International Workshop in Theoretical Approaches to Performance Evaluation, Modeling and Simulation
14-16th December 2016, Granada, Spain

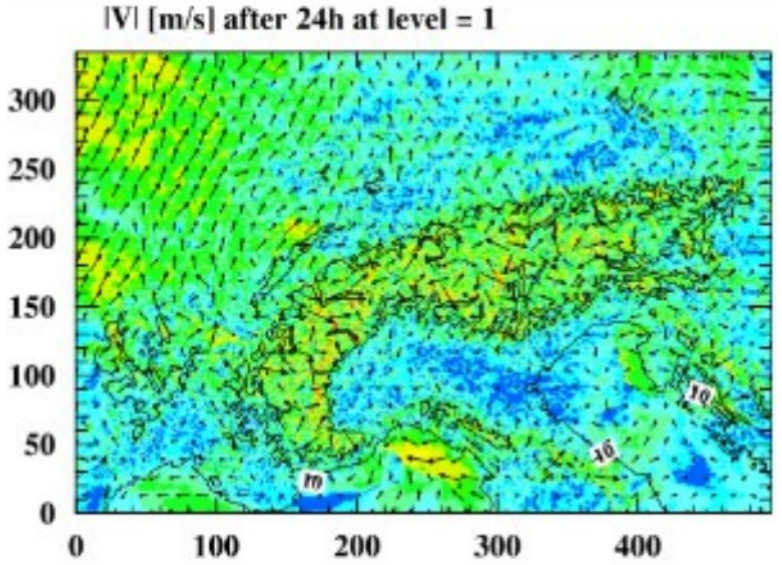
Network-aware Optimization of MPDATA on Homogeneous Multi-core Clusters with Heterogeneous Network

Lukasz Szustak
Roman Wyrzykowski
Czestochowa University of Technology
Poland

Tania Malik
Alexey Lastovetsky
Heterogeneous Computing Laboratory
University College Dublin
Ireland

Motivations for our research

- Our research includes Multidimensional Positive Definite Advection Transport Algorithm, which is one of the main parts of the EULAG model
- MPDATA is a real-life CFD application
- EULAG is an established computational model developed by the group headed by Piotr K. Smolarkiewicz for simulating thermo-fluid flows across a wide range of scales and physical scenarios
- One of the most interesting applications of the EULAG model is numerical weather prediction (NWP)
- In our research, we propose to rewrite the main parts of EULAG and replace standard HPC systems by emerging computing cluster



Motivations for our research

- The efficient utilization of emerging computing platforms becomes a global challenge
- The communication layer of modern HPC platforms is getting increasingly heterogeneous and hierarchical
- In consequence, even on platforms with homogeneous processors, the communication cost of many parallel applications will depend on the arrangement of processes in clusters
- In this work we propose a heuristic solution how solve a problem of mapping MPI processes (ranks) onto computing nodes, taking into account:
  - the network structure and performance
  - the logical communication flow of the application

MPDATA

- MPDATA belongs to the group of forward-in-time algorithms, and performs a sequence of stencil computations
- The whole MPDATA computations in each time step are decomposed into a set of 17 heterogeneous stencils
- In numerical simulation, where MPDATA can be used, the simulation runs for several thousand time steps
- A single MPDATA time step requires 5 input and 1 output matrices
- MPDATA, as a part of EULAG, is interleaved with other important computation in each time step
- MPDATA is a memory-bounded algorithm
- We focus on simulations using **3D grid**
  - the size of grid is **n** by **m** by **l**, where **l=64** or **l=128** for the case of NWP

MPDATA on a single node (shared memory version)

- The methodology of adaptation is based on the following methods:
  - (3+1)D decomposition of MPDATA
  - Improving efficiency of the decomposition by reduction of computation overheads
  - Partitioning of threads into teams
  - Task and data parallelisms
  - Search for the trade-off between computation and inter-cache communication
  - The OpenMP API is used to utilize the computing resources
- Detailed description of adaptation is presented in the papers:
  - L. Szustak, K. Rojek, P. Genere, Using Intel Xeon Phi coprocessor to accelerate computations in MPDATA algorithm, Lect. Notes in Comp. Sci., 8385:582-592, 2014
  - L. Szustak, K. Rojek, T. Olas, and P. Gepner, Adaptation of MPDATA heterogeneous stencil computation to Intel Xeon Phi coprocessor, Scientific Programming, 2015

MPDATA on clusters (distributed memory version)

- The target HPC platforms
  - Clusters with CPU
  - Clusters with Intel Xeon Phi coprocessors
    - Including both 1st and 2nd generations
  - Hybrid clusters with CPU + Intel Xeon Phi

MPDATA on clusters (distributed memory version)

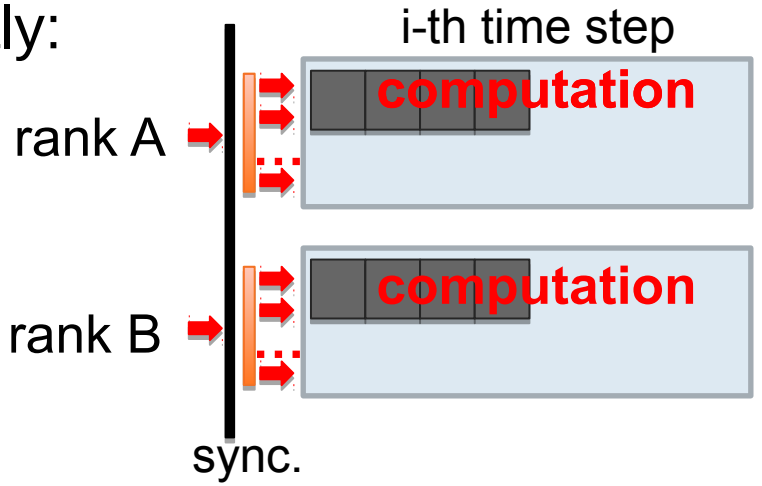
- The target HPC platforms
  - **Clusters with homogeneous CPU with heterogeneous network**
  - Clusters with Intel Xeon Phi coprocessors
    - Including both 1st and 2nd generations
  - Hybrid clusters with CPU + Intel Xeon Phi

MPDATA on clusters (distributed memory version)

- One of the common methods for exploiting the multicore clusters is to employ the hybrid programming model
  - It allows for efficient usage of the distributed and shared memory hierarchies of these systems
- This implies to combine different programming paradigms, such as **MPI** and **OpenMP**
- Such a mixture is successfully utilized for the MPDATA computation:
  - MPI rank is assigned to every multicore node
  - OpenMP threads are employed to utilize the multicore computational resources

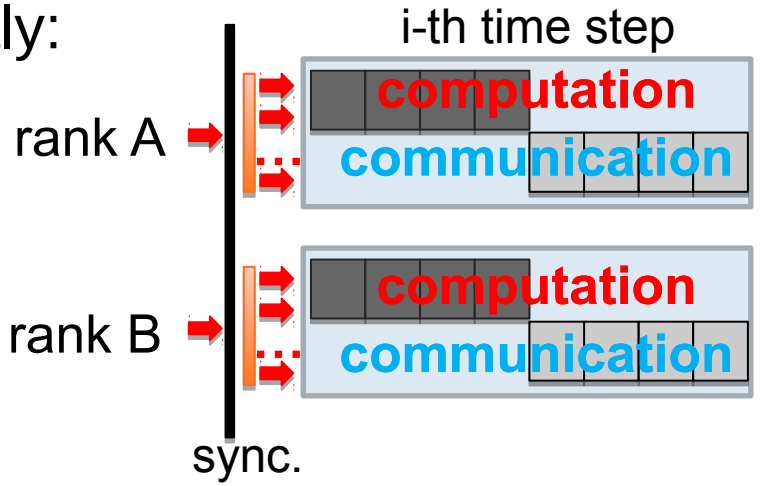
MPDATA on clusters (distributed memory version)

- Currently:



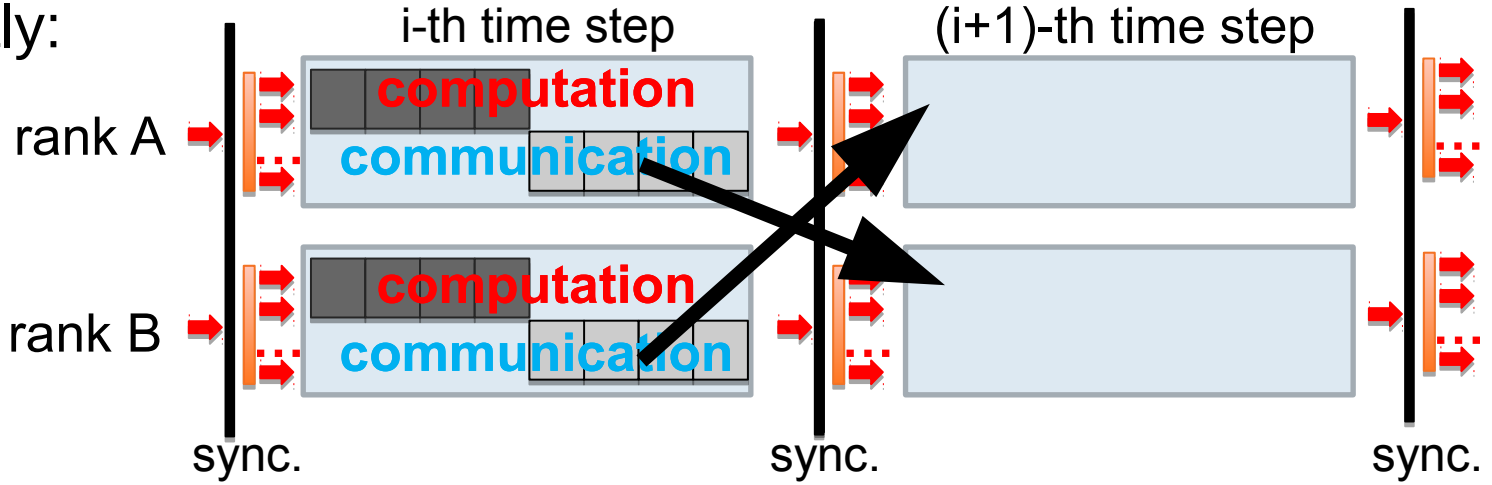
MPDATA on clusters (distributed memory version)

- Currently:



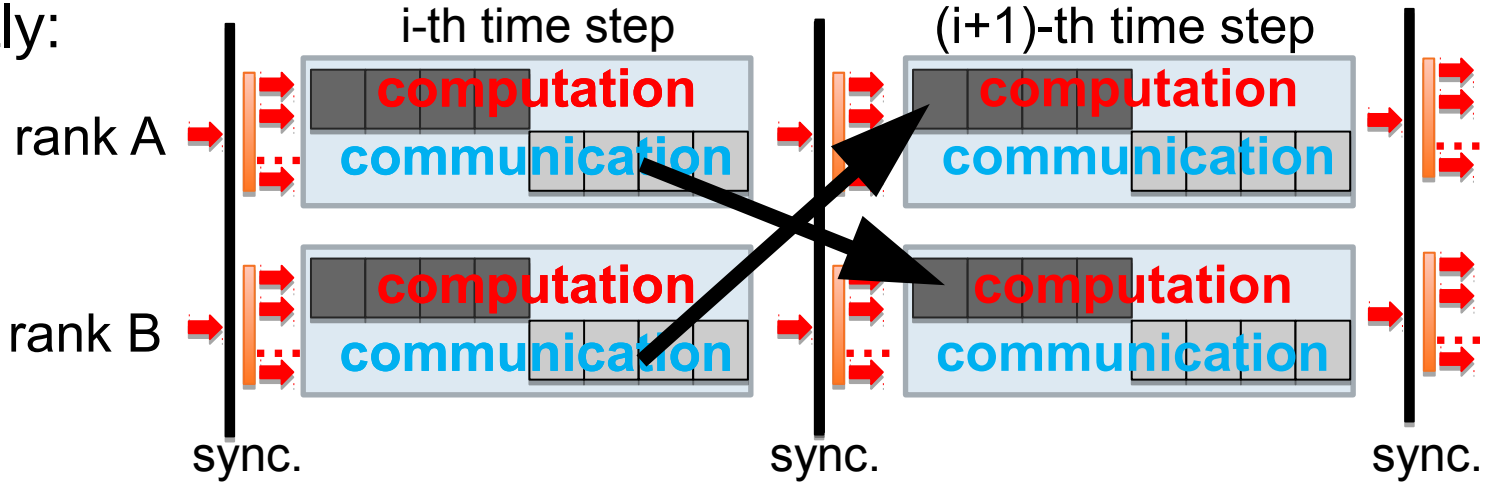
MPDATA on clusters (distributed memory version)

- Currently:



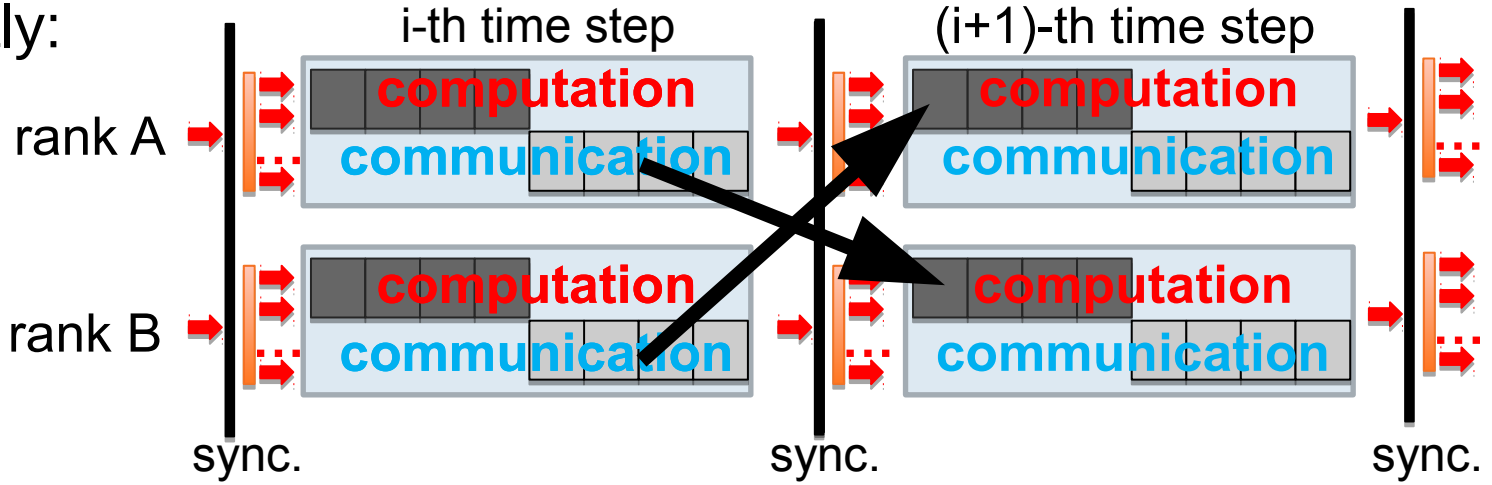
MPDATA on clusters (distributed memory version)

- Currently:

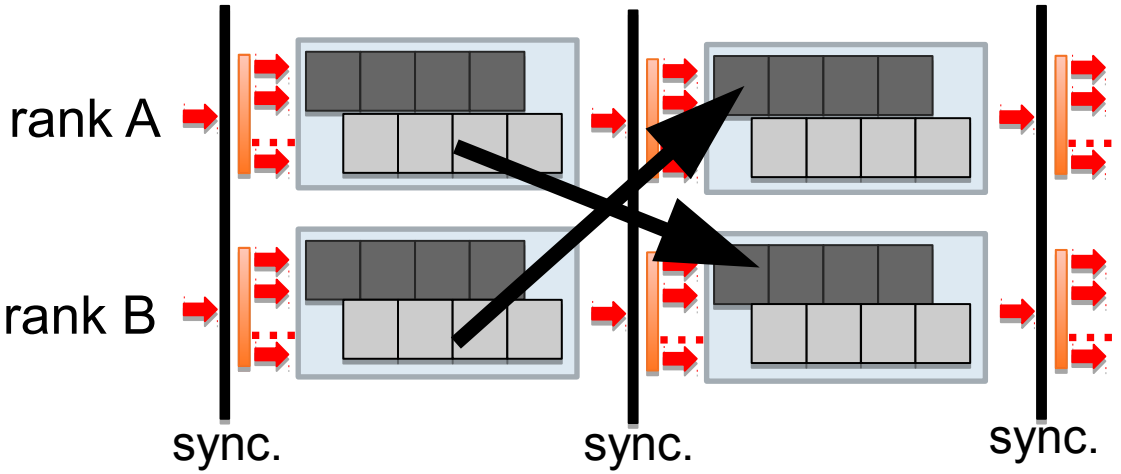


MPDATA on clusters (distributed memory version)

- Currently:

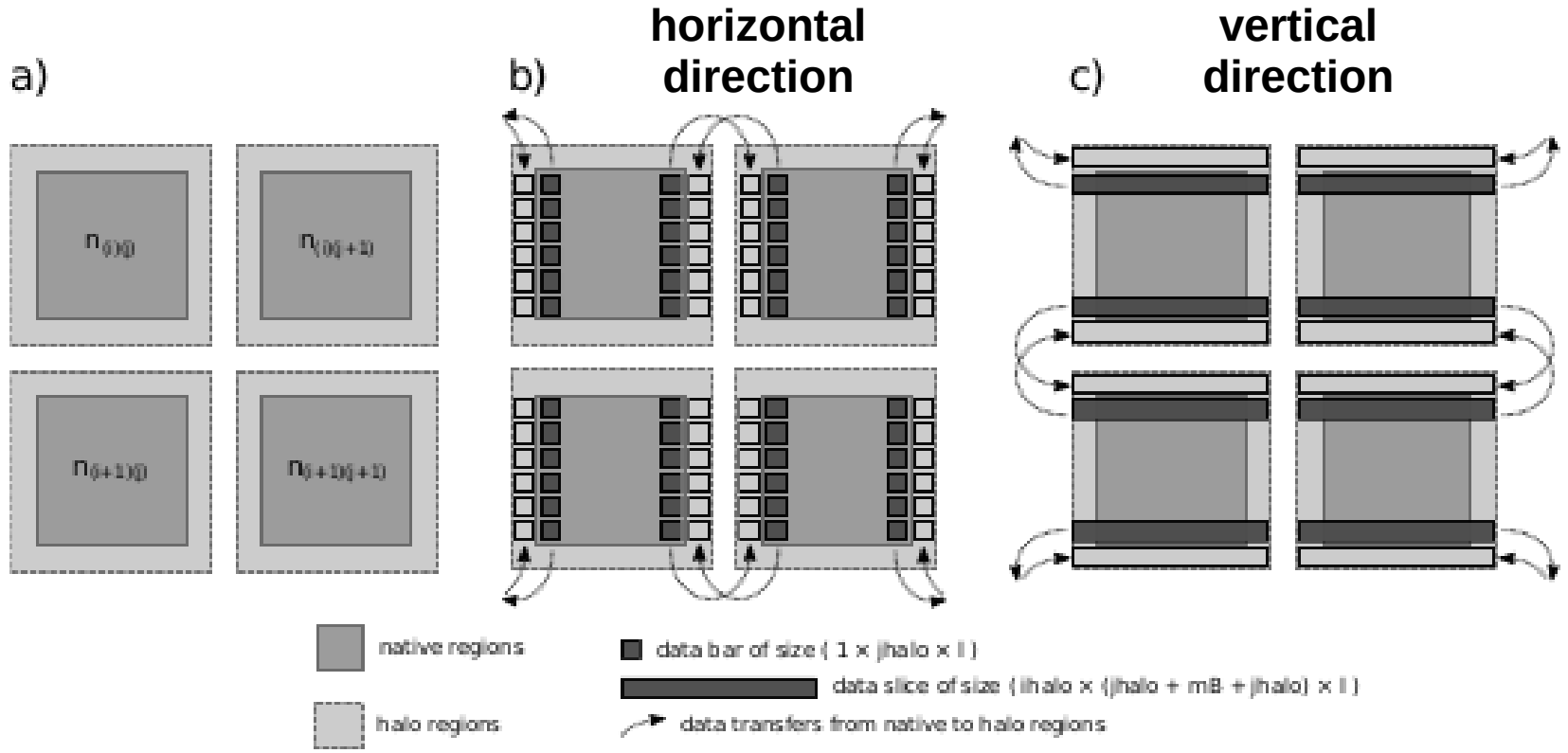


- In the future (*in progress, there are a lot of issues …*):



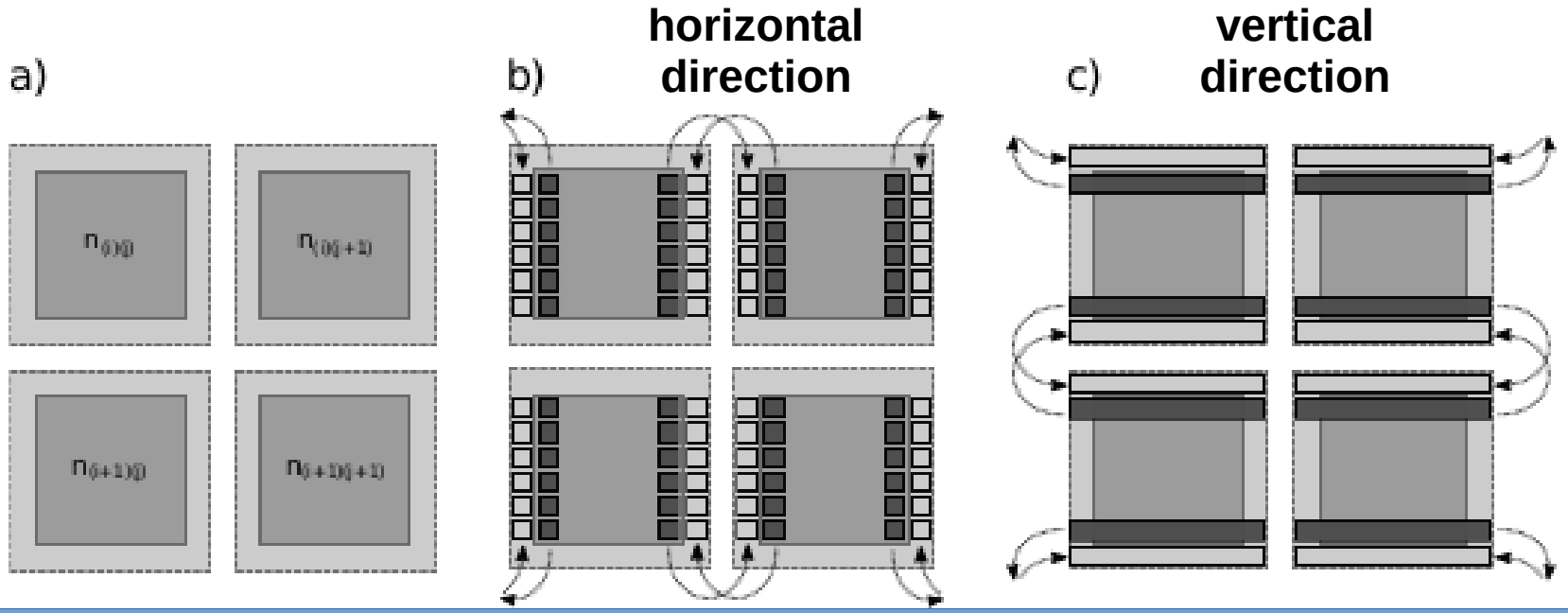
MPDATA on clusters (distributed memory version)

- **The 3D MPDATA domain is partitioned in 2D equal sub-domains that are further one-to-one mapped to nodes**
- Every sub-domain is decomposed according to the (3+1)D decomposition proposed in our previous works (shared memory version)



MPDATA on clusters (distributed memory version)

- **The 3D MPDATA domain is partitioned in 2D equal sub-domains that are further one-to-one mapped to nodes**
- Every sub-domain is decomposed according to the (3+1)D decomposition proposed in our previous works (shared memory version)



Problem: How to map MPI ranks onto computing nodes

Arrangement of MPI ranks in Cluster for MPDATA

- The main goal is to minimize the communication cost of MPDATA
- All configurations can be tested in empirical way ...
- Instead of it, we propose to use the approximate topology-aware heuristic algorithm:
  1. We first, propose an extension of the network-bandwidth-based cost function (see work [1]) to accurately measure the communication cost of the MPDATA application
  2. Then we formulate the heuristic solution that efficiently constructs a near-optimal arrangement for MPDATA by using:
     - information about network topology
     - and the application communication flow

[1] Malik, T., Rychkov, V., Lastovetsky, A.: Network-aware optimization of communications for parallel matrix multiplication on hierarchical hpc platforms. Concurrency and Computation: Practice and Experience 28 (2016) 802–821 cpe.3609.

Cost Function

- We use the cost function to estimate the communication cost incurred by any data partitioning
- The cost functions is based on asymmetric bandwidth
  - MPDATA has different horizontal and vertical communication so our cost function takes two bandwidth values

Cost Function

- Cost function takes two bandwidth values
  - One for horizontal communication

$$cost_H = \sum_{i=1}^{r} \left(h \times \sum_{j=1}^{c} \frac{1}{b_H(Q_{ij}, Q_{i,(j+1)\%c})} \right)$$

  - Other is for vertical one

$$cost_V = \sum_{j=1}^{c} \left(w \times \sum_{i=1}^{r} \frac{1}{b_V(Q_{ij}, Q_{i,(j+1)\%r}} \right)$$

The cost of communication for any arrangement "*A*"

- The communication cost associated with arrangement *A* is represented by two values cost_H (*A*), cost_V (*A*)
- The problem of finding the optimal arrangement can be formulated as minimization of their sum:

$$\text{cost\_H}(A) + \text{cost\_V}(A) \rightarrow \min$$

Heuristic Based on Asymmetric Bandwidth Cost Function

- This is an offline solution (before execution of MPDATA)
- Input:
  - Processors from the same group will follow one other in linear arrangement
  - Horizontal and Vertical Bandwidth between processors
- Output
  - Near-optimal arrangement of MPI ranks in cluster for MPDATA

Input:
Processors, $P_1, P_2, \ldots, P_p \in \mathbb{Z}_{>0}$
Horizontal bandwidth, $b_H(x,y), \forall x,y \in [1,p], b_H(x,y) \in \mathbb{Z}_{>0}$
Vertical bandwidth, $b_V(x,y), \forall x,y \in [1,p], b_V(x,y) \in \mathbb{Z}_{>0}$
Output:
Optimal 2-D arrangement of the processors

Repeat
STEP 1:
for each factor pair $r \times c = p$ **do**
arrange $P_1, ..., P_p$ in r and c by row ranking order $\rightarrow A$
arrange $P_1, ..., P_p$ in r and c by column ranking order $\rightarrow A$
find A such that $cost(A,m,n) = \min\limits_{r,c} cost(A,r,c)$
end for
STEP 2:
generate group permutations of $A_1 \rightarrow A_1^1, \ldots, A_1^{g_1!}$
for each permutation $k := 1$ to $g_1!$ **do**
find k such that $cost_V(A_1^k) = \min$
end for
$A_1^* := A_1^k$
for each column $i := 2$ to n **do**
generate group permutations of $A_i \rightarrow A_i^1, \ldots, A_i^{g_i!}$
for each permutation $k := 1$ to $g_i!$ **do**
find k such that $cost(A_1^*, \ldots, A_{i-1}^*, A_i^k) = \min$
end for
$A_i^* := A_i^k$
end for
generate column permutations of arrangement $A^* \rightarrow A^*(1), \ldots, A^*(n!)$
for each permutation $j := 1$ to $n!$ **do**
find j such that t $cost(A^*(j)) = \min$
end for
$A^* := A^*(j)$

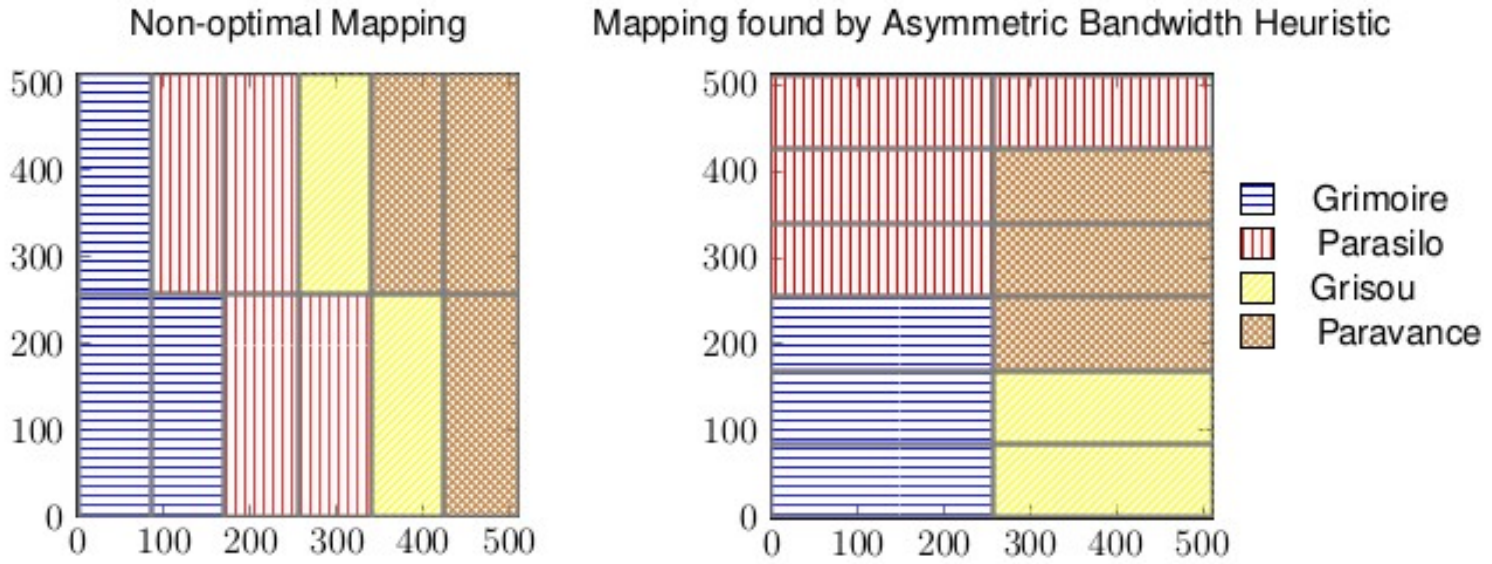
Heuristic Based on Asymmetric Bandwidth Cost Function

- **First Step** allows us to find optimal 2D shape “A”
- **Second Step**: Apply the bandwidth-based algorithm
  - First try permutations of the groups in the first column, and then pick the "*order*" with minimum cost of vertical communication
  - Then, for each following column k = 2, . . . , n, try permutations of the groups in this column, and pick the one that minimizes the sum of vertical and horizontal costs for first ‘k’ columns
- We will feed the new arrangement that we get in step 2 to the first step of next iteration of our heuristic algorithm that will find the optimal m × n arrangement for this new order
- This procedure continues until we find a fixed point of the transformation performed by one iteration of the algorithm
  - If communication cost of the next iteration is greater then the previous one, its mean previous iteration has near-optimal arrangement

Experimental platform

- Performed experiments on Grid 5000
- All clusters have identical Intel Xeon E5-2630 v3 processors with 8 cores per nod
  - Inter-Cluster experiments:
    - Four clusters with 12 nodes in total:
      Grimoire(3), Parasilo(4), Grisou(2), Paravance(3).
    - One MPI process per node
    - Problem size 512 × 512 × 64
  - Intra-Cluster Experiments:
    - 12 nodes from the Grisou cluster
    - One MPI process per node
    - Problem size 512 × 512 × 64

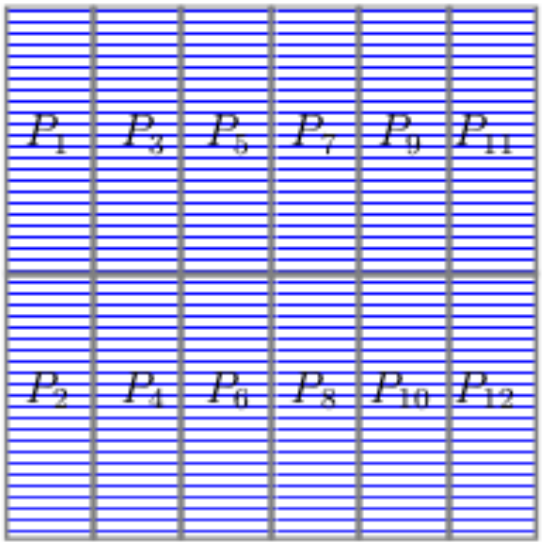
Inter-Cluster Experiments



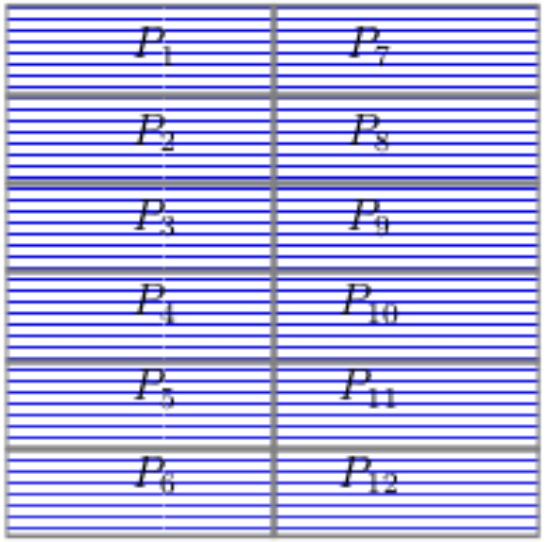
Nodes	Cost		Ratio	Exec. time (sec)		Ratio
	Non-optimal	Heuristic		Non-optimal	Heuristic	
12	22424946	2143978	10.46	994.02	154.20	6.44

Intra-Cluster Experiments

Non-optimal Mapping



Mapping found by Asymmetric bandwidth Heuristic



Nodes	Cost		Ratio	Exec time (sec)		Ratio
	Non-optimal	Heuristic		Non-optimal	Heuristic	
12	65658	18535	3.5	3.86	1.32	3.0

Conclusion and future work

- In this paper, we applied an approach aimed to minimize the communication cost of parallel CFD application using information about network topology/performance and application communication flow
- We also demonstrate that proposed solution provides significant performance gain
- We plan to adapt the proposed solution for the hybrid clusters with Intel Xeon Phi

Thank you for your attention!