FERMILAB-POSTER-24-0297-CSAID

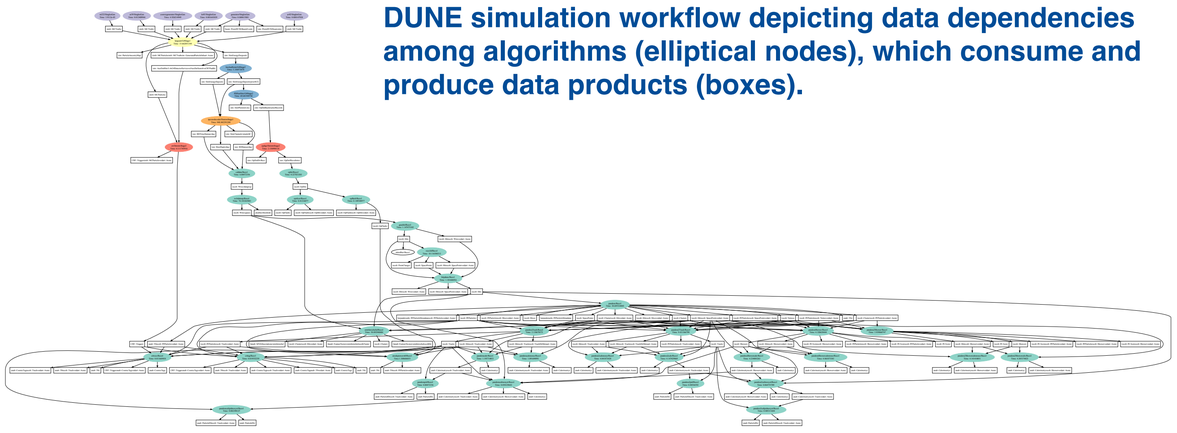
Data-flow parallelism for high-energy and nuclear physics frameworks

Kyle J. Knoepfel, Marc Paterno, Saba Sehrish, Chris Green

Fermi National Accelerator Laboratory

Data-flow parallelism

Computing workflows in high-energy and nuclear physics can generally be expressed as directed acyclic graphs according to the data dependencies among algorithms.



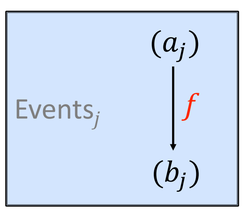
DUNE simulation workflow depicting data dependencies among algorithms (elliptical nodes), which consume and produce data products (boxes).

Graph-based processing approaches are not often used in HENP due to implicit dependencies between algorithms, serialization among thread-unsafe libraries, and difficulties in short-circuiting processing with filters.

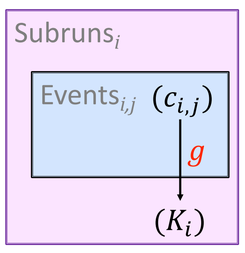
Processing idioms

HENP algorithms tend to very procedural, often obscuring the nature of the computation being performed. However, almost all algorithms can be expressed according to patterns using higher-order functions:

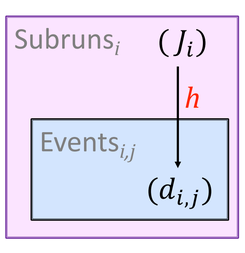
- **Transform:** A user-provided function f is applied to each data product in the sequence (a_j), creating another sequence (b_j).



- **Fold:** A subrun data product K_i is created by applying a user-provided fold operation g on each data product in the sequence $(c_{i,j})$.

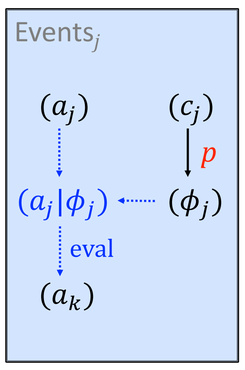


- **Unfold:** A sequence of data products $(d_{i,j})$ is produced by applying a user-provided unfold operation h on one subrun data product J_i.



- **Filter:** A user-provided predicate p is applied to each elements of the sequence (c_j), creating a sequence of Boolean results (ϕ_j).

  The sequences (a_j) and (ϕ_j) are zipped together and evaluated to yield a possibly shorter new sequence (a_k).



This manuscript has been authored by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics.

Collaborating with Intel oneTBB developers

A Fermilab LDRD project (**Meld**) explored using Intel's **oneTBB flow graph** and higher-order functions to process HENP data.

EPJ Web of Conferences **295**, 05014 (2024)

As a result, a working relationship has formed between Fermilab developers and Intel oneTBB flow graph developers to better support HENP.

Intel oneTBB flow-graph spec

Proposed new flow-graph functionalities

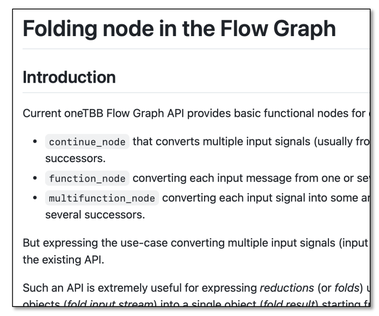
Serializer node: sometimes thread-unsafe software must be invoked from multiple nodes. It's insufficient to specify a "serial" concurrency for each node as separate nodes can still be executed in parallel.

- Each thread-unsafe library has a dedicated node that sends and receives one token—the user's algorithm is not invoked until it receives the token.

https://github.com/oneapi-src/oneTBB/pull/1526

Fold node: accepts multiple input messages (one per sequence element) and outputs one result per sequence.

- Internal atomic counters to ensure fold result emitted at the right time
- oneTBB created RFC to explore adding this to the flow-graph library



Intel RFC for fold nodes

Filtering support: oneTBB considering the addition of a class template `tbb::optional_msg<T>`, with potential short-circuiting behavior for disengaged ("null") objects.

Next steps

Our work was presented 10 October 2024 at the **UXL oneAPI DevSummit**.

The DUNE experiment is pursuing a framework that uses graph-based processing and higher-order functions.

oneTBB flow-graph developers are executing Meld benchmarks to test new ideas.

Some challenges remain (e.g.):

- Pairing physics data with calibration information
- Efficiently and safely invoking Python algorithms from algorithms wrapped by flow-graph nodes
- GPU/resource management
- Re-expressing existing algorithms and behaviors using higher-order functions