https://www.iacs.stonybrook.edu/project/tesse

TESSE, Task-based Environment for Scientific Simulation at Extreme Scale

Damien Genet, Innovative Computing Laboratory,
University of Tennessee, Knoxville

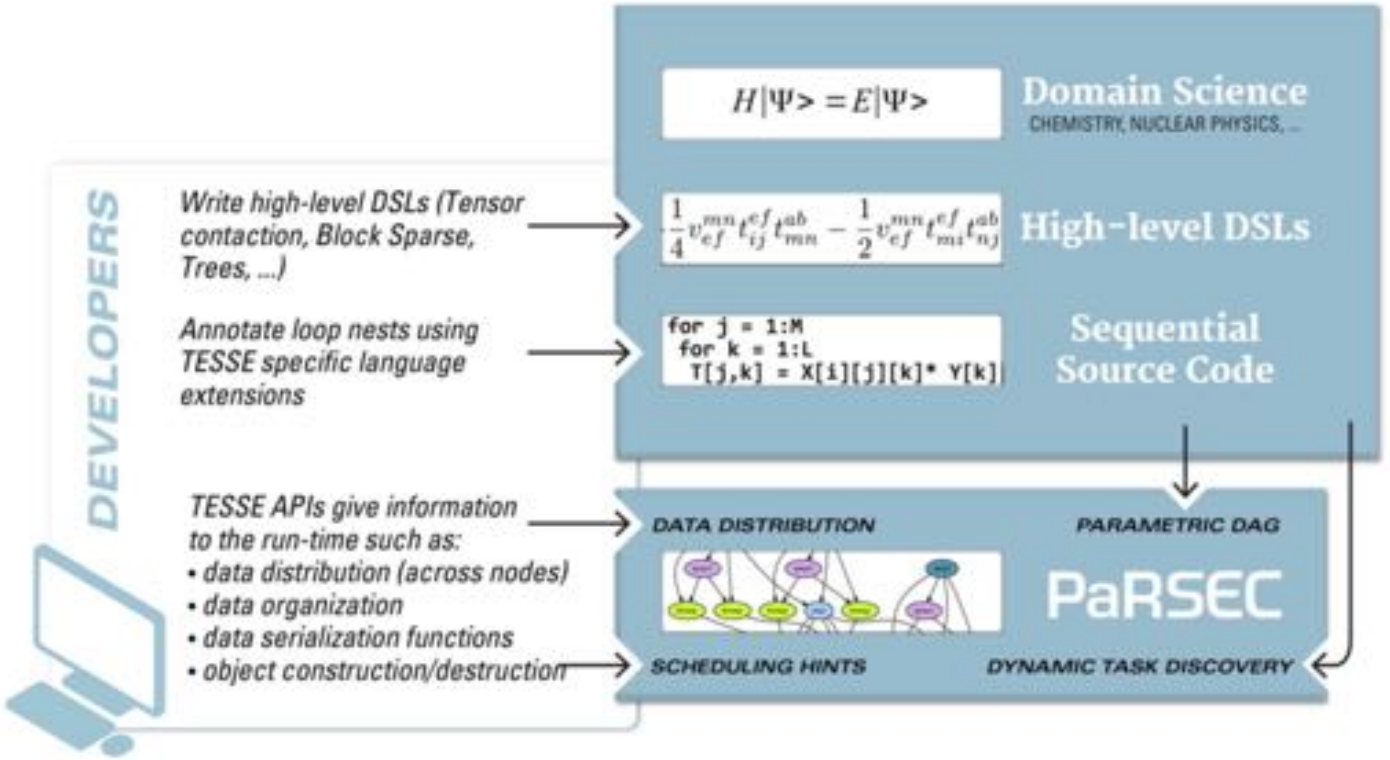
7th JLESC, July 18th, 2017

- Outline
  - TESSE project, main objectives
  - MADNESS, TiledArray, PaRSEC
  - Integration TiledArray+MADNESS and PaRSEC
  - Ongoing work

TESSE

- Task-based environment for scientific simulation at extreme scale
- Stony Brook University
  - Robert J. Harrison
- University of Tennessee
  - George Bosilca and Thomas Herault
- Virginia Tech
  - Eduard Valeev

Application-driven design of a general-purpose and production quality software framework addressing programmer productivity and portable performance for advanced scientific applications on massively-parallel, hybrid, many-core systems of today and tomorrow.



NSF SSI project ACI-1450344 (SBU), ACI-1450262 (VT), ACI-1450300 (UTK)

Main project objectives

- Provide a robust and scalable directed acyclic graph (DAG) execution model and intelligent runtime that can
  - adapt to evolving numerical theories and HPC platforms,
  - enhance scientific productivity
- Transform the scalability of key parts of existing numerical simulation codes by
  - extending domain specific languages (DSLs),
  - utilizing the API of the TESSE runtime,
  - furnishing a migration path for both applications and application programmers
- Demonstrate the feasibility through new science capabilities and proof-of-principle science studies using TESSE-enabled versions of MADNESS, MPQC (TiledArray) and other codes

- Outline
  - TESSE project, main objectives
  - MADNESS, TiledArray, PaRSEC
  - Integration TiledArray+MADNESS and PaRSEC
  - Ongoing work

MADNESS

https://github.com/m-a-d-n-e-s-s/madness

Let

$$\Omega = [-20, 20]^3$$

$$r = x \to \sqrt{x_0^2 + x_1^2 + x_2^2}$$

$$g = x \to \exp(-2 * r(x))$$

$$v = x \to -\frac{2}{r(x)}$$

In

$$\nu = \mathcal{F}\, v$$

$$\phi = \mathcal{F}\, g$$

$$\lambda = -1.0$$

for $i \in [0, 10]$

$$\phi = \phi * \|\phi\|^{-1}$$

$$V = \nu - \nabla^{-2}(4 * \pi * \phi^2)$$

$$\psi = -2 * (-2 * \lambda - \nabla^2)^{-1}(V * \phi)$$

$$\lambda = \lambda + \frac{\langle V * \phi | \psi - \phi \rangle}{\langle \psi | \psi \rangle}$$

$$\phi = \psi$$

print "iter", i, "norm", $\|\phi\|$, "eval", λ

end

End

- Provides a general purpose numerical environment for reliable and fast scientific simulation
- Supports Chemistry, nuclear physics, atomic physics, material science, nanoscience, climate, fusion, …
- Highest-level DSL
- Provides a runtime environment based on futures
- Composes directly in terms of functions and operators with guaranteed precision

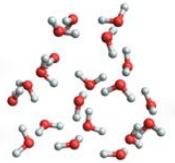
<= This is a Latex rendering of a program to solve the Hartree-Fock equations for the helium atom. The compiler outputs C++ code that without modification can be compiled and run in parallel (threads+MPI)

- Evaluates functions on Haar basis, constant piecewise
- Uses local adaptive refinement until local error measurement is satisfied

- Facile path from laptop to exaflop

TiledArray

https://github.com/ValeevGroup/tiledarray/



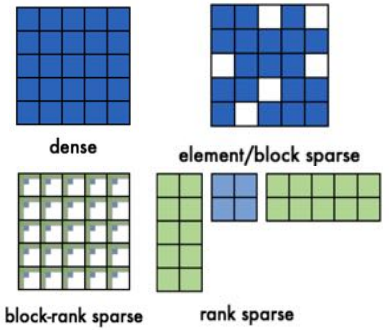
- Generic massively parallel framework for dense and sparse tensor algebra
- State of the art application to electronic structure of chemistry and materials in Massively Parallel Quantum Chemistry (MPQC) package
  - Prototyping platform for DOE Exascale Chemistry App
  - Experimental use by research codes, e.g. ChronusQuantum (Xiaosong Li/UW)
- Reduces communication and load imbalance of sparse tensor algebra using data-driven MADNESS runtime
- High-level DSL from math to C++ expressions:

$$R_{iajb} = G_{iajb} + F_{ac}T_{icjb} + F_{bc}T_{iajc} - F_{ik}T_{kajb} - F_{jk}T_{iakb}$$
$$E = (G_{iajb} + R_{iajb})(2T_{iajb} - T_{ibja})$$

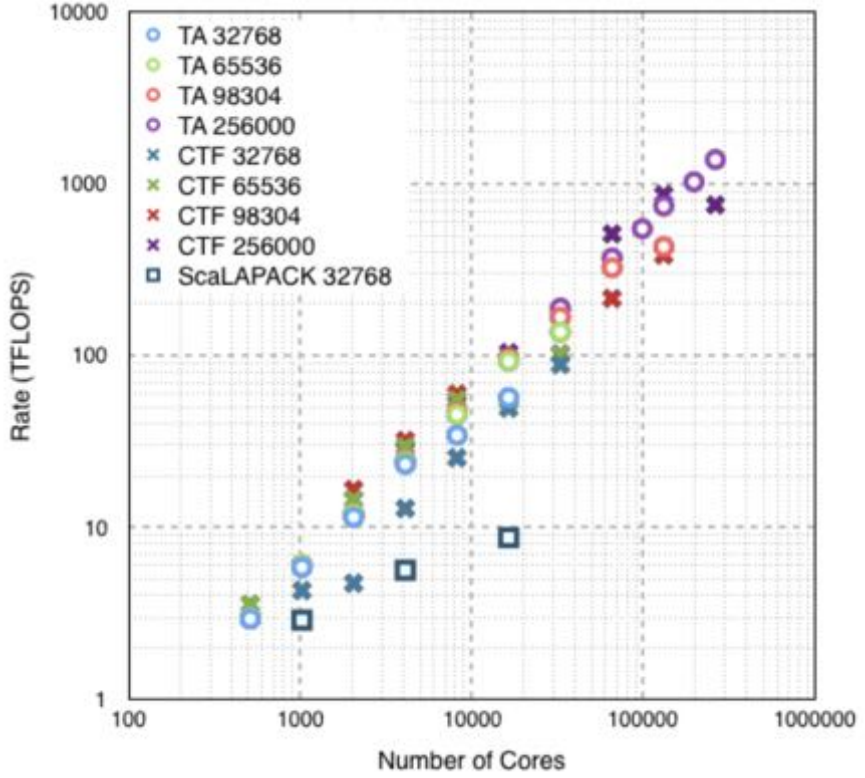
```
TA::TArrayD R(world, ovov);

R("i,a,j,b") = G("i,a,j,b") + Fv("a,c") * T("i,c,j,b") +
               Fv("b,c") * T("i,a,j,c") - Fo("i,k") * T("k,a,j,b") -
               Fo("j,k") * T("i,a,k,b");

double energy =
    (G("i,a,j,b") + R("i,a,j,b")).dot(2 * T("i,a,j,b") - T("i,b,j,a"));
```



Dense square GEMM on IBM Blue Gene/Q

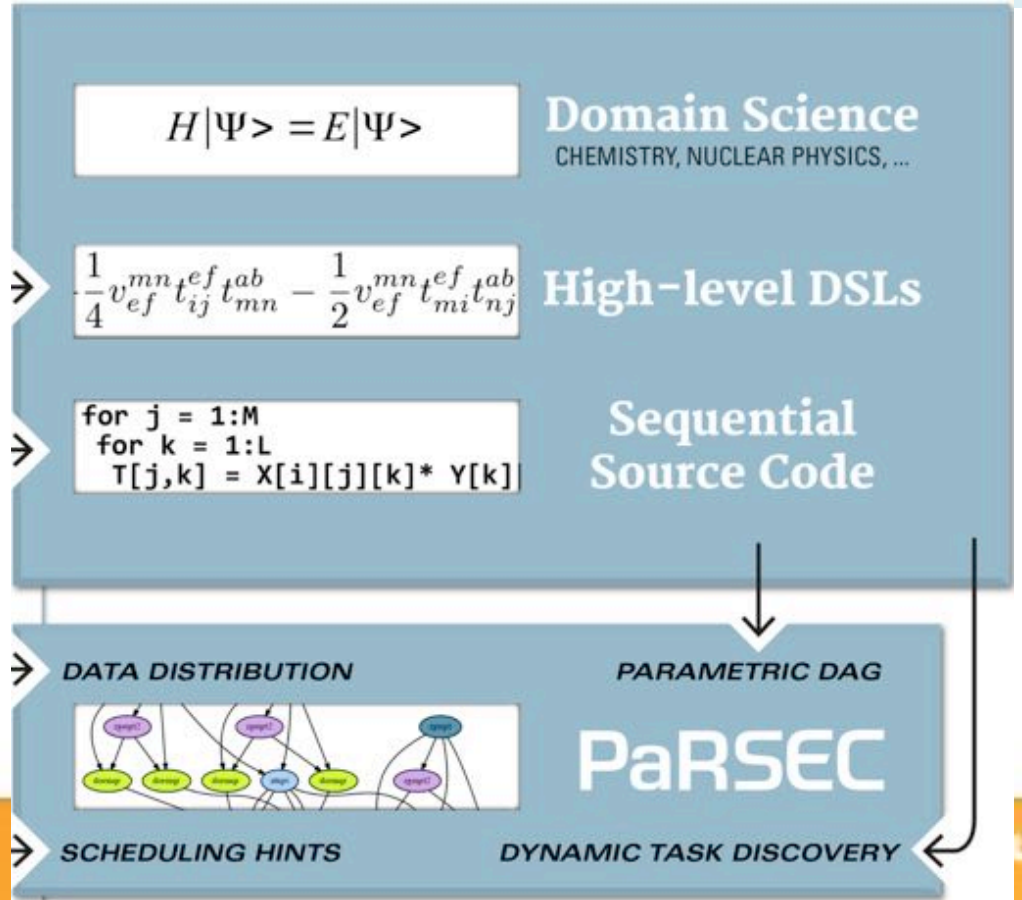


Calvin, Lewis, Valeev, *Proceedings of IA^3 Workshop* (2015)

PaRSEC: a generic runtime system for asynchronous, architecture aware scheduling of fine-grained tasks on distributed many-core heterogeneous architectures

Concepts

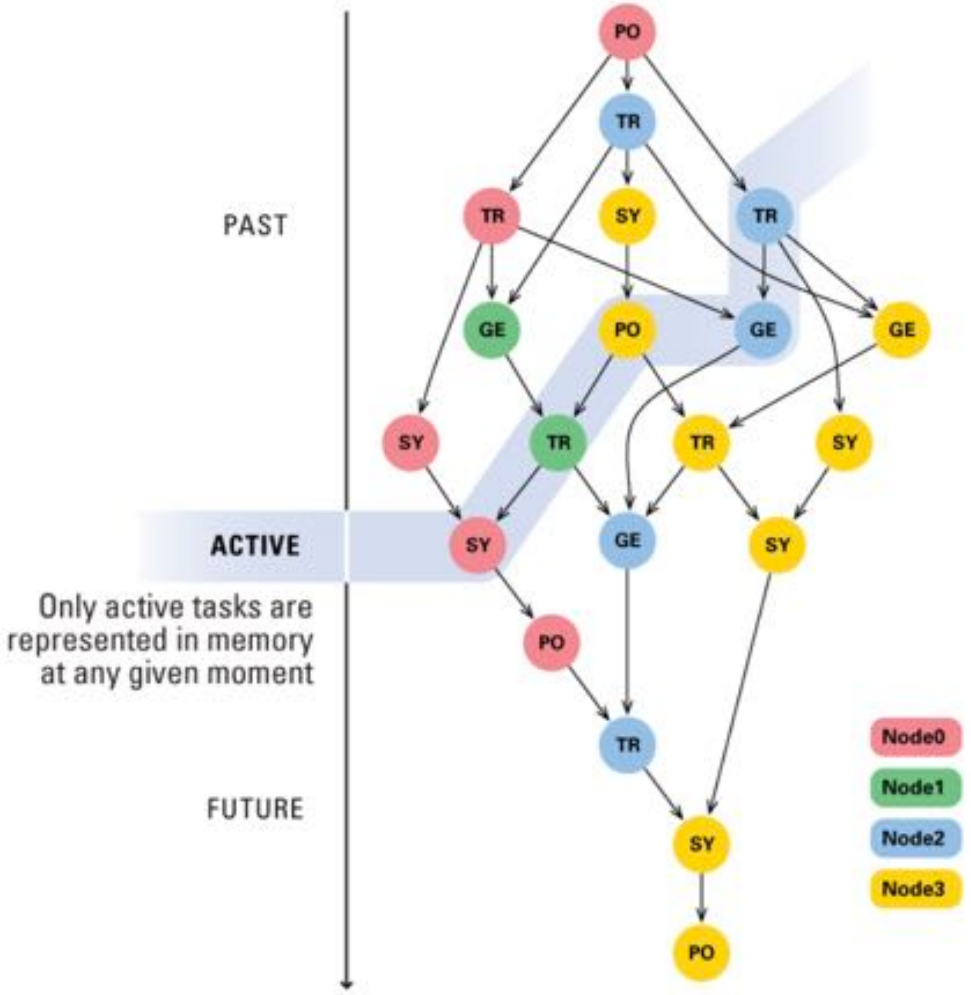
- Clear separation of concerns: compiler optimize each task class, developer describe dependencies between tasks, the runtime orchestrate the dynamic execution
- Interface with the application developers through specialized domain specific languages (PTG/JDF, Python, insert_task, fork/join, …)
- Separate algorithms from data distribution
- Make control flow executions a relic



Runtime

- Portability layer for heterogeneous architectures
- Scheduling policies adapt every execution to the hardware & ongoing system status
- Data movements between producers and consumers are inferred from dependencies. Communications/computations overlap naturally unfold
- Coherency protocols minimize data movements
- Memory hierarchies (including NVRAM and disk) integral part of the scheduling decisions

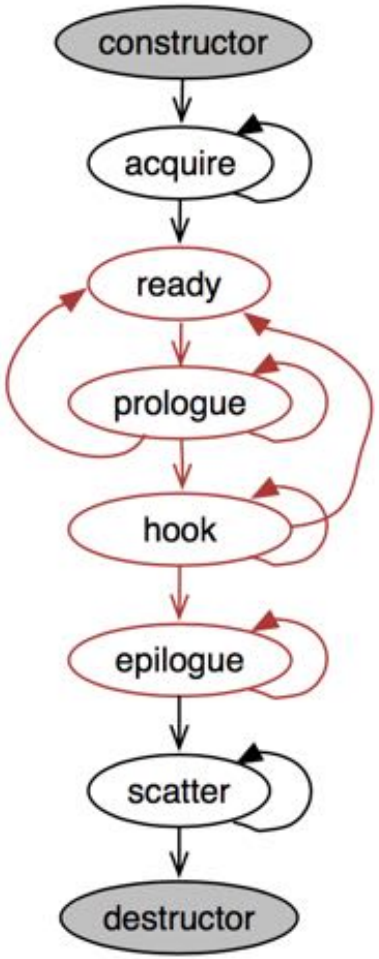
PaRSEC



- Extends Parallel Scheduling and Execution Controller (PaRSEC) to larger classes of dynamic (data-dependent) computation; data distribution; composition and execution of multiple DAGs
- Addresses
  - – heterogeneous hardware by runtime selection between multiple implementations
  - – heterogeneous data distribution by separate specification of data and algorithm, and runtime management of data motion
  - – heterogeneous task duration through lightweight scheduling policies
- Automatic latency hiding enabled by knowledge of the dataflow of the program to enable all communications to occur in the background of the execution itself

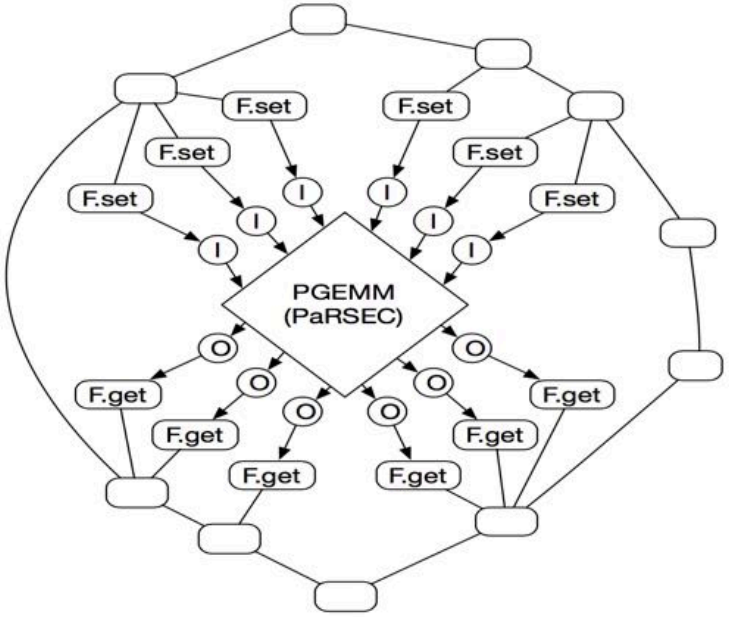
- Outline
  - TESSE project, main objectives
  - MADNESS, TiledArray, PaRSEC
  - Integration TiledArray+MADNESS and PaRSEC
  - Ongoing work

Concept

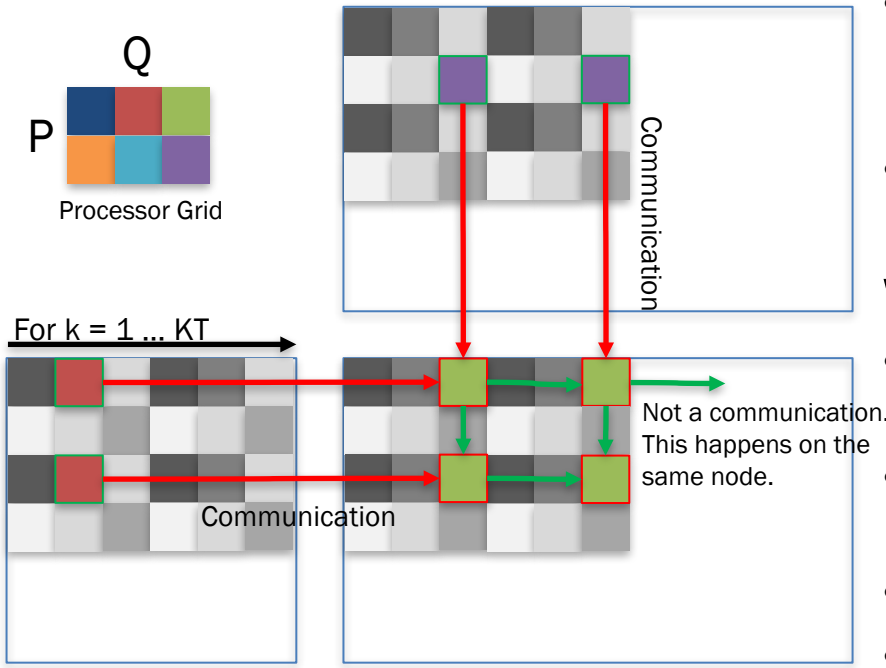


- Use PaRSEC as a support for MADNESS ready-tasks
- Similar to providing another threading management support
  - With benefits of thread binding, NUMA- aware schedulers
- Data movement, dependencies are still managed entirely by MADNESS
- Enables seamless integration of new PaRSEC-enabled components:
  - Interaction between TTG (TESSE) and other MADNESS programming paradigms
  - Interaction between DPLASMA (PaRSEC, Linera Algebra library) and MADNESS operations

- Inputs and Outputs of existing PaRSEC DAGs (e.g. DPLASMA) are presented to MADNESS as Futures
- acquire/scatter steps of the input/output tasks are specialized to expose the task as a MADNESS Future
- Operations are made asynchronous:
  - If a future is not set at acquire time, the task
  - is unscheduled, and a callback to reschedule it is registered with MADNESS when the future is set
  - Get operation on output tasks is empty (does not allocate resource) until the data is ready



PaRSEC side



Summa algorithm in PaRSEC

Legacy data structure for matrices computation,

- Matrix data structure embeds tile size at the top level
- Tile sizes are unique per matrix (mb x nb), which makes communication predictable and enable data mechanism for recycling blocks of memory
- Tile sizes can be finely tuned for a machine

What has been done,

- Tile sizes go down the data structure hierarchy and are embedded by the tile itself
- Each matrix has an irregular tiling along each dimension, communication are all different
- Each tile will have a different size, impacting overall performance
- Tiles may be recursively tiled and the GEMM is resubmitted to the runtime.
- Tiles are re-tiled until the resulting GEMMs completion times fall around the average time for a PaRSEC task (~1ms)

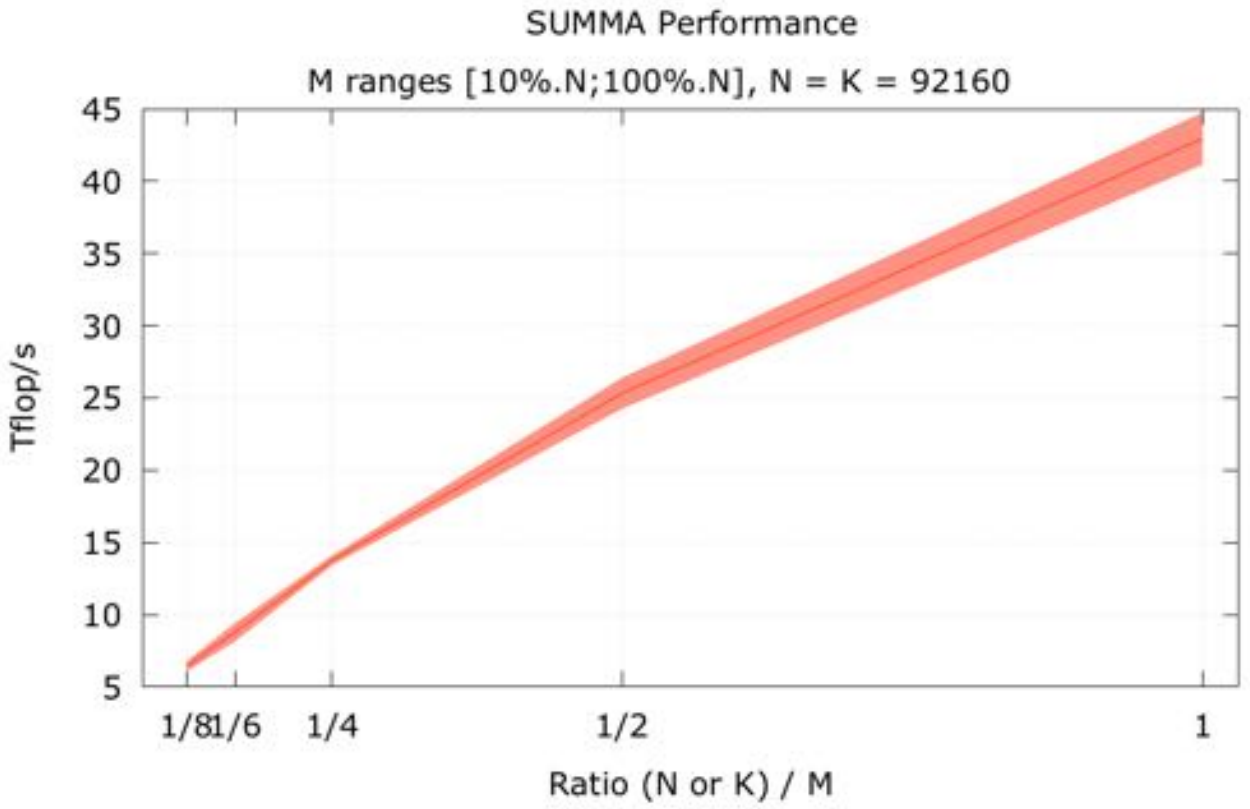
Regular SUMMA with TiledArray data

SUMMA from the PaRSEC driver using TiledArray testcase metadata.

4 nodes, each with 4 Nvidia P100 accelerators
GEMM peak for one card in double precision, 4Tflop/s

Performance as a function of the imbalance between the ‘M’ dimension and the others.

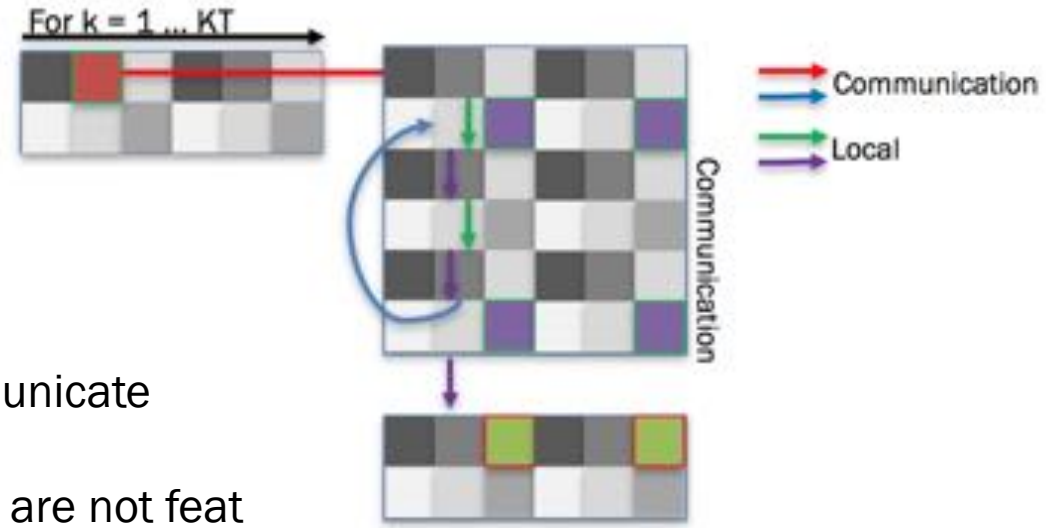
⇒ The regular GEMM is not fit for this problem
⇒ Aggregation of tiles, reduces the parallelism, improves single GEMM performance



PaRSEC side, modified broadcast

TA_cc_abcd, coupled cluster application is a hard problem,
Parameters UOCC = 300, NUOCC = 2 .. 16, OCC = 30, NOCC = 2 .. 5

- All the tiles are irregular;
- B is square (UOCC^2 x UOCC^2, tiled by NUOCC^2 on each dimension);
- A and C are wide but small (M dimension is OCC^2 tiled in NOCC^2 pieces);



- Tiles on dimension M are really small (OCC / NOCC)
- Tiles of C and A are small, meaning that they are cheap to communicate
- compared to B.
- Regular GEMM or SUMMA algorithm localizing computation on C are not feat for this problem.
- B should be fix, computation localized on B.
- A and C tiles move, we set up a reduction on C using a ring, targetting the rank supposed to write C ”on disk”.

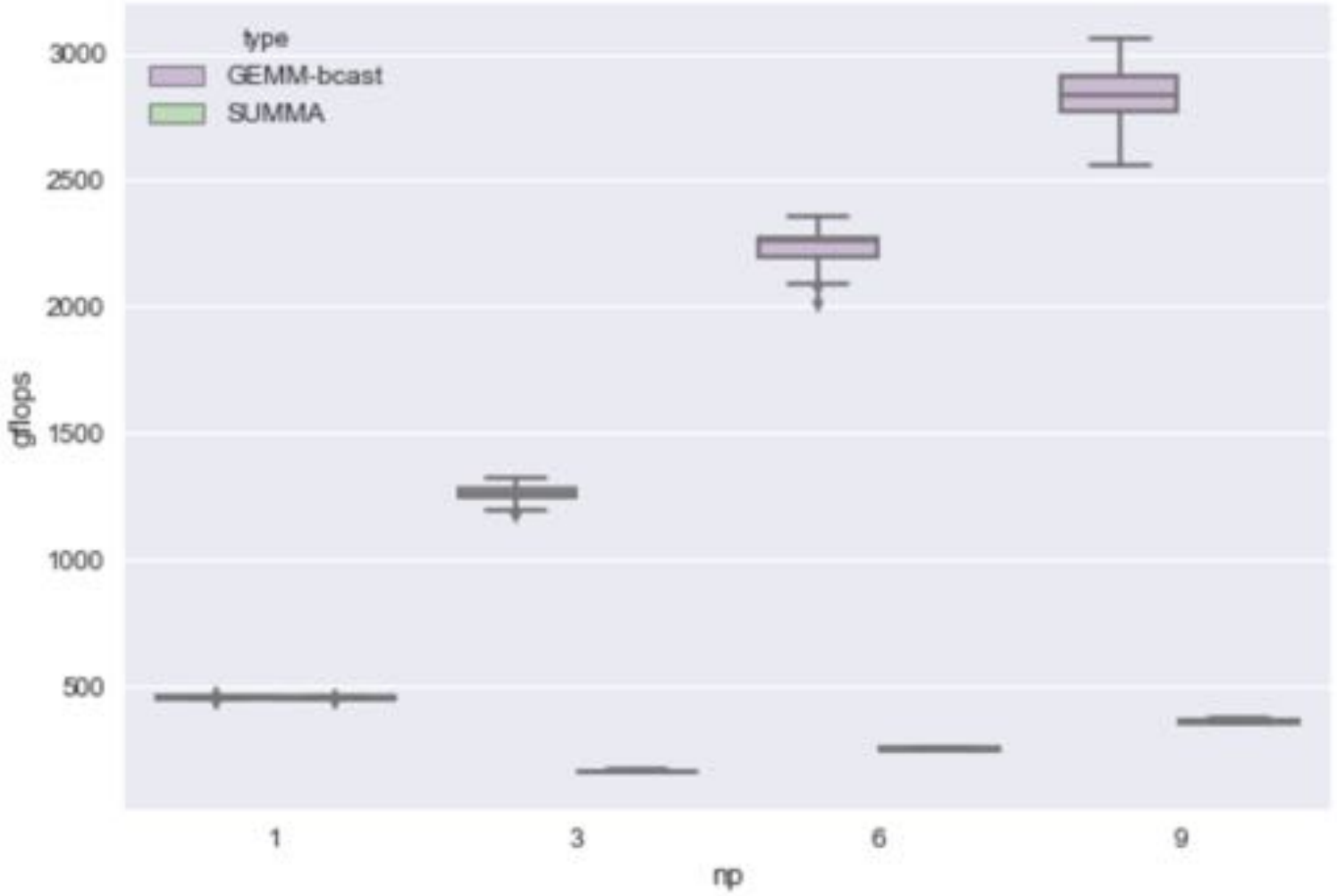
Results

TA_cc_abcd, coupled cluster application
Running with MADNESS and PaRSEC

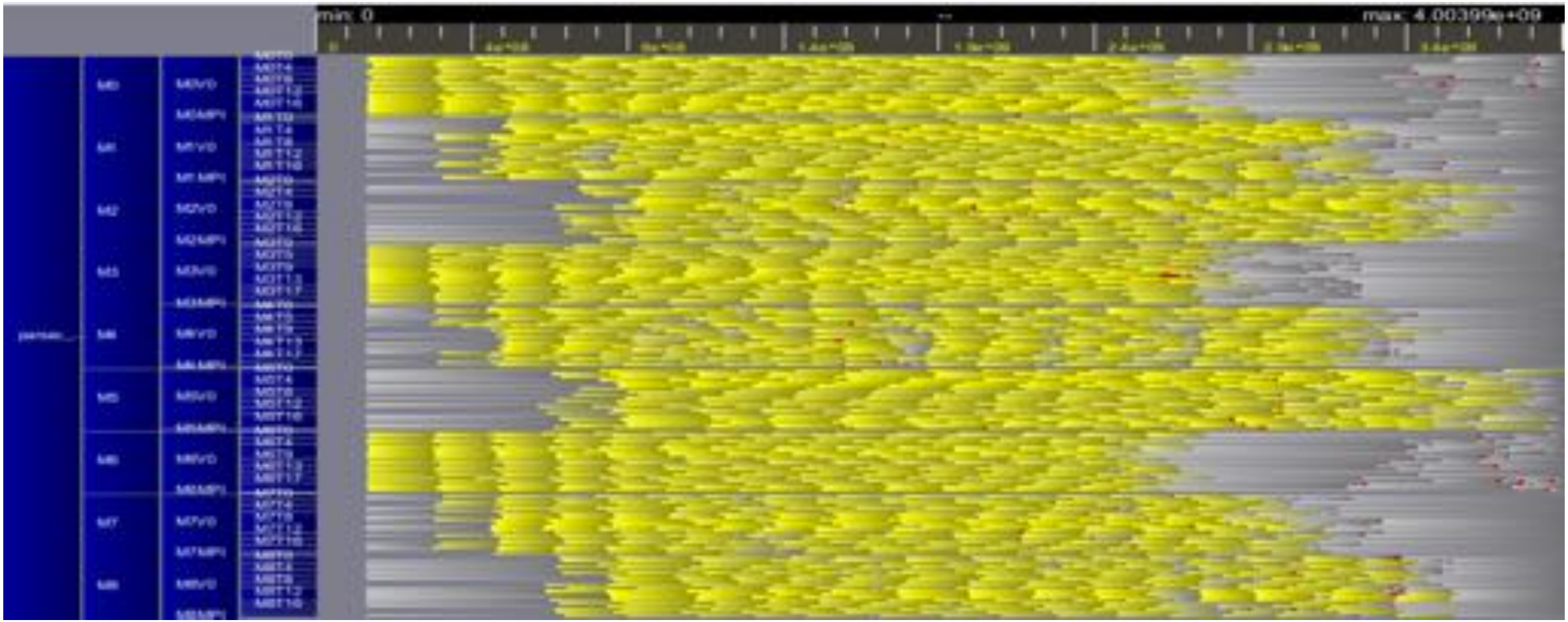
OCC: 30
NOCC: 3
UOCC: 250
NUOCC: 5

9 nodes (arc):
20 cores Intel Xeon CPU E5-2650 v3

GEMM peak: 453Gflops/node



Trace of the previous result



- Outline
  - TESSE project, main objectives
  - MADNESS, TiledArray, PaRSEC
  - Integration TiledArray+MADNESS and PaRSEC
  - Ongoing work

Ongoing work, Collaborations

- TTG, Templated Task Graph
  - C++ backend for incoming DSL
  - Set of classes mapping on the runtime internal structure, that describes best a directed acyclic graph

    $\Rightarrow$ Development of a frontend to generate the DAG from a domain specific language

- Expand PaRSEC capabilities to support multiple data representation (low-rank tiles) with operators to change representation, and dynamic algorithm for block sparse.
- Larger runs of TiledArray / ta_cc_abcd coupled cluster

- Open Collaborations
  - GEMM strategies for irregular, highly ill-shaped tiles
  - Aggregation strategies that will not destroy performance