Memory-Aware Scheduling of Tasks Sharing Data on Multiple GPUs

Maxime Gonthier, Loris Marchal, Samuel Thibault

Memory-Aware Scheduling of Tasks Sharing Data on Multiple GPUs

Maxime GONTHIER

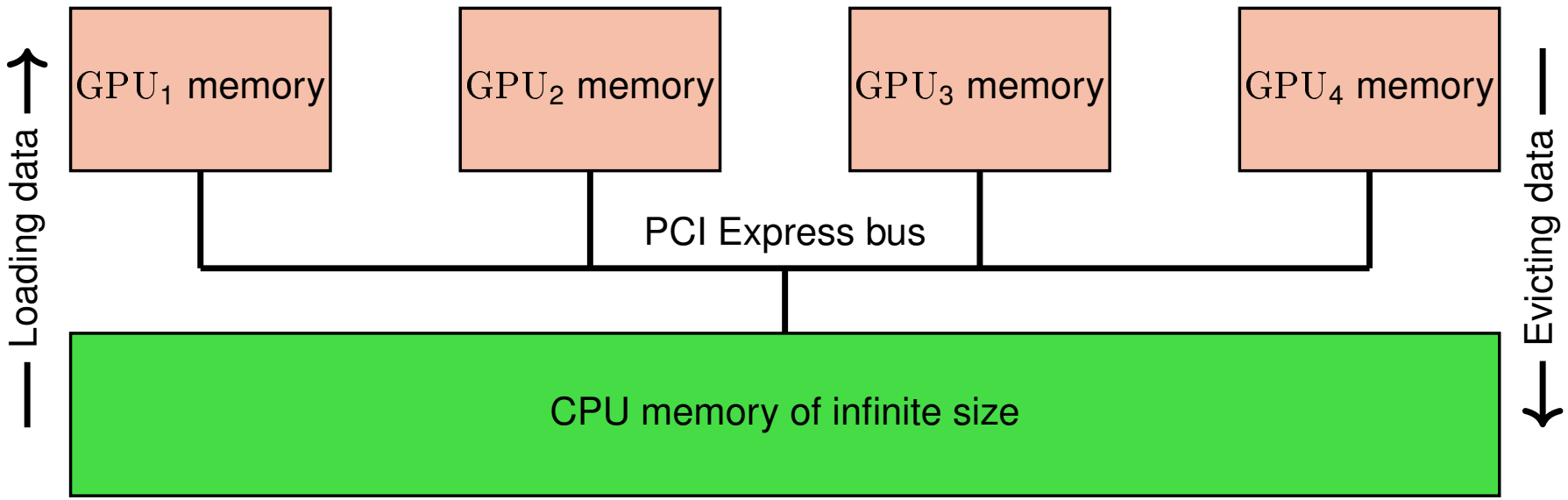
Supervised by Loris MARCHAL and Samuel THIBAULT
maxime.gonthier@ens-lyon.fr

LIP - ROMA - LaBRI - STORM - Inria
ENS de Lyon - Université de Bordeaux

March 23, 2023

ENS DE LYON université de BORDEAUX

Why focus on memory-aware scheduling ?

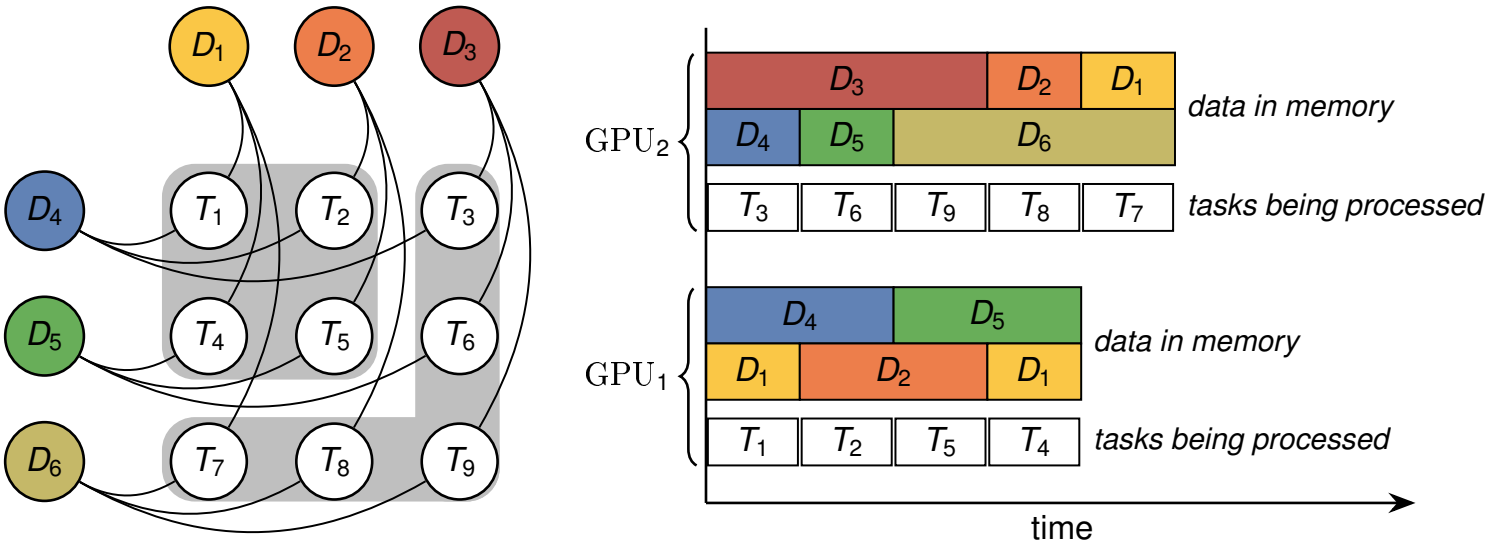


GPUs are quick but have a limited memory

They share the same PCI bus with limited bandwidth

ENS DE LYON université de BORDEAUX

Example with 2D grid dependencies



GOALS:

- **Balancing** tasks among GPUs → Reduce total execution time
- **Ordering** tasks inside each GPU → Reduce data transfers
- Having a **generic** scheduler → Efficient with any application or memory limitation

ENS DE LYON université de BORDEAUX

Dynamic scheduler of StarPU: DMDAR

Strategy

Schedule tasks so their completion time is minimal based on computation + communication times

+ **Ready** reordering heuristic on GPU_k

From the list L of tasks allocated on GPU_k, it will search the task $T \in L$ requiring the fewest data transfers

ENS DE LYON université de BORDEAUX

Data-Aware Reactive Task Scheduling: DARTS

The scheduling is done by GPU_k when it needs new tasks.

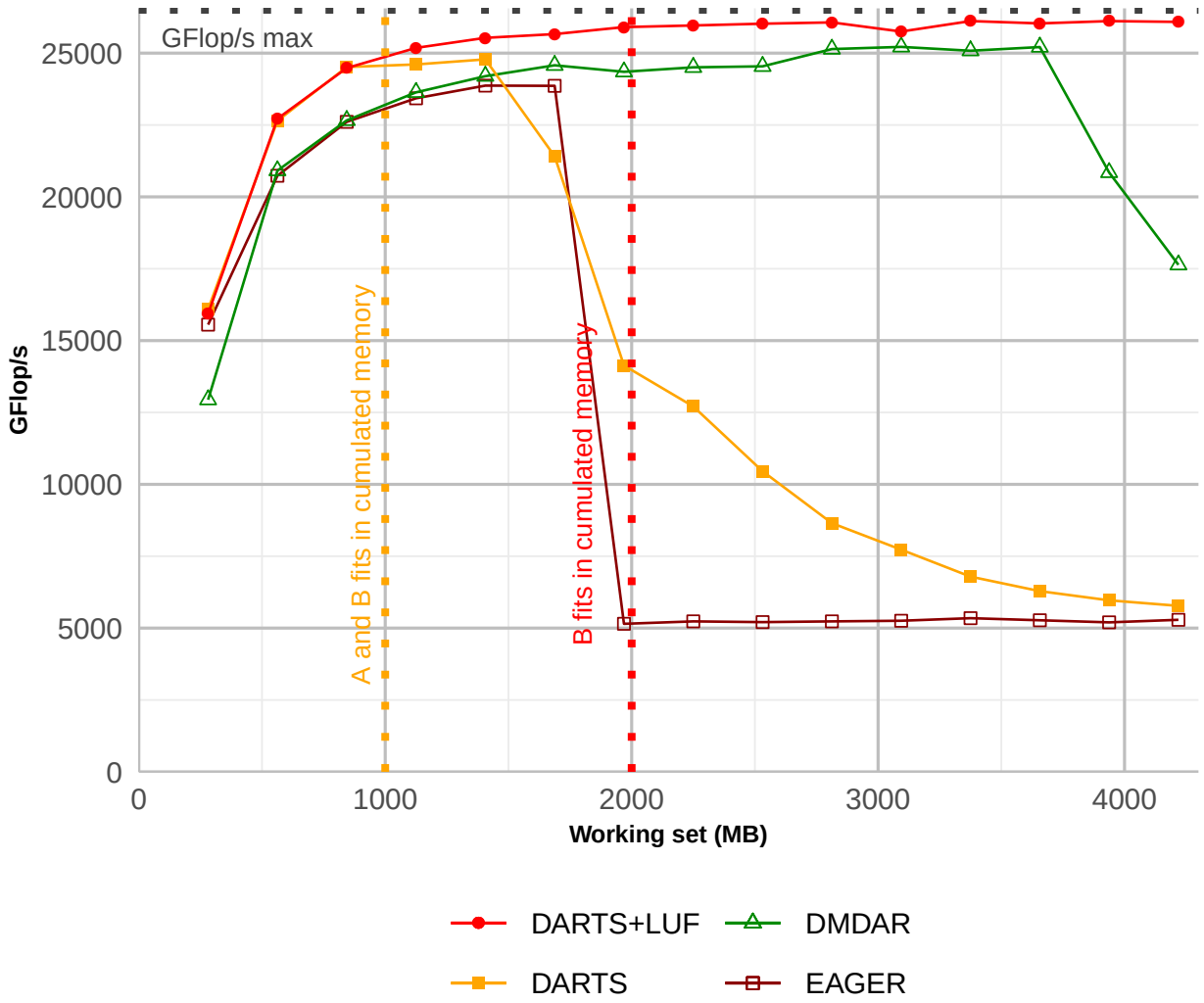
- Find $D_{opt} \in dataNotInMem_k$ such that the number of tasks depending on D_{opt} and on other data already in memory is maximum (if there is a tie: choose with transfer time, priority of the associated tasks and most total unprocessed task)
- $plannedTasks_k \leftarrow$ set of unprocessed tasks depending only on D_{opt} and on other data already in memory

Our eviction policy: LUF (Least Used in the Future)

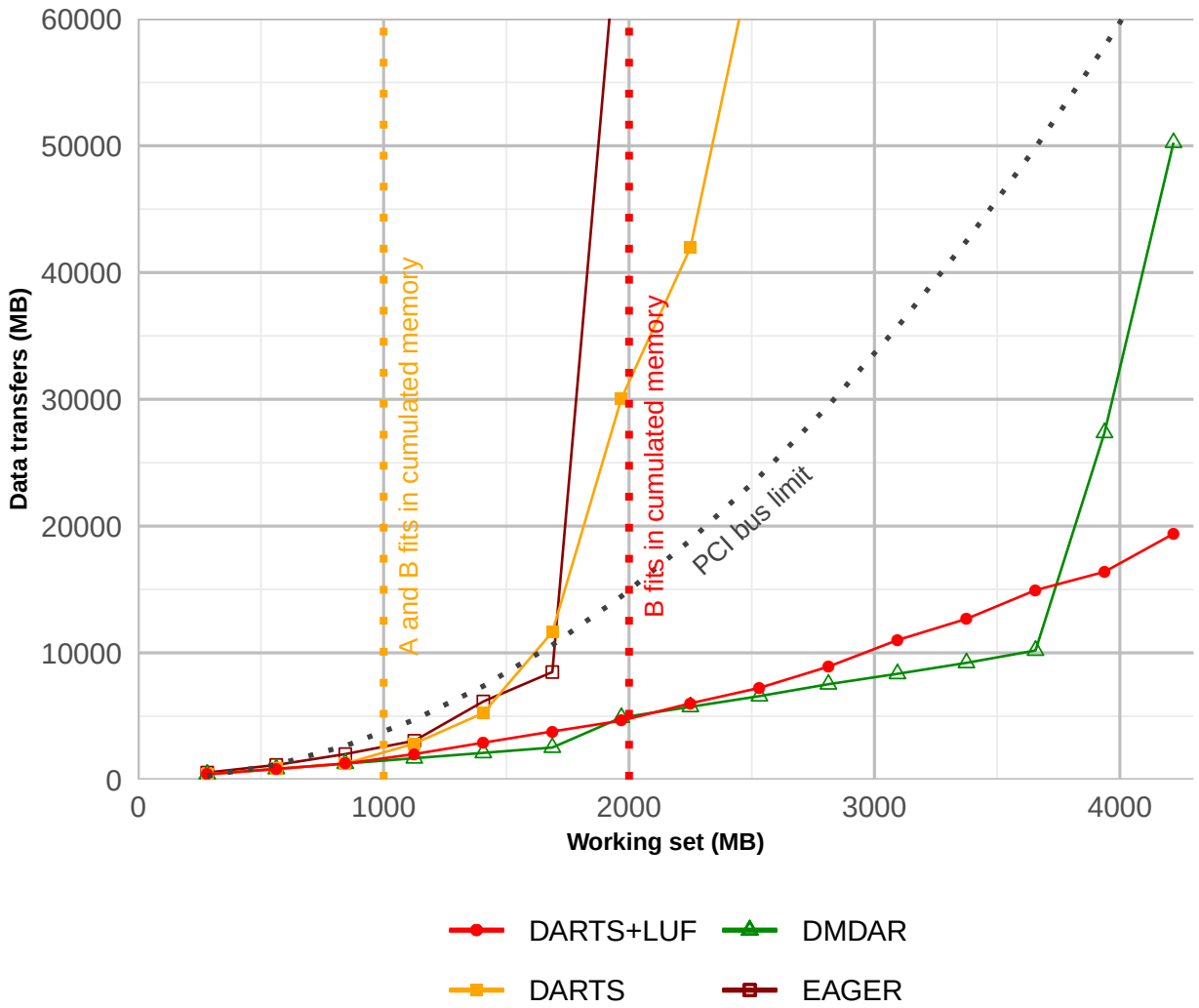
- Try to evict a data not useful for any task already in the GPU's buffer.
- Try to evict a data used by a minimal number of tasks in the set of task planned by the scheduler.

ENS DE LYON université de BORDEAUX

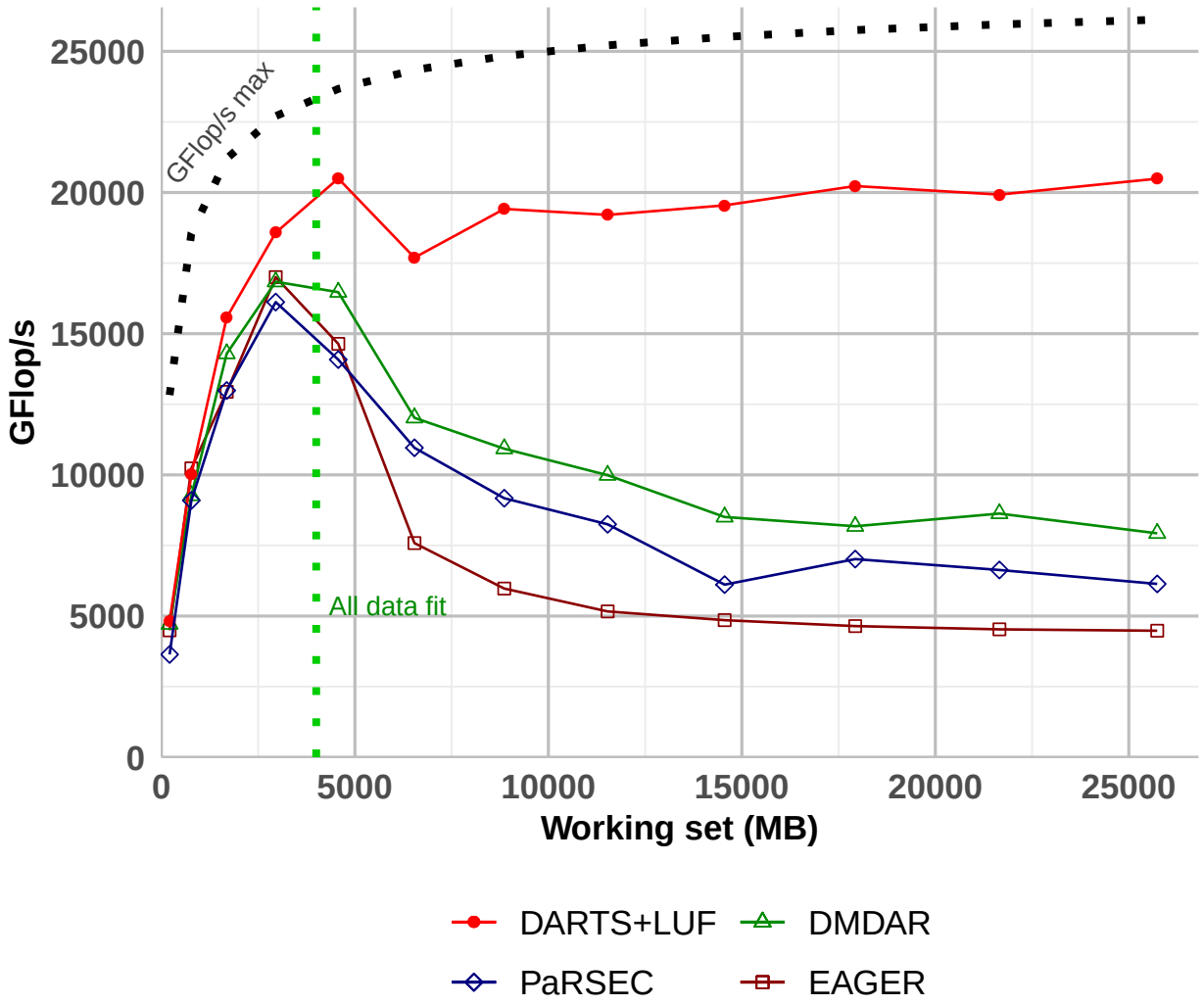
2D matrix multiplication with 2 Tesla V100 GPUs



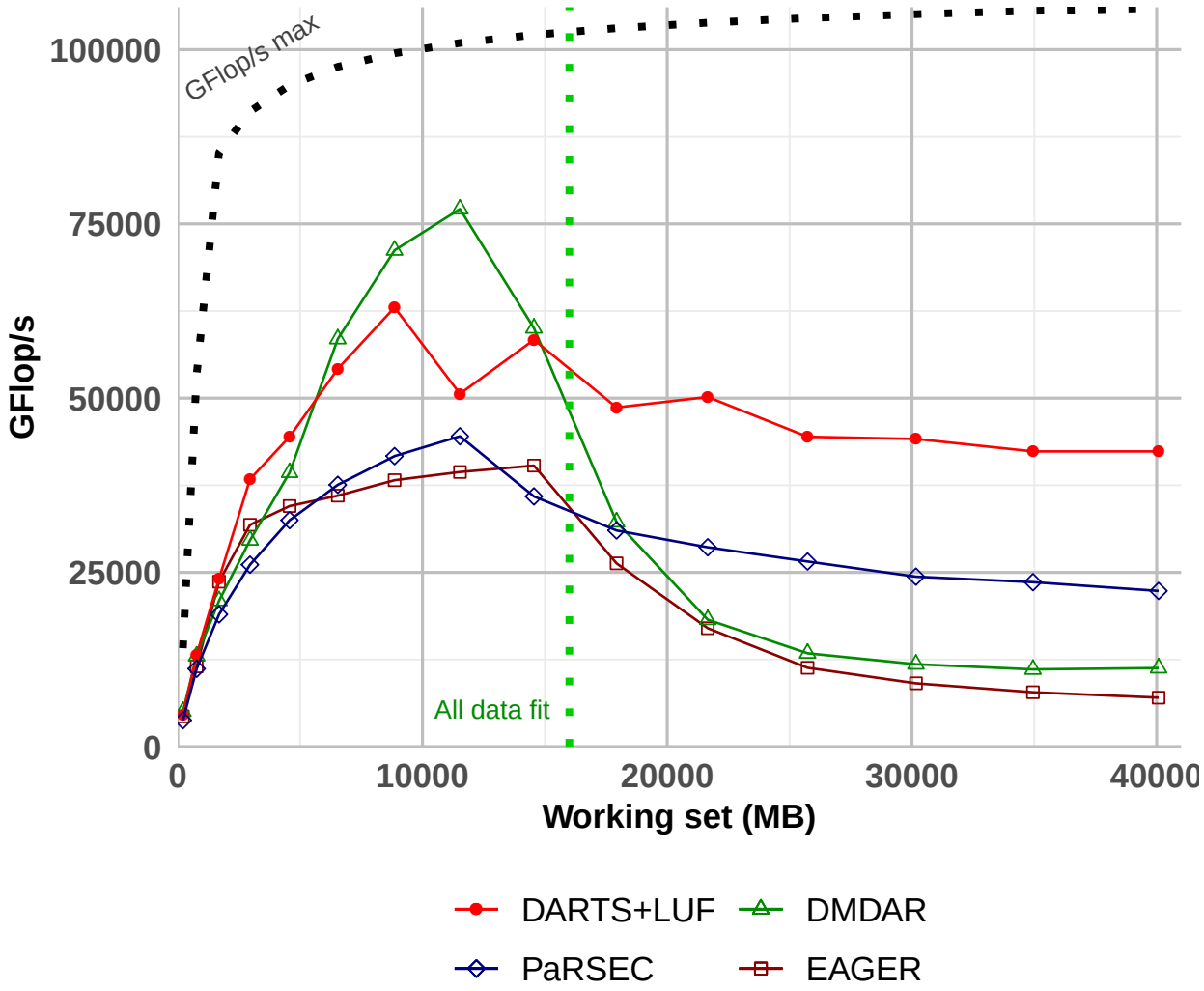
Amount of data transfers on the 2D matrix multiplication with 2 Tesla V100 GPUs



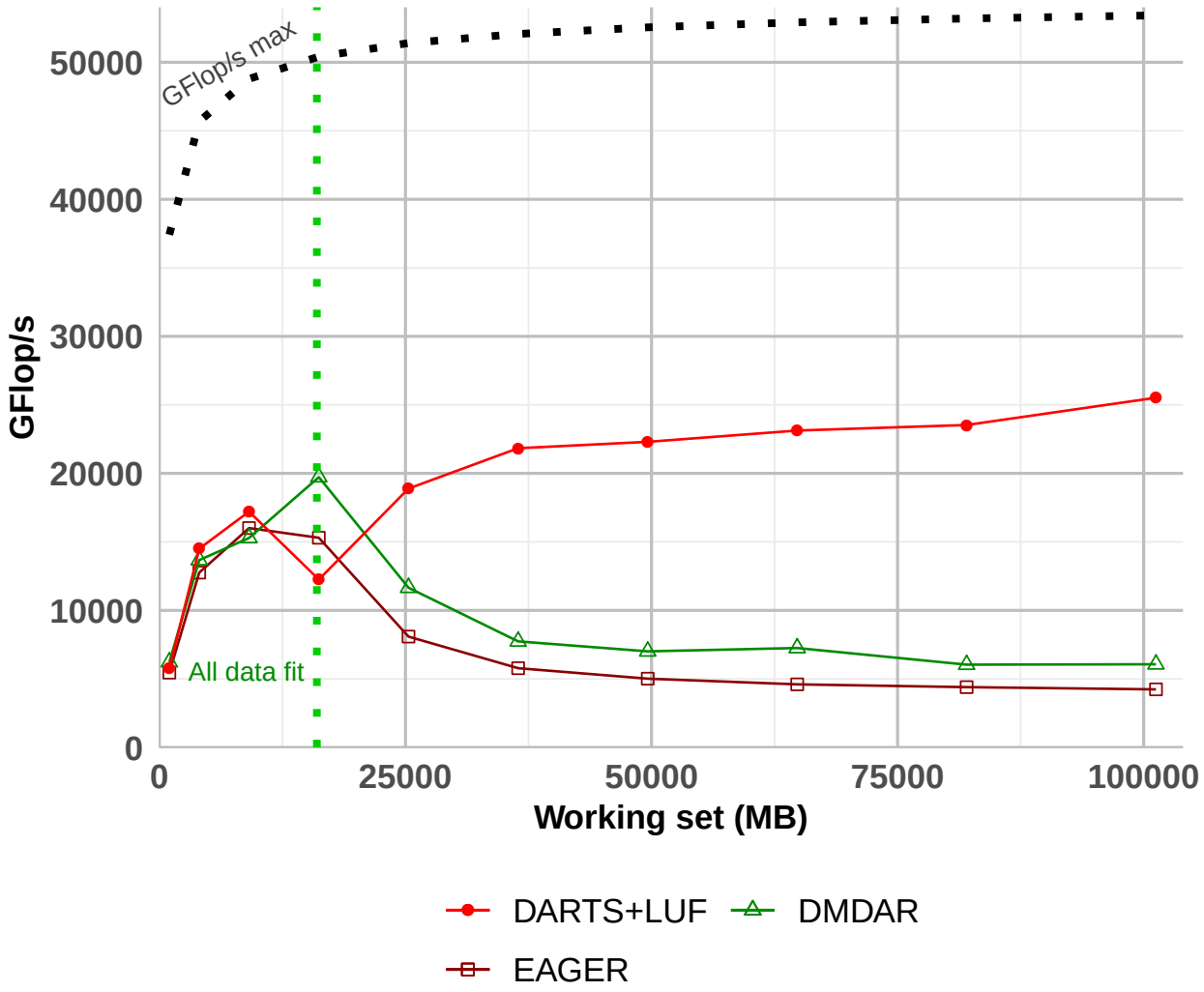
Cholesky decomposition with 2 GPUs



Cholesky decomposition with 8 GPUs



LU with 8 GPUs



Conclusion

Limiting data movements is crucial

→ **New strategy: DARTS, focused on data movement**

Tested on 2D matrix multiplication, Cholesky decomposition, LU, GEMM and sparse matrix multiplication → DARTS can achieves good performance when the memory is not constrained and always get very good performance when it is a scarce resource

Areas for improvement

- Improve computational complexity. It's holding us back to get good performances on Gemms and large matrics of the outer product
- Manage multiple MPI nodes
- **Export it to other out-of-core applications. DARTS is not specific to GPUs!**

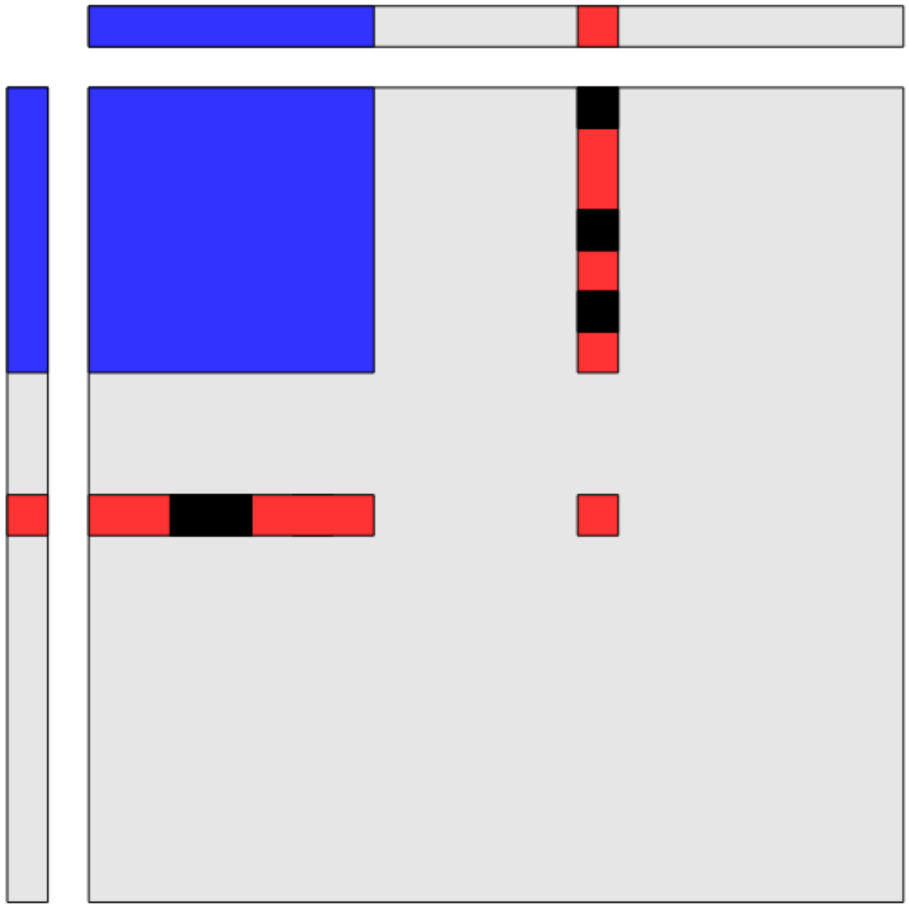
If you have an open postdoc position contact me :)

Contact:

maxime.gonthier@ens-lyon.fr
http://perso.ens-lyon.fr/maxime.gonthier/

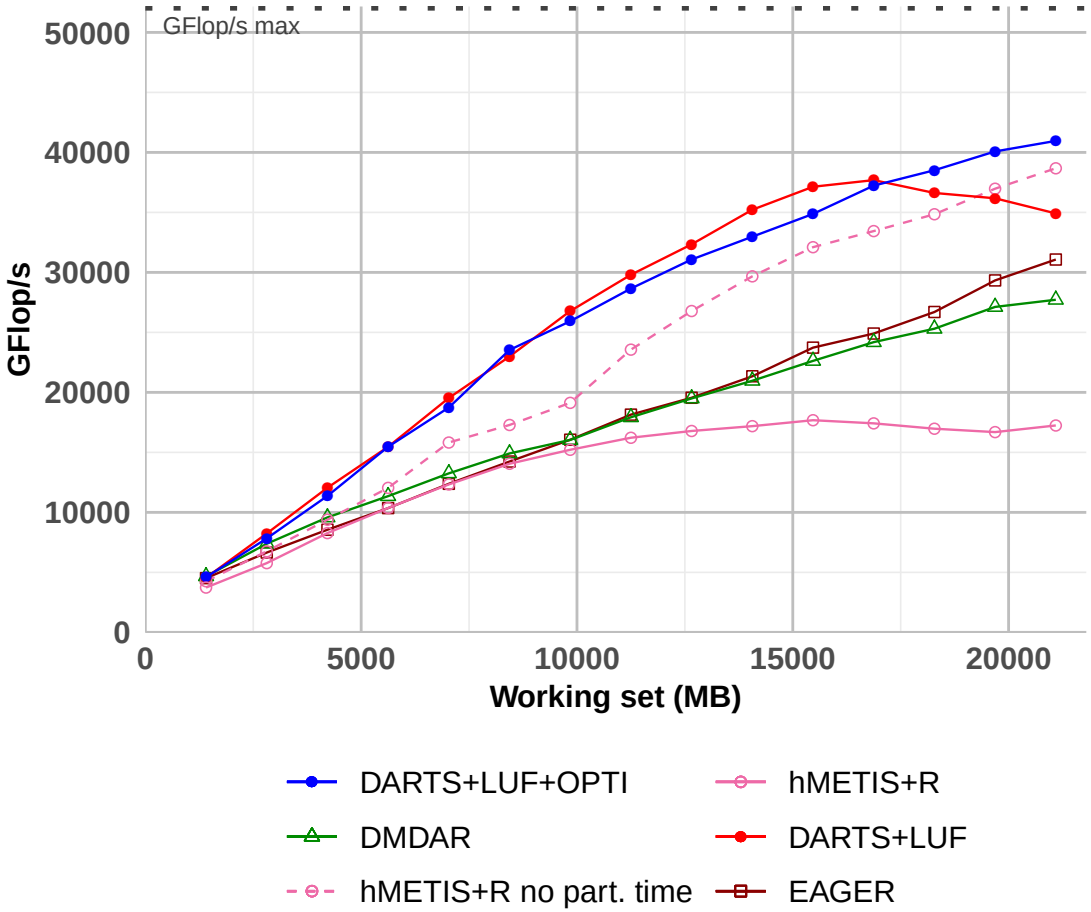
ENS DE LYON université de BORDEAUX

Joker slide



Source: Analysis of Dynamic Scheduling Strategies for Matrix Multiplication on Heterogeneous Platforms - Marchal - Beaumont

Sparse 2D matrix multiplication without memory limitation (32GB by GPU) with 4 Tesla V100 GPUs



- No memory constraint
- DARTS produces a **processing order that best distributes transfers over time**