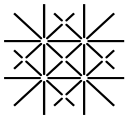
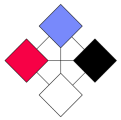


University of Basel

Department of Mathematics and Computer Science



DAPHNE

DMI HPC

Multilevel Scheduling in Action for Data Analysis Pipelines with DAPHNE

Florina M. Ciorba
Department of Mathematics and Computer Science
University of Basel
ITU Resource-Aware Data Science Day, February 13, 2023

Joint work with Ahmed Eleliemy

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement number 957407.

Presentation Overview

- Multilevel Scheduling
- DAPHNE
- Results
- Next Steps

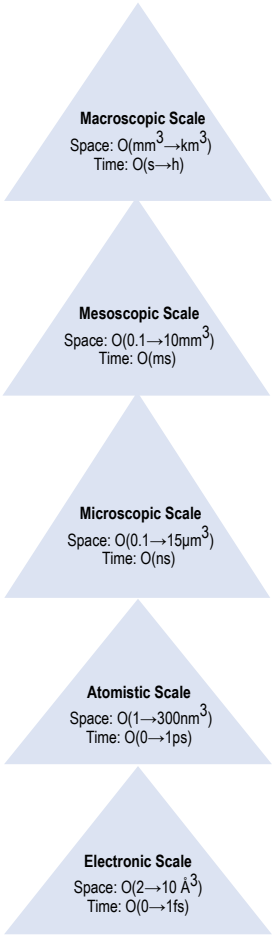
Presentation Overview

- **Multilevel Scheduling**
- DAPHNE
- Results
- Next Steps

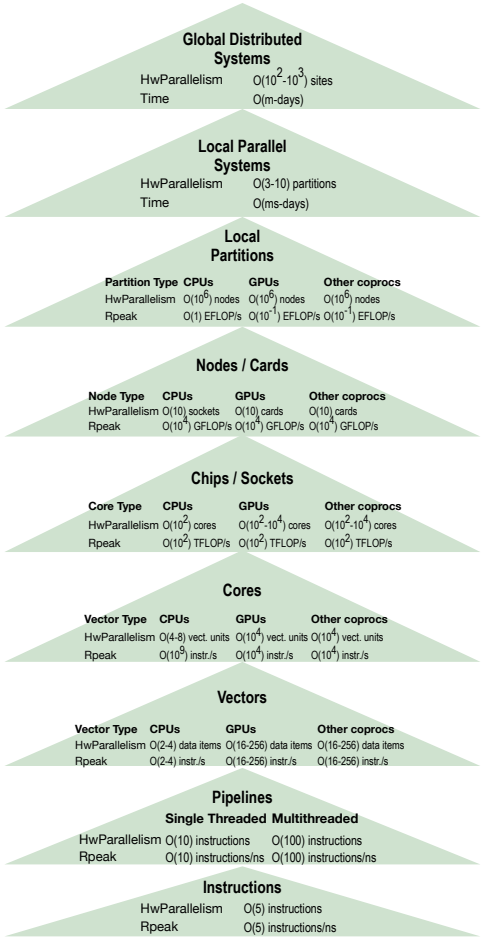
Multilevel Parallelism

DAPHNE

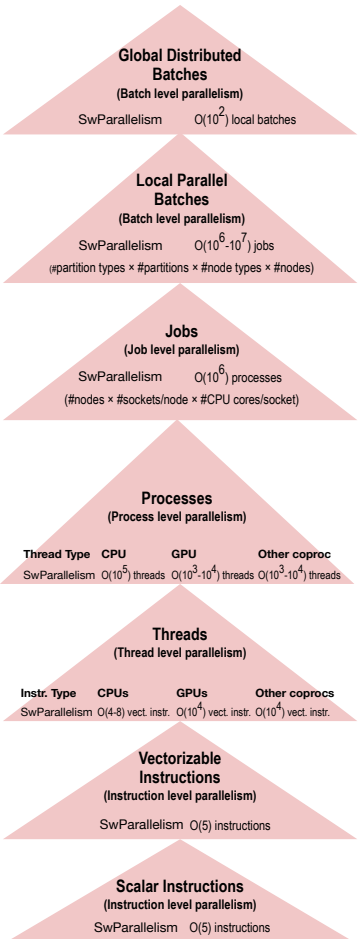
Multiscale Modeling



Multilevel Hardware Parallelism



Multilevel Software Parallelism



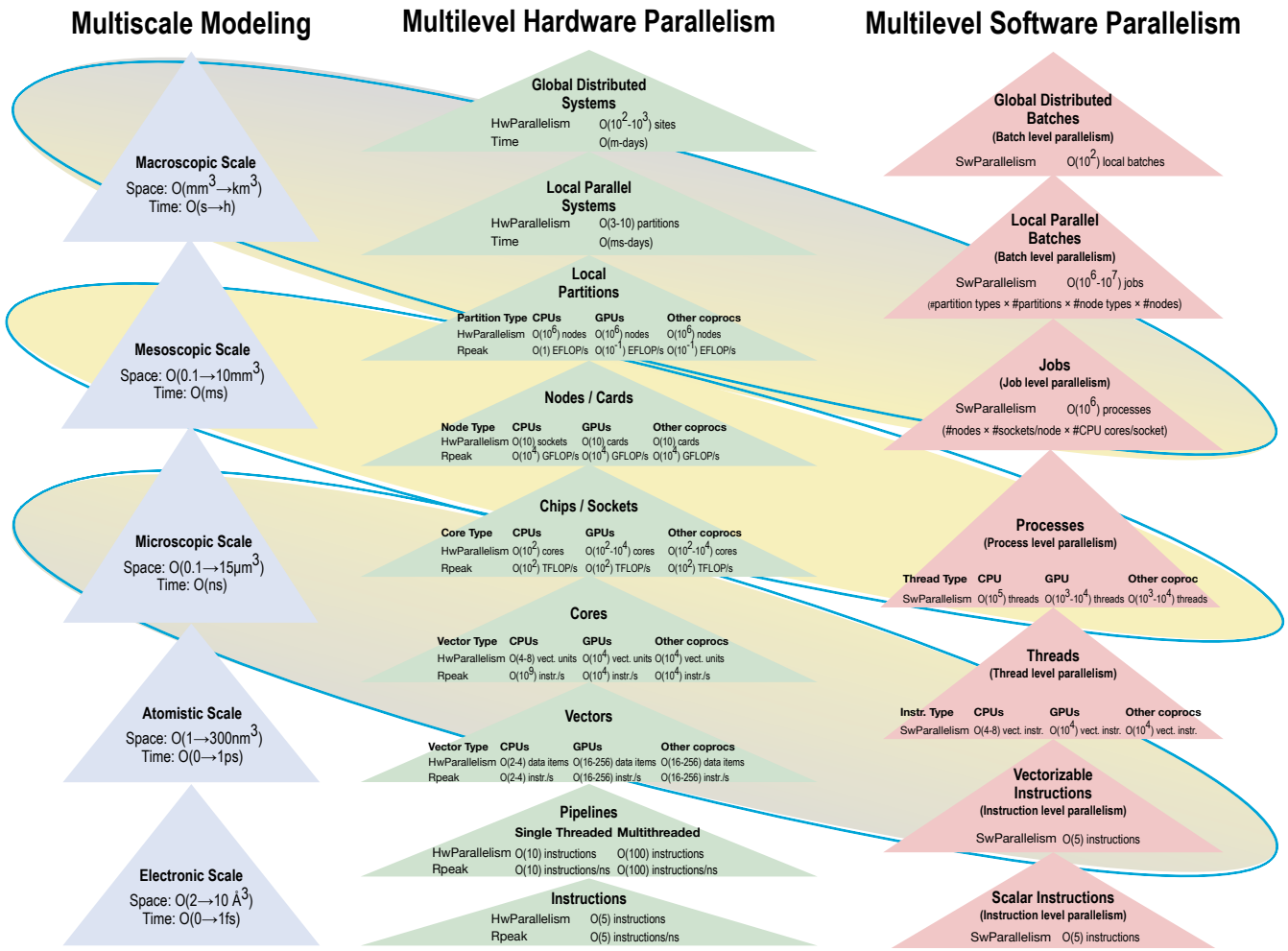
Performance requires complex interplay of

Massive
Multilevel
Heterogeneous
Hardware and Software
Parallelism

How to use it all to solve the world's most challenging problems?

Multilevel Parallelism

DAPHNE



Multilevel Scheduling

Macroscopic scale

⬆

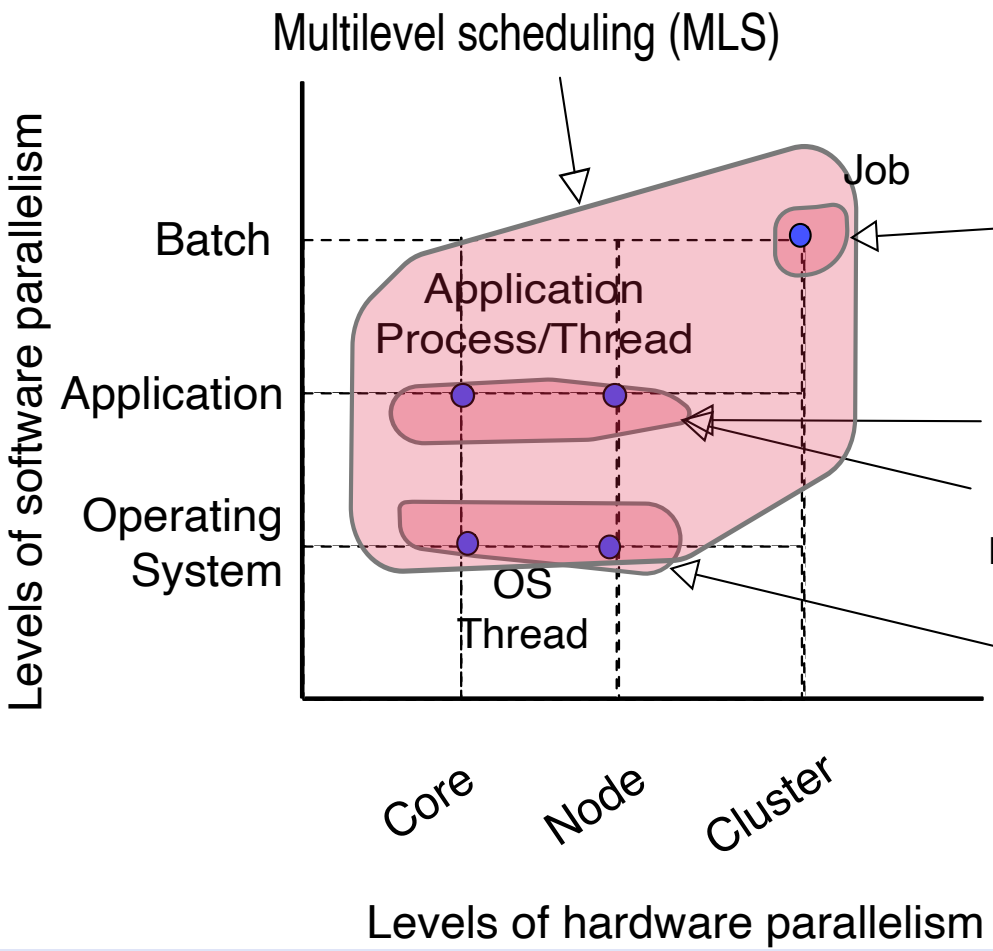
Mesoscopic scale

⬇

Microscopic scale

Multilevel Scheduling

DAPHNE



List Scheduling Algorithms: FCFS, SJF, LSF, EDF, etc.
Schedulers: SLURM, Mesos, Torque, Borg, Omega, etc.

Batch level scheduling

Process level scheduling

Thread level scheduling

Application level scheduling

OS level scheduling

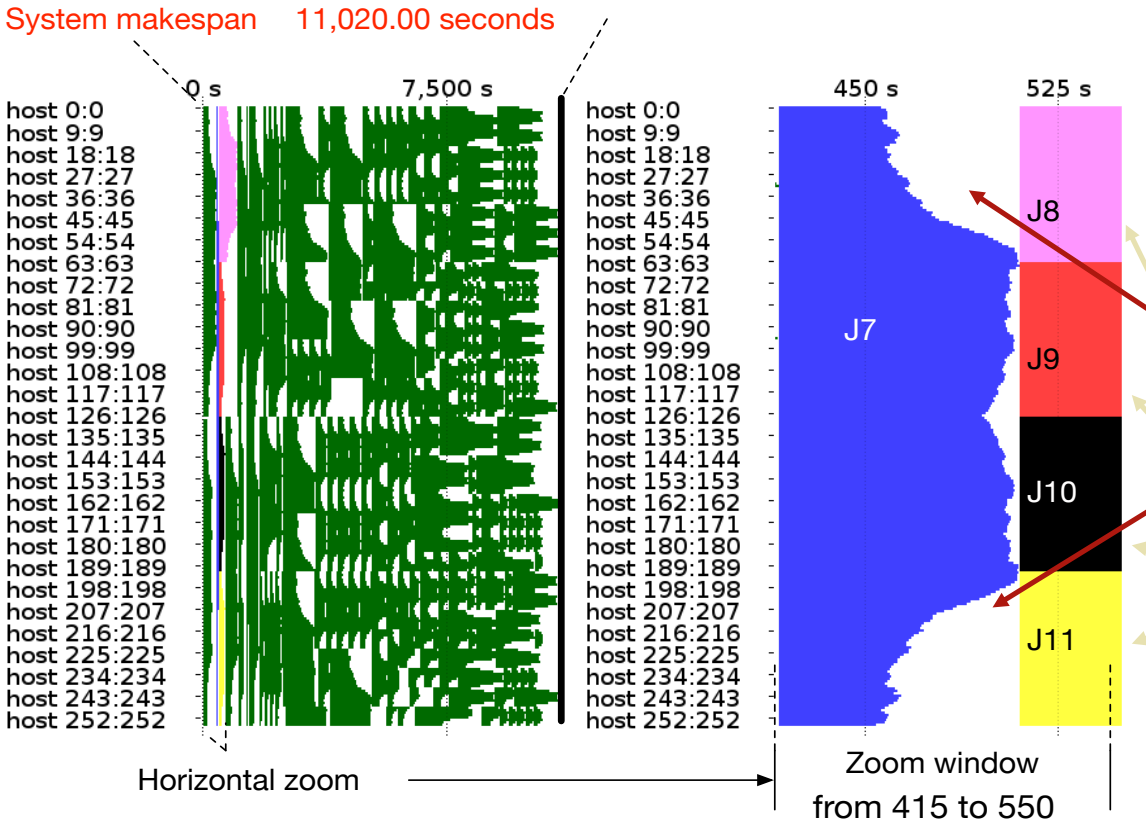
Multilevel Scheduling *exploits massive, multilevel, heterogeneous parallelism.*

Requires Coordination *to expose and exchange information across levels:*
- *idle resources*
- *remaining work*

Self-Scheduling Algorithms: SS, FSC, FISS, VISS, GSS, TSS, FAC, WF, AWF, BOLD, etc.
Libraries: LB4OMP, LB4MPI
Algorithm selection: Expert, RL-based

Interference Between Batch and Application Level Scheduling

DAPHNE



Effective System Performance (ESP): Mandelbrot
Jobs: 230
Requested hosts by each job: 2-256
Batch scheduling: FCFS+BF (SLURM simulation)
Application scheduling: STATIC (SimGrid simulation)

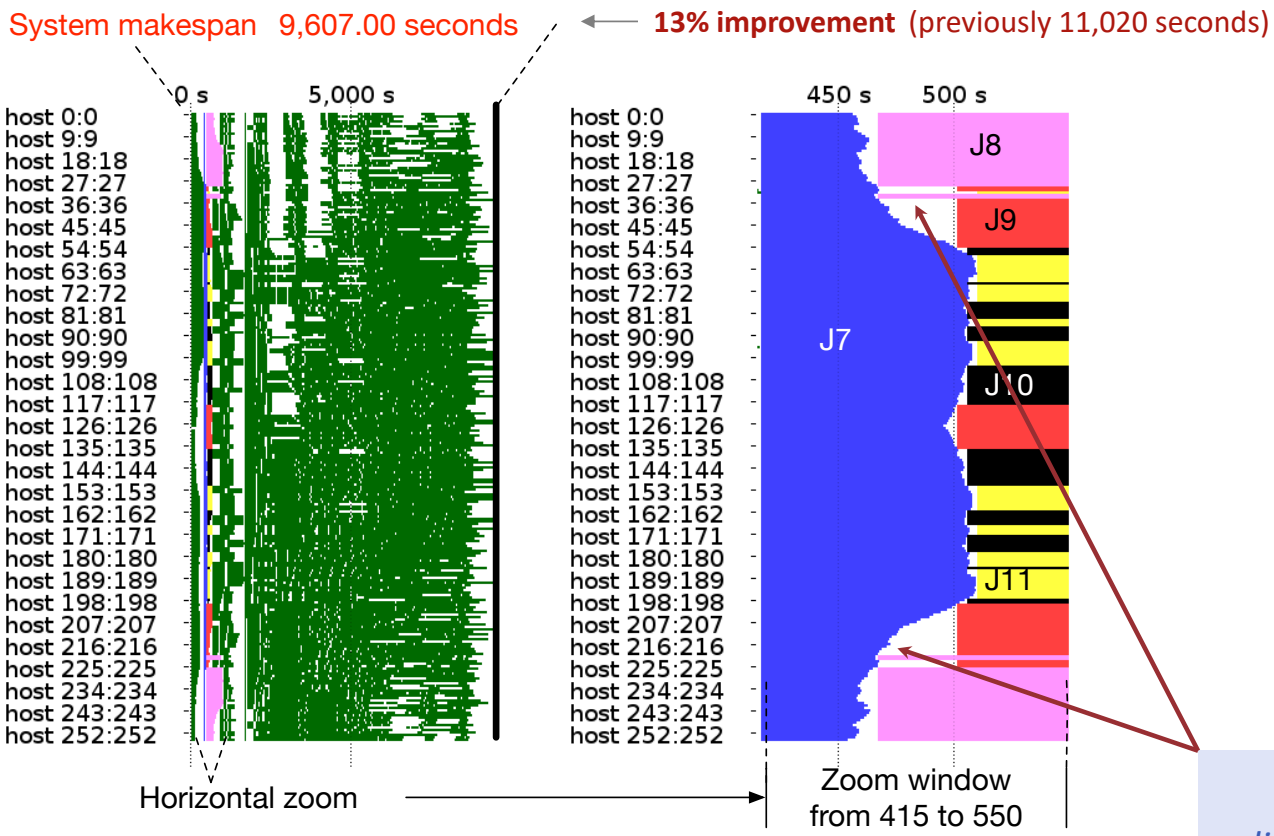
Application load imbalance causes system **load imbalance**

System load imbalance: ready jobs **wait** while free resources exist

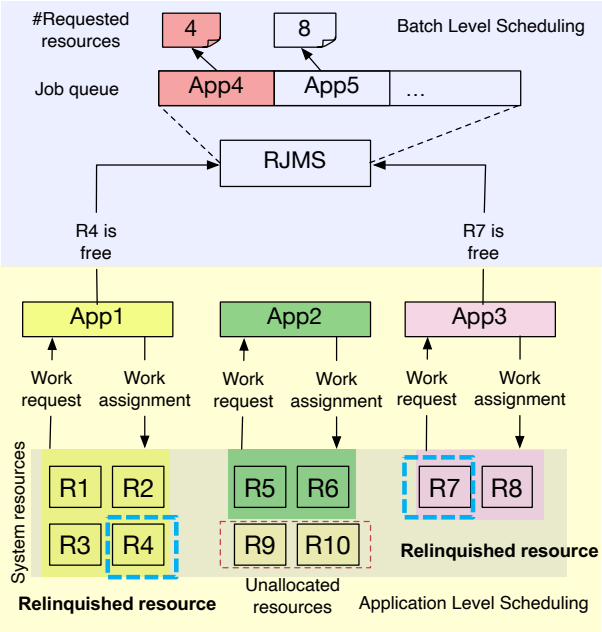
Need to reduce idleness
Can applications relinquish the resources that are no longer needed (no more work)?

Multilevel Scheduling (MLS) to Reduce Idleness

DAPHNE



Resourceful Coordination Approach (RCA)

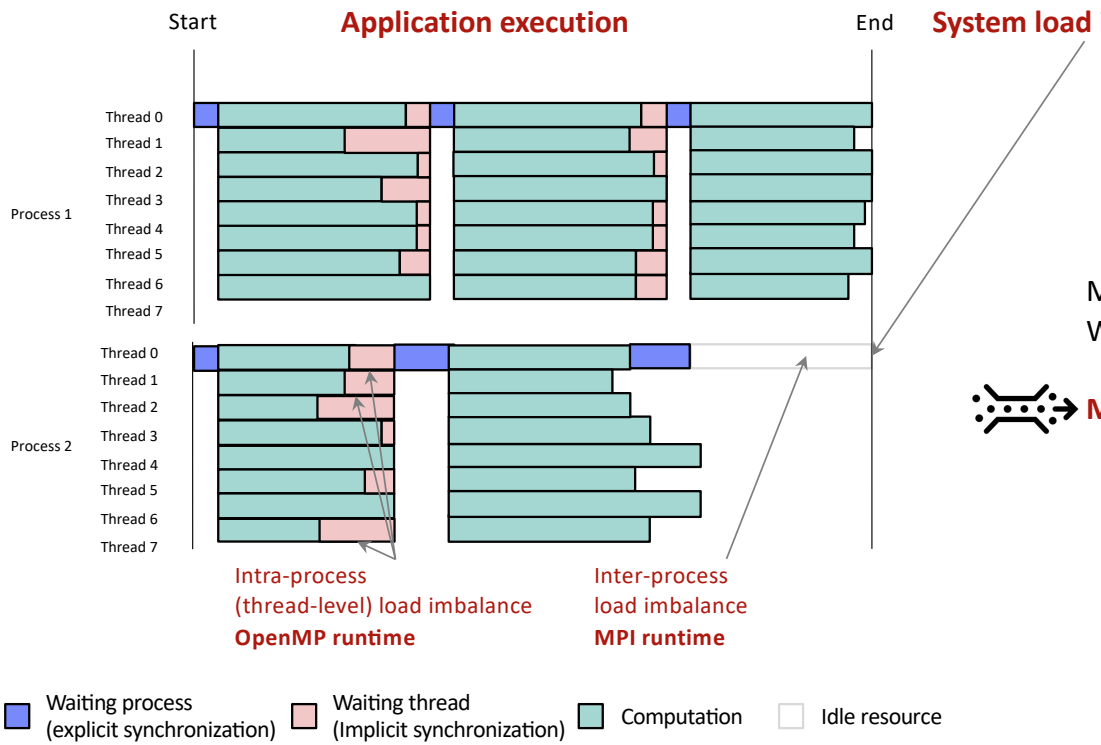


Reduced idleness *by application relinquishing no longer needed resources.*

A. Eleliemy and F. M. Ciorba. "A Resourceful Coordination Approach for Multilevel Scheduling". International Conference on High Performance Computing & Simulation (HPCS 2020)

Interference Between Application Process and Thread Level Scheduling

DAPHNE



Master (process/thread) **partitions** and **assigns** work upon workers' request.
Workers only request, **wait** for, and **execute** received work.

Master is a bottleneck and induces waiting and idleness.

Need to reduce waiting

Can work partitioning be separate from assignment?

Bottleneck icon by Stephen Plaster from thenounproject.com

Multilevel Scheduling (MLS) to Reduce Waiting

DAPHNE

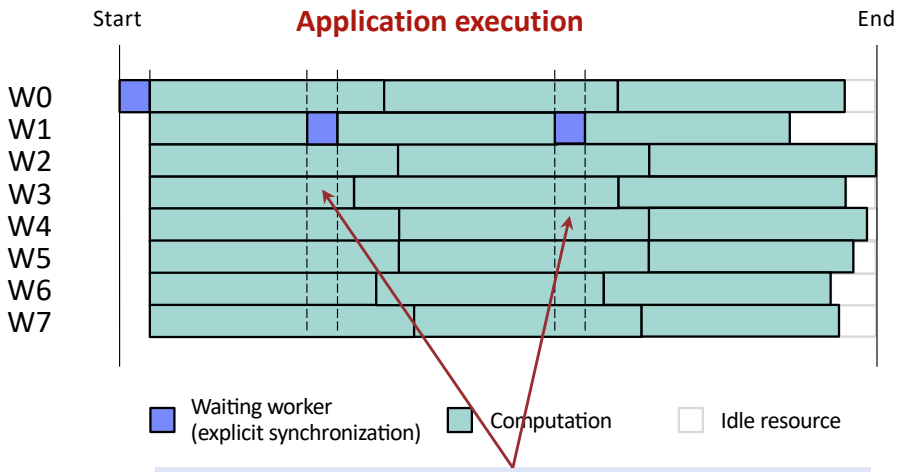
- **Workers help out master by separating work partitioning and work assignment**
  - Workers (threads or processes) **self-partition work** (K_i), then **self-assign it** (i^{th} chunk)
  - Implementation: atomic operations (threads) or RMA get-put operations (processes)
- **Master** ensures atomic updates on "i" (threads) or maintains RMA window (processes)

Centralized chunk calculation approach (CCA)

$$K_i^{FISS} = K_{i-1}^{FISS} + constant$$ ← **Recursive (difficult to parallelize)**

Distributed chunk calculation approach (DCA)

$$K_i'^{FISS} = K_0^{FISS} + i * constant$$ ← **Not recursive (easy to parallelize)**



***Reduced waiting** by eliminating master and establishing worker cooperation.*

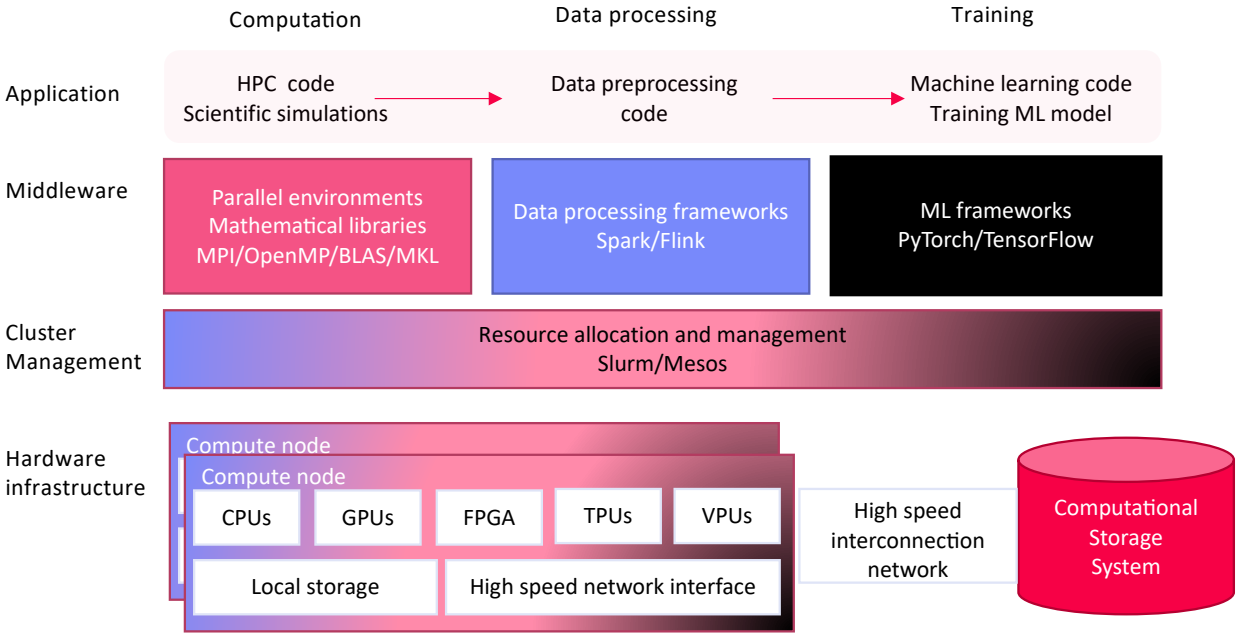
A. Eleliemy and F. M. Ciorba. "A Distributed Chunk Calculation Approach for Self-scheduling of Parallel Applications on Distributed-memory Systems". Int. J. of Computational Science, 2021.

Presentation Overview

- Multilevel Scheduling
- **DAPHNE**
- Results
- Next Steps

Data Analysis Pipelines

DAPHNE



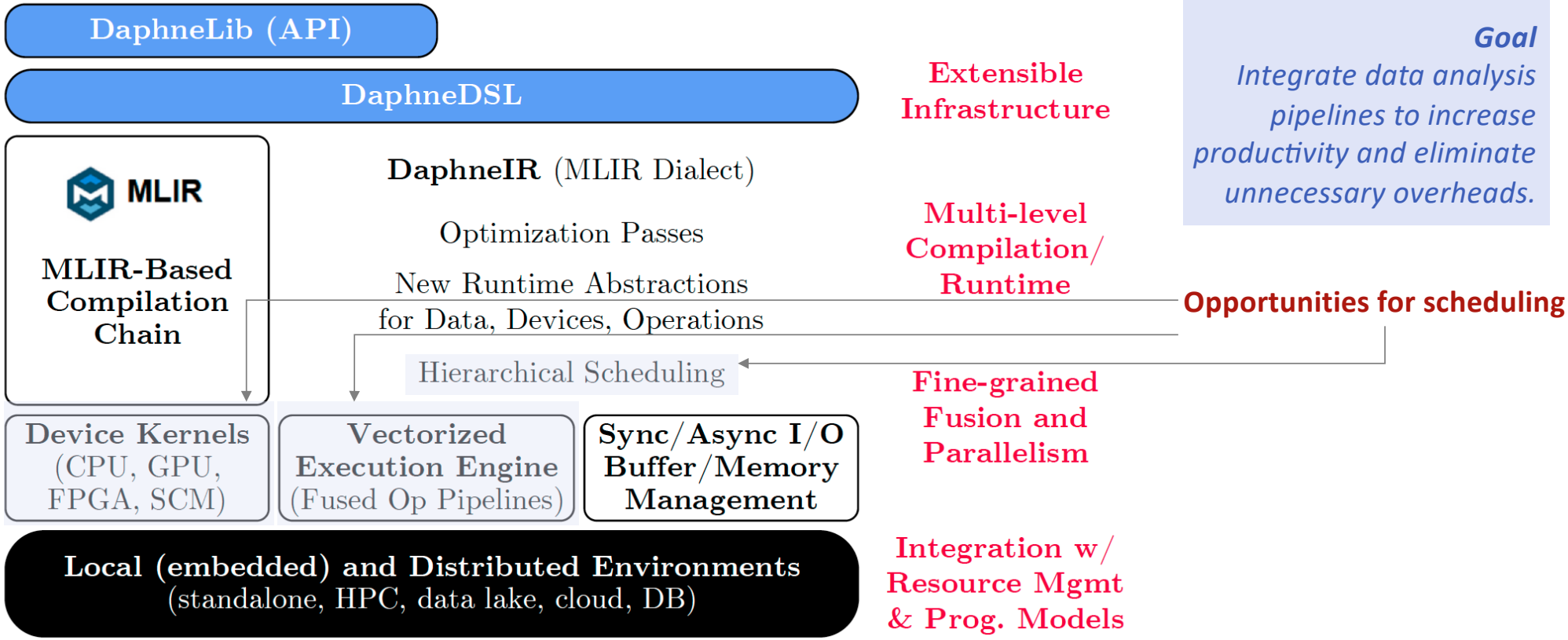
How to efficiently schedule Integrated Data Analysis Pipelines?
Aka how to exploit massive, multilevel, heterogeneous parallelism?

- **Pipelines**: data management, query processing, high performance computing, complex simulations, training and scoring for multiple machine learning models
- **Integration**: increasingly common, sharing compilation, runtime techniques, and converging cluster hardware

N. Ihde et al. "A Survey of Big Data, HPC and Machine Learning Benchmarks". TPCTC 2021, Copenhagen, Denmark, August 2021. Open Access here.

DAPHNE: An Open and Extensible System Infrastructure for Integrated Data Analysis Pipelines

DAPHNE



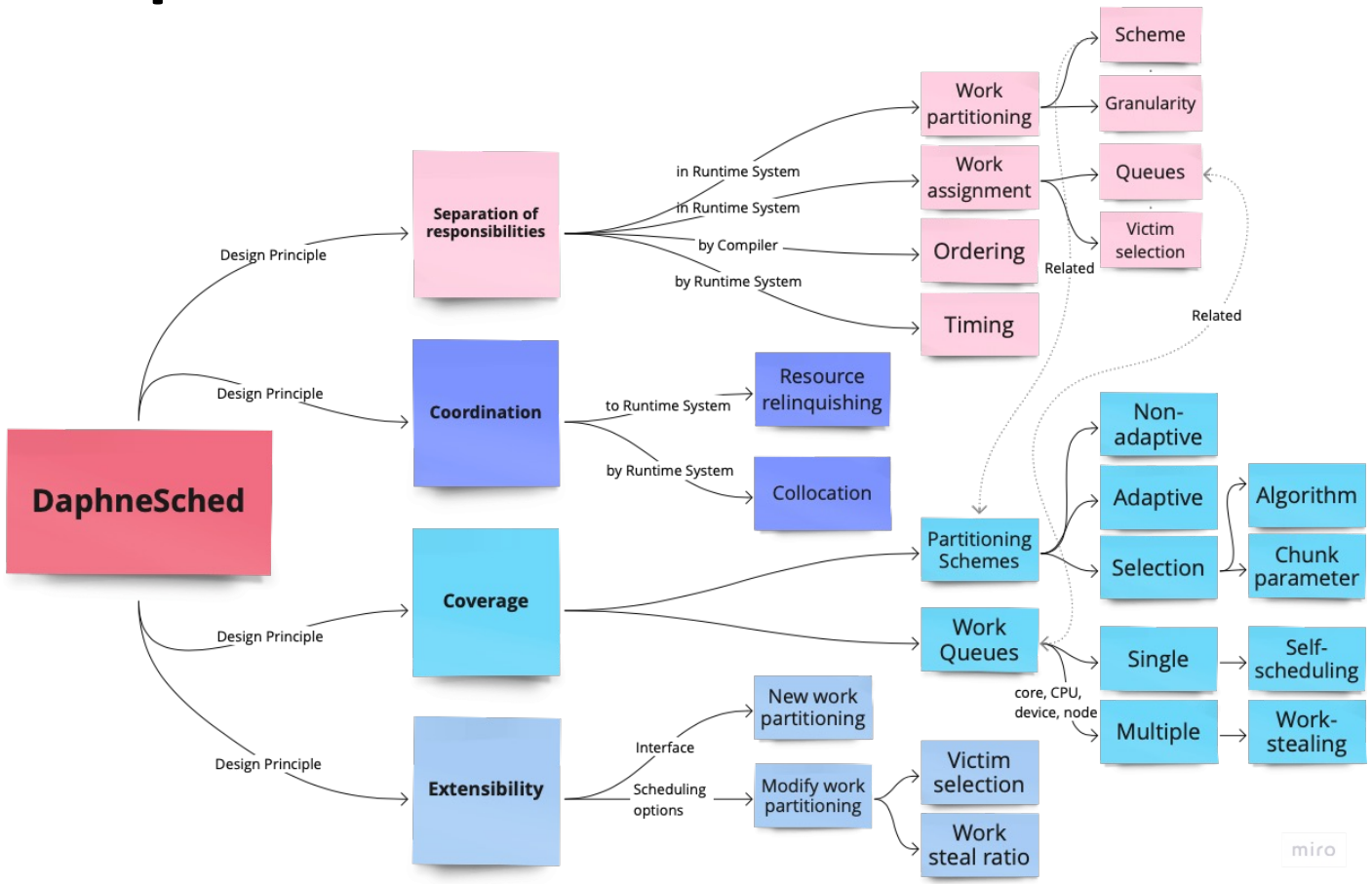
Goal
Integrate data analysis pipelines to increase productivity and eliminate unnecessary overheads.

P. Damme et al. DAPHNE: An Open and Extensible System Infrastructure for Integrated Data Analysis Pipelines. CIDR '22. Open Access here.

https://github.com/daphne-eu/daphne

DaphneSched

DAPHNE



Four Design Principles

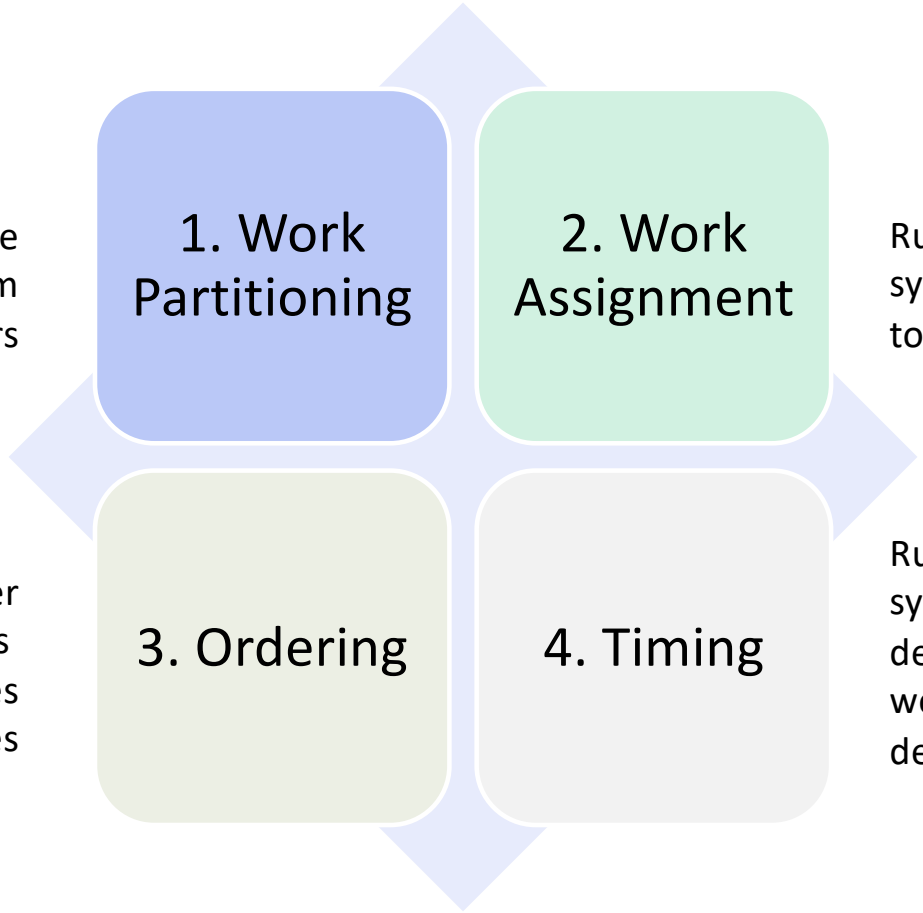
First two incubate multilevel scheduling of threads, processes, pipelines across cores, sockets, devices, nodes

Last two driven by the DAPHNE philosophy

A. Eleliemy and F. M. Ciorba. "DaphneSched: Scheduler for Integrated Data Analysis Pipelines". *In preparation*, 2023.

Separation of Responsibilities (à la MLS)

DAPHNE



Runtime system for workers

Runtime system to workers

Compiler identifies and organizes dependencies

Runtime system deploys work onto devices

Data parallelism
Same operator, multiple data

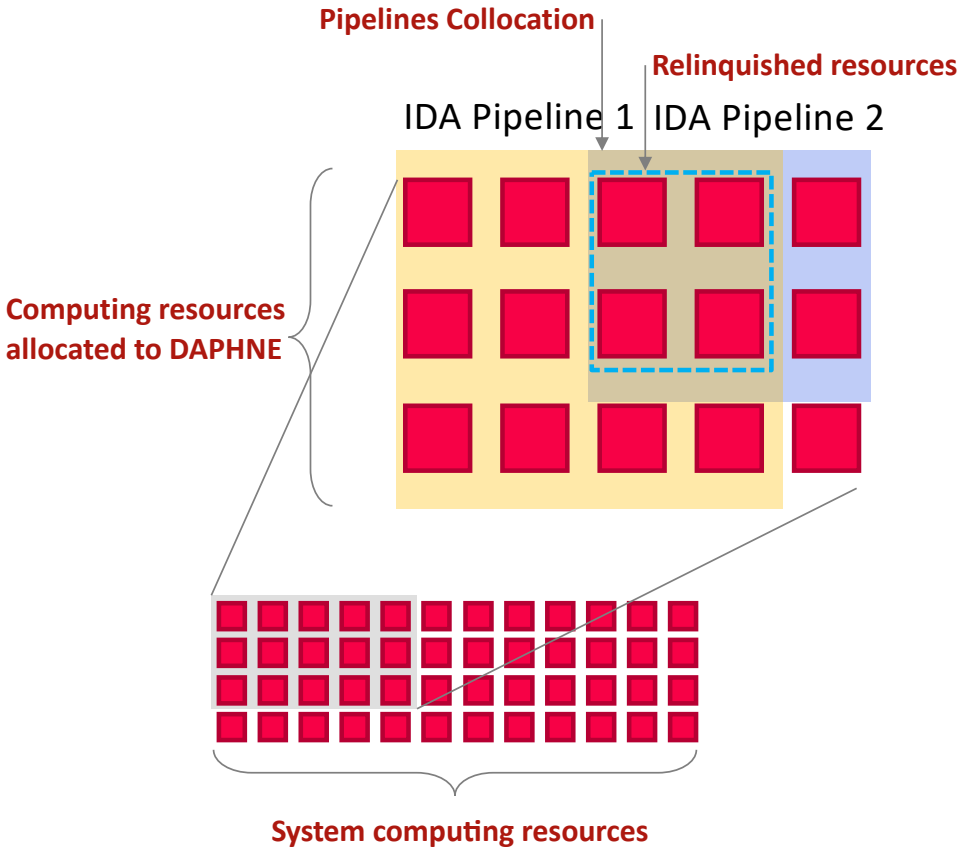
Functional parallelism
Different operators, multiple data

Reduce waiting
Compiler and runtime system have separate responsibilities

task = operators on data (smallest work unit)

Coordination (à la MLS)

DAPHNE



Reduce idleness via

Resource relinquishing

DaphneSched relinquishes resources no longer needed by IDA pipeline 1 to DAPHNE RT

followed by

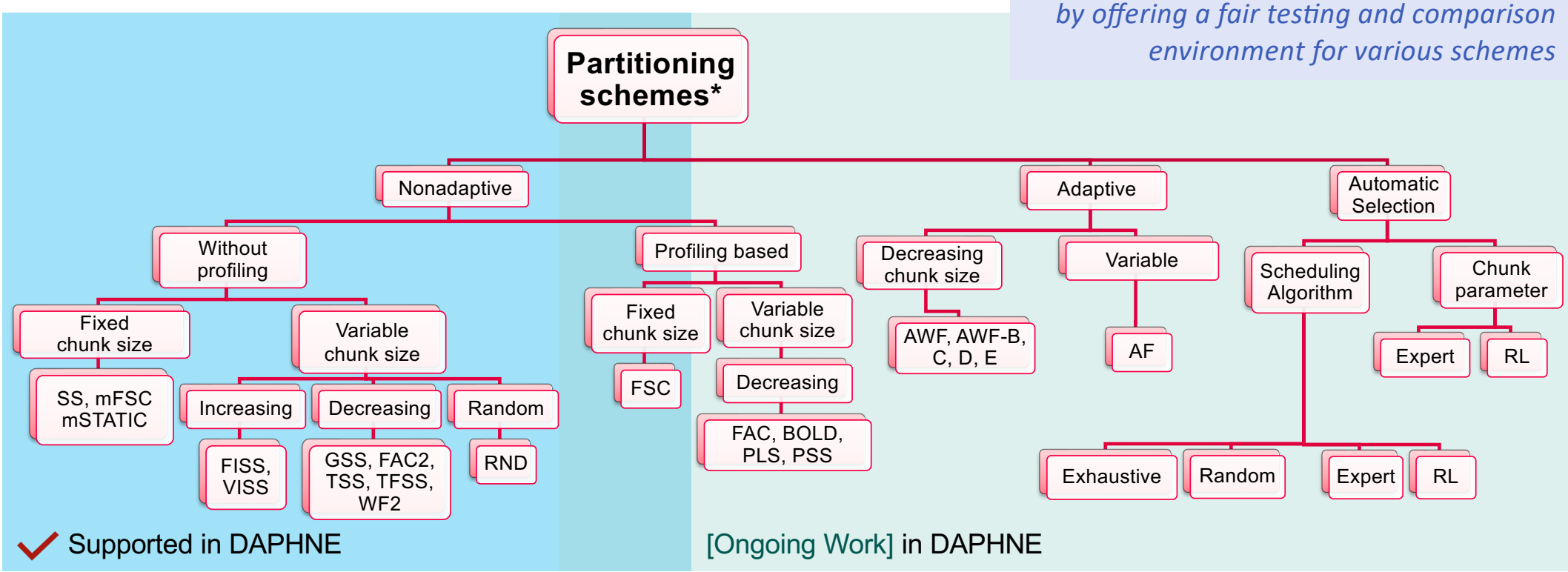
Collocation

DAPHNE RT collocates IDA pipeline 2 on resources just relinquished by DaphneSched from IDA Pipeline 1

Wide Range Coverage (à la DAPHNE)

DAPHNE

Benefits Scheduling Research
by offering a fair testing and comparison environment for various schemes

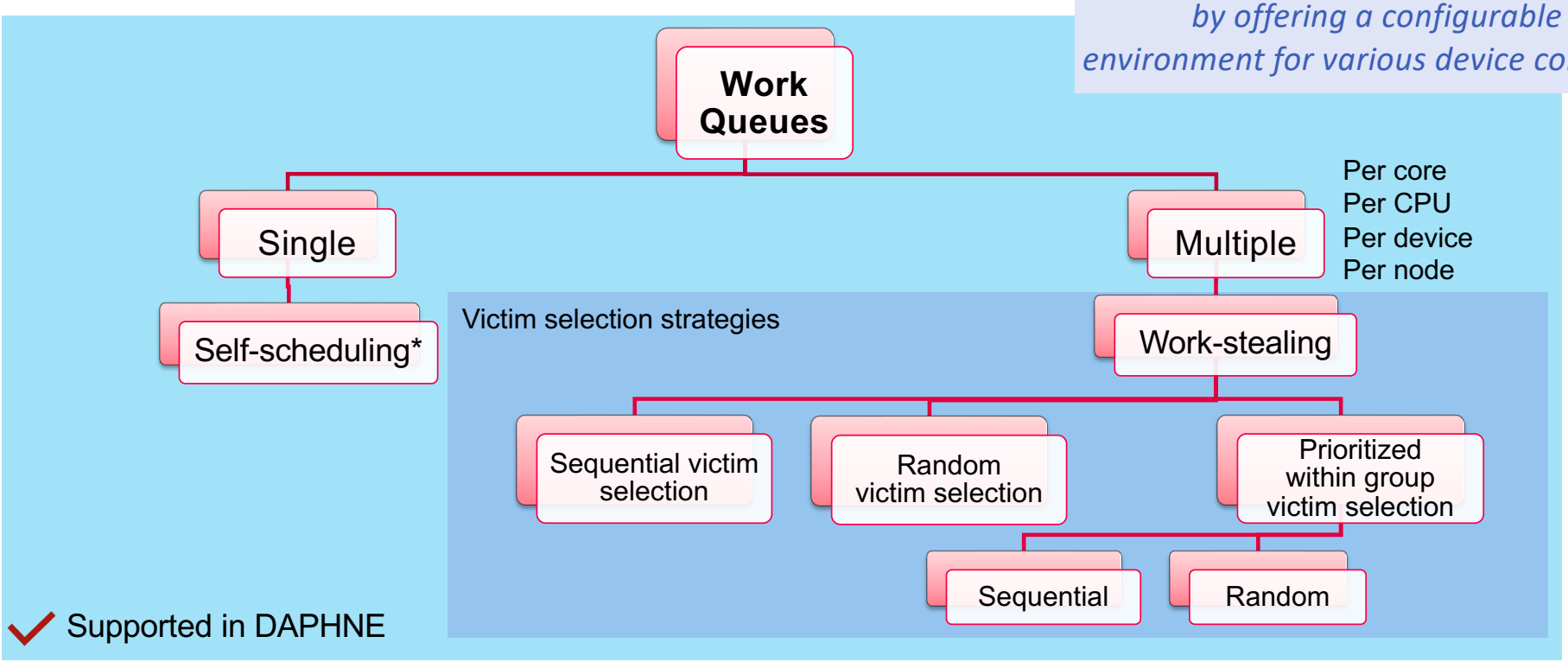


* Work partitioning uses the chunk calculation formulae of the various Dynamic Loop Self-Scheduling schemes

Wide Range Coverage (à la DAPHNE)

DAPHNE

Benefits Adaptive Resource Management
by offering a configurable deployment environment for various device configurations



* Self-scheduling as a principle for work self-assignment.
(Not to be confused with the self-scheduling method that partitions and self-assigns work).

Easily Extensible (à la DAPHNE)

DAPHNE

- DaphneSched open and easily extensible via
  - New **work partitioning** schemes, e.g., **MYTECH**
    - enum SelfSchedulingScheme
    - uint64_t getNextChunk()
    - opt<SelfSchedulingScheme> taskPartitioningScheme
  - Customize existing **work assignment** schemes (mainly work-stealing)
    - Victim selection: --SEQ, --SEQPRI, --RANDOM, --RANDOMPRI
    - Work stealing ratio: --SS, --GSS, --TSS, --FAC2, --TFSS, --FISS, --VISS, --PLS, --MSTATIC, --MFSC, --PSS, --**MYTECH**

New work partitioning
via changes to three functions

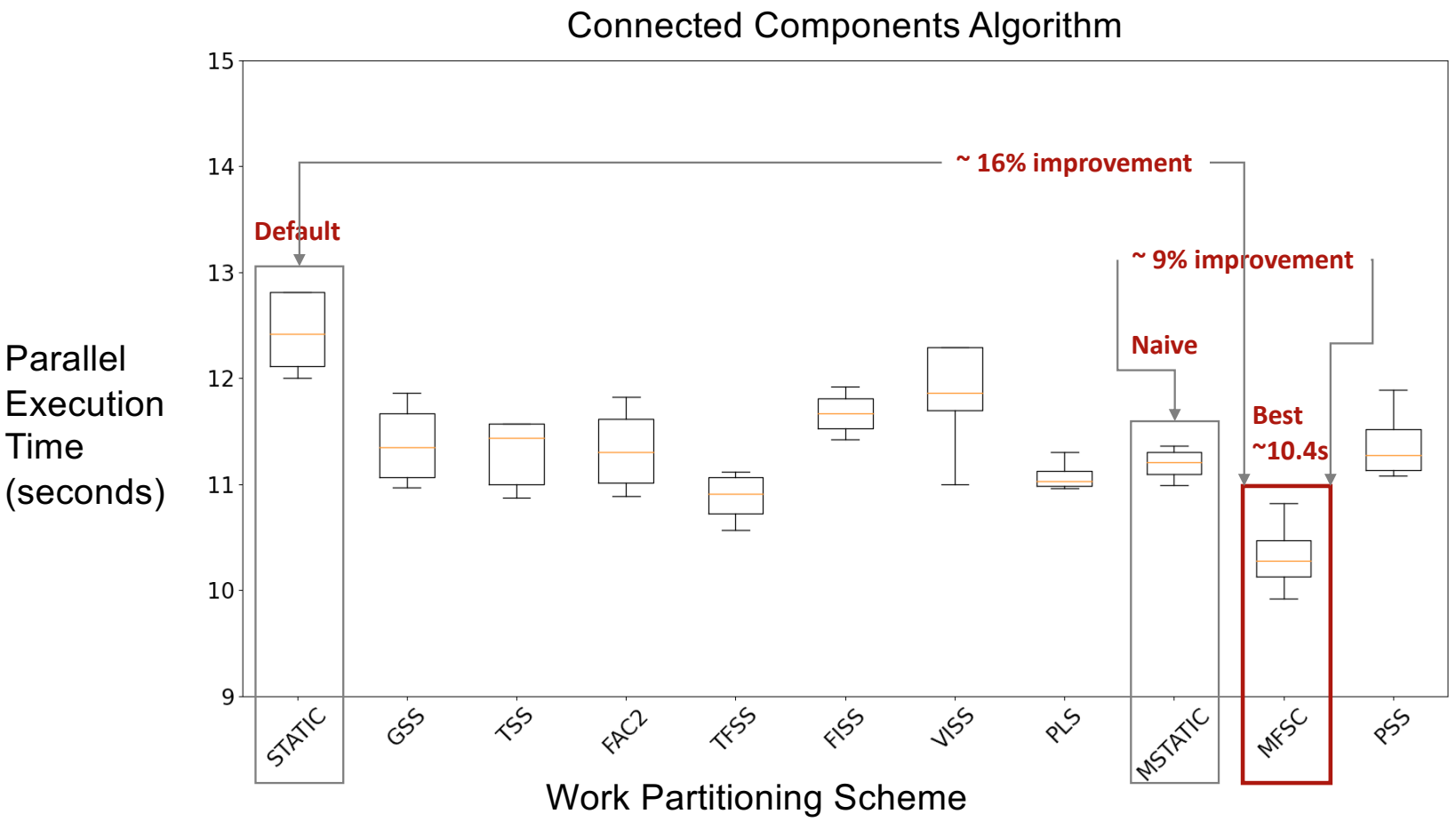
Custom work assignment
via two knobs

https://github.com/daphne-eu/daphne/blob/main/doc/SchedulingOptions.md

Presentation Overview

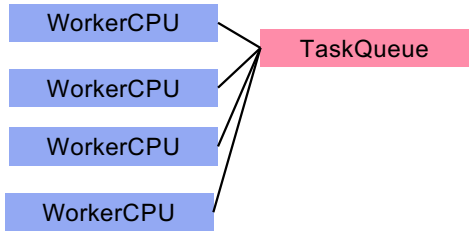
- Multilevel Scheduling
- DAPHNE
- **Results**
- Next Steps

DaphneSched of a Data Analysis Pipeline



DAPHNE

Centralized Work Queue (Self-scheduling)



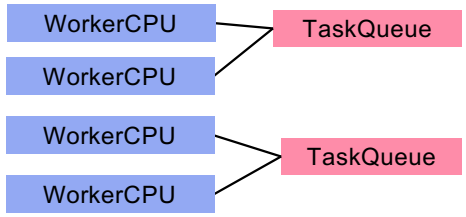
Problem size: 403'394 * 50 vertices
Processor type: Intel Xeon E5-2640, 64 GB RAM, 2.4 GHz CPU
Hardware parallelism: 2 CPUs x 10 cores
Software parallelism: 20 threads (no hyperthreading)

DaphneSched of a Data Analysis Pipeline

DAPHNE

Connected Components Algorithm

Multiple Work Queues
(per CPU socket)



23% improvement

Work Assignment (work stealing with victim selection)

	STATIC	GSS	TSS	FAC2	TFSS	FISS	VISS	PLS	MSTATIC	MFSC	PSS
SEQ	12.34	11.12	10.96	11.27	11.03	11.25	11.75	10.23	11.59	10.42	11.3
SEQPRI	11.52	12.16	11.47	11.02	10.52	11.61	11.44	10.95	11.7	10.36	11.48
RANDOM	11.89	10.21	10.3	10.09	10.61	10.37	11.12	10.95	10.05	9.27	10.08
RANDOMPRI	11.42	11.26	11.51	10.26	10.67	10.37	11.81	10.57	10.12	11.11	11.58

Parallel Execution Time (seconds)

9.3 — 16.9

Work Partitioning Scheme

Best ~9.3s
vs. ~10.4s
(same scheme)

Problem size: 403'394 * 50 vertices
Processor type: Intel Xeon E5-2640, 64 GB RAM, 2.4 GHz CPU
Hardware parallelism: 2 CPUs x 10 cores
Software parallelism: 20 threads (no hyperthreading)

Presentation Overview

- Multilevel Scheduling
- DAPHNE
- Results
- **Next Steps**

DAPHNE

Next Steps

Distributed DAPHNE Runtime: MLS incubation (in progress)

Research on **new scheduling approaches** (data-vs. task-driven)

Deploy hierarchical scheduling (local and distributed runtimes) with **automated algorithm selection** per level

Research on **multi-tenancy** for multiple IDA pipelines across heterogeneous devices

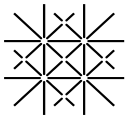
Takeaways

Multilevel Scheduling is key for exploiting massive, multilevel, heterogeneous parallelism

Separation of responsibilities and Coordination reduce synchronization overhead and resource idleness

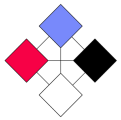
DaphneSched incubates MLS and offers wide-range **Coverage** and easy **Extensibility**

DAPHNE integrates data analysis pipelines, increasing their performance and user productivity



University of Basel

Department of Mathematics and Computer Science



DAPHNE

DMI HPC

Multilevel Scheduling in Action for Data Analysis Pipelines with DAPHNE

Florina M. Ciorba
Department of Mathematics and Computer Science
University of Basel
ITU Resource-Aware Data Science Day, February 13, 2023

Joint work with Ahmed Eleliemy

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement number 957407.