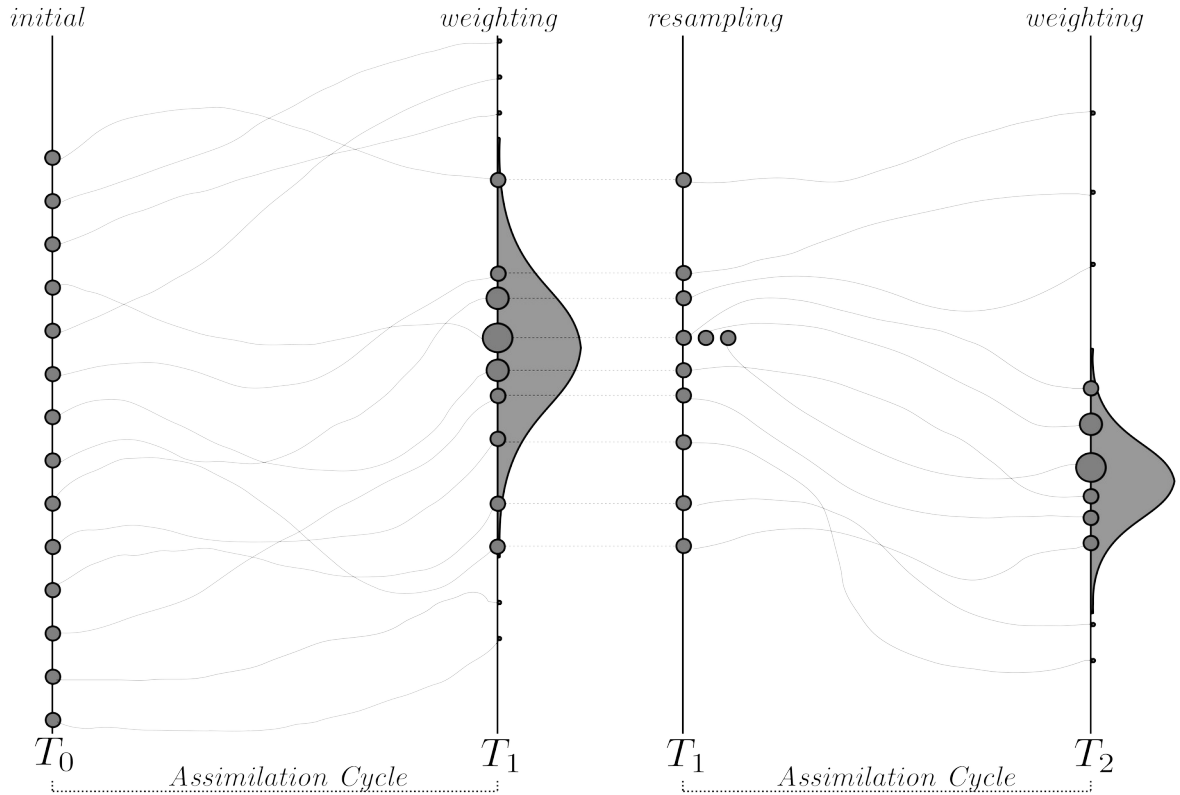
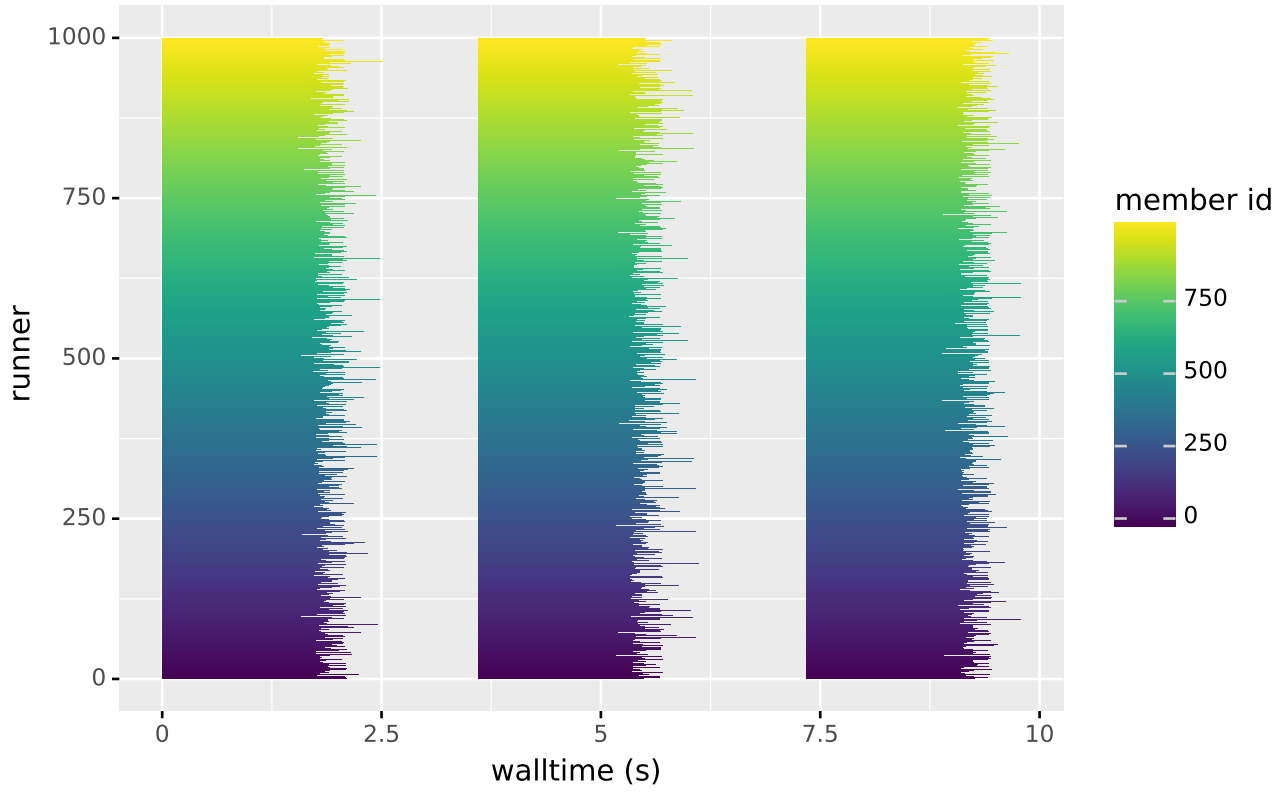
Elastic Large Scale Ensemble Data Assimilation with Particle Filters for Continental Weather Simulation

Sebastian Friedemann*, Kai Keller†, Yen-Sen Lu‡, Bruno Raffin*, Leonardo Bautista-Gomez†
*Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, 38000 Grenoble, France
{sebastian.friedemann, bruno.raffin}@inria.fr
†Barcelona Supercomputing Center, 08034 Barcelona, Spain
{kai.keller, leonardo.bautista}@bsc.es
‡Forschungzentrum Juelich, 52428 Juelich, Germany
ye.lu@fz-juelich.de

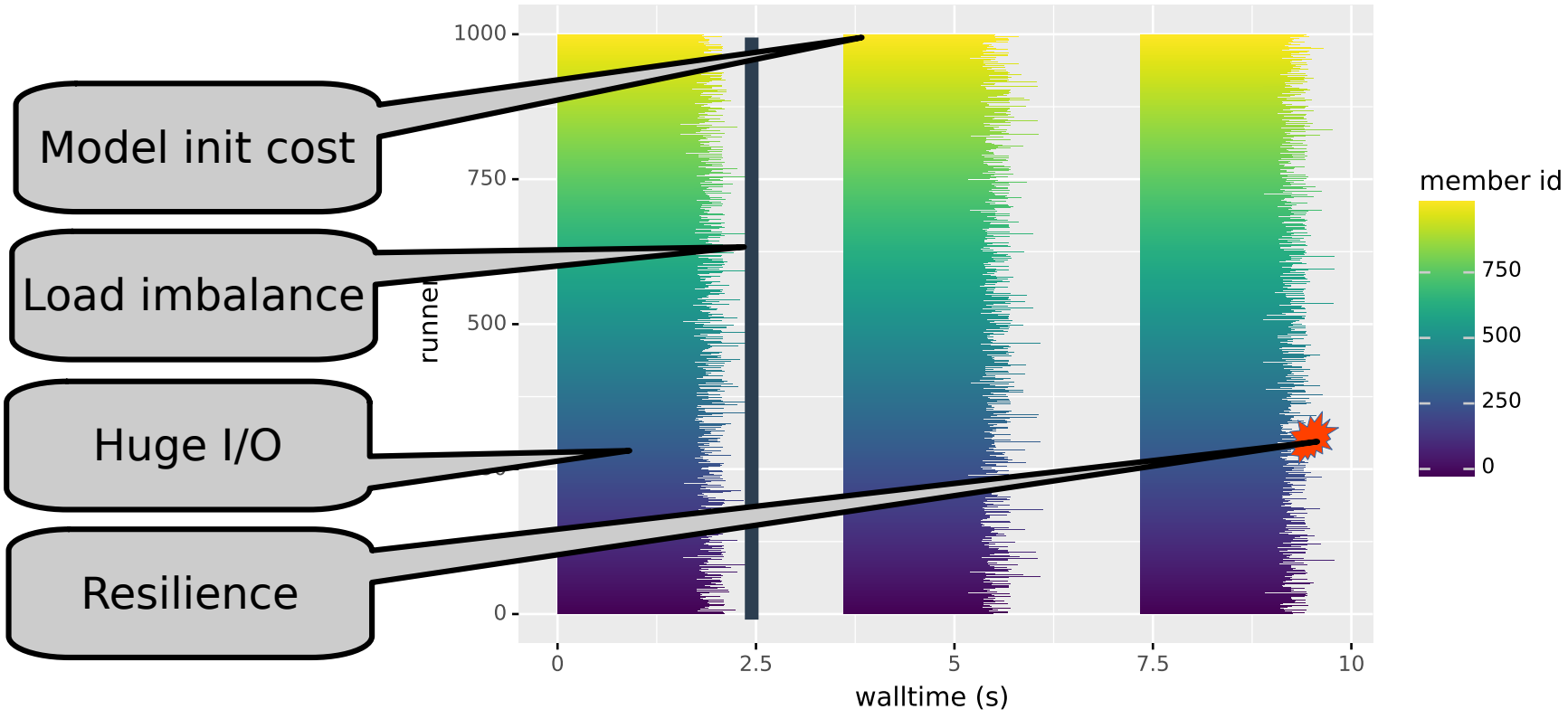
Particle filters



Particle Filters on HPC



Challenges


Model init cost
Load imbalance
Huge I/O
Resilience
1000
750
500
0
runner
0
2.5
5
7.5
10
walltime (s)
member id
750
500
250
0

Our solution: Particle Virtualization

- **Instrument existing model code to allow for yielding to different particles**
  - Exchange model state with another particle
  - One model instance can calculate multiple particles per time step

    → **Runner**

    1 Runner = 1 resource allocation = calculates 1 particle at a time
  - **BUT: can calculate multiple particles for each timestep**

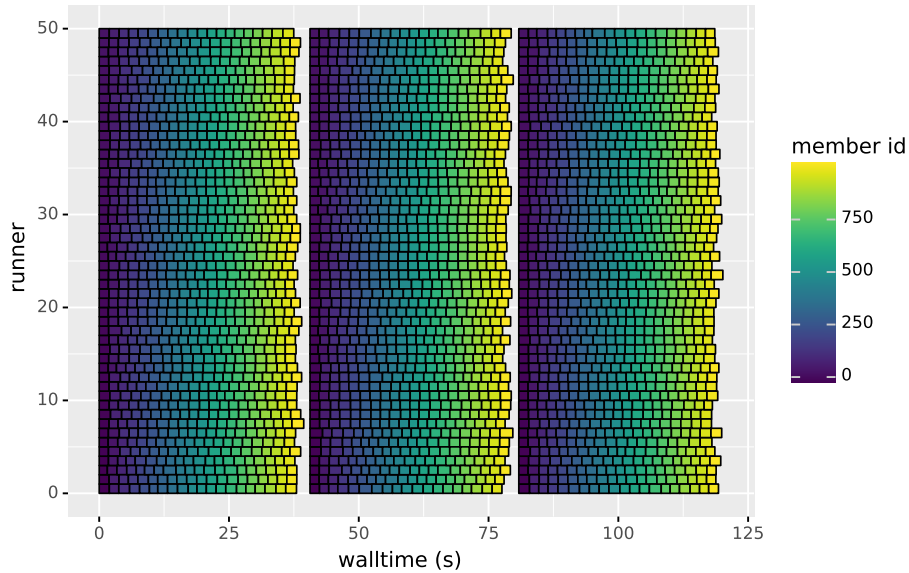
Load balancing, avoiding initialization costs

- **Particle propagation tasks are scheduled one by one to each runner**

→ Load Balanced

- **Runners start up only once**

→ Initialization time amortizes



Handling a huge amount of particle state data

- **Helper core per runner node handling**
  - Local cache (RAM disk / NVRAM)
  - Shared state storage (PFS)

  } Distributed Cache

- **Data aware scheduling of particles to runners**

  In Particle Filters: one particle of previous assimilation cycle is ‘parent’ for multiple particles

  → completely avoid > 60% of PFS reads

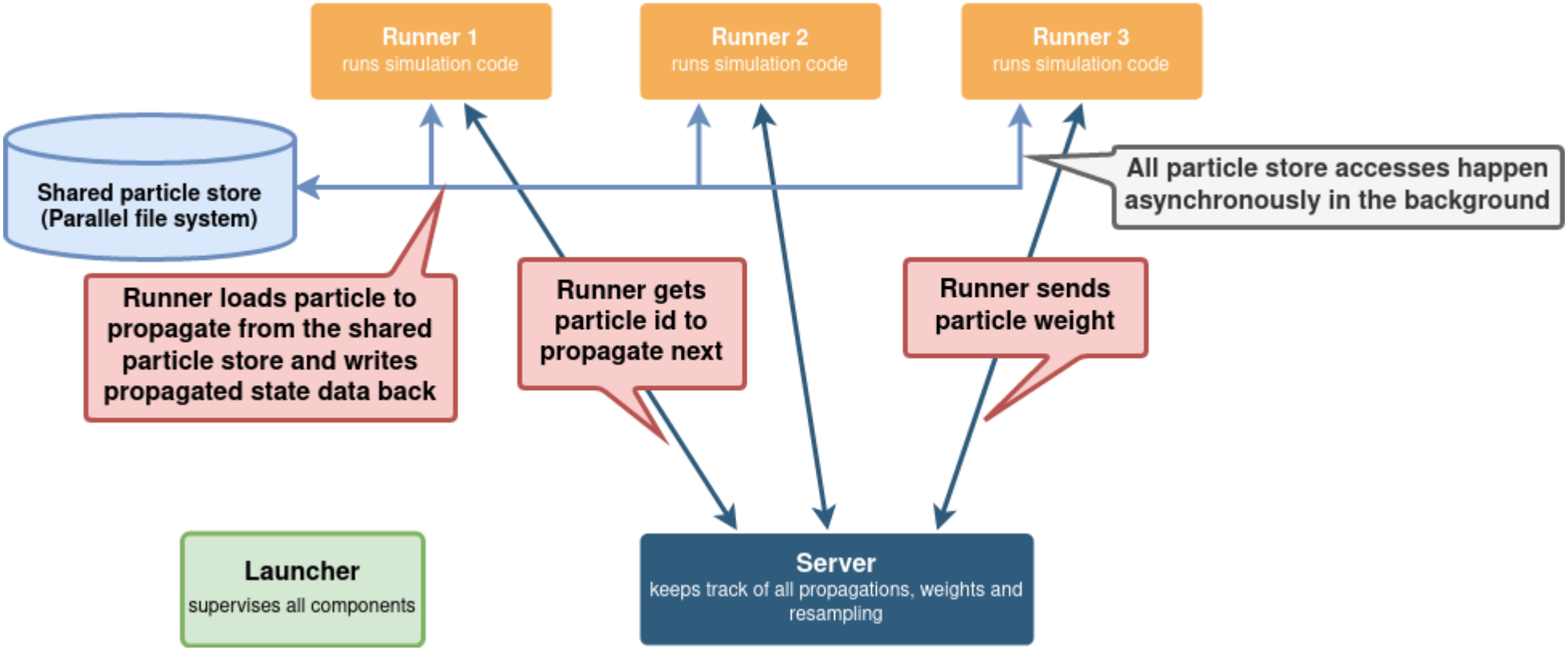
Resilience, Elasticity

- **Runners may fail independently**
- **FTI**
- **Same mechanism can be used for elastic execution**
  - adding/ removing runners at runtime

Implementation

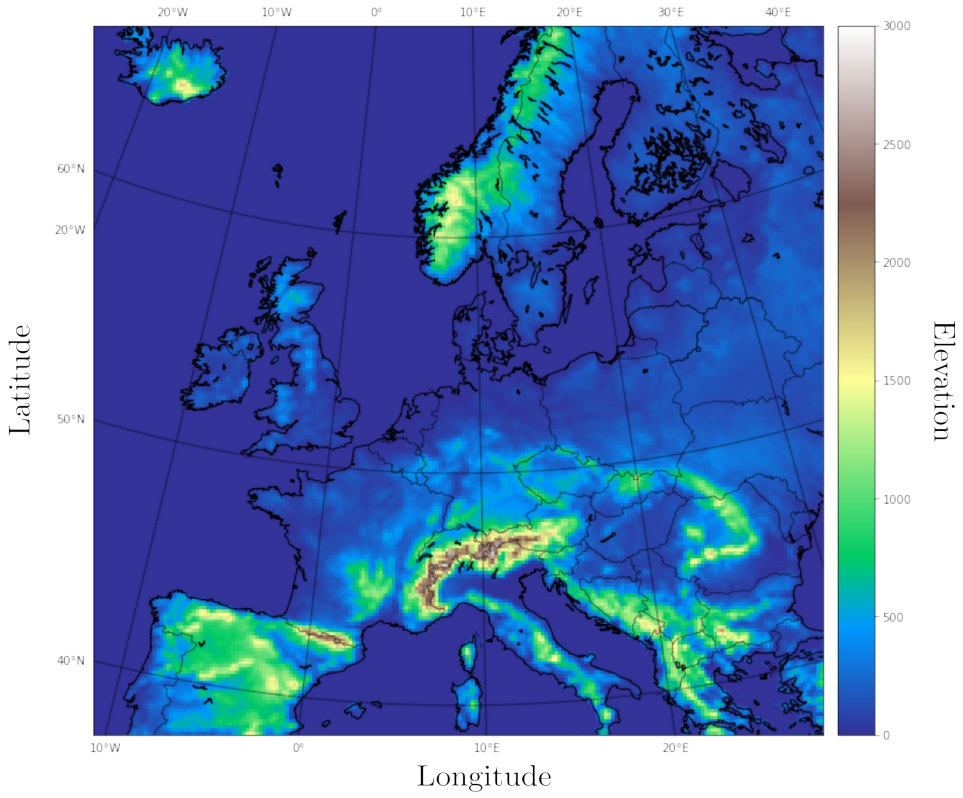
- **Melissa: very large scale In Situ ensemble runs**
- **Instrumented simulations (Runners) communicate as clients with a central server**
- **Runners and server are started up by a Launcher component**
  - interacts with machine's batch scheduler
- **Grown to the multi purpose Melissa software family**
  - Sensitivity analysis
  - Data Assimilation
  - Deep surrogate training

Architecture

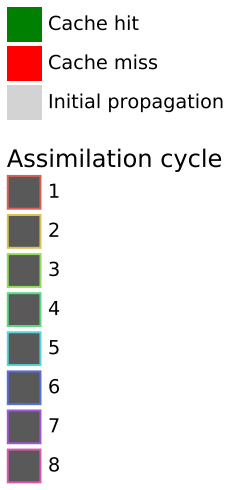


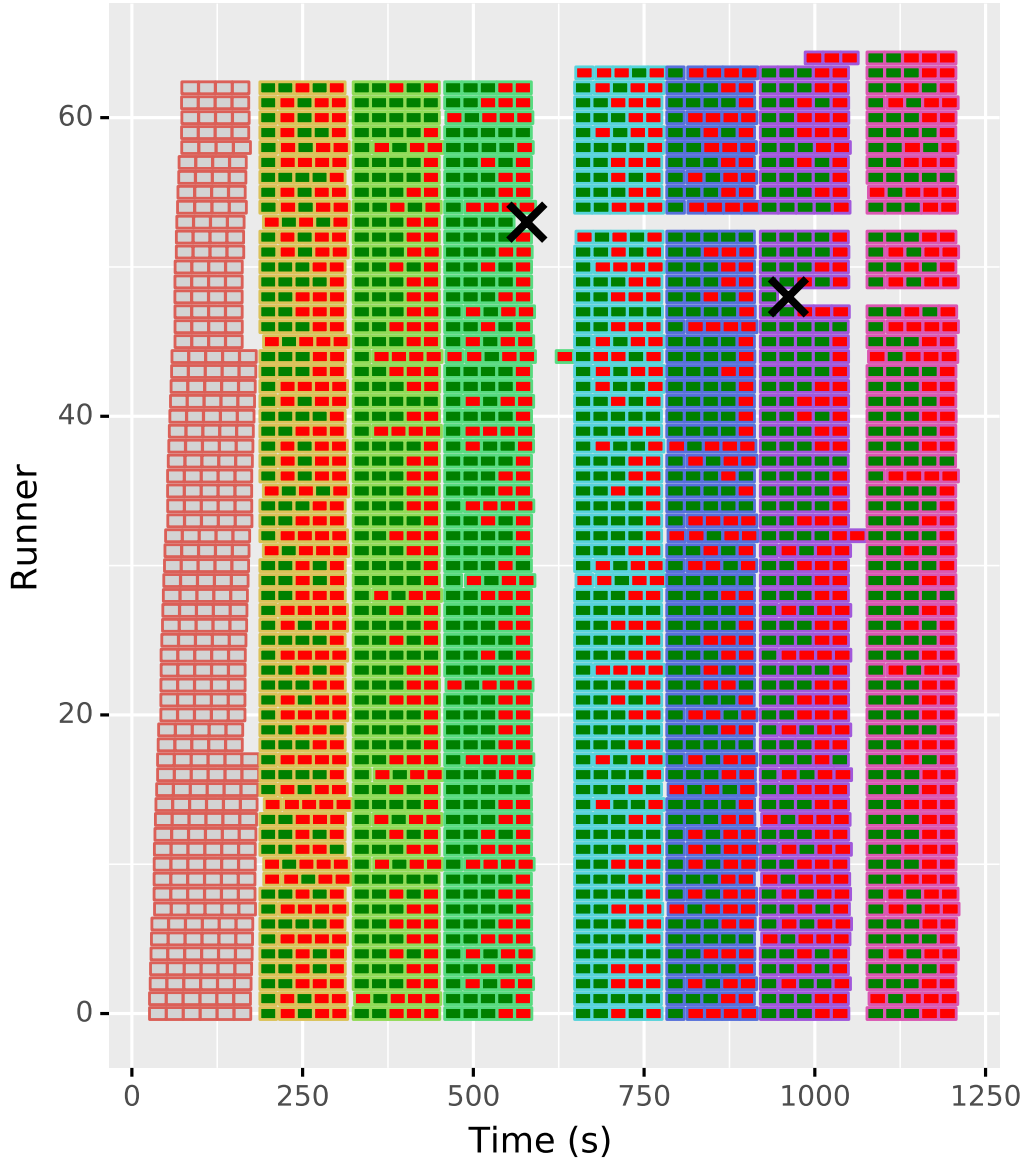
Use case

- **WRF, European domain**
- **15 km horizontal resolution**
- **49 vertical levels → 2.5GiB per particle**
- **Boundary conditions: ERA5 Dataset (ECMWF)**
- **Observations: CFRACT from EUMETSAT CMSAF**
- **Hourly resampling for high stress test**
- **Jean-Zay Supercomputer (40 cores @ 2.5GHz per node, up to 512 nodes)**



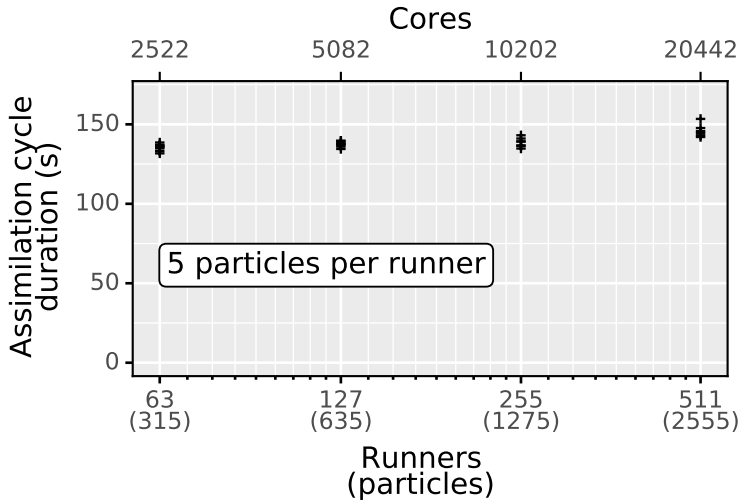
Experimental results

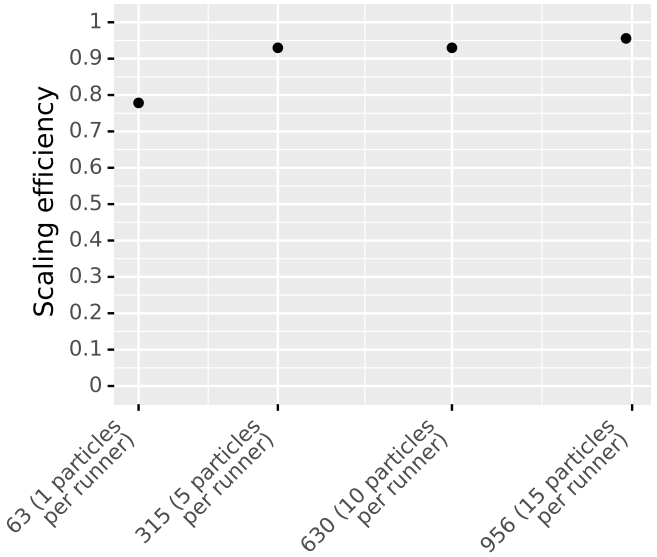




Scaling efficiency

> 90 % when at least 10 particles per runner (amortizing resampling time)





Conclusion, future work

- **Particle virtualization enables**
  - Amortize model initialization cost
  - Load balance
  - Hide large data transfers
  - Resilience

  - Architecture handles very large particle ensembles (≥ 2555 particles, scaling efficiency > 90% if more than 10 particles per runner)

- **Ongoing work: Unsync resampling (speculative particle execution)**

More information

- **Visit us online: gitlab.inria.fr/melissa/ , gitlab.inria.fr/melissa/melissa-da**

  Links to papers, preprints, other presentations

- **Contact: sebastian.friedemann@inria.fr**

Q? & A!