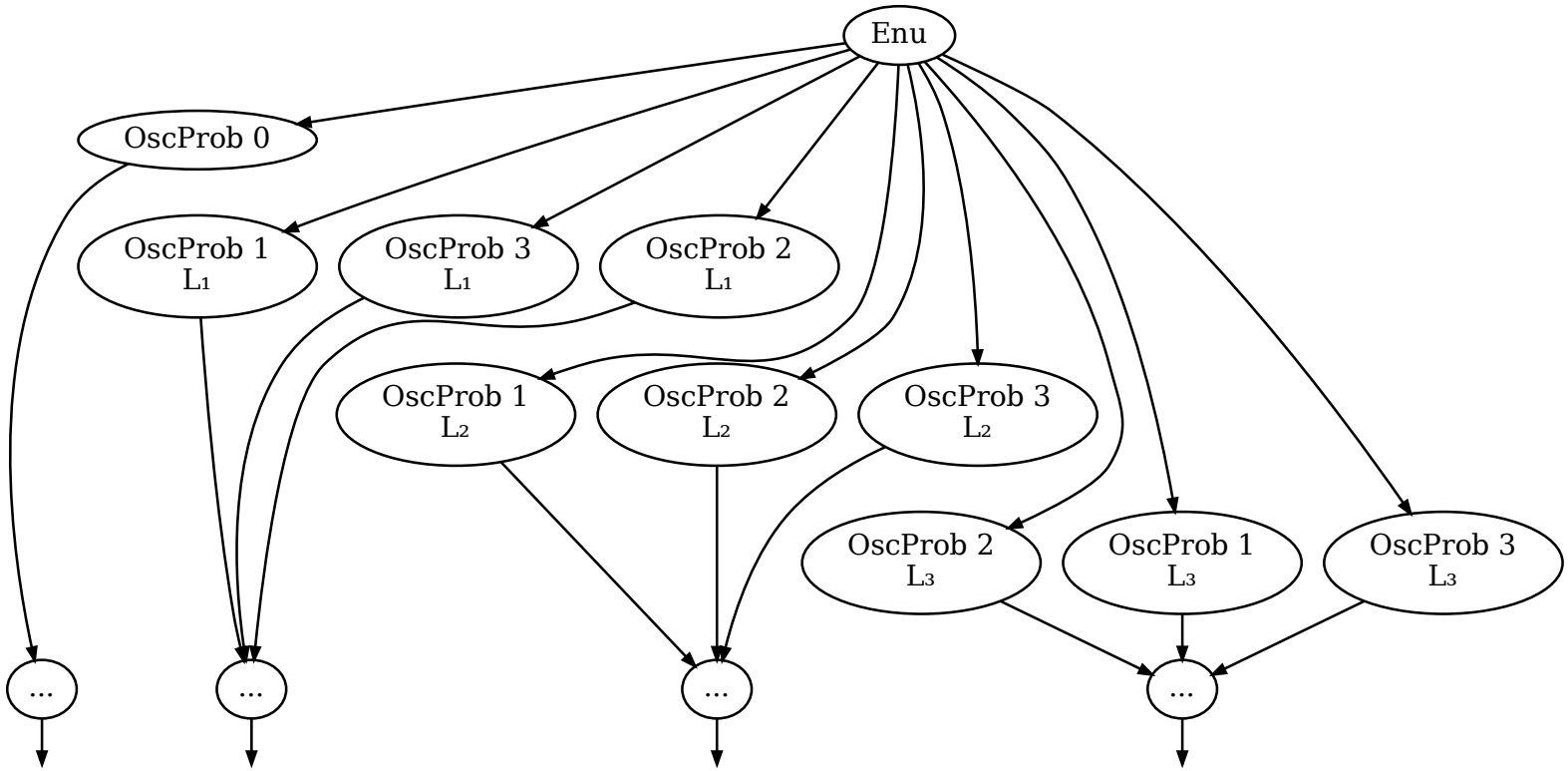
- Outputs are usually arrays.
- Input is a view to output of other corresponding transformation.
- Transformation results are cached.

Two stages of computations:

1. Building the computational graph — occurs once:
   - Check inputs types.
   - Infer output types.
2. Evaluation on demand.

Computational graph

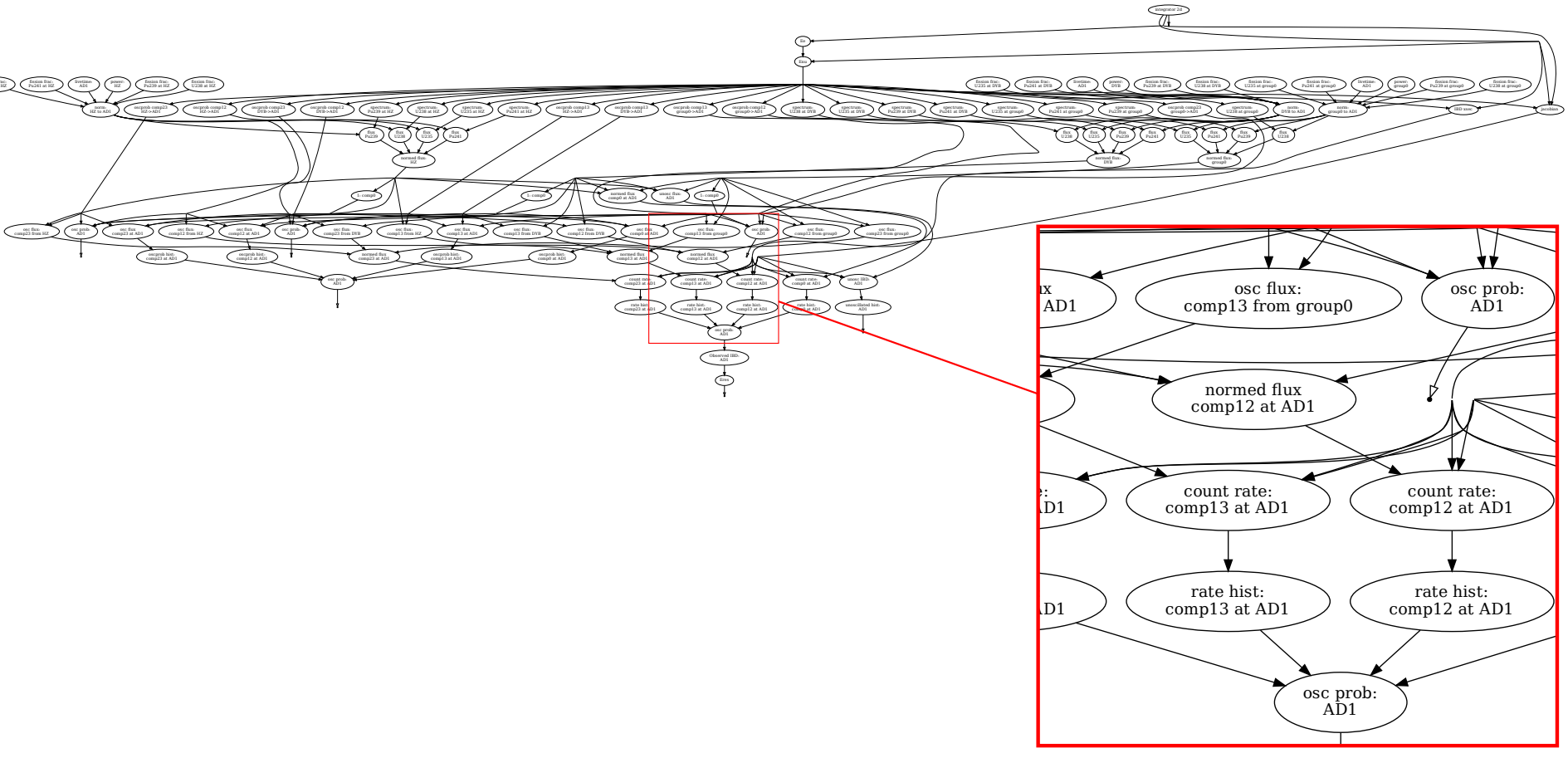
A part of JUNO graph



A part of JUNO computational graph. Oscillation probability with different input parameters and single input vector with energy values.

Computational graph

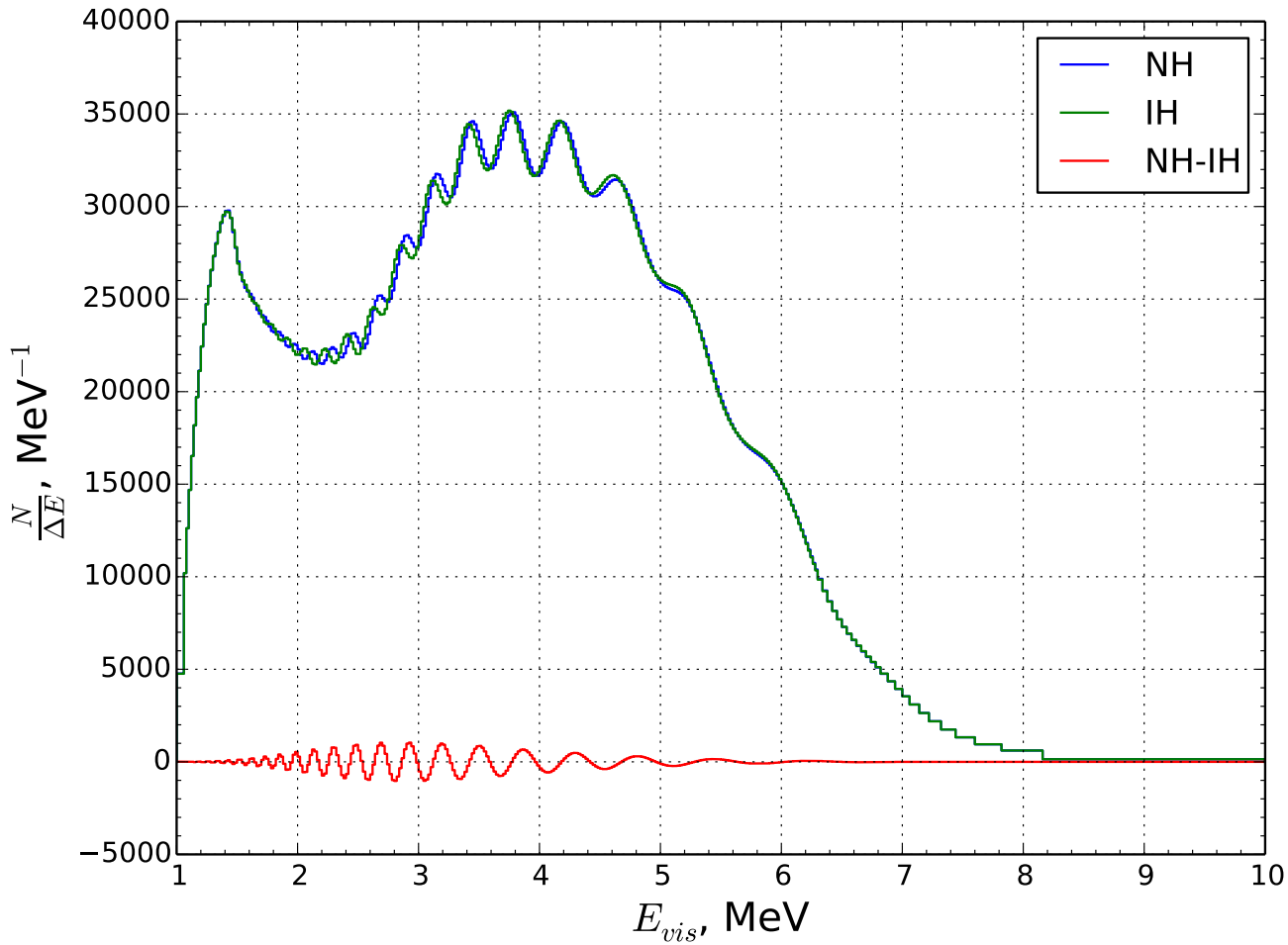
The whole JUNO graph



- The JUNO graph contains 110 nodes and 174 edges.
- It produces a histogram of 500 bins.

Computational graph

Example of JUNO graph output



Antineutrino spectra expected to be observed in JUNO experiment for different mass hierarchies.

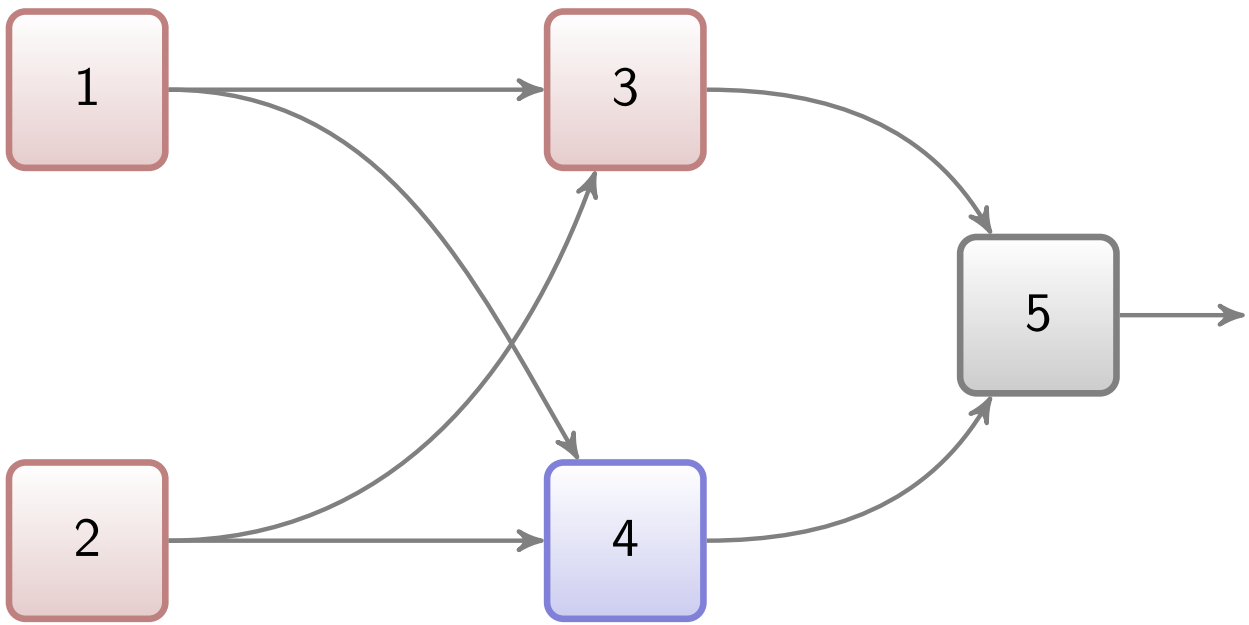
Features

Lazy evaluation

Computations are executed only in case the value is used.

Caching

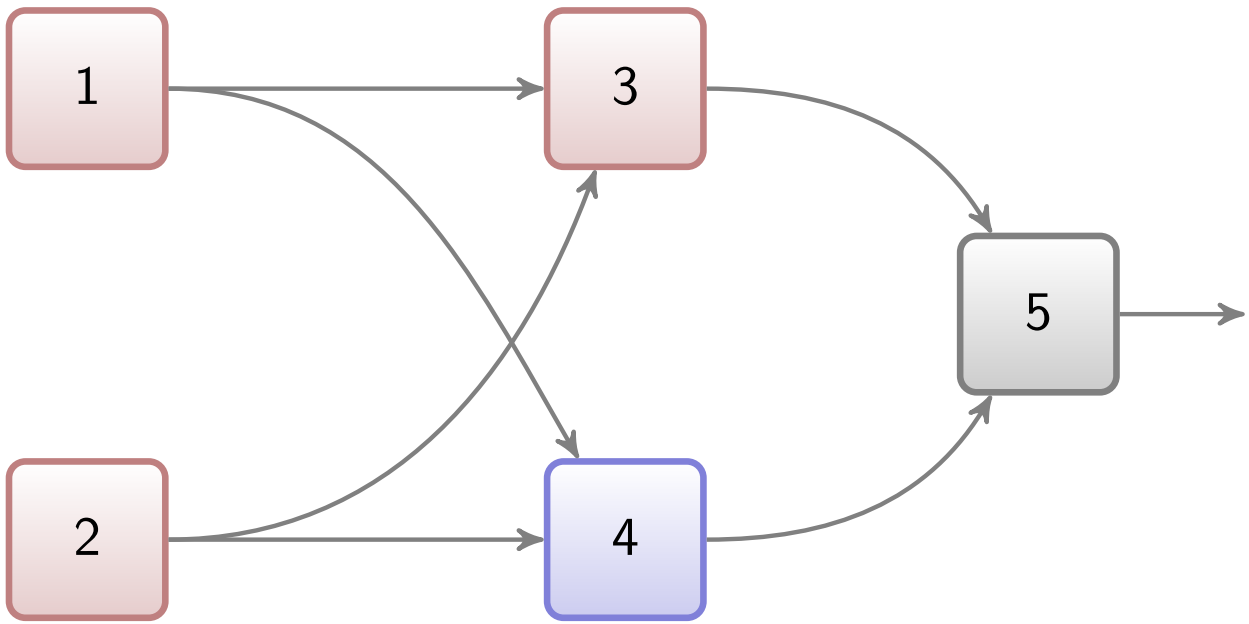
Transformation is computed once and the result may be reused.



Features

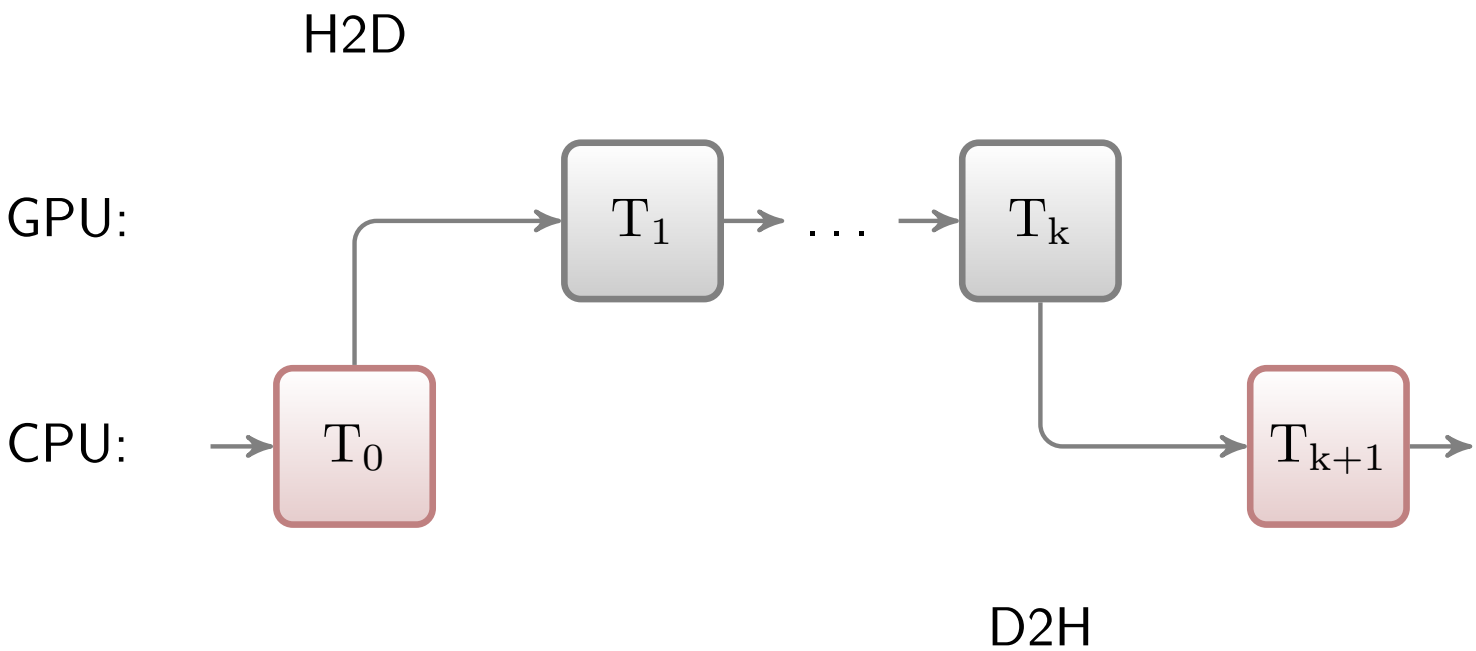
Multithreading

- Transparent for the end-user.
- Is being developed now.
- Threads are created once.
- Prevent race conditions.



Features — CUDA support

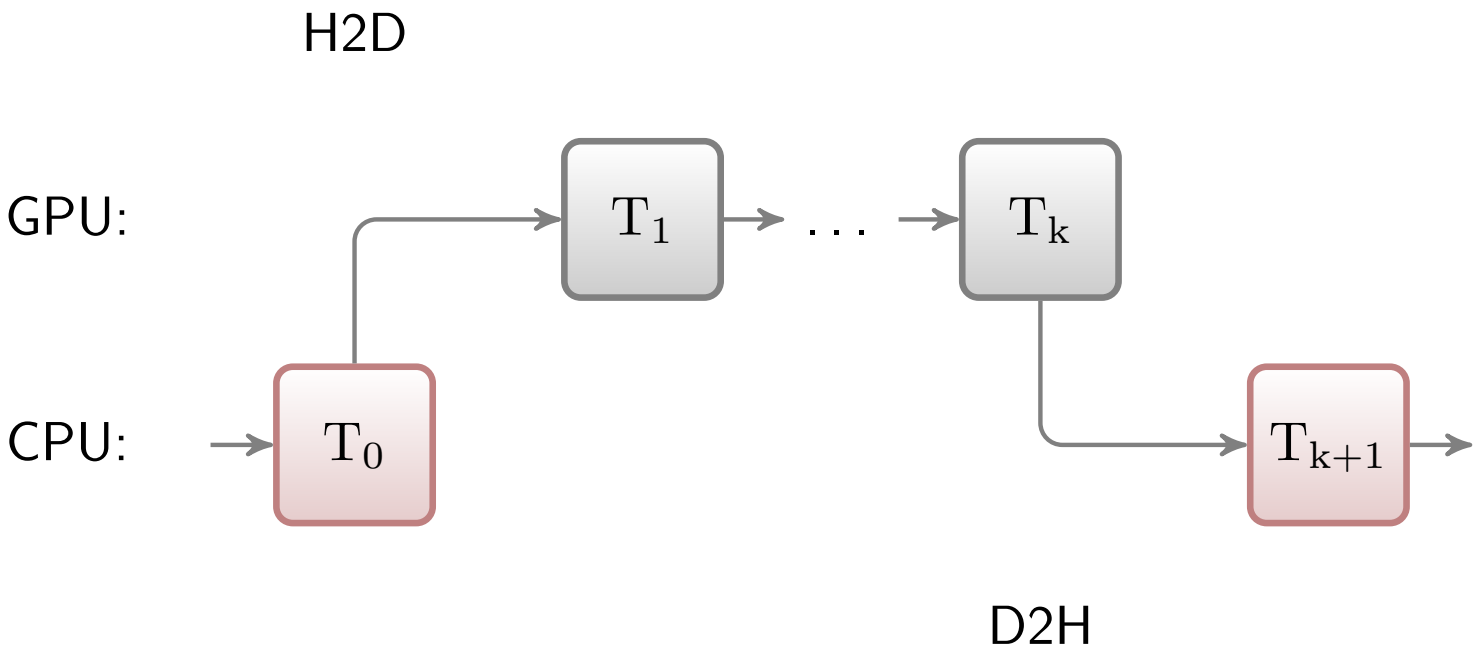
- Transparent for the end-user GPU support.
- Separate library inside the framework.
- If you don't need it — switch it off. No overhead will be produced.



,,CUDA Support in GNA Data Analysis Framework" (ICCSA 2018).
https://arxiv.org/abs/1804.07682

Features — CUDA support

- Provides a wrapper for data arrays with automatic data management.
- Achieved x20 acceleration for oscillation probability transformation (double precision values).



,,CUDA Support in GNA Data Analysis Framework" (ICCSA 2018).
https://arxiv.org/abs/1804.07682

Prospects and summary

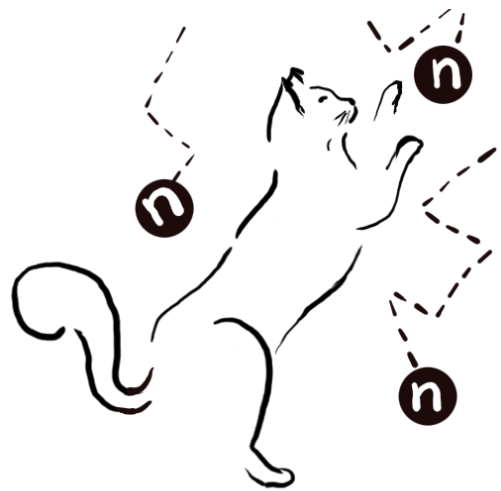
The framework is being actively developed now. Our plans include:

- Implementation of the Daya Bay and NOvA oscillation analysis.
- Multicore CPU + GPU systems support.
- Porting the existing transformations on the GPU.
- Unit-tests and general documentation.
- Global analysis of neutrino data produced by several experiments.

Current status:

- Flexible framework for data analysis of neutrino experiments.
- May be extended by user-defined transformations.
- The framework is used for the JUNO sensitivity study by several groups.

Thank you for your attention!



Any questions?