Programming heterogeneous architectures using divisible tasks

Mathieu Faverge, Nathalie Furmento, Abdou Guermouche, Gwenolé Lucas, Raymond Namyst, Pierre-André Wacrenier

December 2020

Plan

Introduction

Context

- Task-based programming.
- Sequential Task Flow model.

Introduction

Context

- Task-based programming.
- Sequential Task Flow model.

Some Limitations

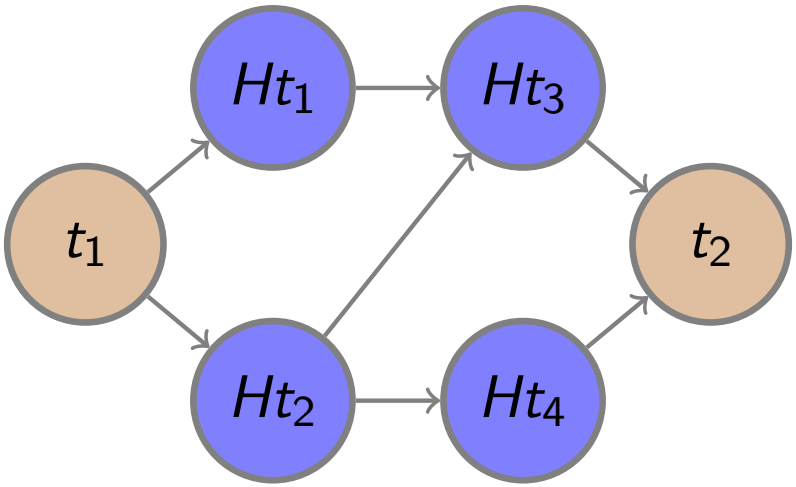
- Requires the submission of a static graph at the start of the application.
- The sequential insertion of the tasks can lead to a large overhead.

Introduction

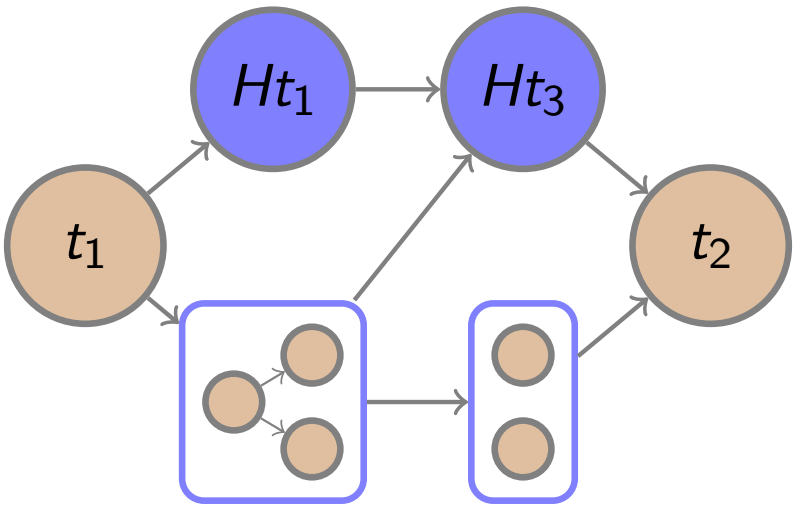
Objectives

- Allow StarPU to dynamically adapt the granularity in order to satisfy the different cores of a heterogeneous architecture.
- Select task implementation dynamically.
- Submit less tasks at startup.
- Insert tasks in parallel.

Hierarchical Tasks

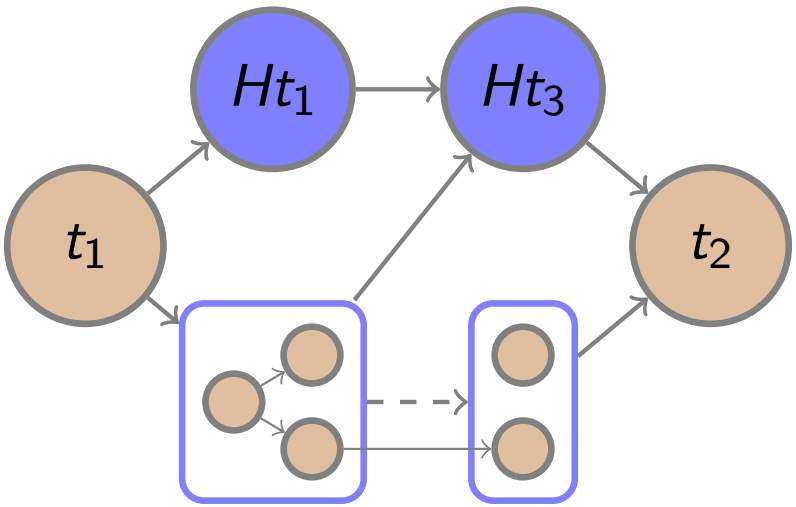


Hierarchical Tasks



- When executed, a hierarchical task can insert a subgraph.

Hierarchical Tasks



- When executed, a hierarchical task can insert a subgraph.
- The dependencies in the subgraphs are infered from the datas.

Application to Dense Linear Algebra

To validate this model, we applied it to Chameleon's Cholesky factorisation DAG.

Early Simplifications

Application to Dense Linear Algebra

To validate this model, we applied it to Chameleon's Cholesky factorisation DAG.

Early Simplifications

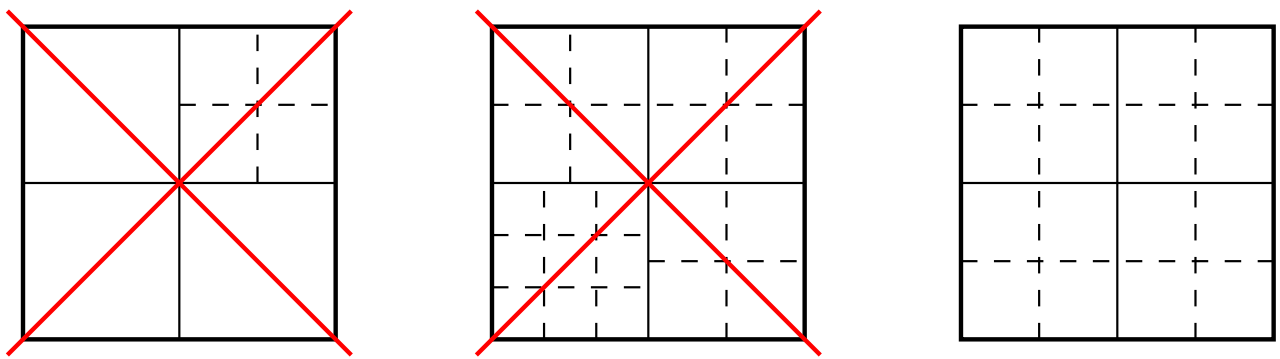
- The matrix partitioning is decided beforehand and not dynamically.

Application to Dense Linear Algebra

To validate this model, we applied it to Chameleon's Cholesky factorisation DAG.

Early Simplifications

- The matrix partitioning is decided beforehand and not dynamically.
- It is also uniform, each tile must be split in the same way.



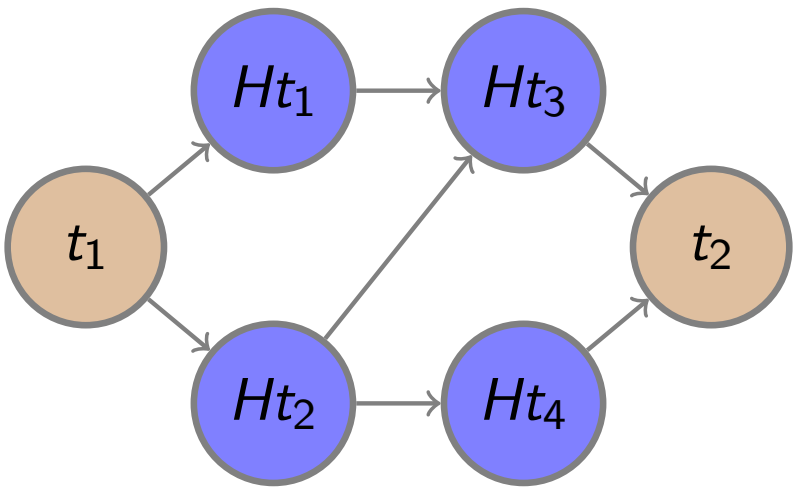
Updating Chameleon Kernels

Recursive Algorithm

- During the initial DAG submission, each task is a hierarchical task.
- The behaviour of these tasks at runtime depends on the tiles they received :
  - If the tiles are partitioned, the hierarchical task submit a subgraph of the kernel operation.
  - Else, the kernel is executed as intended.
- The subgraphs follow the same rules as the initial DAG.

Parallel Task Insert

- In StarPU, all tasks are submitted by a single thread, which can lead to a significant overhead when dealing with applications using a large number of tasks.
- With hierarchical tasks, we can reduce the amount of tasks initially submitted and insert subgraph-tasks in parallel (for example, Ht_1 and Ht_2 below), decreasing this overhead.



Benchmarks

The tests are running on PlaFRIM's `bora` nodes :

- 2x 18-core Cascade Lake Intel® Xeon® Skylake Gold 6240 @ 2,6 GHz.
- 5.3 Go of memory per core (@2933 MHz).

Labelling

The labels on the curves correspond to the partitioning of the matrix in the experiment, by stating the level of partitioning followed by the size of the tiles on each level.

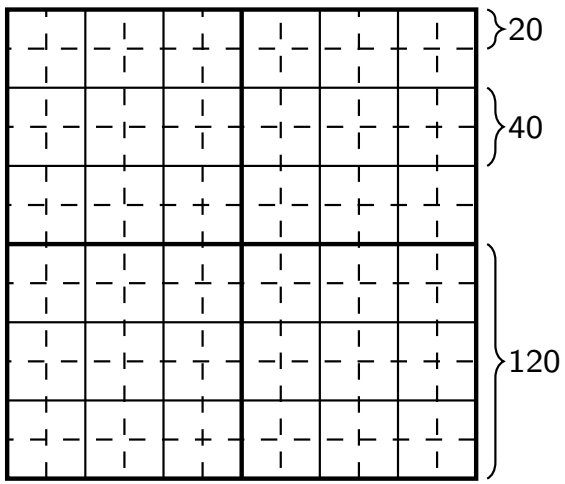


Figure 1: Partitioning matching the label `3lvl-120:40:20`

First results on `bora` nodes

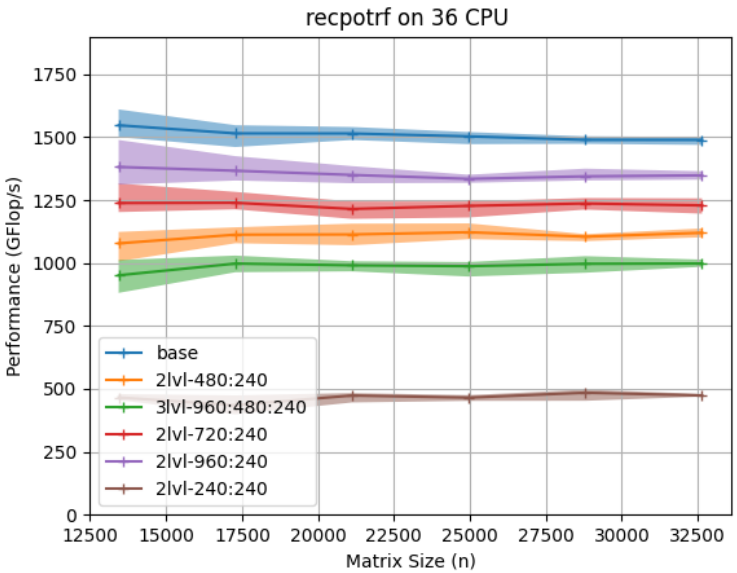


Figure 2: Without parallel insertion

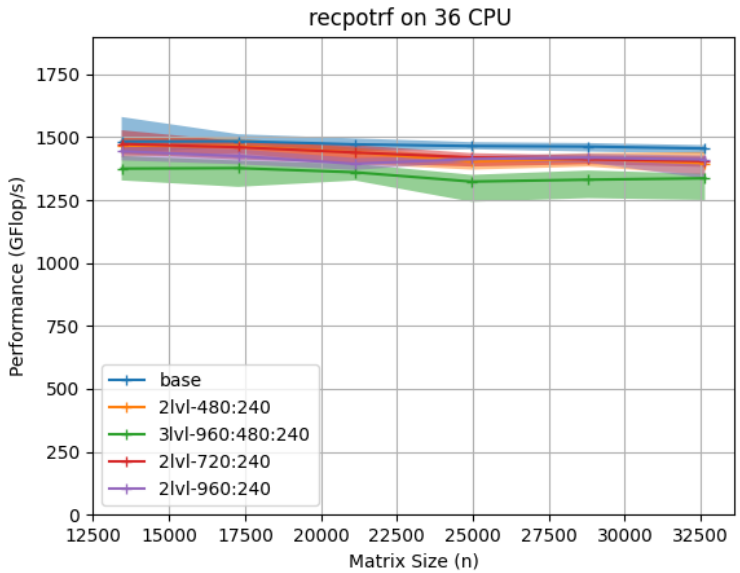


Figure 3: With parallel insertion (potrf only)

Parallel insertion impact

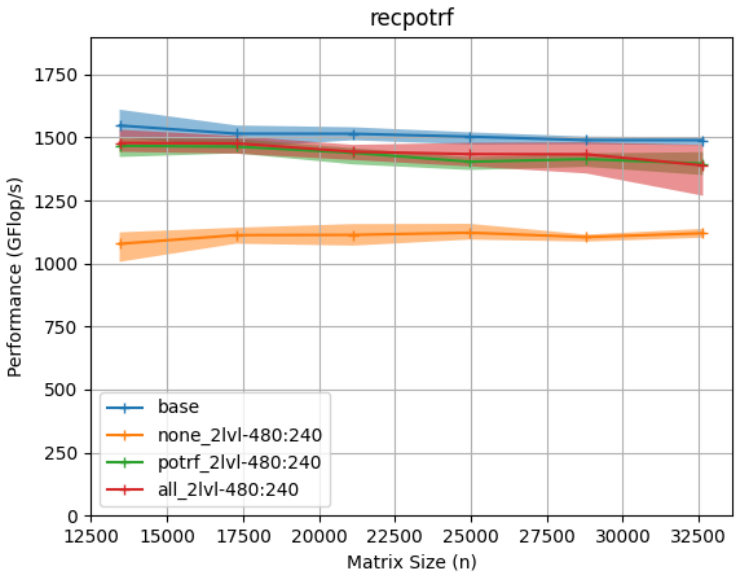


Figure 4: Tiles partitionned in 4

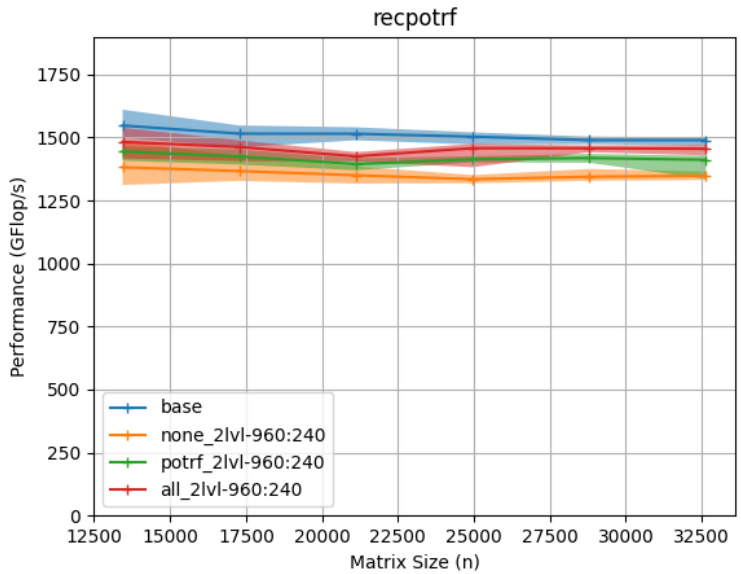


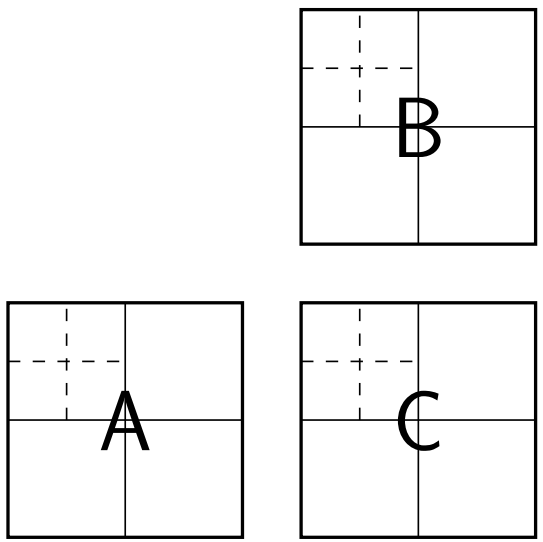
Figure 5: Tiles partitionned in 16

Automatic Partitioning

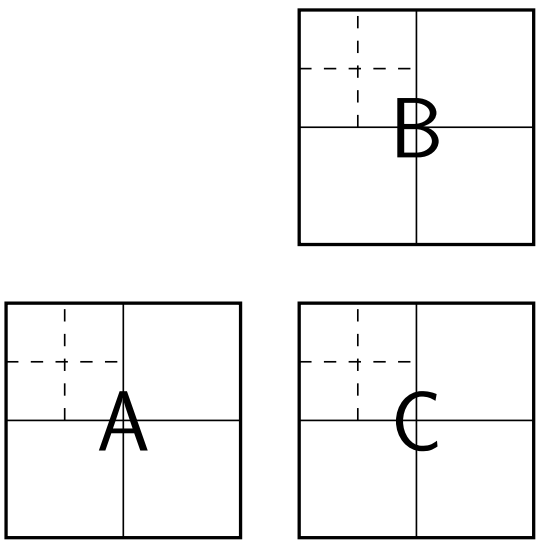
In order to extend our method to the use of GPUs, we need to take into account data partitioning.

More linear algebra : matrix multiplication example

$$C = C + \alpha.A \times \beta.B$$

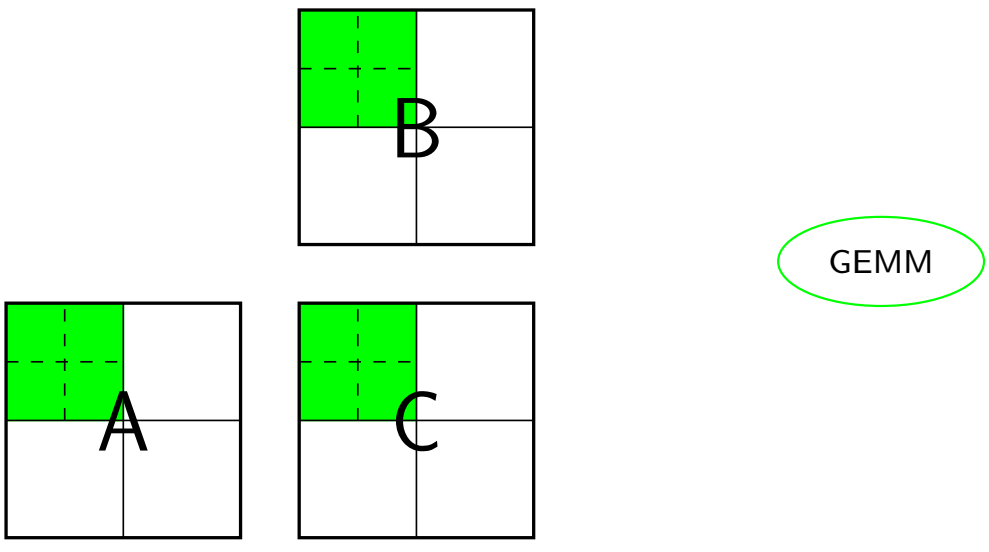


Automatic Partitioning - Example



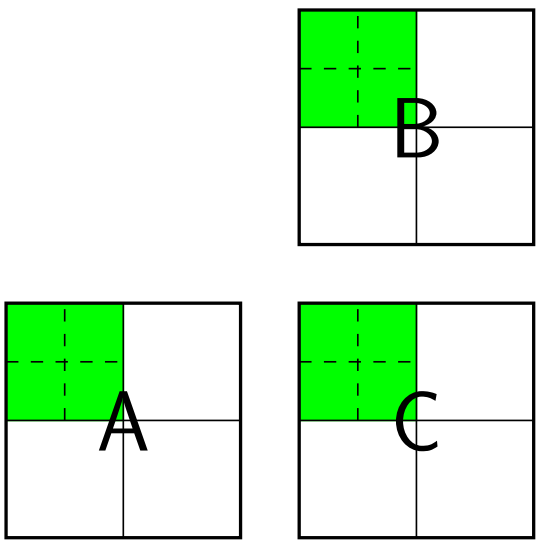
Automatic Partitioning - Example

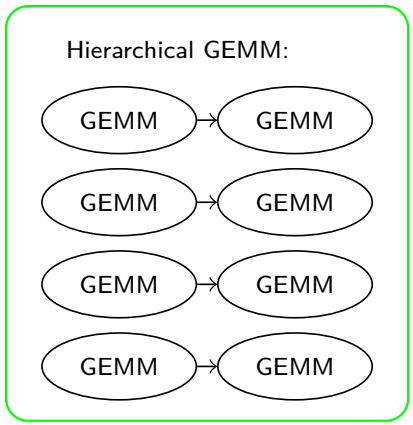
All tiles are partitioned. . .



Automatic Partitioning - Example

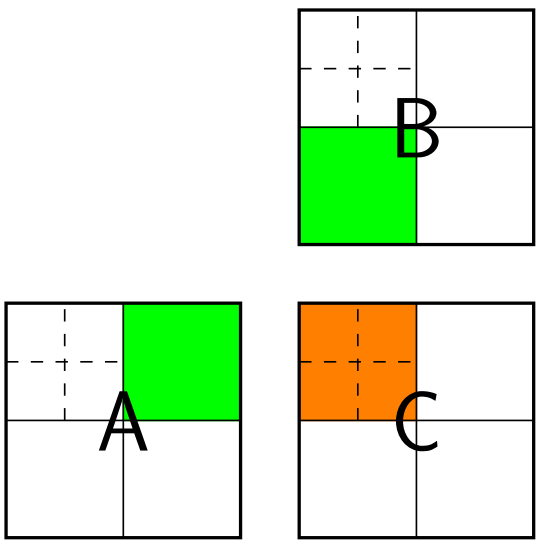
. . . The hierarchical task insert a subgraph.

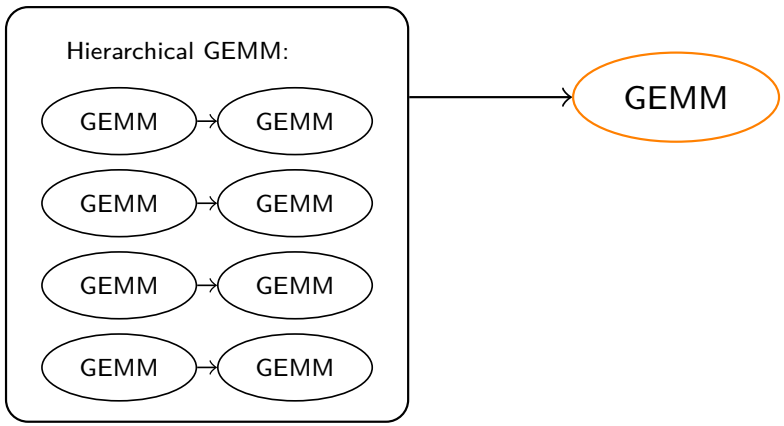




Automatic Partitioning - Example

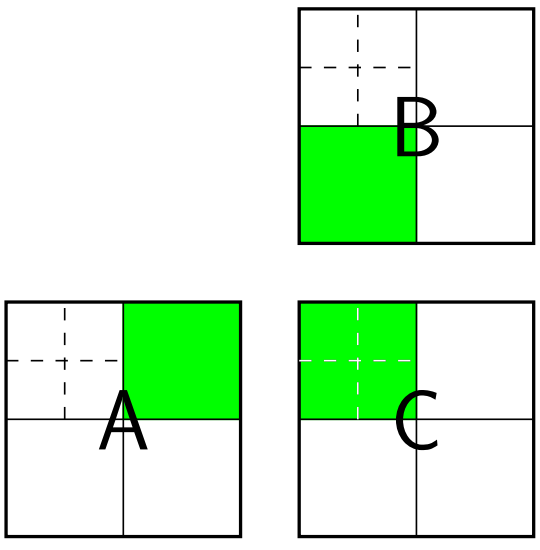
A single tile is partitioned...

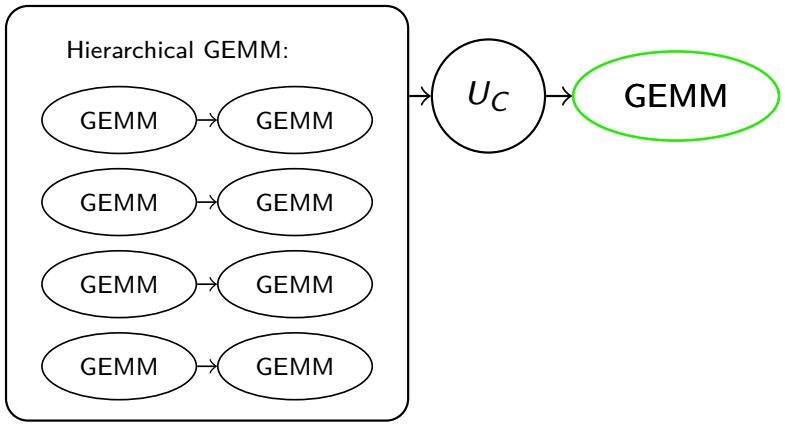




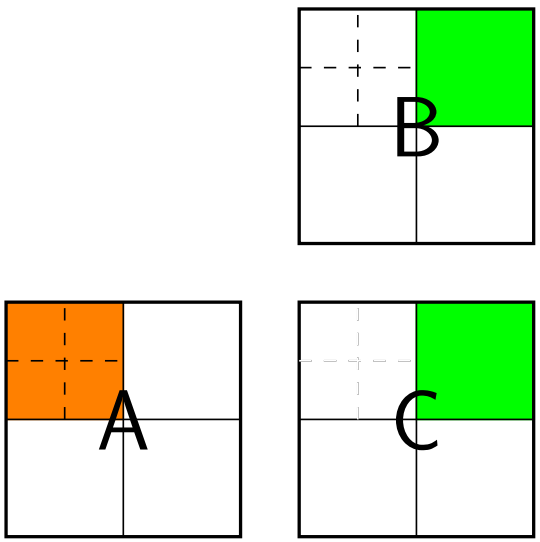
Automatic Partitioning - Example

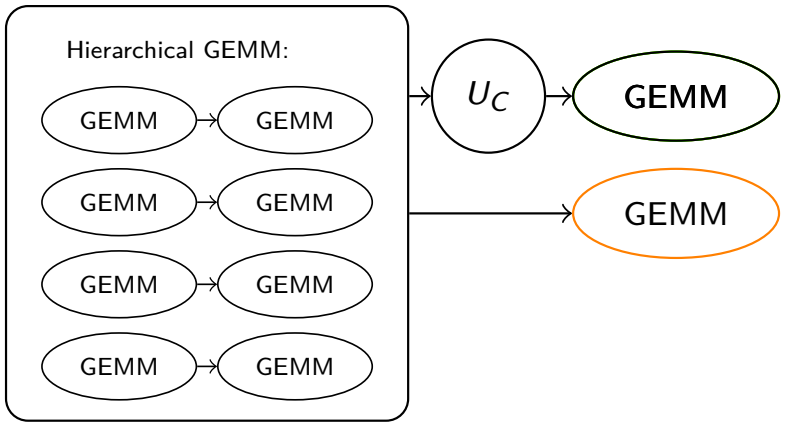
How to insert this U_C ?





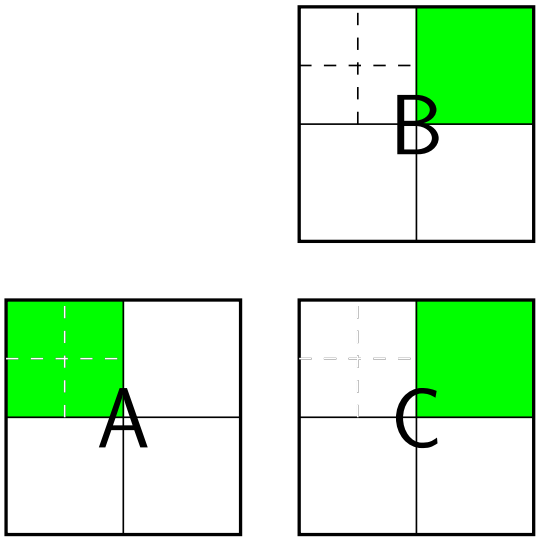
Automatic Partitioning - Example

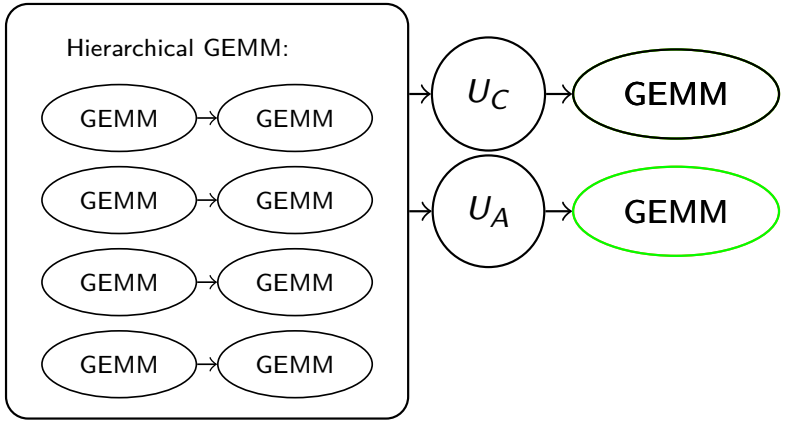




Automatic Partitioning - Example

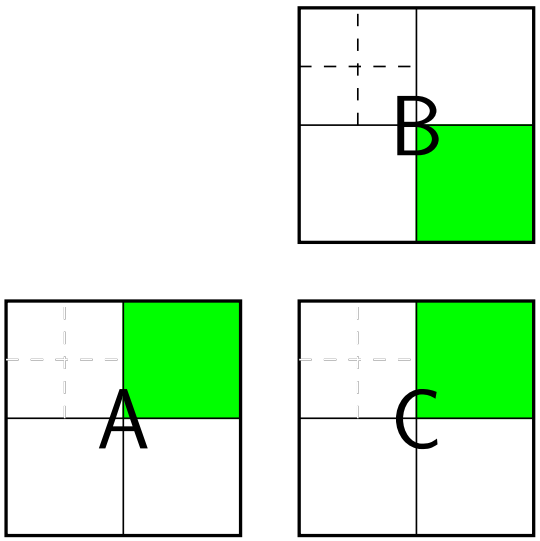
When a task use both partitioned and not-partitioned datas, StarPU automatically insert the unpartition tasks needed.

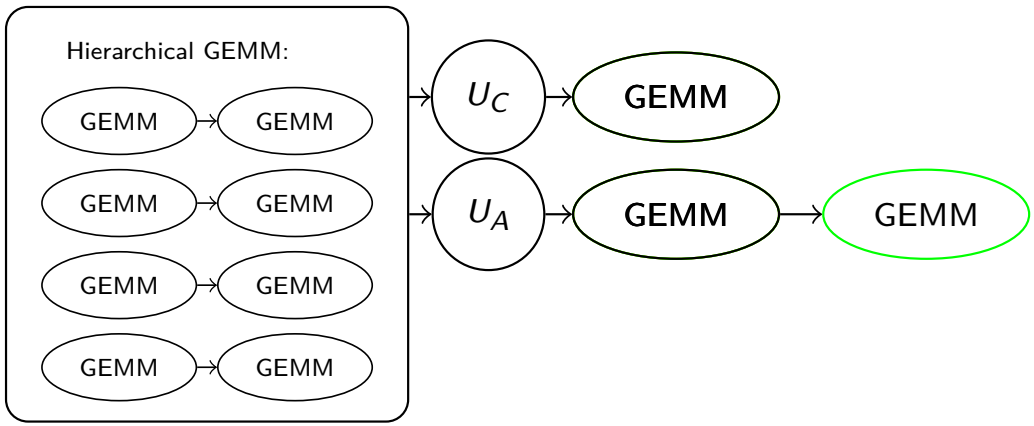




Automatic Partitioning - Example

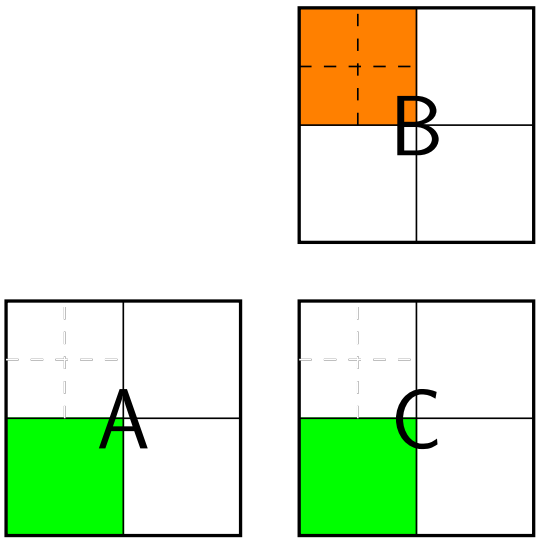
When a task use both partitioned and not-partitioned datas, StarPU automatically insert the unpartition tasks needed.

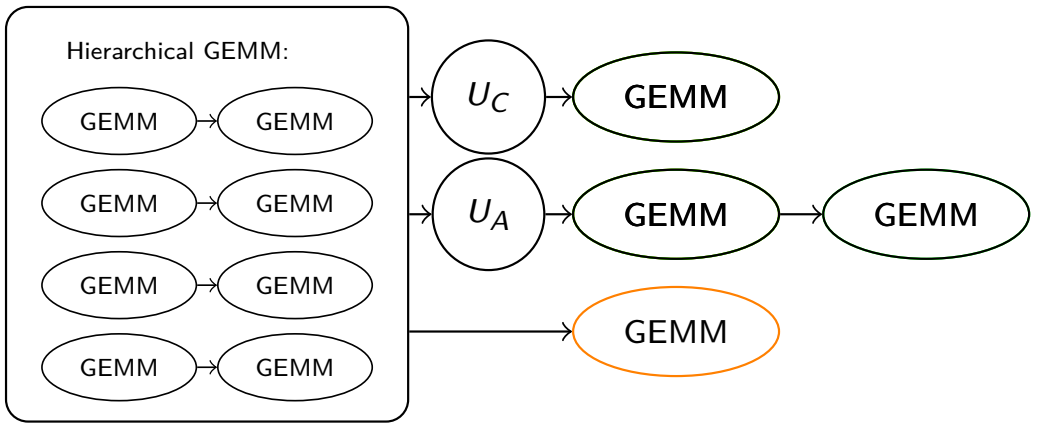




Automatic Partitioning - Example

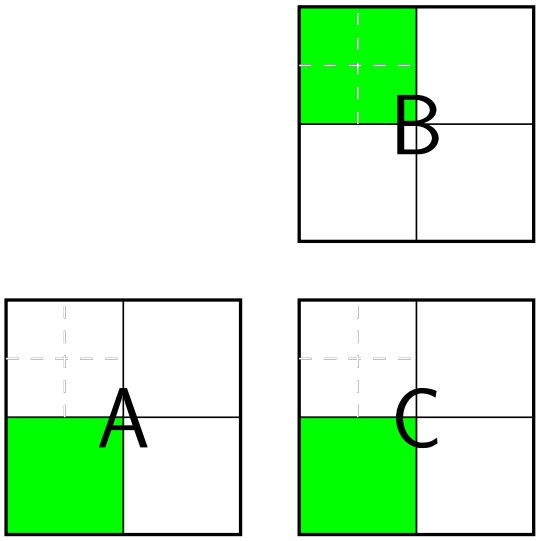
When a task use both partitioned and not-partitioned datas, StarPU automatically insert the unpartition tasks needed.

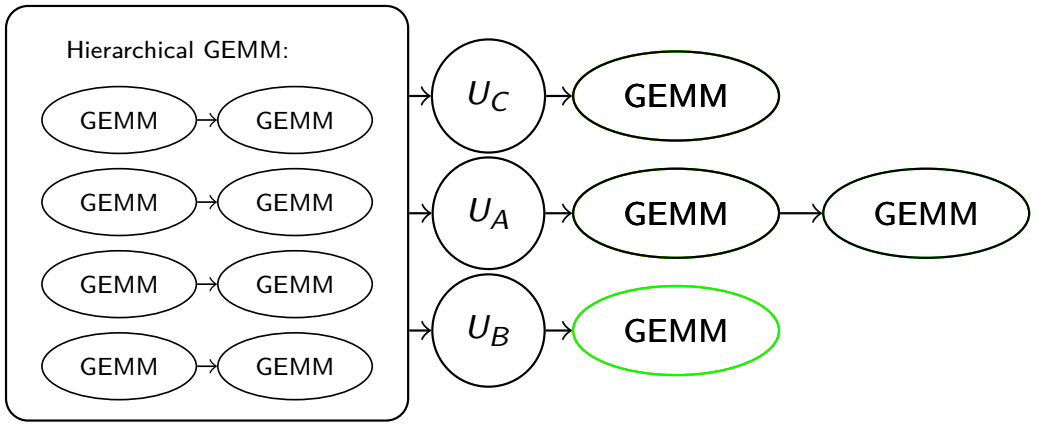




Automatic Partitioning - Example

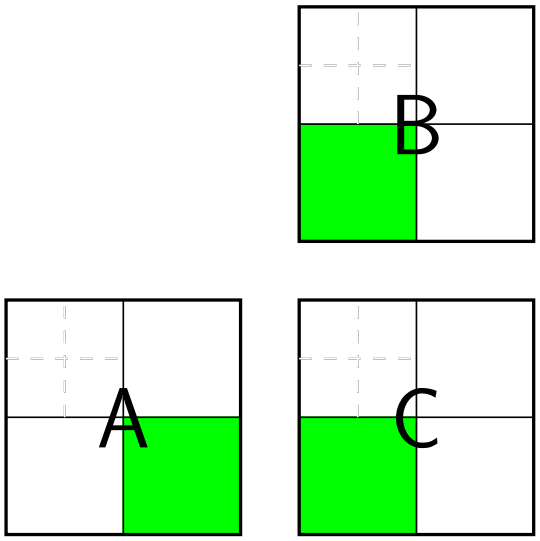
When a task use both partitioned and not-partitioned datas, StarPU automatically insert the unpartition tasks needed.

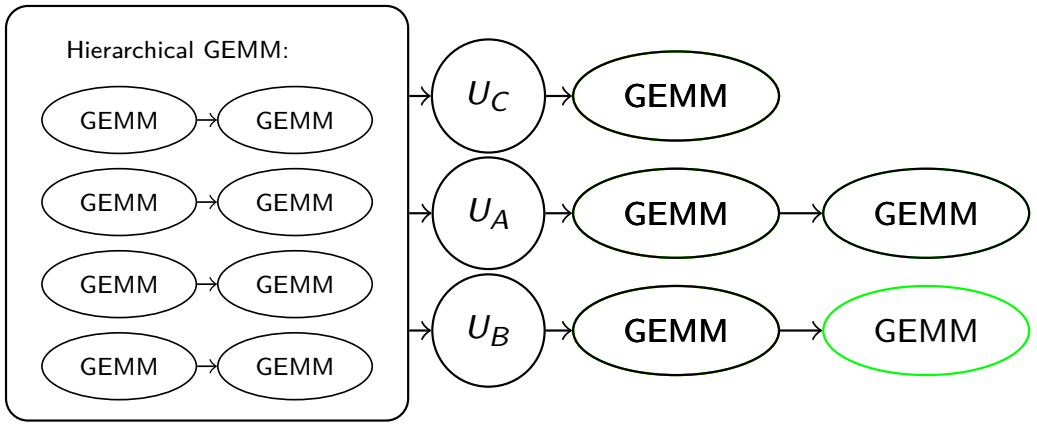




Automatic Partitioning - Example

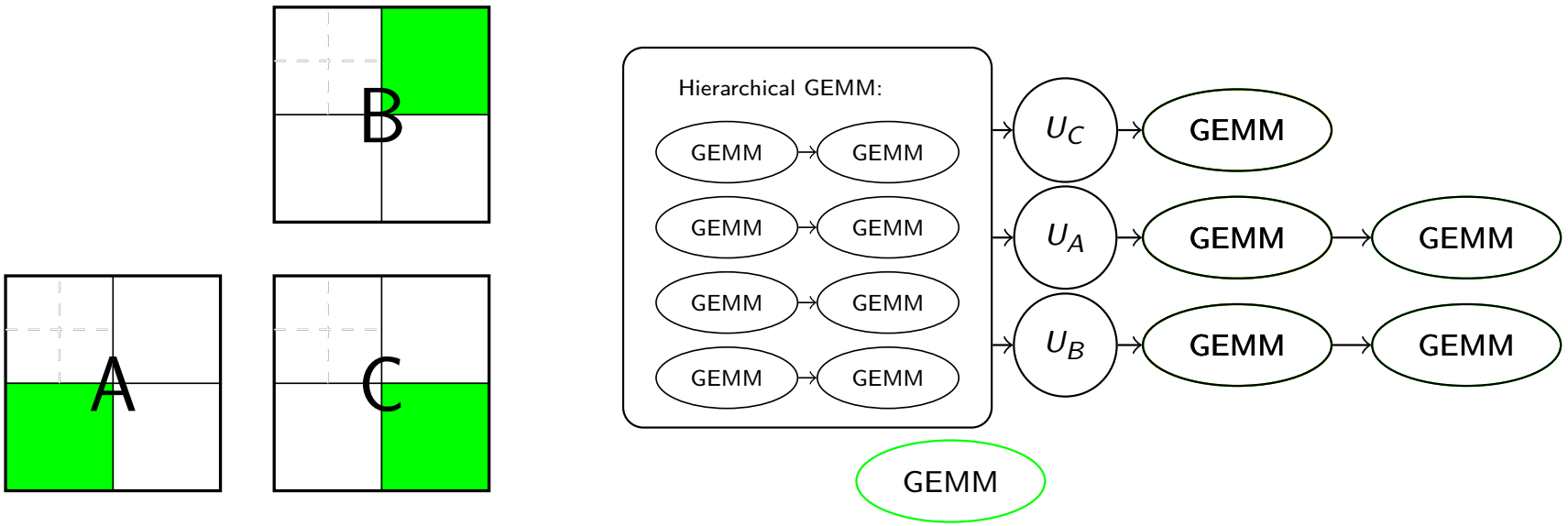
When a task use both partitioned and not-partitioned datas, StarPU automatically insert the unpartition tasks needed.





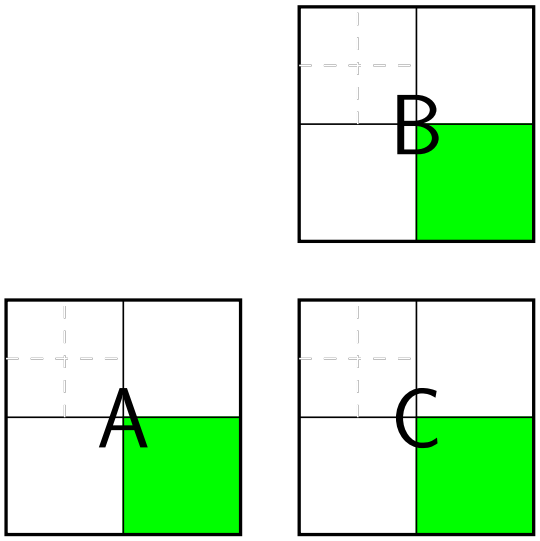
Automatic Partitioning - Example

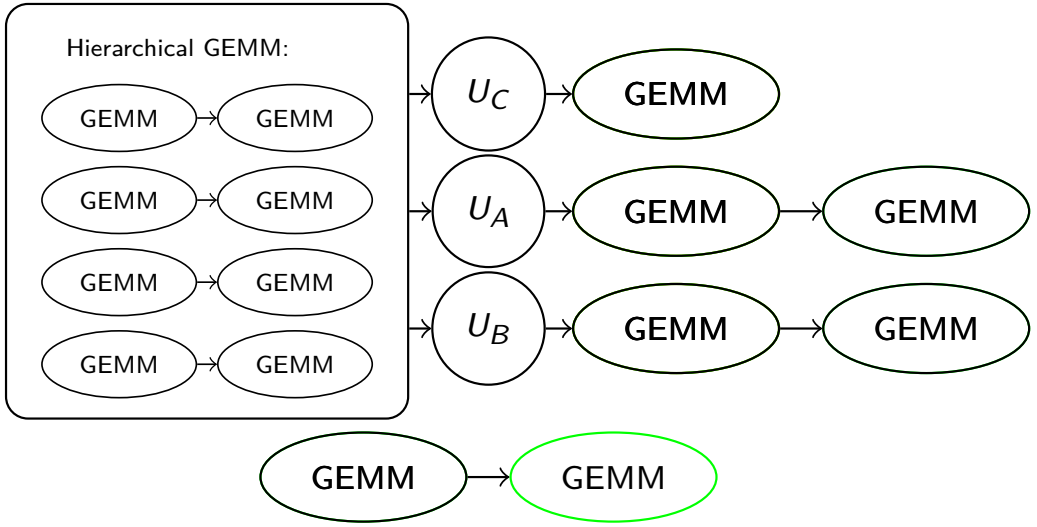
When a task use both partitioned and not-partitioned datas, StarPU automatically insert the unpartition tasks needed.



Automatic Partitioning - Example

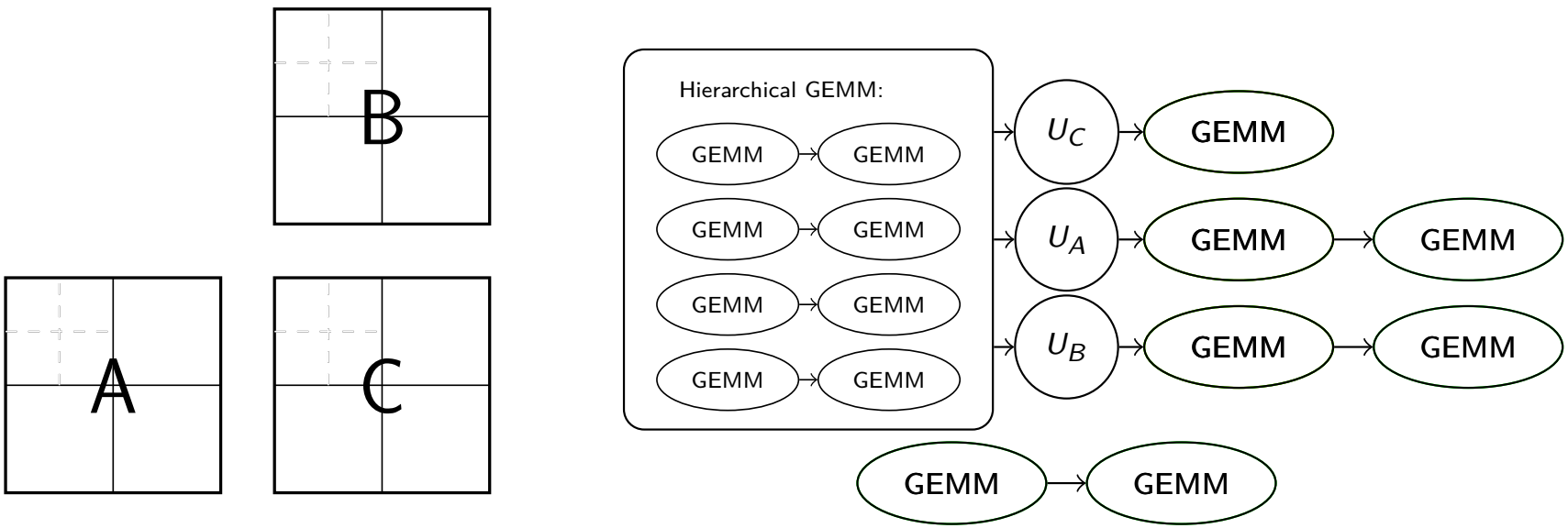
When a task use both partitioned and not-partitioned datas, StarPU automatically insert the unpartition tasks needed.





Automatic Partitioning - Example

When a task use both partitioned and not-partitioned datas, StarPU automatically insert the unpartition tasks needed.



Automatic Partitioning - Rules

- Both partition and unpartition tasks are automatically inserted.

Automatic Partitioning - Rules

- Both partition and unpartition tasks are automatically inserted.
- Unpartitions are added before regular tasks, if needed.

Automatic Partitioning - Rules

- Both partition and unpartition tasks are automatically inserted.
- Unpartitions are added before regular tasks, if needed.
- Partitions are added before hierarchical tasks, if needed.

Automatic Partitioning - Rules

- Both partition and unpartition tasks are automatically inserted.
- Unpartitions are added before regular tasks, if needed.
- Partitions are added before hierarchical tasks, if needed.
- All subgraphs must use subdatas of the data the parent hierarchical task was given.

Conclusion

- The model has been validated using Cholesky factorisation.
- Parallel insertion can help reduce the overhead of hierarchical tasks.
- Data partitioning with hierarchical tasks can be automated.

Future Work

Make the code more robust.

Future Work

Make the code more robust.

Scheduling

- When should we insert a subgraph ?
- Where should we execute it ?
- Using what implementation ?

We need to adapt scheduling strategies to answer those questions.

Future Work

Make the code more robust.

Scheduling

- When should we insert a subgraph ?
- Where should we execute it ?
- Using what implementation ?

We need to adapt scheduling strategies to answer those questions.

Testing other applications

Sparse Solvers, Low Rank Approximation, ...

Thank You!

Any questions ?