

# Graph partitioning techniques for load balancing of coupled simulations

Maria Predari & Aurélien Esnard (Inria, University of Bordeaux, France)

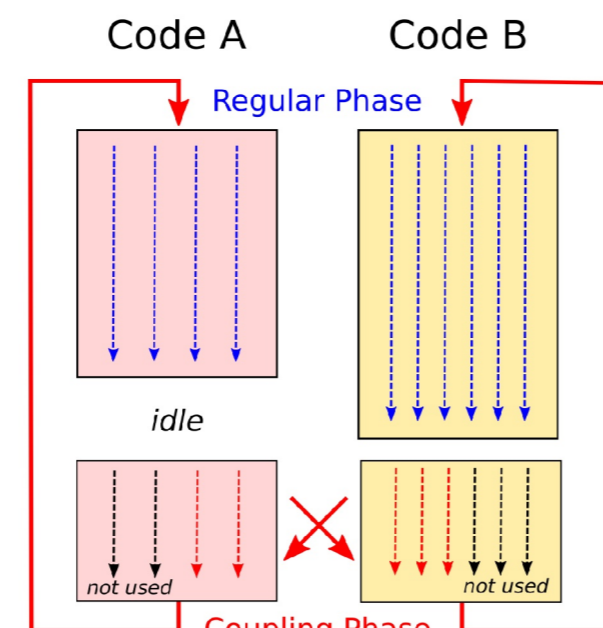
## Introduction

**Context** – In the field of parallel computing, the load balancing is a crucial issue which determines the performance of parallel applications. A very common approach to solve the load-balancing problem is based on graph model. To equilibrate the load between  $k$  processors and minimize the communication cost, one performs a *graph partitioning*, each part being assigned to a given processor.

**Challenges** – Multi-physics or multi-scales applications that emerge nowadays in real-life problems often have more complex structure and consist of numerous parallel codes, running concurrently and periodically coupled together.

**Issue** – The traditional load balancing strategy is not adequate. The naive partitioning of each code independently will not balance the coupling phase, leading to bad inter-codes exchange.

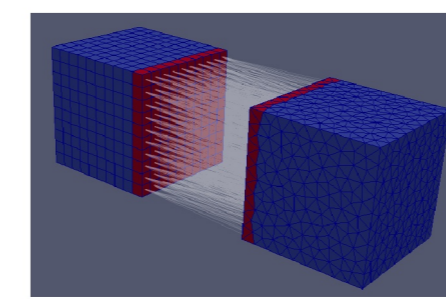
**Our goal** - Finding a *coupling-aware* partitioning that takes into account the coupling phase explicitly as a trade-off to the partitioning during the regular phase.



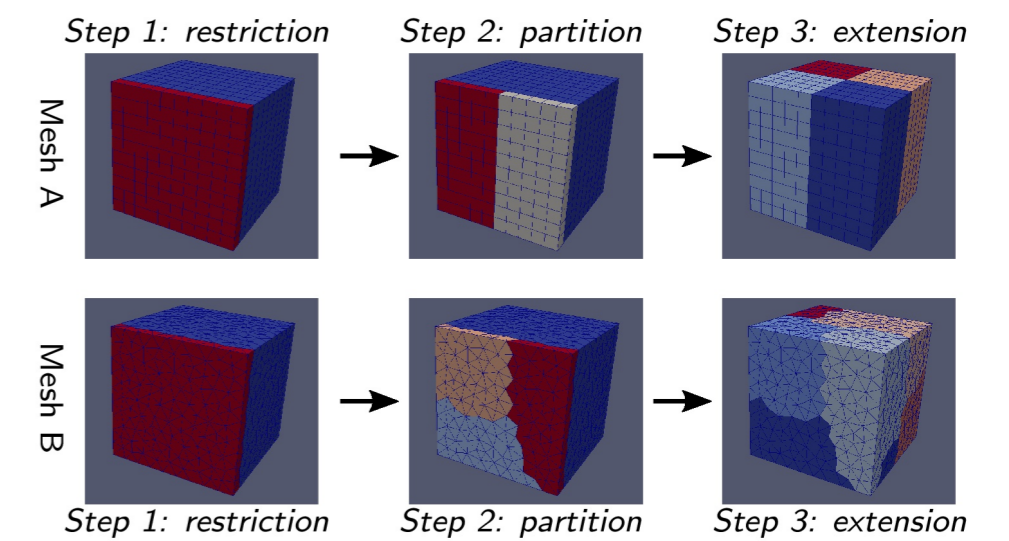
Model: A and B are running the regular phase on  $K_A = 4$  and  $K_B = 6$  processors, while less processors are involved in the coupling phase that acts as a synchronization phase.

## Copartitioning Algorithms

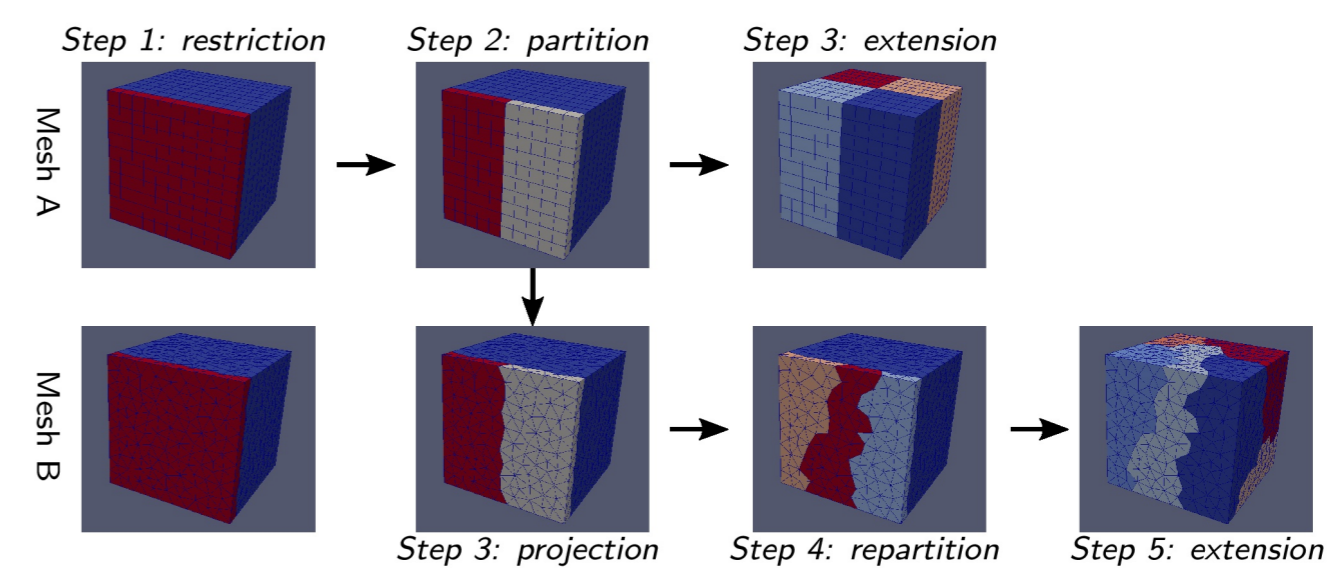
Given two coupled meshes  $A$  and  $B$  (or graphs  $G_A$  &  $G_B$ ) and interedges  $I_{AB}$ , we propose two algorithms that compute the output partitions  $P_A$  and  $P_B$  in respectively  $K_A$  and  $K_B$  parts.



Meshes A & B with coupling interedges.



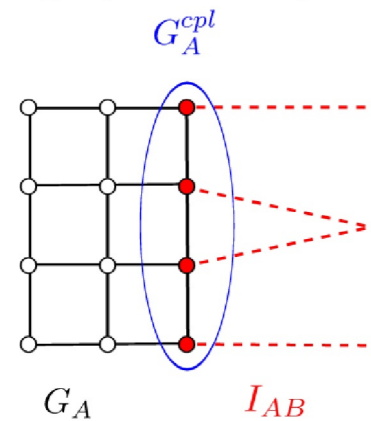
**ProjReport**: each code is also aware of the other coupling interface.



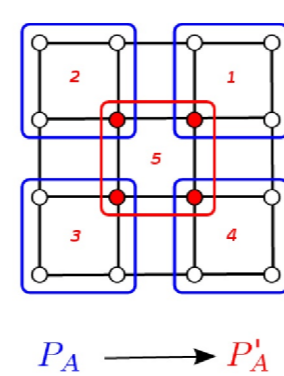
## Graph Operators

Let consider, two graphs  $G_A$  and  $G_B$  coupled by interedges  $I_{AB}$ . Our copartitioning algorithms are described as a sequence of graph operators.

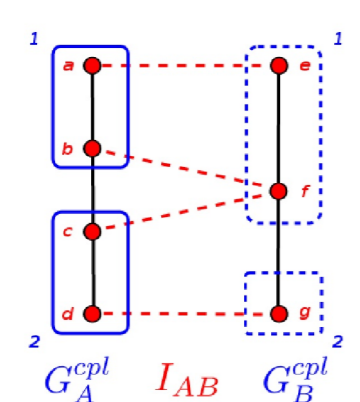
**Restriction**  
find the subgraph in coupling interface



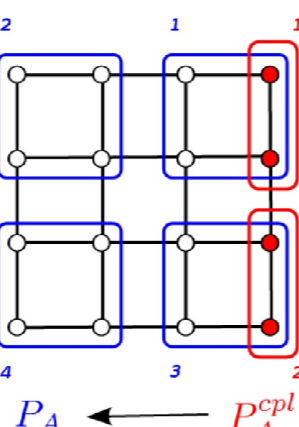
**Repartition**  
change the number of parts in a partition



**Projection**  
project a partition from one mesh to another



**Extension**  
extend a partition to the rest of the graph



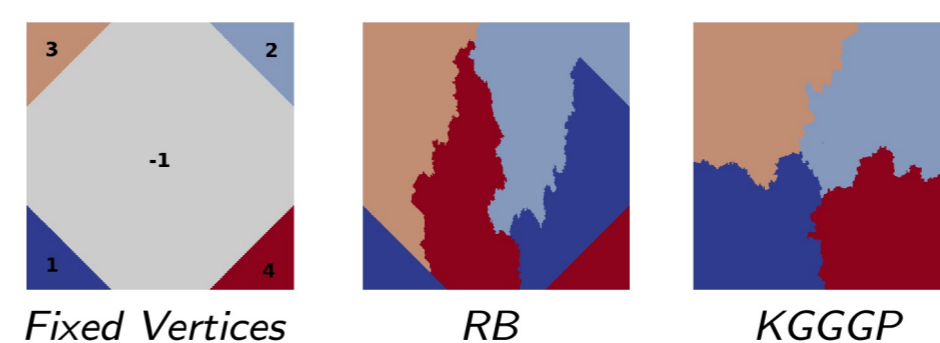
The implementation is based on biased graph partitioning with fixed vertices.

## Graph Partitioning with Fixed Vertices

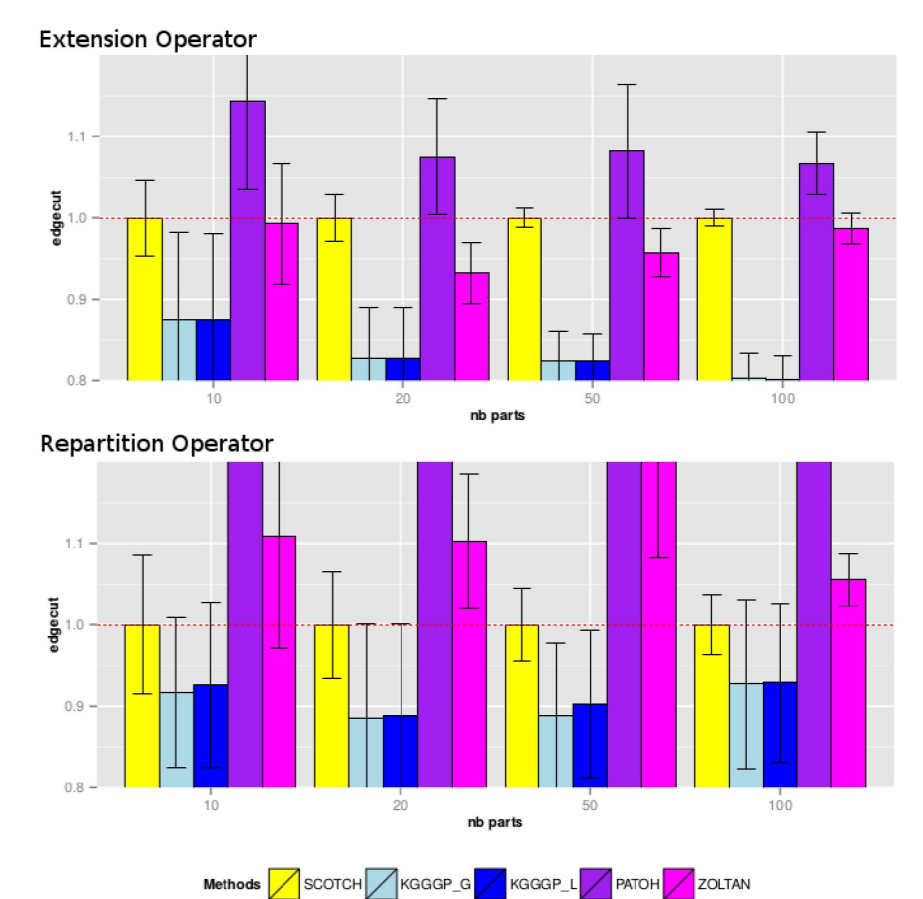
**Motivation** – Extension, Projection and Repartition operators need to handle vertices that must remain in place during the partitioning procedure (fixed vertices).

**Issue** – The state-of-the-art recursive bisection (RB) algorithms fail to handle properly the additional constraint of fixed vertices. So, we propose a different algorithm KGGGP (k-way Greedy Graph Growing Partitioning) to overcome this issue.

**Limitation of Recursive Bisection** – Given initial fixed vertices in the corners, comparison of KGGGP against RB (without multilevel framework) for a simple 2D grid, illustrating the inherent limitation of RB.



**Experimental results** – Evaluation of graph operators with different algorithms on the *Dimacs/Walshaw* graph collection.



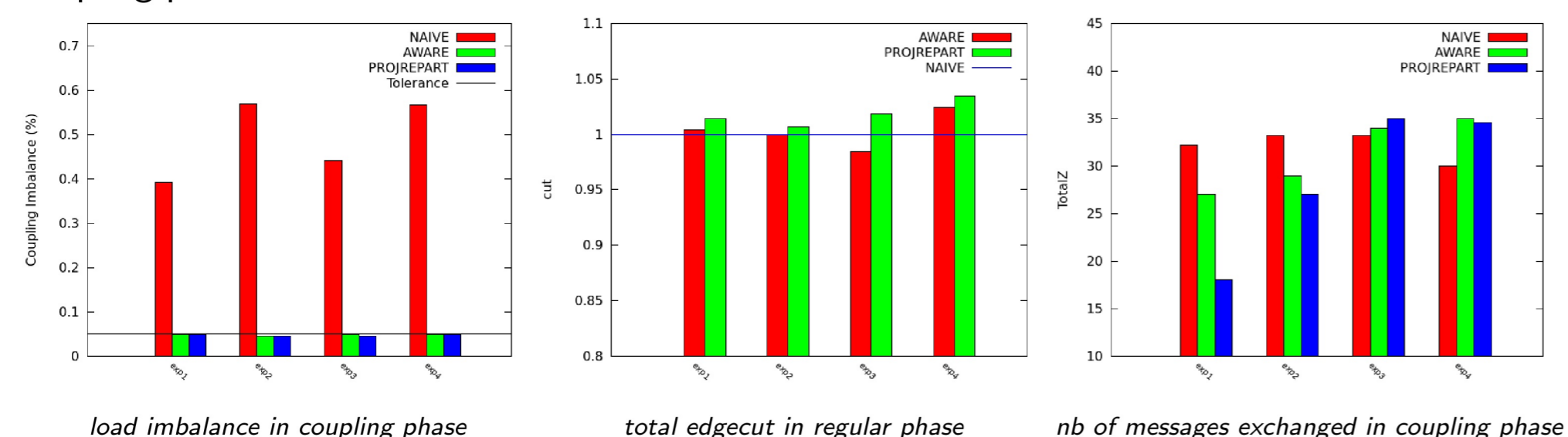
KGGGP achieves better partitioning quality as RB methods with similar performance.

## Copartitioning Results

**Experimental Setup** – Surface coupling between two cubic meshes using different discretizations (hexahedron or tetrahedron elements).  $K_A = 16$  and  $K_B = 48$ .

Exp.	Mesh A	Mesh B
exp1	hexa 25x25x25	hexa 100x100x100
exp2	hexa 25x25x25	hexa 70x70x70
exp3	tetra 40630	hexa 100x100x100
exp4	tetra 40630	tetra 486719

**Results** – Comparison of copartitioning algorithms (**Aware** and **ProjReport**) (using KGGGP) against the **Naive** approach, where each graph is partitioned independently, ignoring the coupling phase.



- Naive: highly imbalanced during coupling phase, best edgcut in the regular phase.
- Aware/ProjReport: load balance respected in the coupling phase at a slight increase of edgcut.
- ProjReport: the number of *major* messages exchanged in coupling phase is minimized (not shown).

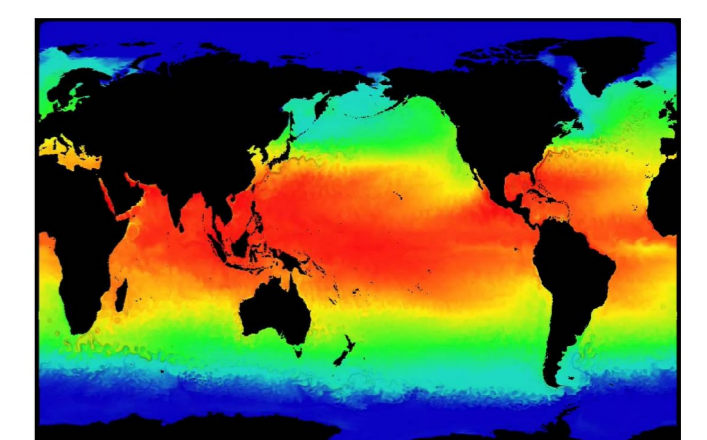
## Conclusion

### Our Contributions

- two new load-balancing algorithms, **Aware** and **ProjReport**, for coupled simulations
- a new algorithm for graph partitioning **KGGGP**, that handles properly initial fixed vertices
- the code is publicly available in the **MetaPart** library at [metapart.gforge.inria.fr](http://metapart.gforge.inria.fr)

### Future Work

- validate our copartitioning algorithms with real-life 3D coupled simulations with both surface and volume coupling
- implement a *multithread* version of KGGGP algorithm based on OpenMP



POP (Parallel Ocean Program) simulation coupling ocean and atmosphere models.

### References

- A *k*-way Greedy Graph Partitioning with Initial Fixed Vertices for Parallel Applications. M. Predari, A. Esnard. PDP 2016.
- Coupling-Aware Graph Partitioning Algorithms: Preliminary Study. M. Predari, A. Esnard. HIPC 2014.
- Graph Repartitioning with both Dynamic Load and Dynamic Processor Allocation. C. Vuchener, A. Esnard. ParCo 2013.
- Dynamic Load-Balancing with Variable Number of Processors based on Graph Repartitioning. C. Vuchener, A. Esnard. HIPC 2012.