# Hands-on tutorial on PaStiX with GPU

**11/02/2021**

Tony Delarue — Mathieu Faverge — Pierre Ramet

# 1

# Introduction

# Problem and context

**Problem: Solve $Ax = b$**

- Factorize $A = LU$, where $A$ is a sparse matrix
- Solve $Ly = b$
- Solve $Ux = y$

# Problem and context

## Problem: Solve $Ax = b$

- Factorize $A = LU$, where $A$ is a sparse matrix
- Solve $Ly = b$
- Solve $Ux = y$

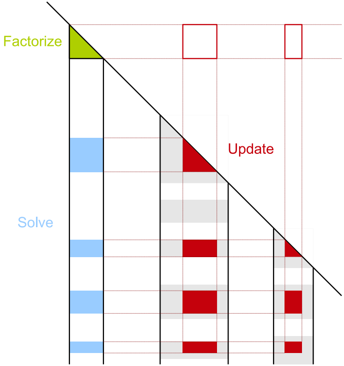
## Main steps of the sparse solver

1. Reorganize the unknows to reduce the fill-in.
2. Create the symbolic matrix $L$.
3. Factorize the matrix.
4. Solve the linear system.

# PaStiX factorization principle

## Algorithm for a column-block

1. Factorize the diagonal block (POTRF/GETRF).
2. Solve extra-diagonal blocks (TRSM).
3. Update the other column-blocks (GEMM).

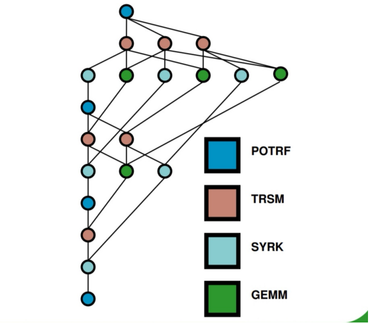


# PaStiX functionalities

## Functionalities by scheduler

| | Seq/Static/Dynamic | PaRSEC/StarPU |
|---|---|---|
| POTRF (Cholesky) | SHM/MPI/LR | SHM/MPI/LR/GPU |
| PXTRF ($LL^t$ for complex) | SHM/MPI/LR | SHM/MPI/LR/GPU |
| HETRF ($LDL^h$) | SHM/MPI/LR | SHM/MPI/LR/GPU |
| SYTRF ($LDL^t$) | SHM/MPI/LR | SHM/MPI/LR/GPU |
| GETRF ($LU$) | SHM/MPI/LR | SHM/MPI/LR/GPU |

# Runtime presentation




## StarPU et PaRSEC

- Create a task diagram. It allow us to anticipate dependencies between tasks.
- Share the datas on the different computation devices.
- Take care of computer heterogeneity.

# 2

## Obtaining better peformances with PaStiX-GPU

# Obtaining better peformances - models

## Define your performance model

- PaStiX allows you to define you performance model.
- POTRF/GETRF time kernel estimation:
  $a3 * N^3 + a2 * N^2 + a1 * N + a0$
- TRSM time kernel estimation:
  $a5 * M * N^2 + a4 * M * N + a3 * N^2 + a2 * M + a1 * N + a0$
- GEMM time kernel estimation:
  $a7 * M * N * K + a6 * M * K + a5 * K * N + a4 * M * N + a3 * M + a2 * N + a1 * K + a0$
- Need to be coherent with your hardware.
- Default value : $a7 = 2./1.2e12$

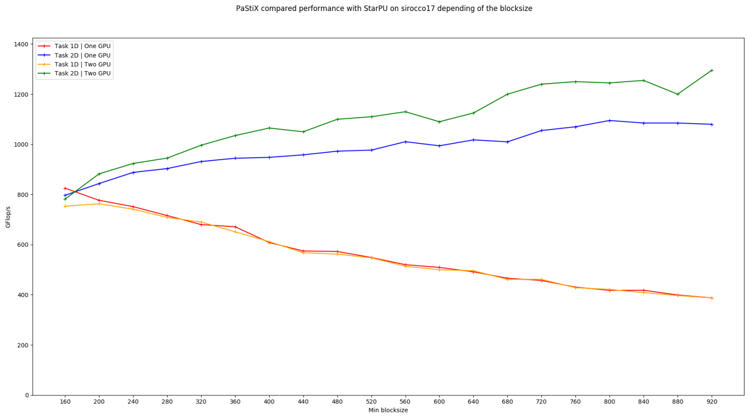
# Obtaining better peformances - granularity

## 1D or 2D task

We can play with the granularity of the computation tasks.

- 1D if we consider a block-column.
- 2D if we consider only blocks.

# Obtaining better peformances - granularity



# Obtaining better peformances - PaRSEC

## PaRSEC

In your home directory, you can create a $HOME/parsec/mca-params.conf file to better configure PaRSEC.

- device_show_capabilities = 1
- device_show_statistics = 1
- device_cuda_max_streams = 10 # $\geq 3$ #
- device_cuda_max_events_per_stream = 4
- runtime_comm_short_limit = 0

# Obtaining better performances - STARPU

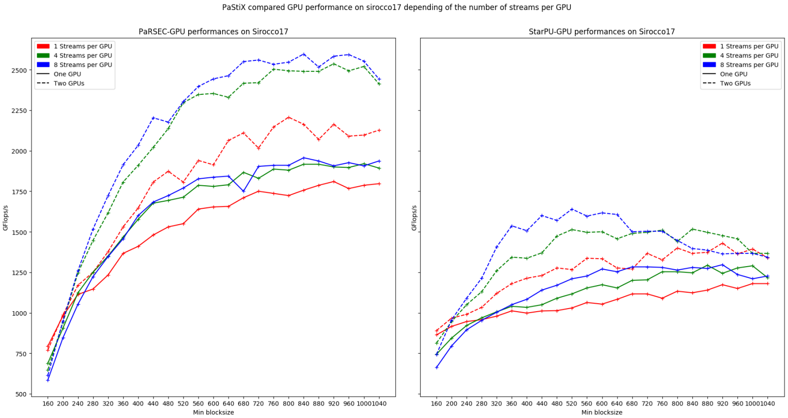
## STARPU

STARPU contains a set of environment variables to define its comportement with GPUs:

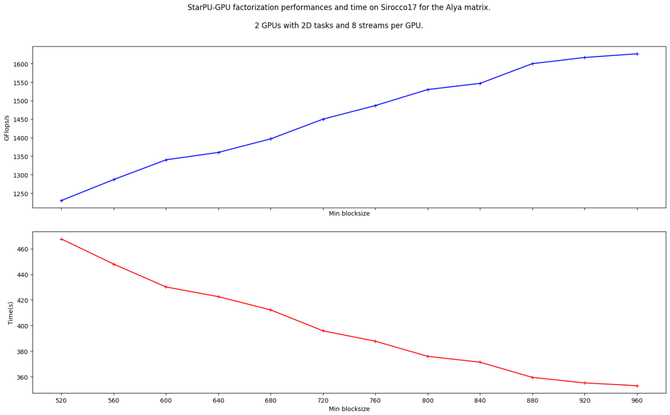
- STARPU_CUDA_PIPELINE=4
- STARPU_NWORKER_PER_CUDA=8
- STARPU_CUDA_THREAD_PER_WORKER= $[0||1]$

You can either export them or call them at the beginning of your command line.

# Obtaining better performances - Experiments



# Experiments with EoCoE matrix



# 3

# Conclusion

# Conclusion

## Conclusion

- Creation of user tutorials for PASTIX-GPU use.
- Highligh the parameters to look at according to your GPU.
- Give performance results of EoCoE matrices.

## Futur works

- Make PASTIX-GPU scale with PARSEC.
- Understand the gap of performances between PARSEC and STARPU.
- Improve PASTIX-MPI implementation with runtimes to be efficient with GPUs.

**Merci pour votre attention !**