DAPHNE

Quentin GUILLOTEAU

Univ. Grenoble Alpes, INRIA, CNRS, LIG
firstname.lastname@inria.fr

2023-08-29

Context

- Integrated Data Analysis (IDA) pipelines
- e.g., HPC simulation $\rightsquigarrow$ Big Data processing $\rightsquigarrow$ ML training
- would require different tools, methods, expertises, platforms, etc.

Context

- Integrated Data Analysis (IDA) pipelines
- e.g., HPC simulation $\rightsquigarrow$ Big Data processing $\rightsquigarrow$ ML training
- would require different tools, methods, expertises, platforms, etc.

$\hookrightarrow$ **Daphne wants to be an infra to develop & deploy IDA pipelines**

Context

- Integrated Data Analysis (IDA) pipelines
- e.g., HPC simulation ⇝ Big Data processing ⇝ ML training
- would require different tools, methods, expertises, platforms, etc.

↪ **Daphne wants to be an infra to develop & deploy IDA pipelines**

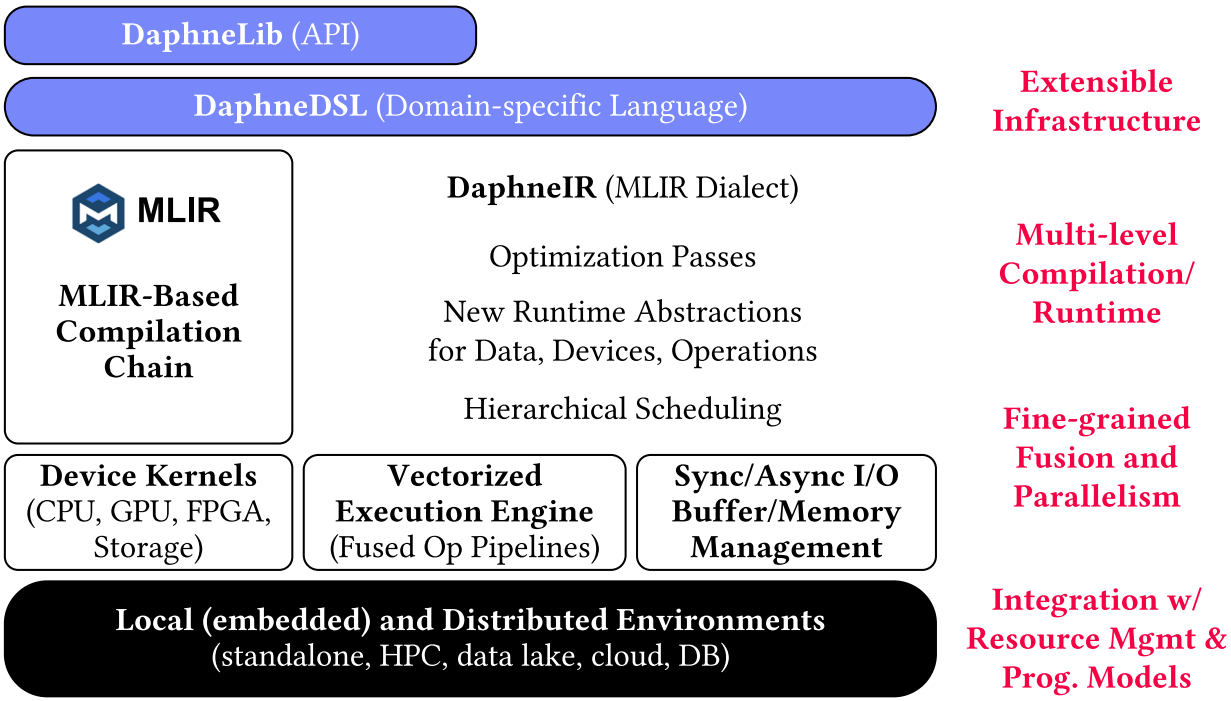

Figure: Daphne infrastructure [1]

- DaphneDSL ≃ Python, Julia, R
- reuse Python code w/ DaphneLib
- MLIR ⇝ allows to use existing kernels
- wants to be extensible
- *Scheduling opportunities*

The Scheduling Challenges (1/2)

How to efficiently schedule IDA pipelines?

The Scheduling Challenges (1/2)

How to efficiently schedule IDA pipelines?

- Execution ordering: generated by compiler, FCFS
- Execution timing: start task immediatedly, $\rightsquigarrow$ defer
- having a master thread partition and assign work to workers
  $\Longrightarrow$ waiting and idleness

The Scheduling Challenges (1/2)

How to efficiently schedule IDA pipelines?

- Execution ordering: generated by compiler, FCFS
- Execution timing: start task immediatedly, $\rightsquigarrow$ defer
- having a master thread partition and assign work to workers
  $\Longrightarrow$ waiting and idleness

  $\hookrightarrow$ **instead**: the workers partition and self-assign the work

The Scheduling Challenges (1/2)

How to efficiently schedule IDA pipelines?

- Execution ordering: generated by compiler, FCFS
- Execution timing: start task immediatedly, $\rightsquigarrow$ defer
- having a master thread partition and assign work to workers
  $\Longrightarrow$ waiting and idleness

  $\hookrightarrow$ **instead**: the workers partition and self-assign the work

Local scheduler [2, 3]

- Self-Scheduling: State-of-Practice + State-of-the-Art
- Work-queues: centralized, decentralized (per core, per CPU)
- Work-Stealing: seq, random, + same but with NUMA priority
- [5] evaluation of *DaphneSched*
- extensible!

The Scheduling Challenges (2/2)

But what about several nodes?

The Scheduling Challenges (2/2)

But what about several nodes?

- need to consider locality, cost of data transfers, etc.
- interaction between IDA pipelines
- coarser grain idleness
- even more heterogeneous

The Scheduling Challenges (2/2)

But what about several nodes?

- need to consider locality, cost of data transfers, etc.
- interaction between IDA pipelines
- coarser grain idleness
- even more heterogeneous

Global Scheduler [4]

- interaction with the batch-scheduler
- opportunity for collocation, relinquish resources, monitoring, etc.
- communications between the global and local scheds
- $\rightsquigarrow$ not done yet in Daphne?

Questions

- when decentralized work queue, workers *pull* from the "main" queue. Can the main queue *push* to the workers, and then let them steal work?
- at the cluster level, when running 2 IDA pipelines, how many instances of Daphne?
  - if 1, Daphne runtime is a wrapper around batch sched?
  - if 2, how to communicate between the instances?

Technical task

src/runtime/local/vectorized/LoadPartitioning.h

```
+ cstChunk = 0;
+ if (schedulingMethod == CST){
+     if (const char* env_cst_size =\
            std::getenv("DAPHNE_CST_TASK_SIZE")){
+         cstChunk = std::stoi(env_cst_size);
+     }
+     else{
+         schedulingMethod = STATIC;
+     }
+ }
// ...
+ case CST:{
+     chunkSize=cstChunk;
+     break;
+ }
```

Demo

Demo!

(see a GIF of the demo here: `https://github.com/GuilloteauQ/daphne-toy-sched#the-toy-scheduler`)

A word on reproducibility

- in Grenoble (DATAMOVE team), we are very interested in reproductibility of experiments
- in CS this notably goes through reproducibility of software (and others)
- was annoying to build (MLIR & ANTLR)
- use tools such as Nix/Guix (see issue #580)
- packaged (rougthly) Daphne in Nix: see daphne-nix
- created a binary cache (cachix): daphne-nix.cachix
- allows for reproducible, sharable, precisely customizable softwares
- (see those slides for more)

References I

[1] P. Damme et al., "Daphne: an open and extensible system infrastructure for integrated data analysis pipelines," in Conference on innovative data systems research (2022).

[2] DAPHNE, "D5.2 Prototype of Pipelines and Task Scheduling Mechanisms," 2022.

[3] DAPHNE, "D5.2 Scheduler Design for Pipelines and Tasks," 2022.

[4] A. Eleliemy et al., "A resourceful coordination approach for multilevel scheduling," arXiv preprint arXiv:2103.05809 (2021).

[5] A. Eleliemy et al., "Daphnesched: a scheduler for integrated data analysis pipelines," arXiv preprint arXiv:2308.01607 (2023).