Fine-grained data caching approaches to speedup a distributed RDataFrame analysis

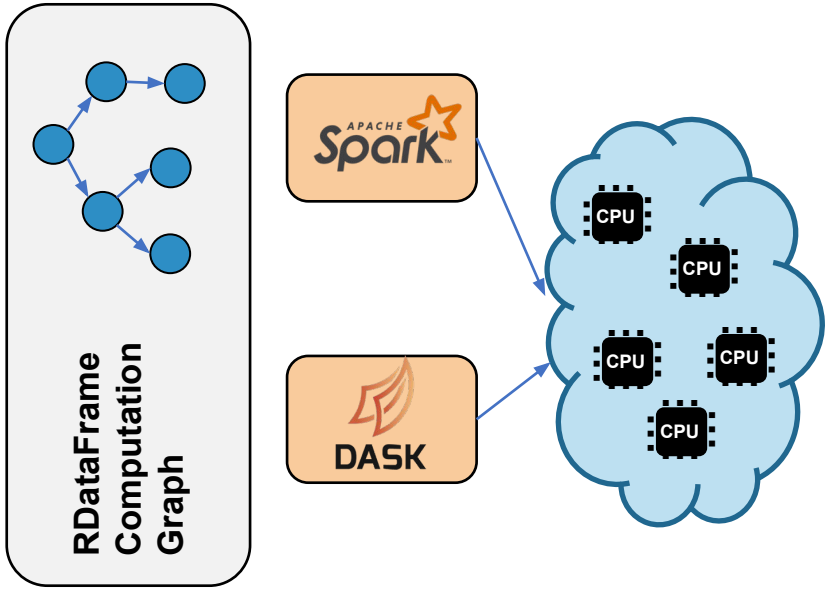
ROOT

Data Analysis Framework

https://root.cern

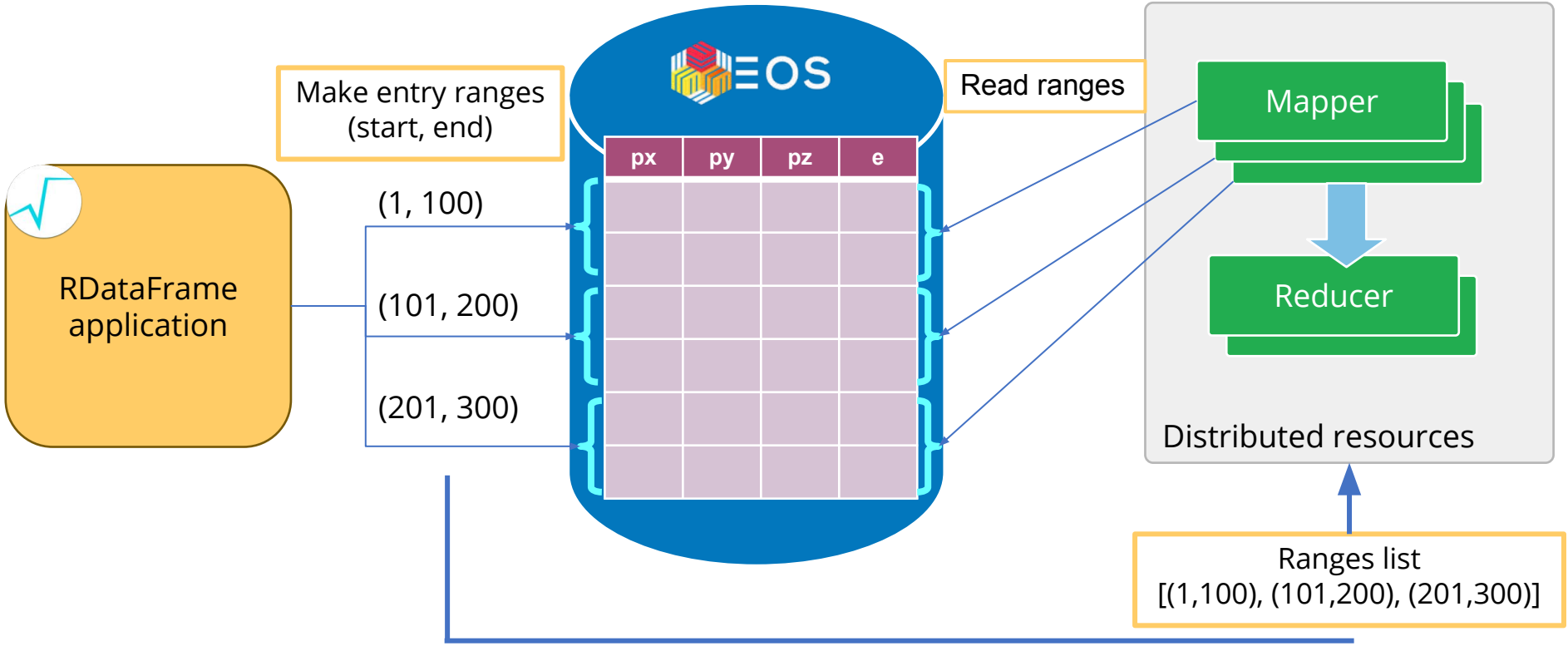
Distributed ROOT RDataFrame

- Enable **interactive large scale distributed** data analysis with RDataFrame
- Can run with different schedulers
  - Currently supporting **Spark**, support for **Dask** coming soon
- Analysis from start to end in a single interface



[Tutorial: distributed RDataFrame with Spark]

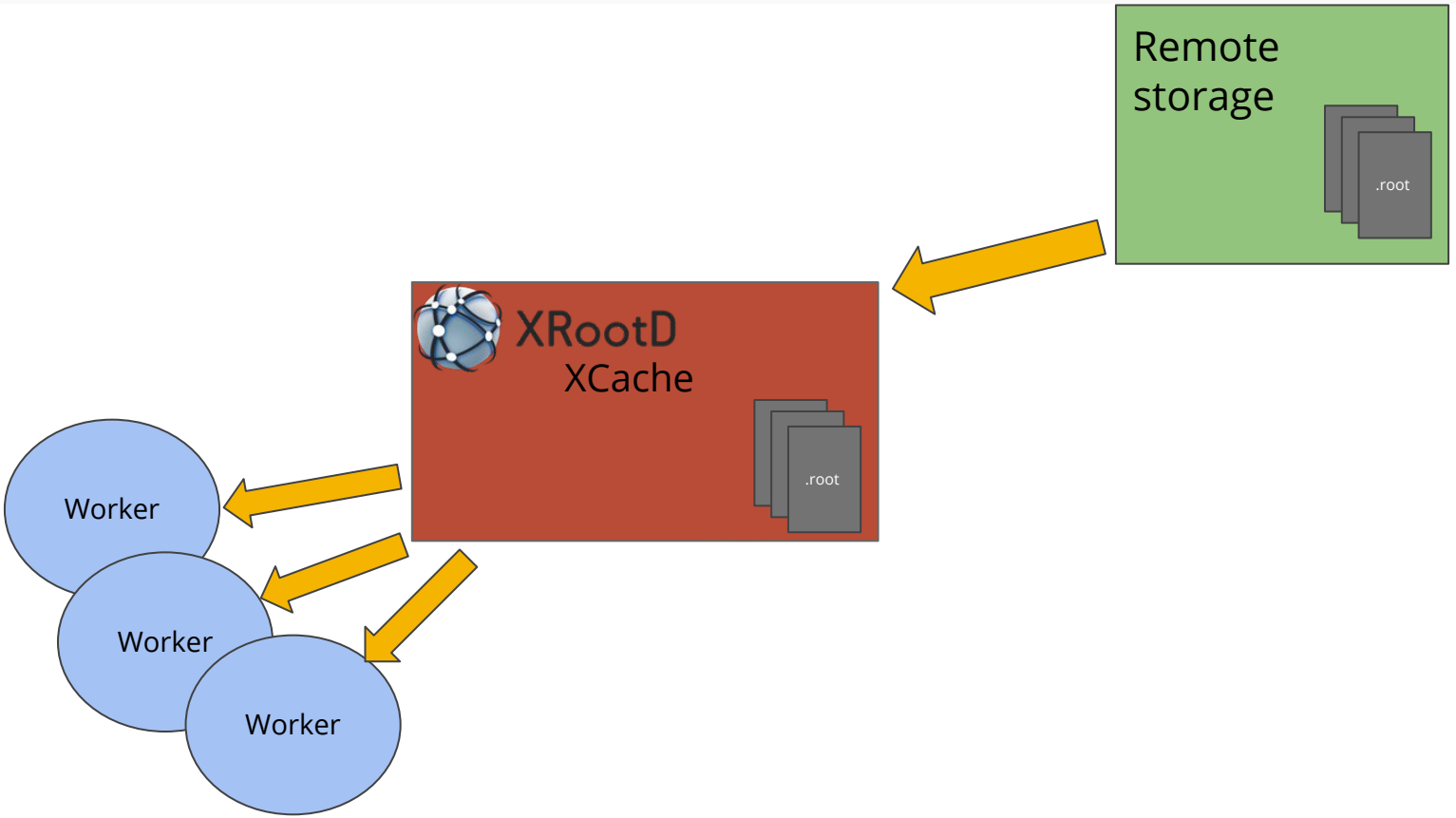
Behind distributed RDataFrame



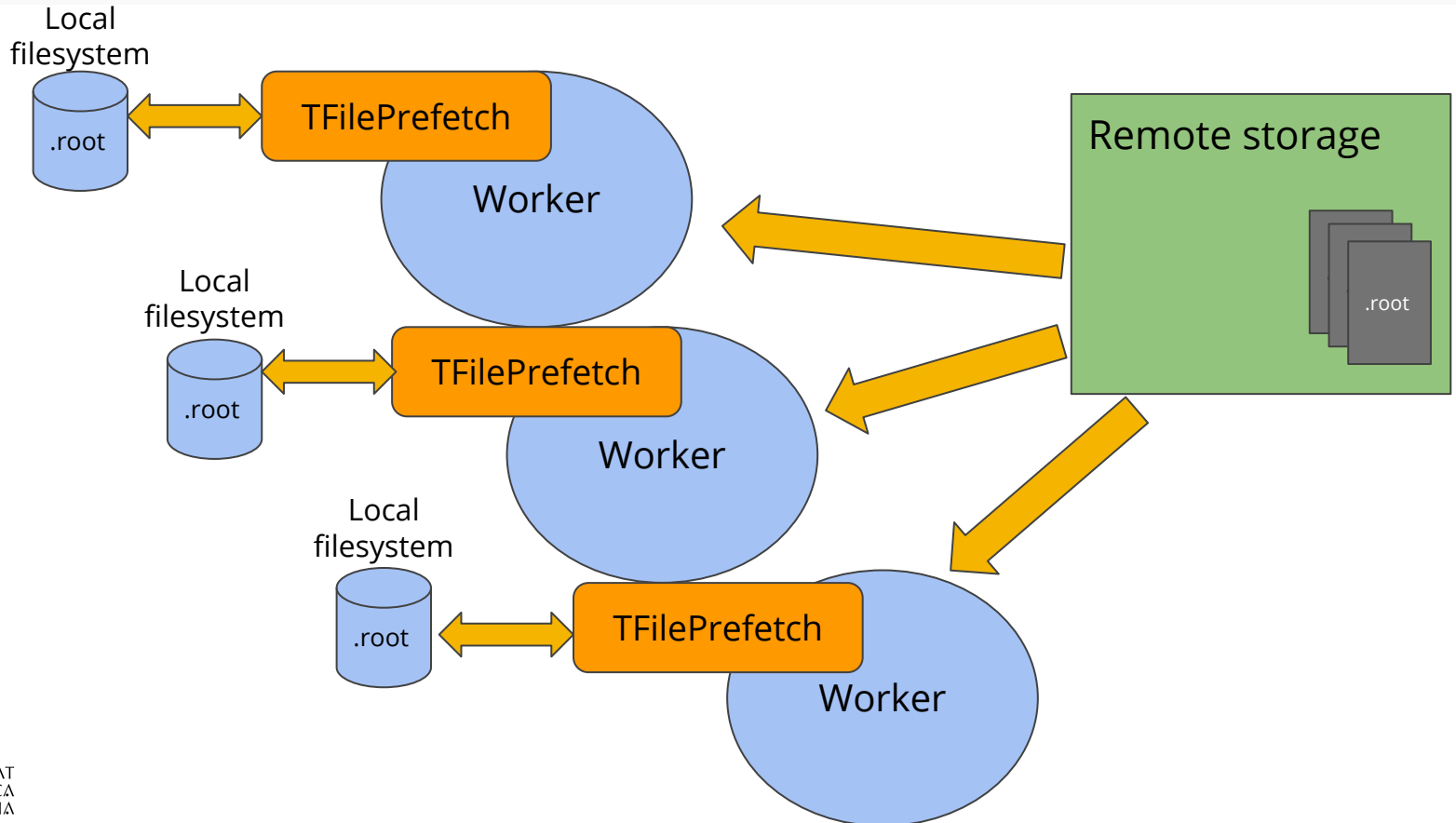
How/why data caching

- Exploratory analysis, repetitive workflow
- Data is often remote → **caching** can prove beneficial
- Sparse data access → **granularity** required
- Two approaches compared:
  - **Separate** cache **server** (`XRootD XCache`)
  - **Local caches** (`ROOT TFilePrefetch`) on the computing nodes

XRootD XCache server



TFilePrefetch local cache



Test application

Reference dataset

- **1 file**, 100M entries, 5 columns
- File size: 1.8 GB
- Cached column size: **~700MB**
- Stored in **EOS**

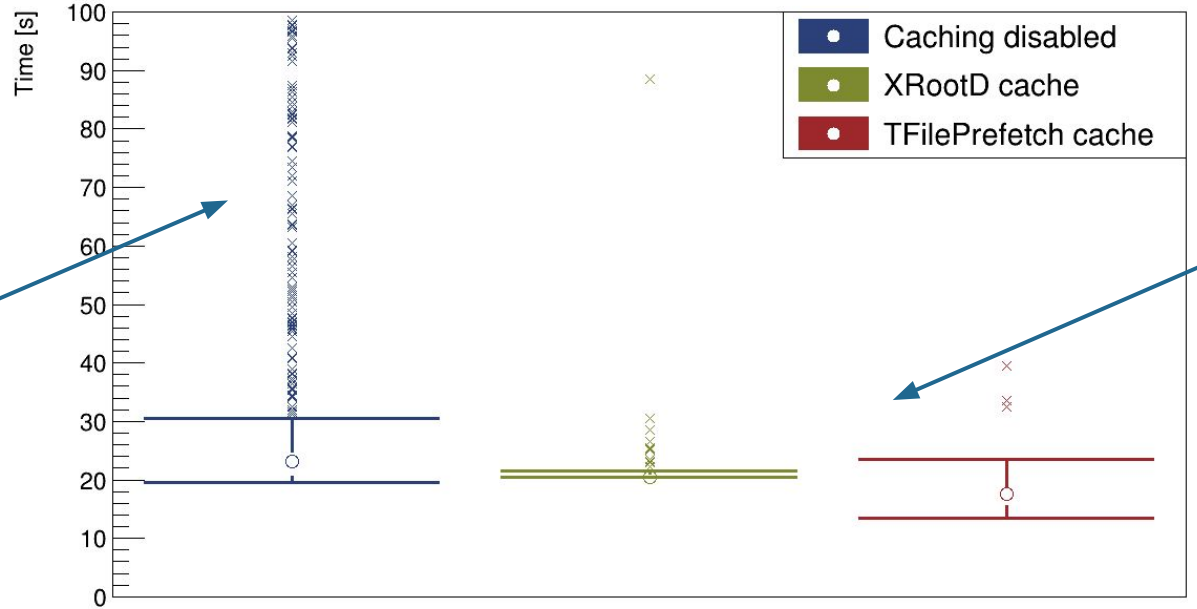
Test runs

- Simple operation on a column
- Single node / distributed
- Remote data / cached data

Github repository

Single Node application

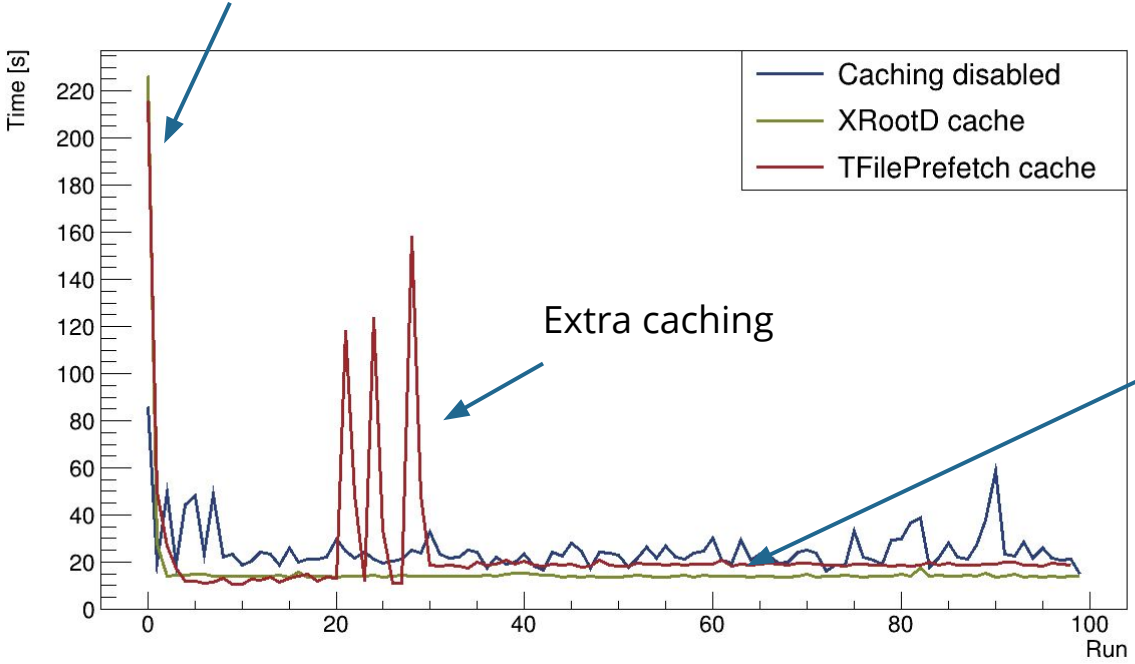
Very high runtime variability due to remote IO



XRootD cache shows the best consistency

Distributed application
non-deterministic task scheduling

First run, caches are populated



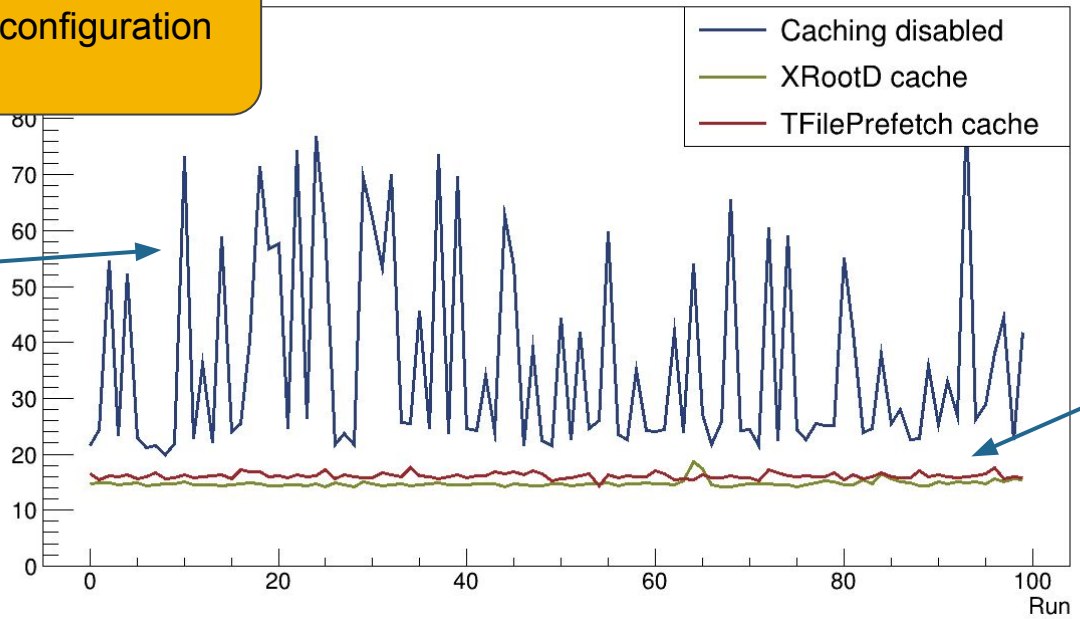
Extra caching

Stable, lower runtimes with caching

Distributed application
locality aware task scheduling

Omitting first run of each configuration

Very high variability when reading from EOS!



Cached runtimes are quite similar

BUT local caches can perform better

Conclusions and next steps

- Caching in HEP needs careful consideration: **granularity**, cache system **layout**, data **locality**
- XCache and TFilePrefetch show similar performance
  - On the **dataset** and workflow analysed and the **hardware** used
  - If data locality is **guaranteed** on the computing nodes

For the future:

- Multiple applications caching concurrently
- Different compression algorithms, TTree vs RNTuple
- Investigate issues with more latency (geographical distance)
- Ad-hoc caching mechanisms for RDataFrame analysis