Pipelining Computational Stages of the Tomographic Reconstructor for Multi-Object Adaptive Optics on a Multi-GPU System

Ali Charara^a, Hatem Ltaief^a, Damien Gratadour^b, David Keyes^a, Arnaud Sevin^b, Ahmad Abdelfattah^a, Eric Gendron^b, Carine Morel^b, Fabrice Vidal^b

^a Extreme Computing Research Center – King Abdullah University of Science and Technology, ^b Observatoire de Paris (LESIA, CNRS, UPD, UPMC)

1. BACKGROUND

European Extreme Large Telescope (E-ELT) is a high priority project in **ground based astronomy** that aims at constructing the **largest telescope** ever built. MOSAIC is an instrument proposed for E-ELT using **Multi-Object Adaptive Optics (MOAO)** technique for astronomical telescopes, which **compensates** for effects of atmospheric turbulence on **image quality**, and operates on patches across a large FoV.

2. MOTIVATION

- **Real time simulation** needed.
- **Extremely compute intensive.**
- Core computation in constructing **Tomographic Reconstructor (TR)**.
- **Drives in real-time** the Deformable Mirror (DM).
- **Repeatedly** computed by inverting **huge dense symmetric** covariance matrix (order of **100k x 100k**)

© ESO, https://www.eso.org/sci/facilities/develop/ao/ao_modes.html

3. CONTRIBUTION

- **Regularize** the **numerically unstable** problem for TR computation [1], by accounting for the **natural noise** from WFS.
- Positive definite **invertible** matrix, allowing for the **less expensive direct Cholesky** based matrix inversion.
- Operate on **tile algorithms** – exposing **more concurrency** – through MORSE dense linear algebra library [2].
- **Accelerate** computation using HPC (High Performance Computing) tools that can utilize the hardware resources (CPU cores and GPUs) to near optimal performance.
- Deploy concepts of **dataflow programming** model to **facilitate** data management, **dynamic scheduling** and **synchronization-reducing flow** of task execution. Established by **pipelining** different stages of computation through the **dynamic runtime system** StarPU [3].
- **Unprecedented speed** and **scale** on **shared memory** system.

4. IMPLEMENTATION

- TR computation of 4 main stages: Cholesky factorization (**DPOTRF**), Cholesky-based inversion (**DTRTRI** & **DLAUUM**), symmetric matrix-matrix multiplication (**DSYMM**). breaks down to smaller subtasks as shown in the DAG figure.
- **Tile-based** implementations by MORSE over both **CPU & GPU**.
- StarPU builds a **DAG** representing tasks and their dependencies.
- **Asynchronously schedules** these tasks **out of order**.
- **Scheduling policy** that is **data aware**, relies on **history based performance models** (calibrated and adapted to available hardware).
- **High priority** to tasks at the **critical path**.