TAPEMS, 2017

CCGrid2017

Formal modeling and performance evaluation of a run-time rank remapping technique in Broadcast, Allgather and Allreduce MPI collective operations

Jesús M. Álvarez, Juan C. Díaz, Juan A. Rico

University of Extremadura

Cáceres, Spain

http://gim.unex.es

This work is financed by the Junta de Extremadura and the European Union (ERDF funds) through the support funds to research groups GR1517

FEDER GOBIERNO DE EXTREMADURA Consejería de Economía, Competitividad e Innovación

Este trabajo está financiado por la Junta de Extremadura y por la Unión Europea (fondos FEDER) a través de los fondos de ayuda a grupos de investigación GR1517

Contents

- Introduction and motivation
- Modeling the influence of mapping in MPI collectives
- Run time reordering
- Performance evaluation
- Conclusions and future work

Introduction and motivation

- Current HPC systems are multi-core clusters, composed of multi-core nodes connected by networks.
- Performance of inter-process communications depends on processes' locations.
- Processes that communicate more should be located as *near* as possible.
- We focus on MPI applications over multi-core clusters.

Introduction and motivation

- MPI applications rely on MPI collective operations between processes:
  - *MPI_Allgather, MPI_Allreduce, MPI_Broadcast*
- MPI collective operations rely on one or more collective algorithms:
  - *Recursive Doubling Algorithm (RDA), Ring,…*
- Collective algorithms rely on communication patterns, based on rank numbers.
  - Mapping of rank numbers to processors has influence on performance of algorithms → collectives → applications

Introduction and motivation

- Types of mappings on MPI applications:

  - **Sequential**, SEQ (default on OpenMPI): contiguous ranks assigned to processes in the same node.

  - **Round Robin**, RR (default on MPICH): ranks assigned ringing over the nodes.

  - **Custom**: non *regular* mapping.

Introduction and motivation

- *MPI_Allreduce* MPICH implementation:
  - RDA (for small messages)→ Optimal with RR.
  - Ring (for large messages)→ Optimal with SEQ.

- Custom mapping can be used to optimize communications for an application (as in Jeannot *TreeMatch*): is an average optimization.

- **Even if global communications are *average* optimized, each of individual collective could be not.**

Introduction and motivation

- **We propose** optimization of each individual collective by means of **temporal rank reordering ("remapping")**.
  - Temporary change RR to SEQ or vice versa when suitable.
  - Without physical migration.
  - Transparent to application.
  - Not incompatible with other optimizations.

- Profit quantified using LogGP model.
- Tested it in a real cluster.

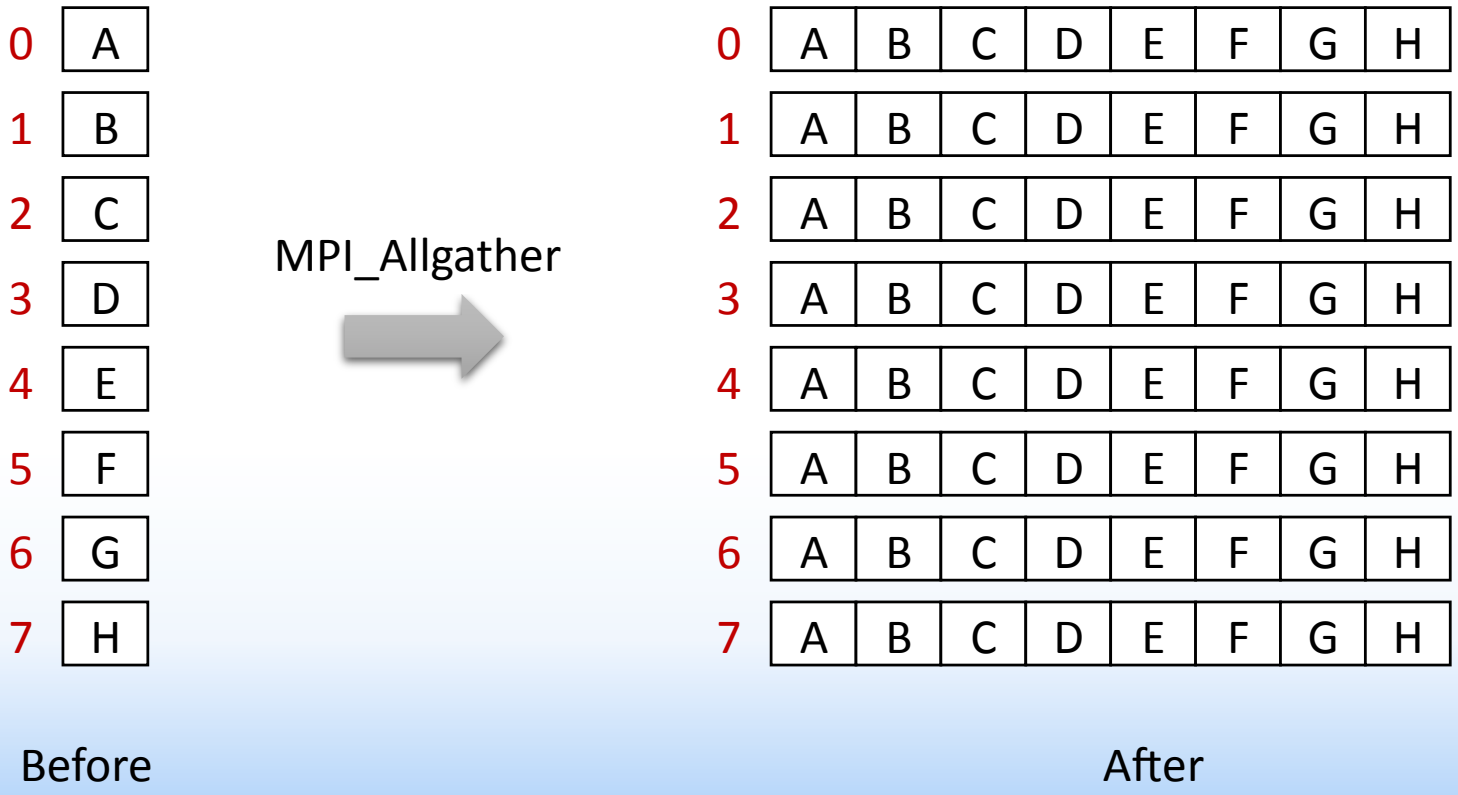
Modeling the influence of mapping

- What is the influence of rank-to-processor mapping in the cost of MPI collectives in a multicore cluster?

- Let us study the case of MPI_Allgather primitive:
  - RDA/Ring algorithms.
  - SEQ/RR mapping.

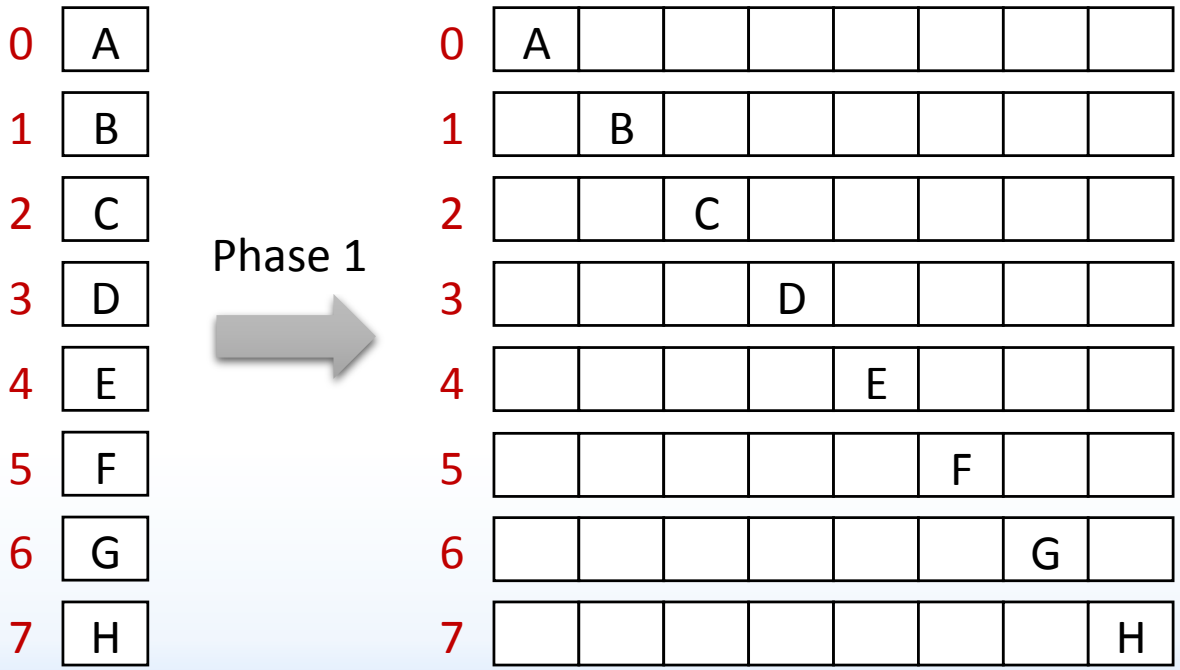
Modeling the influence of mapping

- *MPI_Allgather*: all processes gather all data from the rest (P=8 processes):



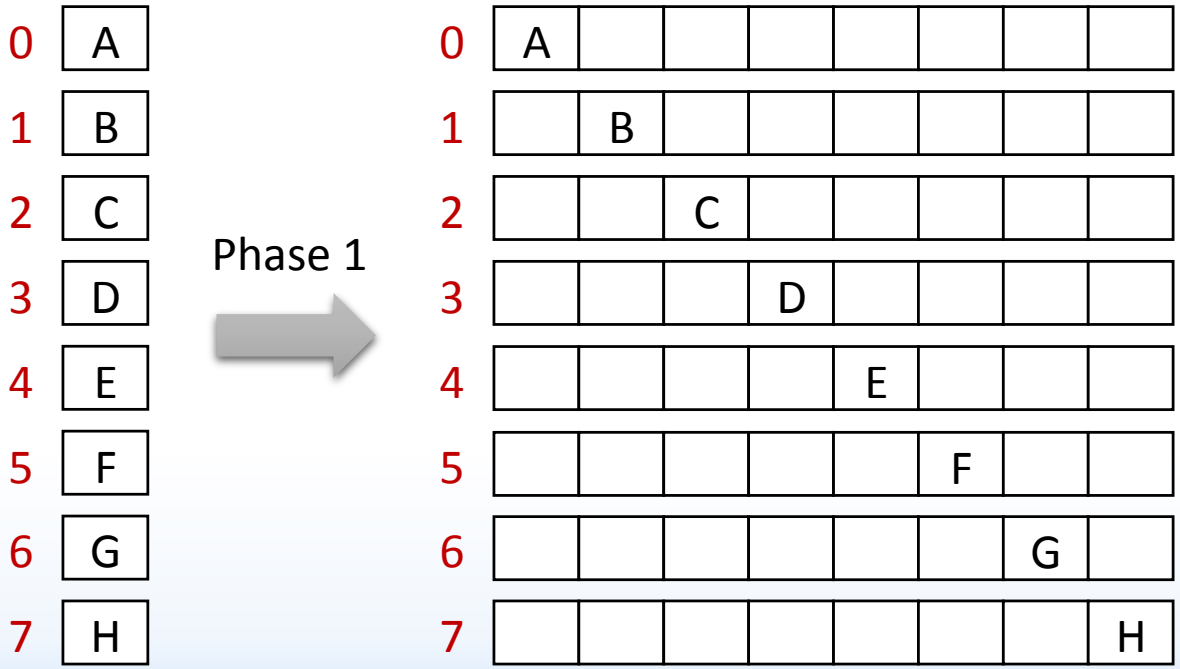
Modeling the influence of mapping

- *MPI_Allgather* with RDA: phase 1: copy own data



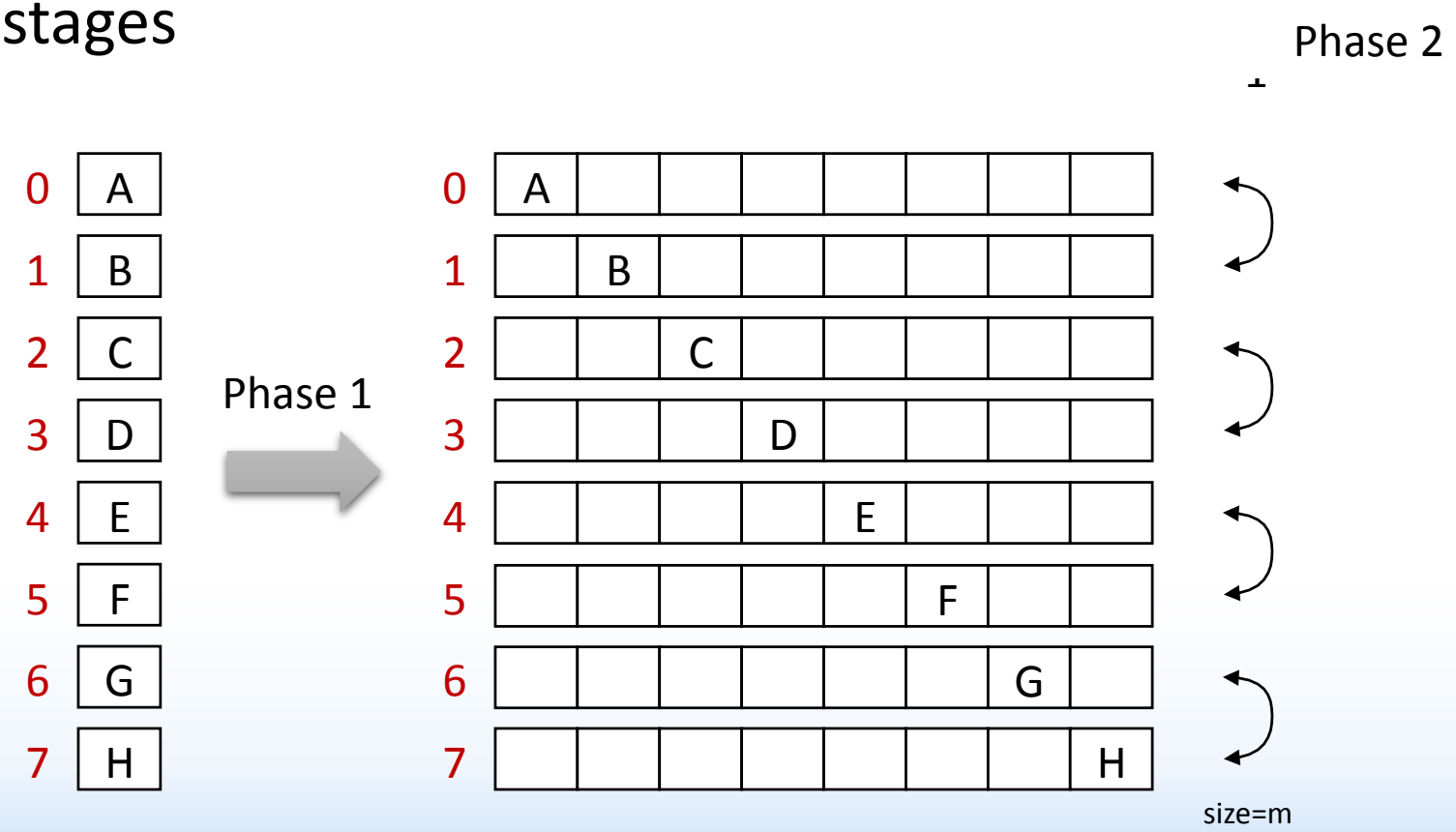
Modeling the influence of mapping

- *MPI_Allgather* with RDA: phase 2: $\log_2 P$ exchange stages



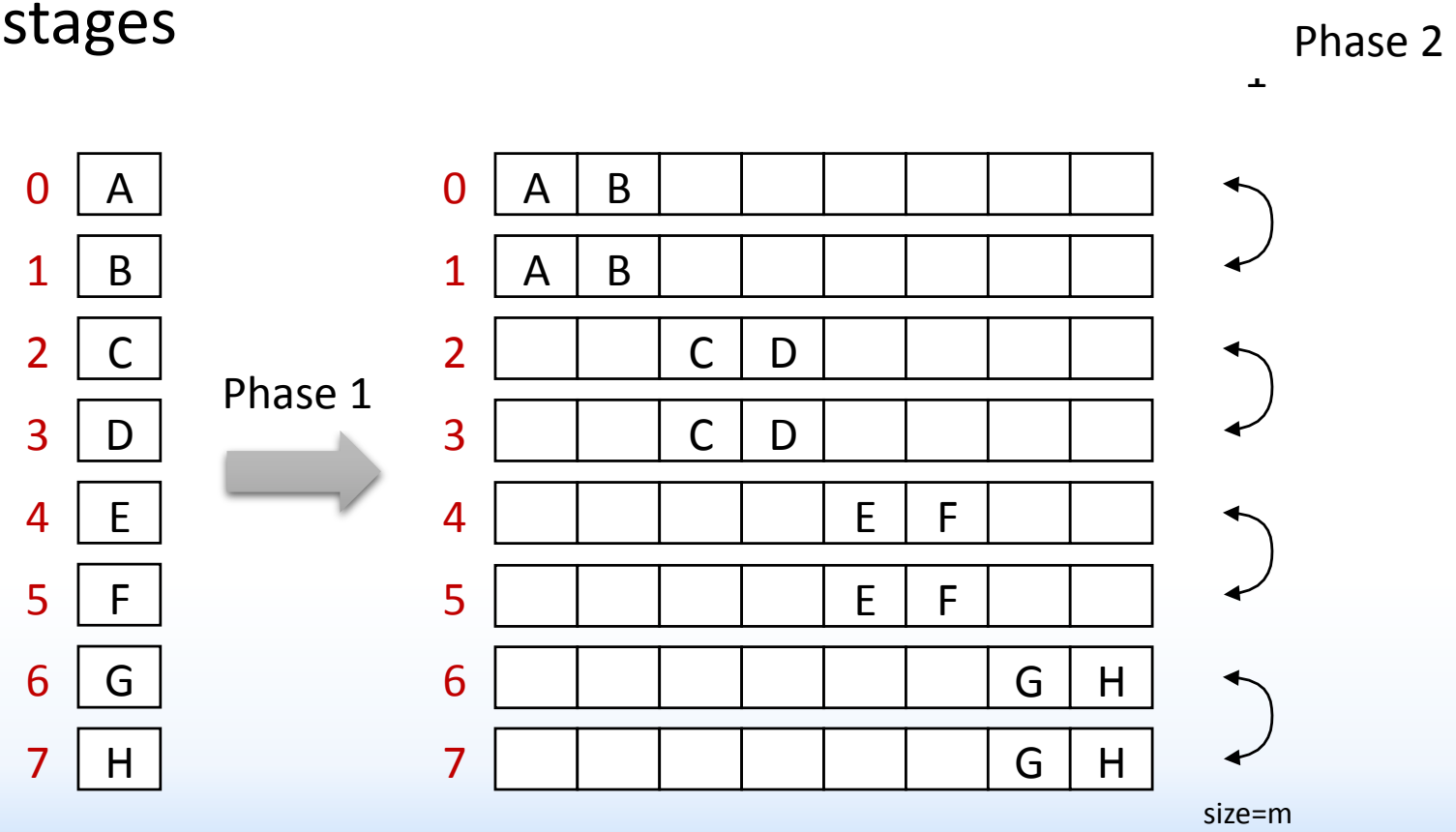
Modeling the influence of mapping

- *MPI_Allgather* with RDA: phase 2: $\log_2 P$ exchange stages



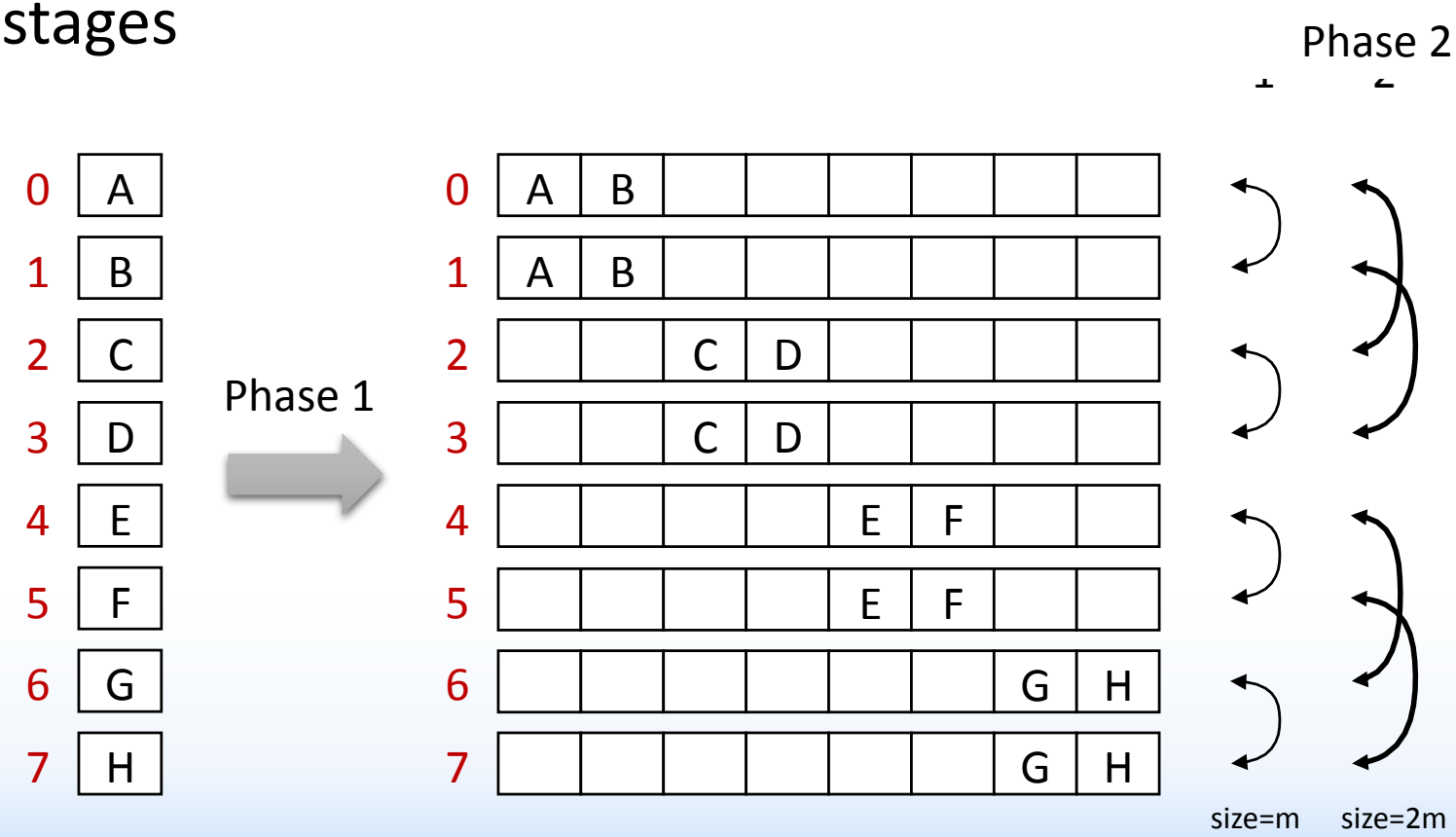
Modeling the influence of mapping

- *MPI_Allgather* with RDA: phase 2: $\log_2 P$ exchange stages



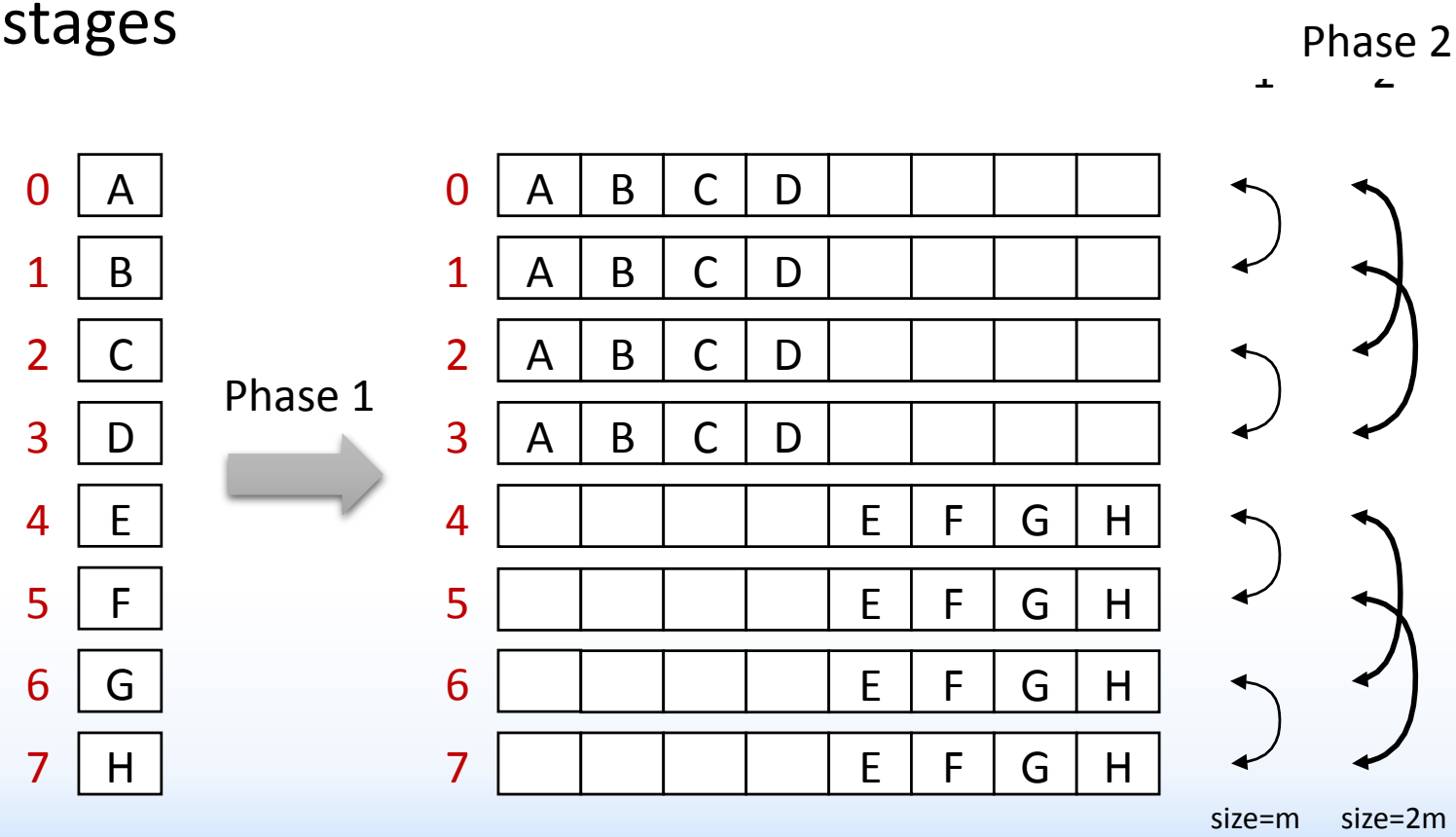
Modeling the influence of mapping

- *MPI_Allgather* with RDA: phase 2: $\log_2 P$ exchange stages



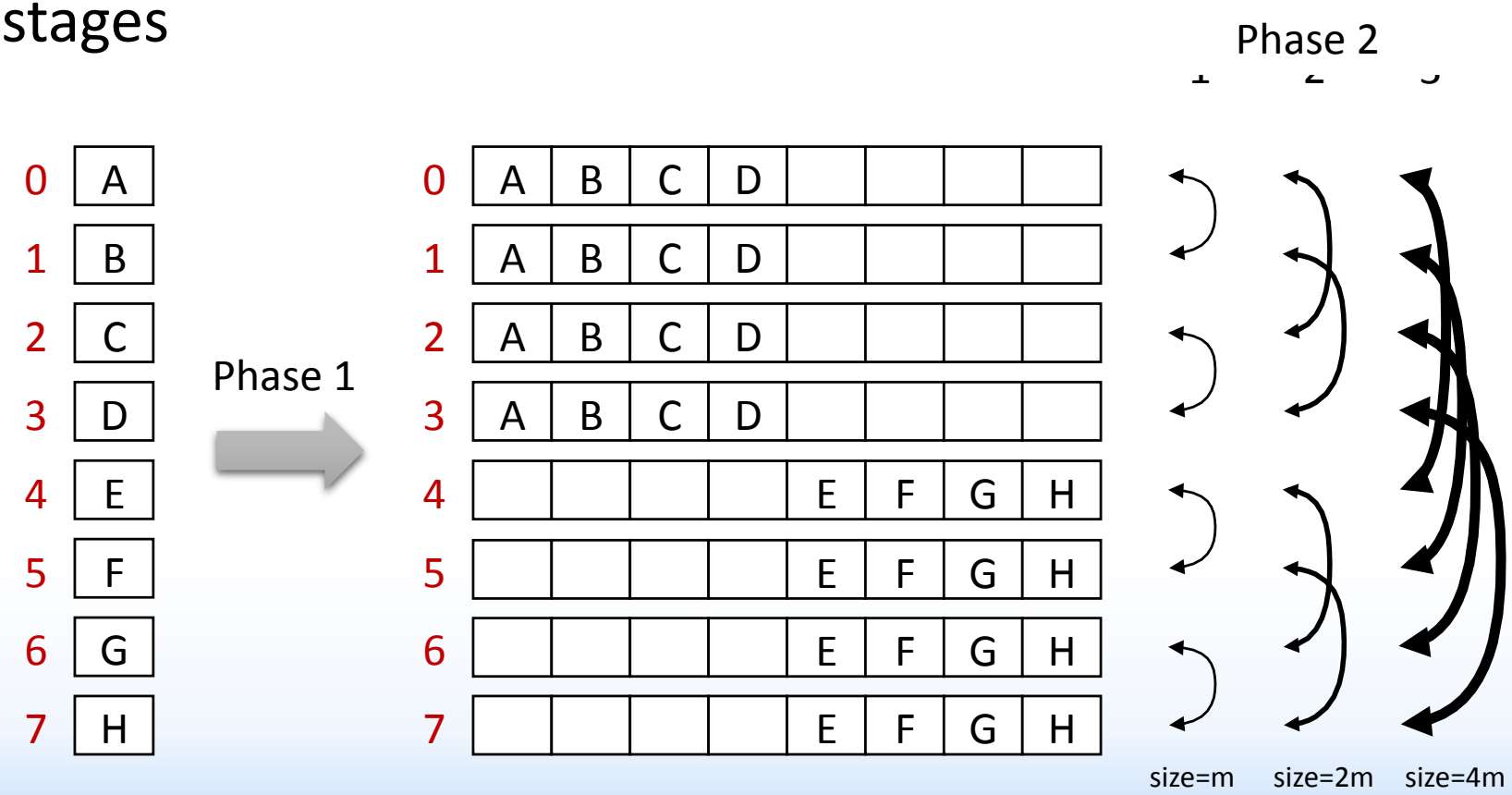
Modeling the influence of mapping

- *MPI_Allgather* with RDA: phase 2: $\log_2 P$ exchange stages



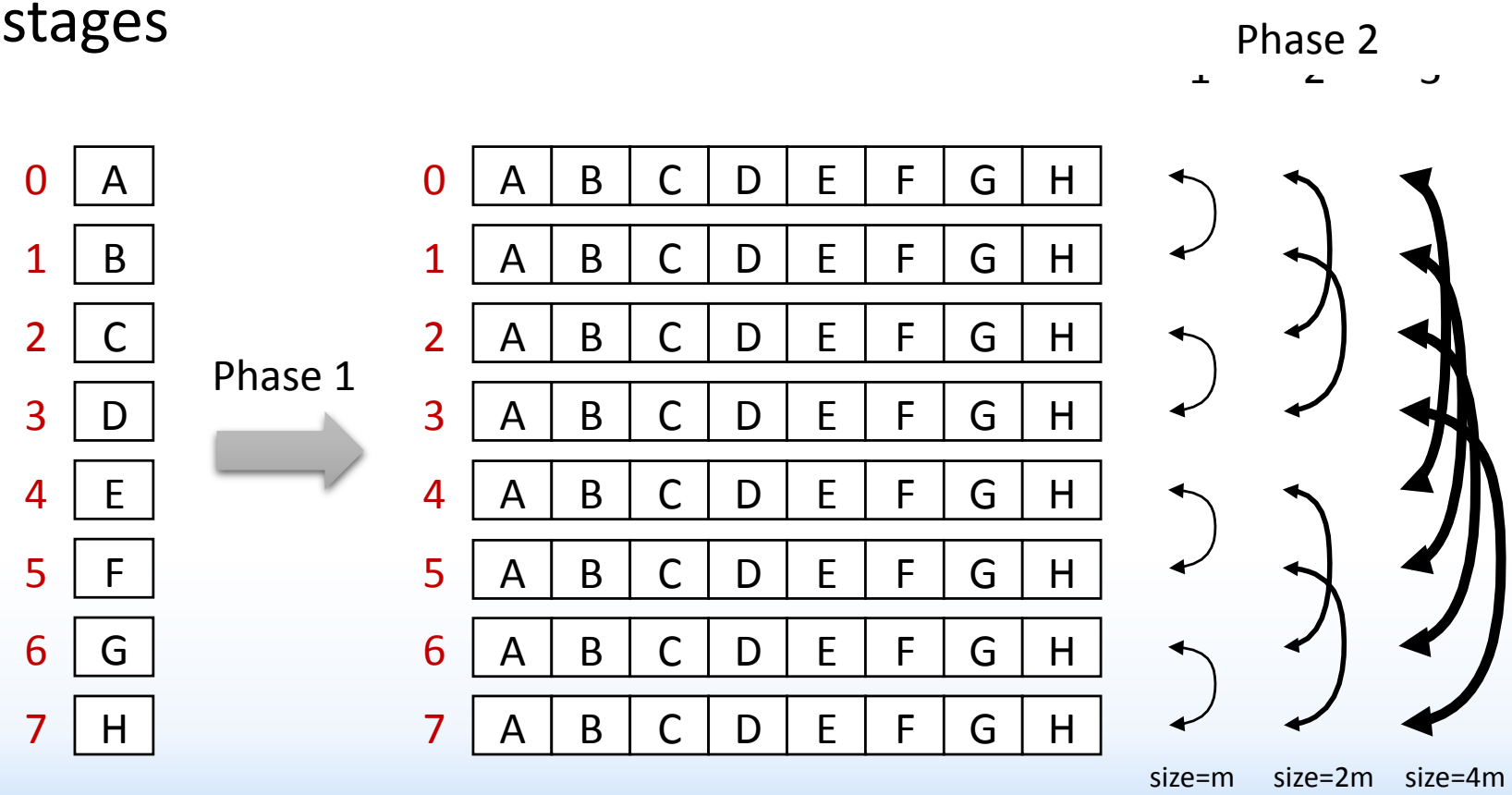
Modeling the influence of mapping

- *MPI_Allgather* with RDA: phase 2: $\log_2 P$ exchange stages



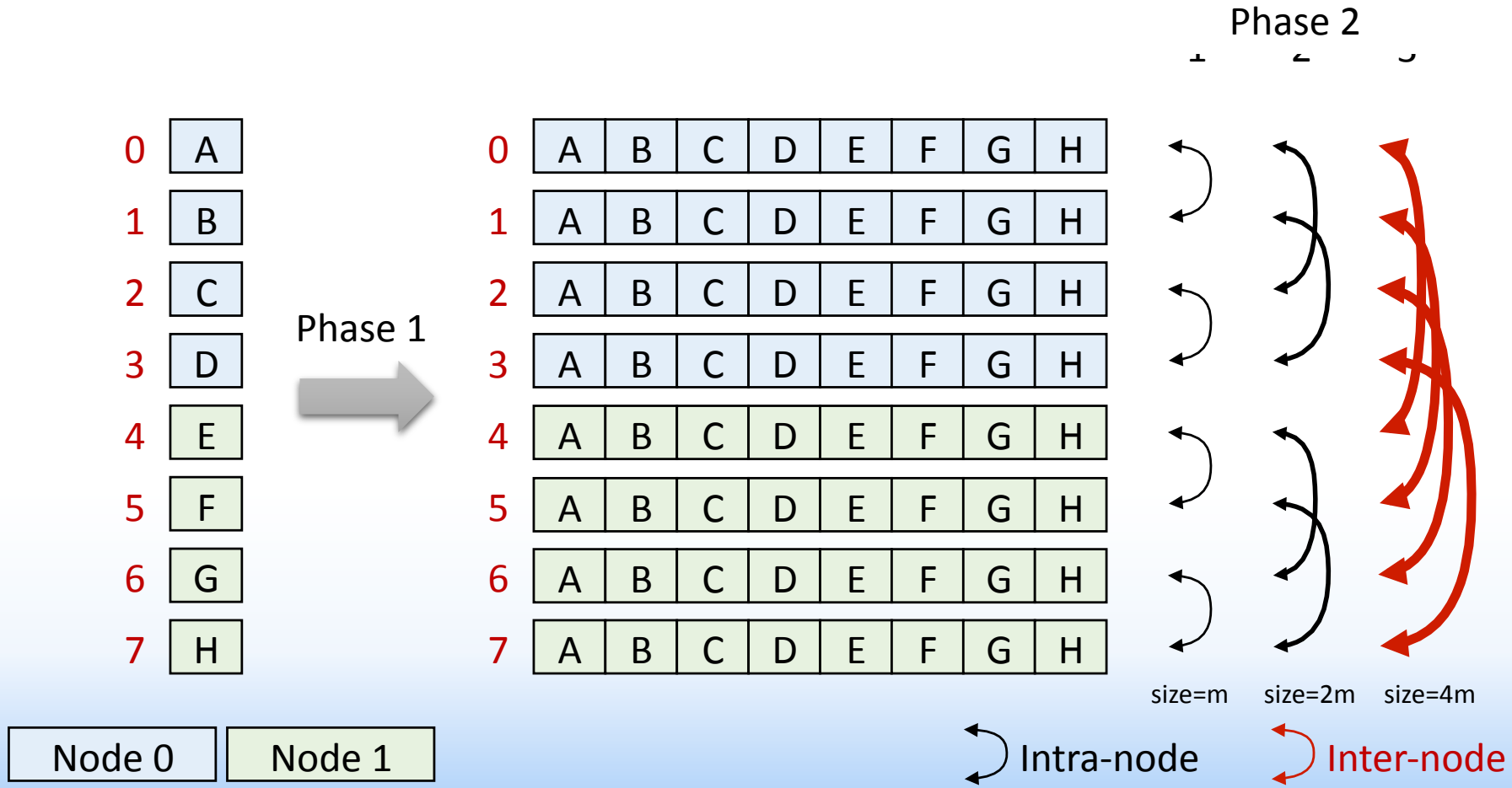
Modeling the influence of mapping

- *MPI_Allgather* with RDA: phase 2: $\log_2 P$ exchange stages



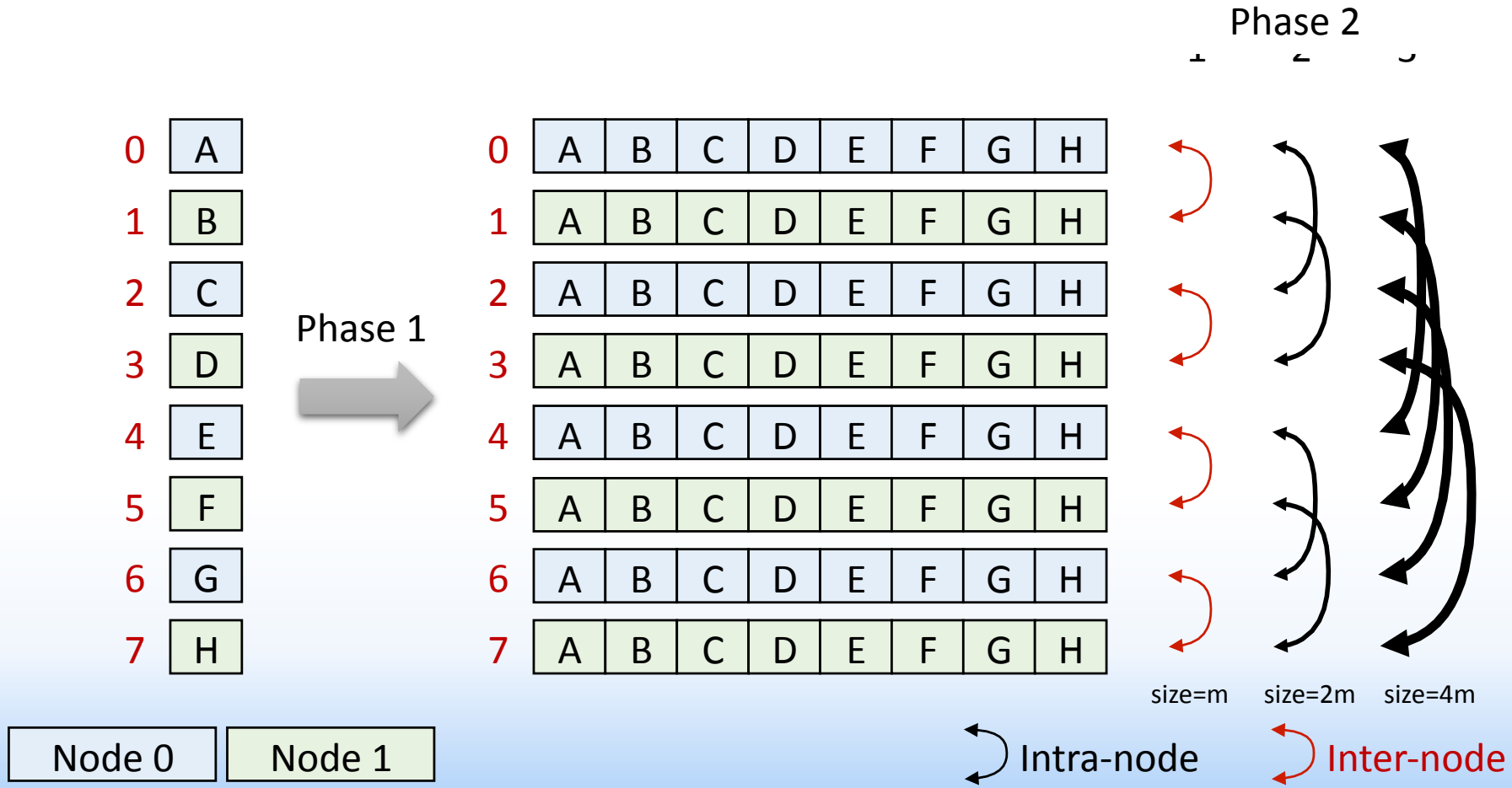
Modeling the influence of mapping

- *MPI_Allgather* with RDA: multi node with **SEQ mapping**



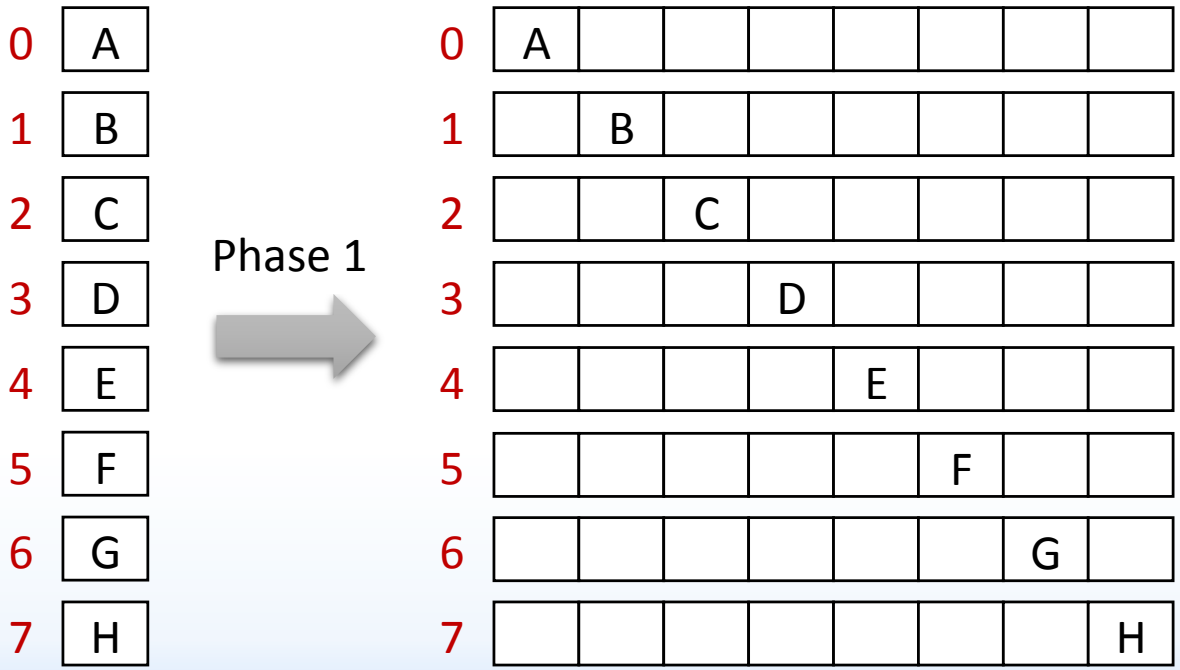
Modeling the influence of mapping

- *MPI_Allgather* with RDA: multi node with **RR mapping**



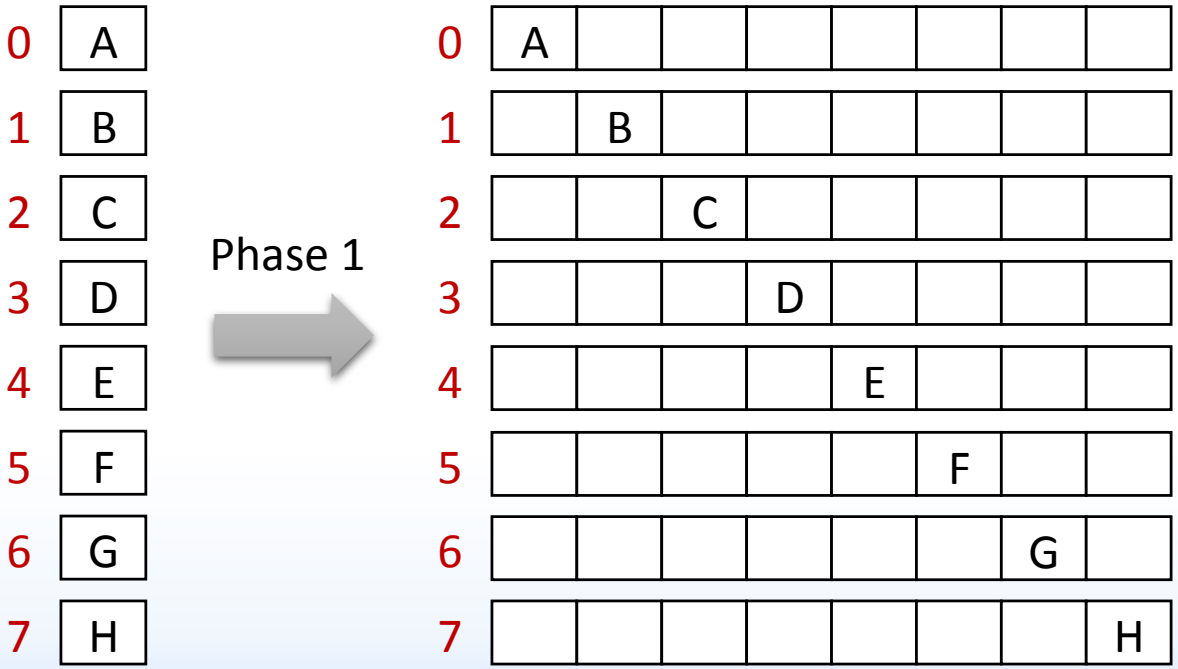
Modeling the influence of mapping

- *MPI_Allgather* with Ring: phase 1: copy own data



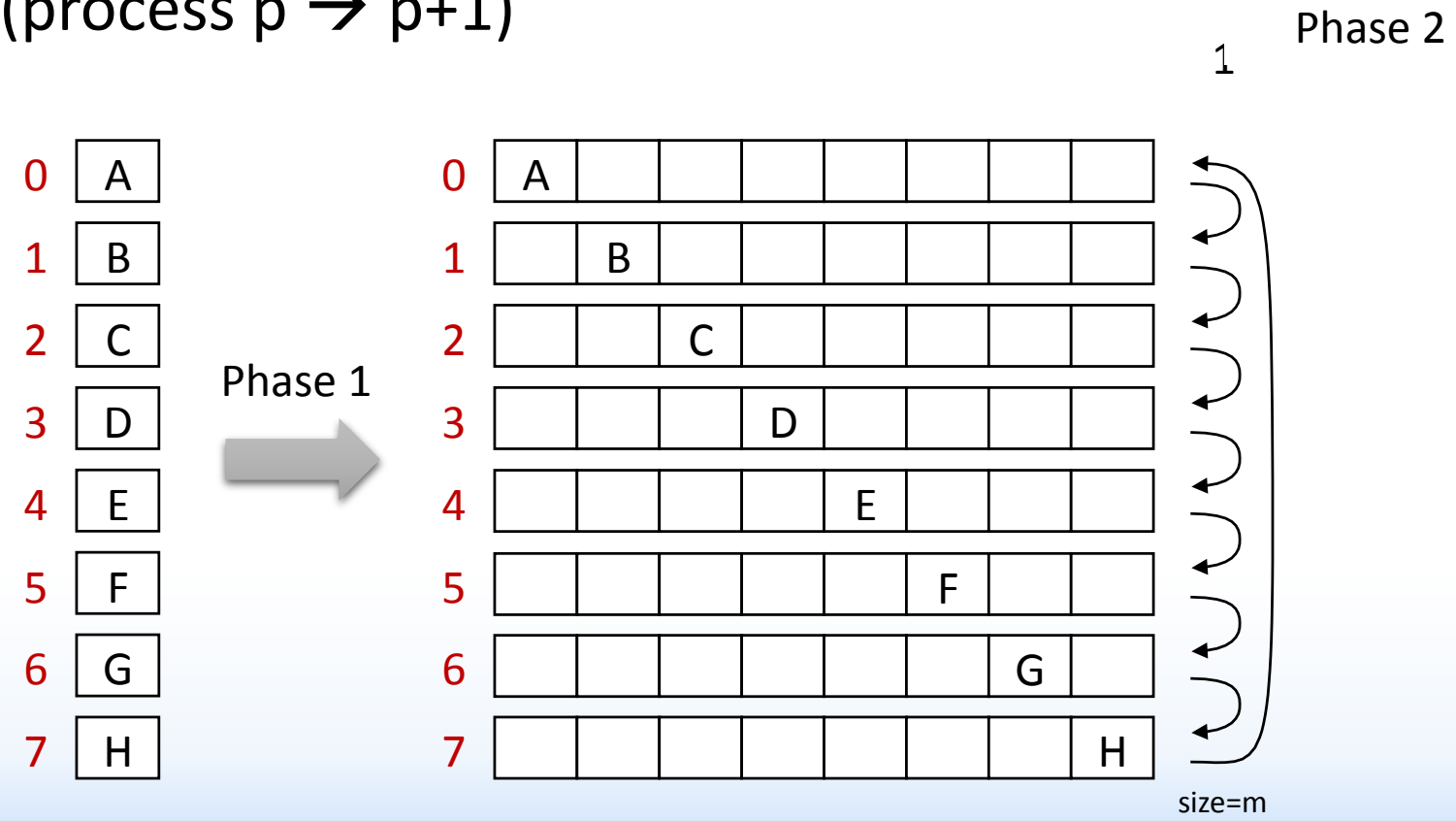
Modeling the influence of mapping

- *MPI_Allgather* with Ring: phase 2: P-1 send stages (process p → p+1)



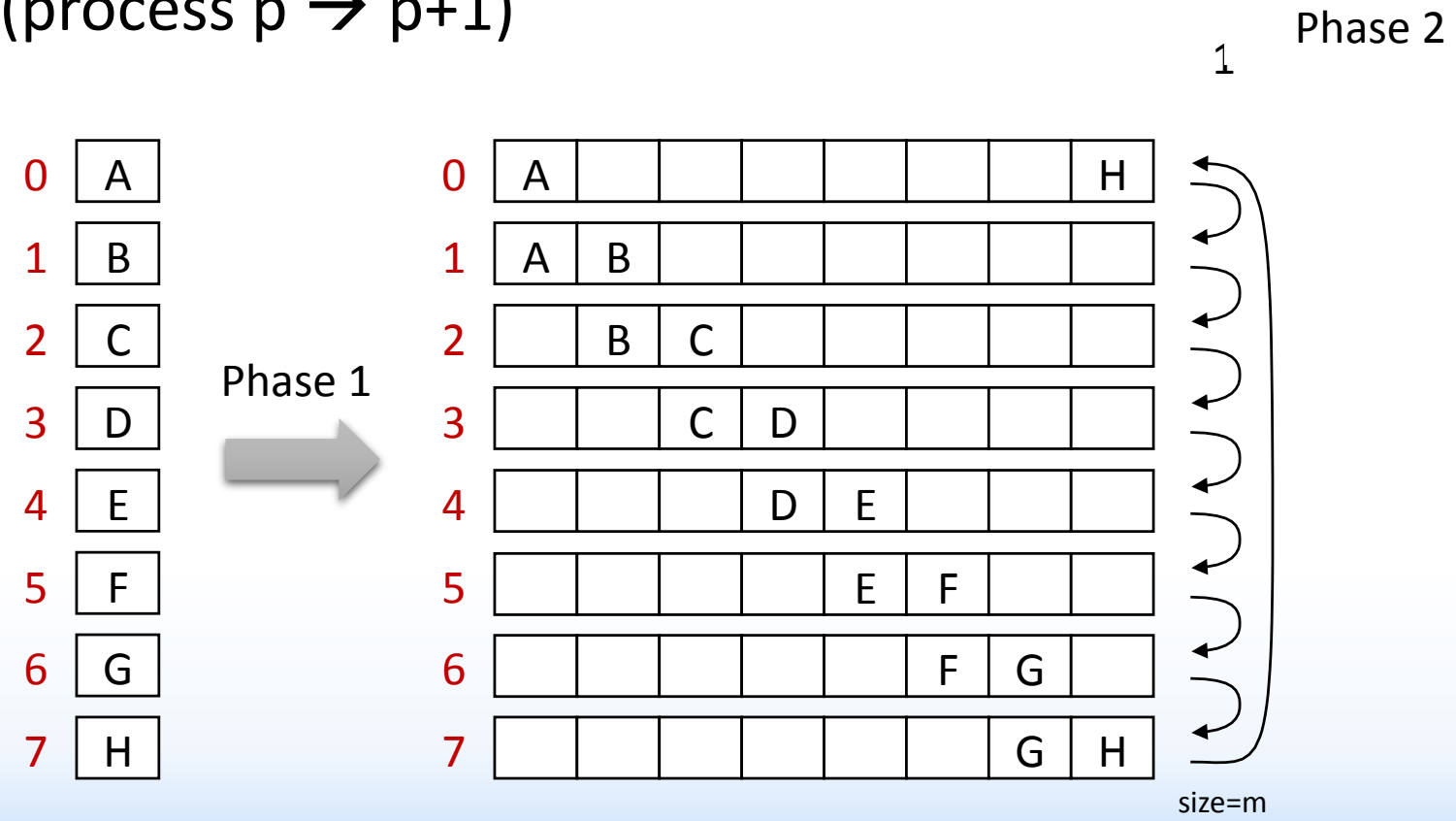
Modeling the influence of mapping

- *MPI_Allgather* with Ring: phase 2: P-1 send stages (process p → p+1)



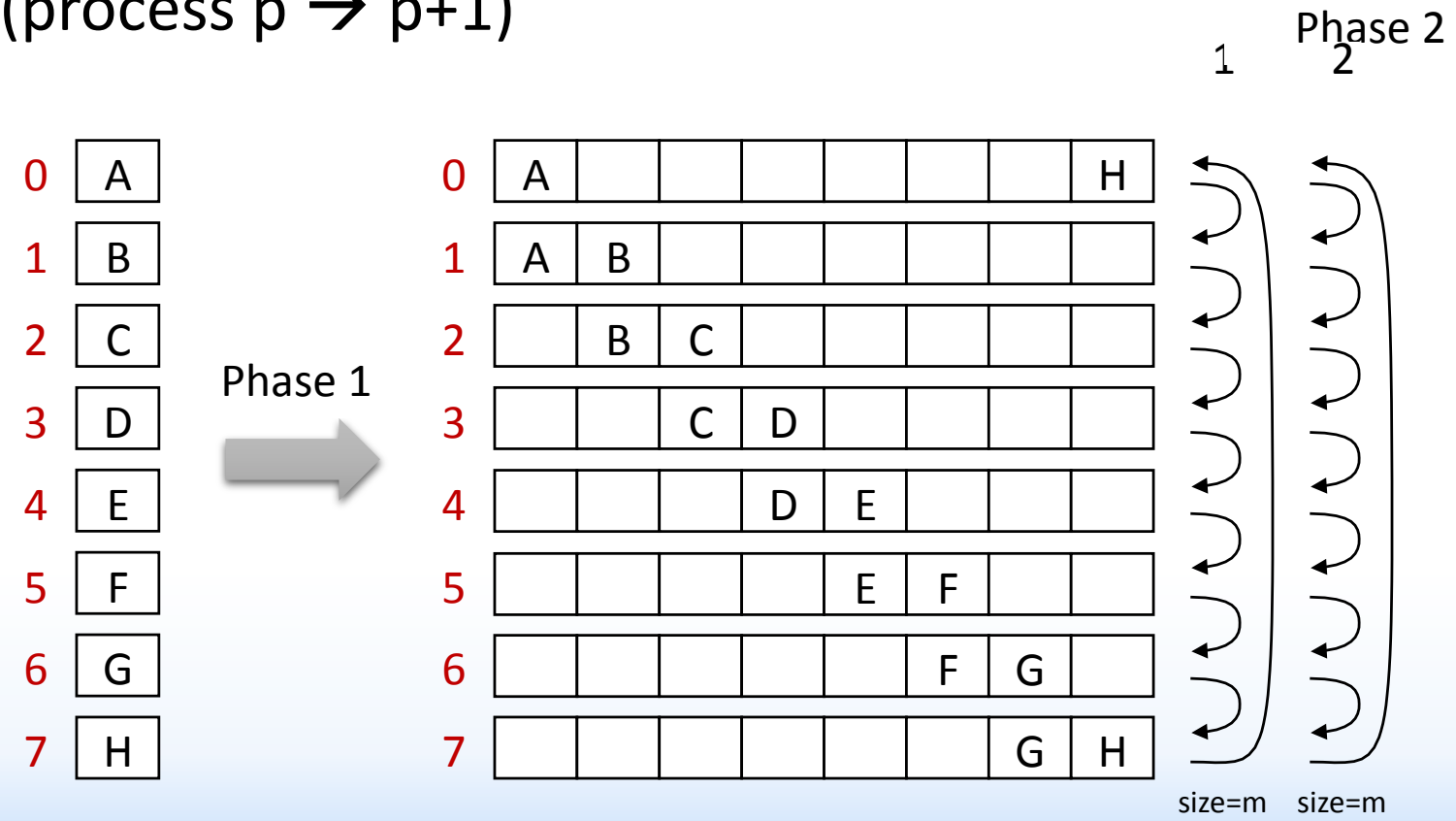
Modeling the influence of mapping

- *MPI_Allgather* with Ring: phase 2: P-1 send stages (process p → p+1)



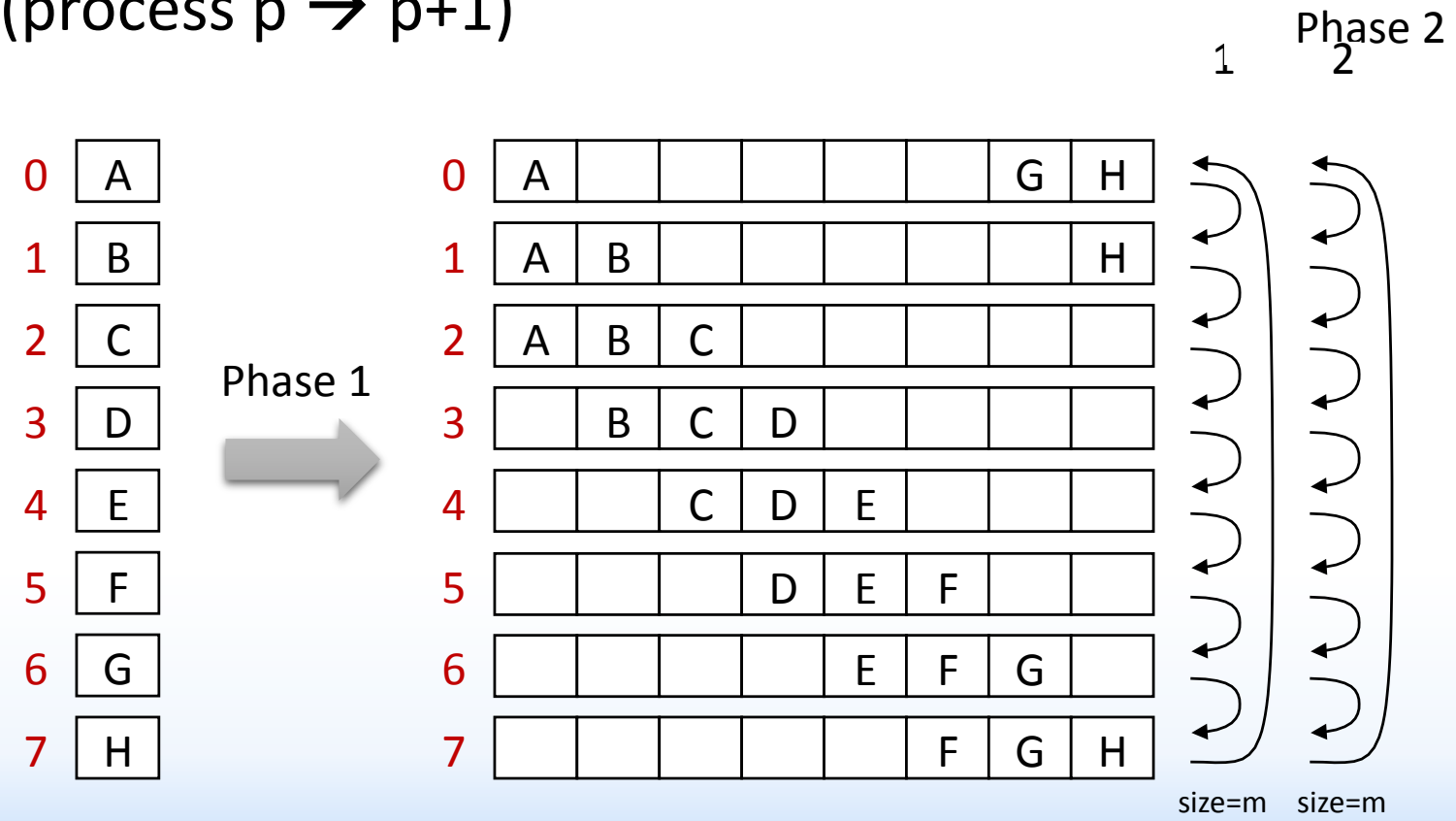
Modeling the influence of mapping

- *MPI_Allgather* with Ring: phase 2: P-1 send stages (process p → p+1)



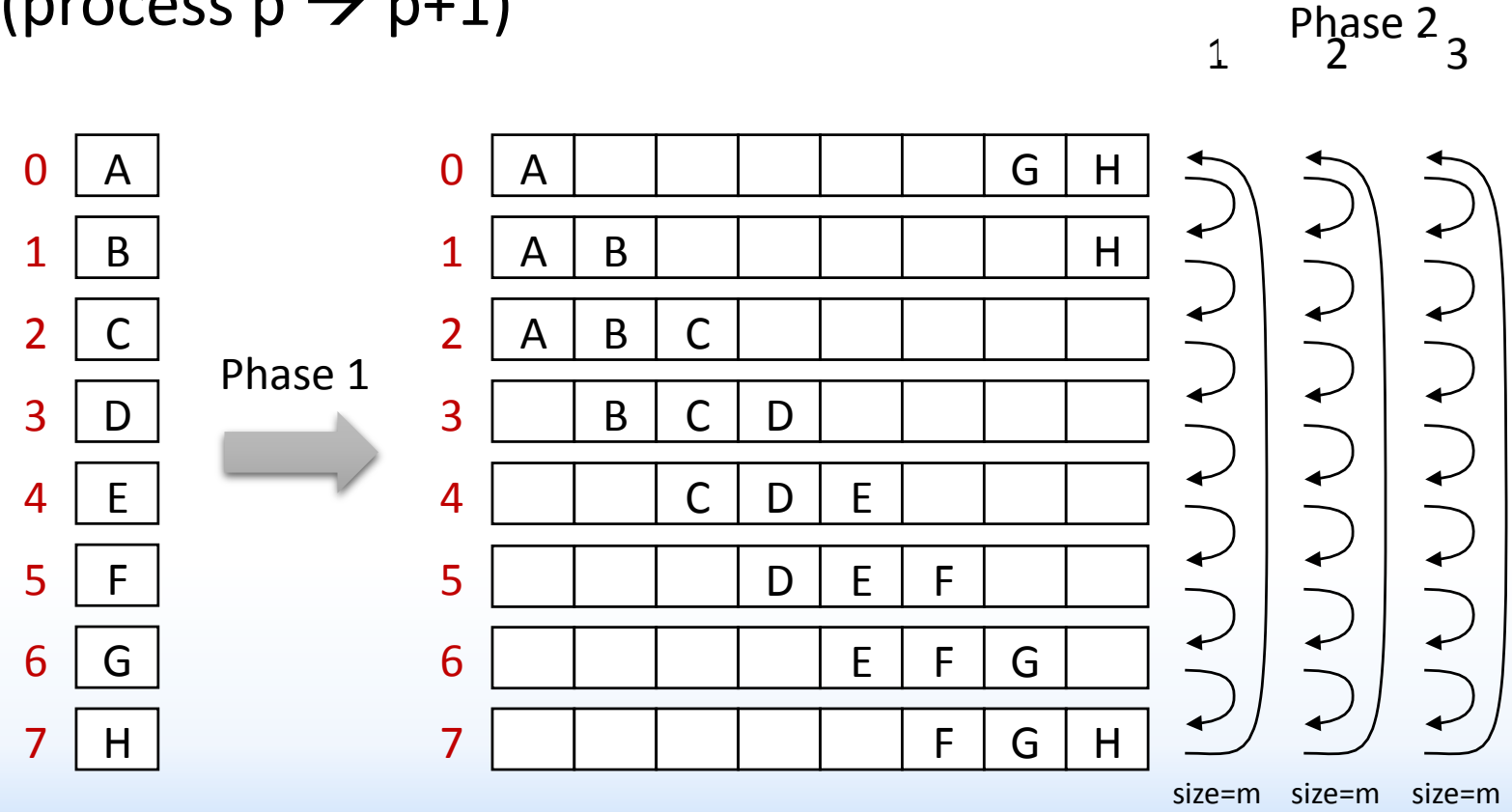
Modeling the influence of mapping

- *MPI_Allgather* with Ring: phase 2: P-1 send stages (process p → p+1)



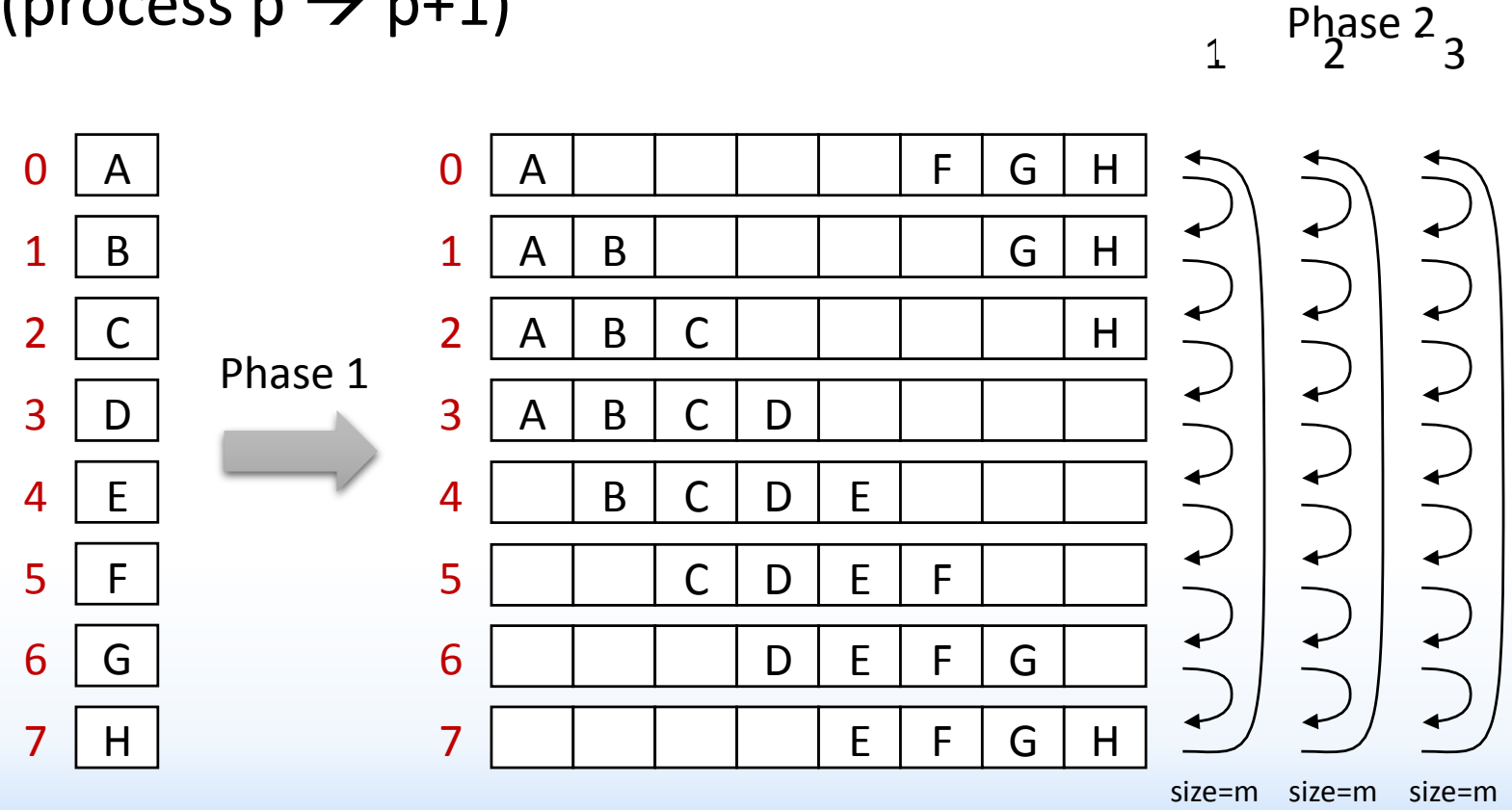
Modeling the influence of mapping

- *MPI_Allgather* with Ring: phase 2: P-1 send stages (process p → p+1)



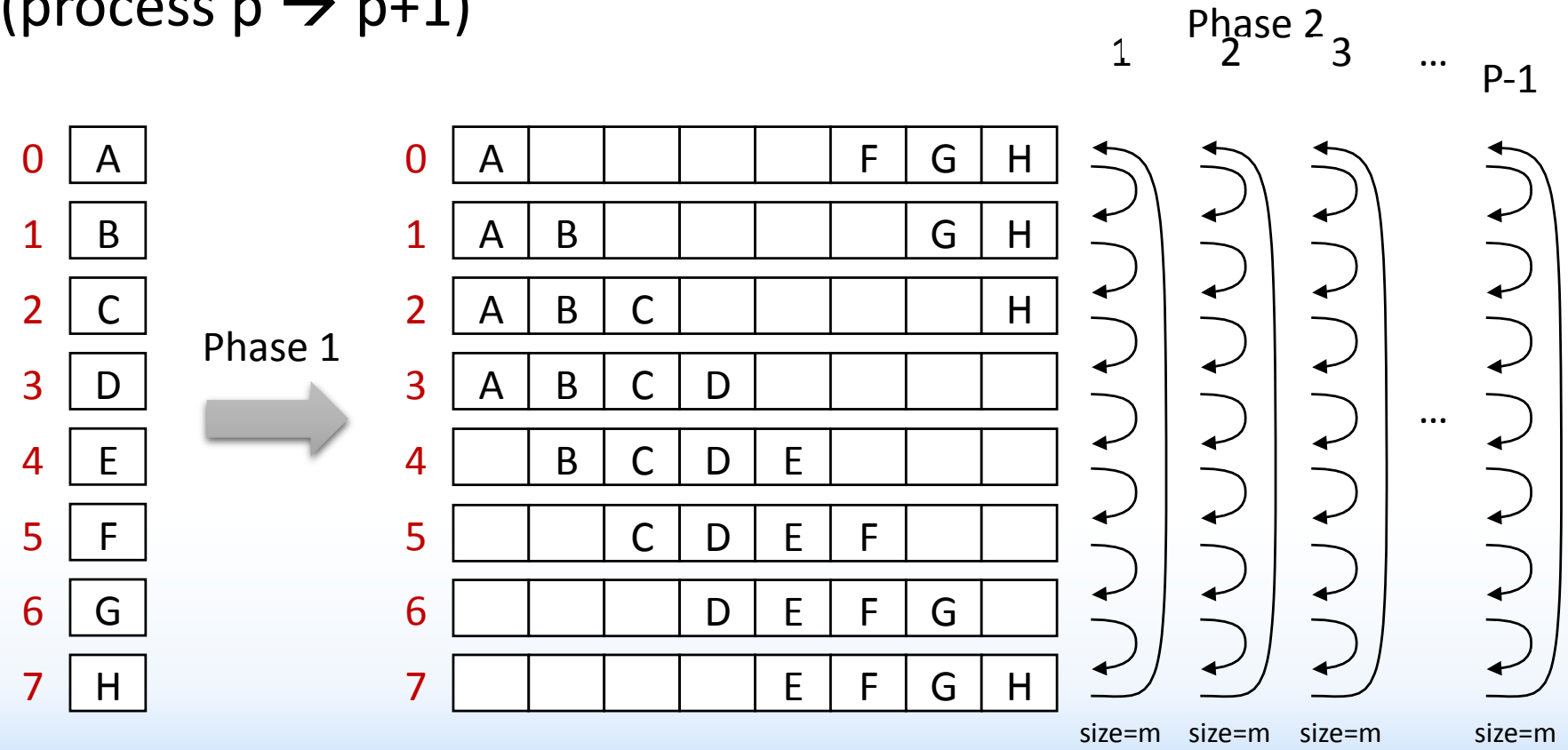
Modeling the influence of mapping

- *MPI_Allgather* with Ring: phase 2: P-1 send stages (process p → p+1)



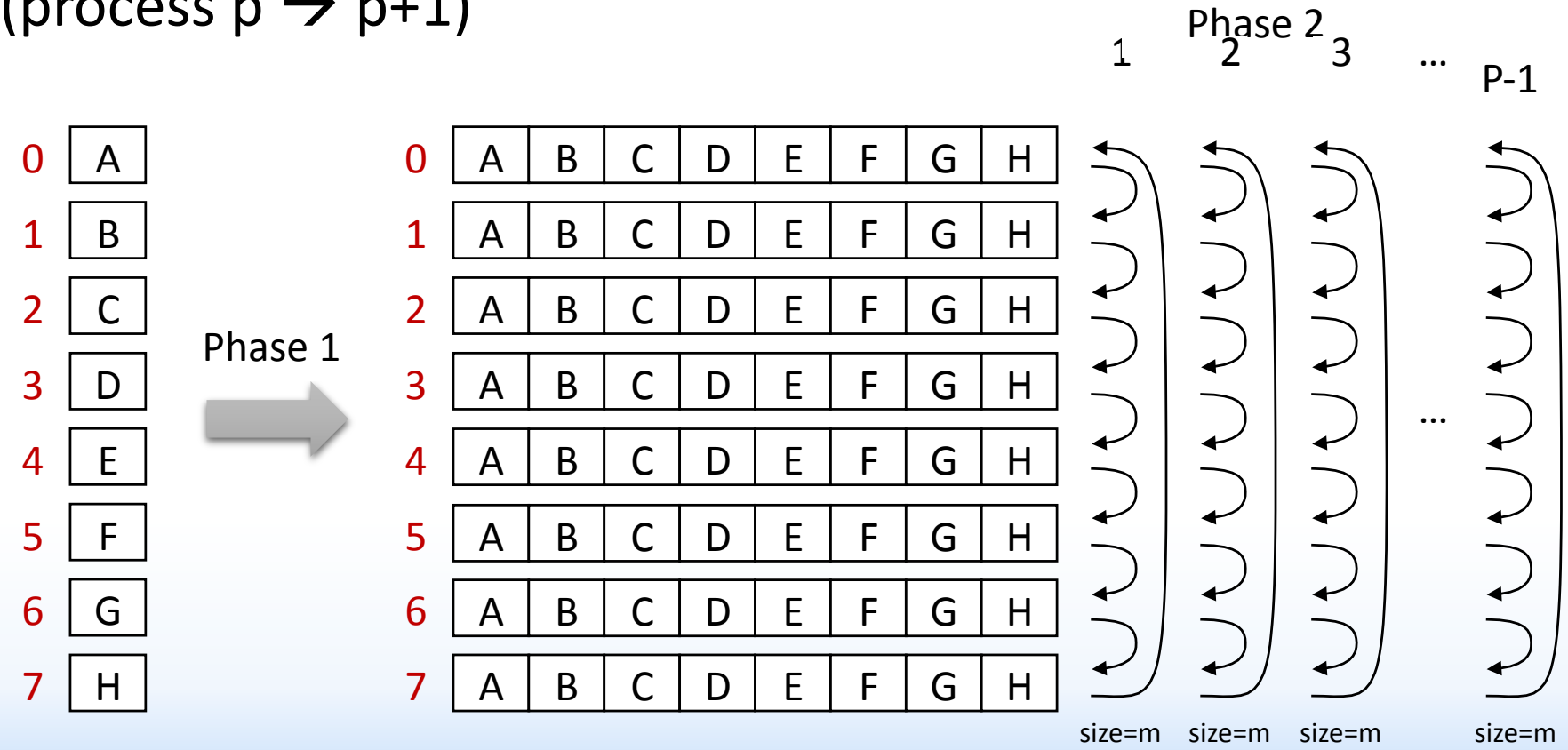
Modeling the influence of mapping

- *MPI_Allgather* with Ring: phase 2: P-1 send stages (process p → p+1)



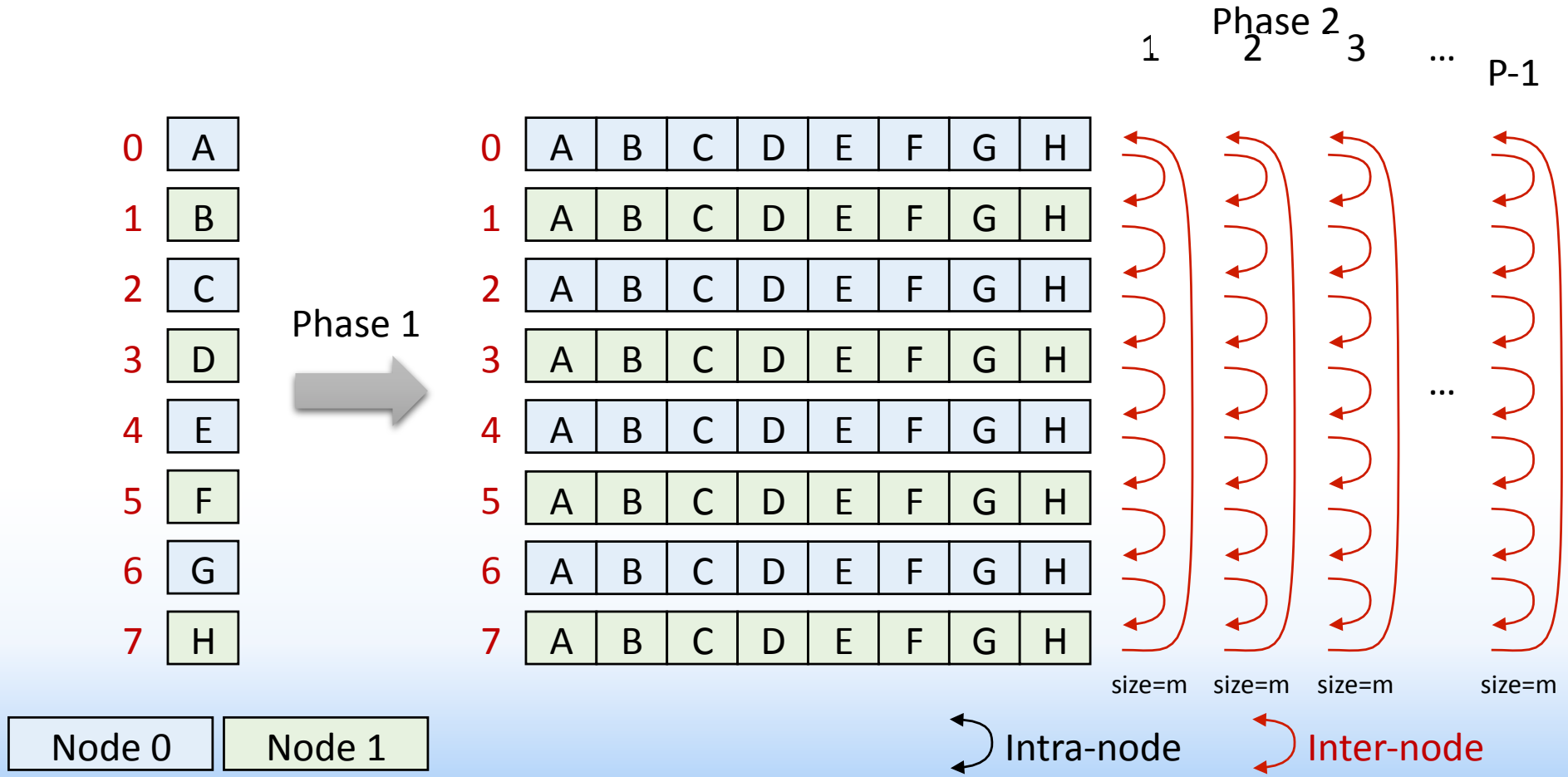
Modeling the influence of mapping

- *MPI_Allgather* with Ring: phase 2: P-1 send stages (process p → p+1)



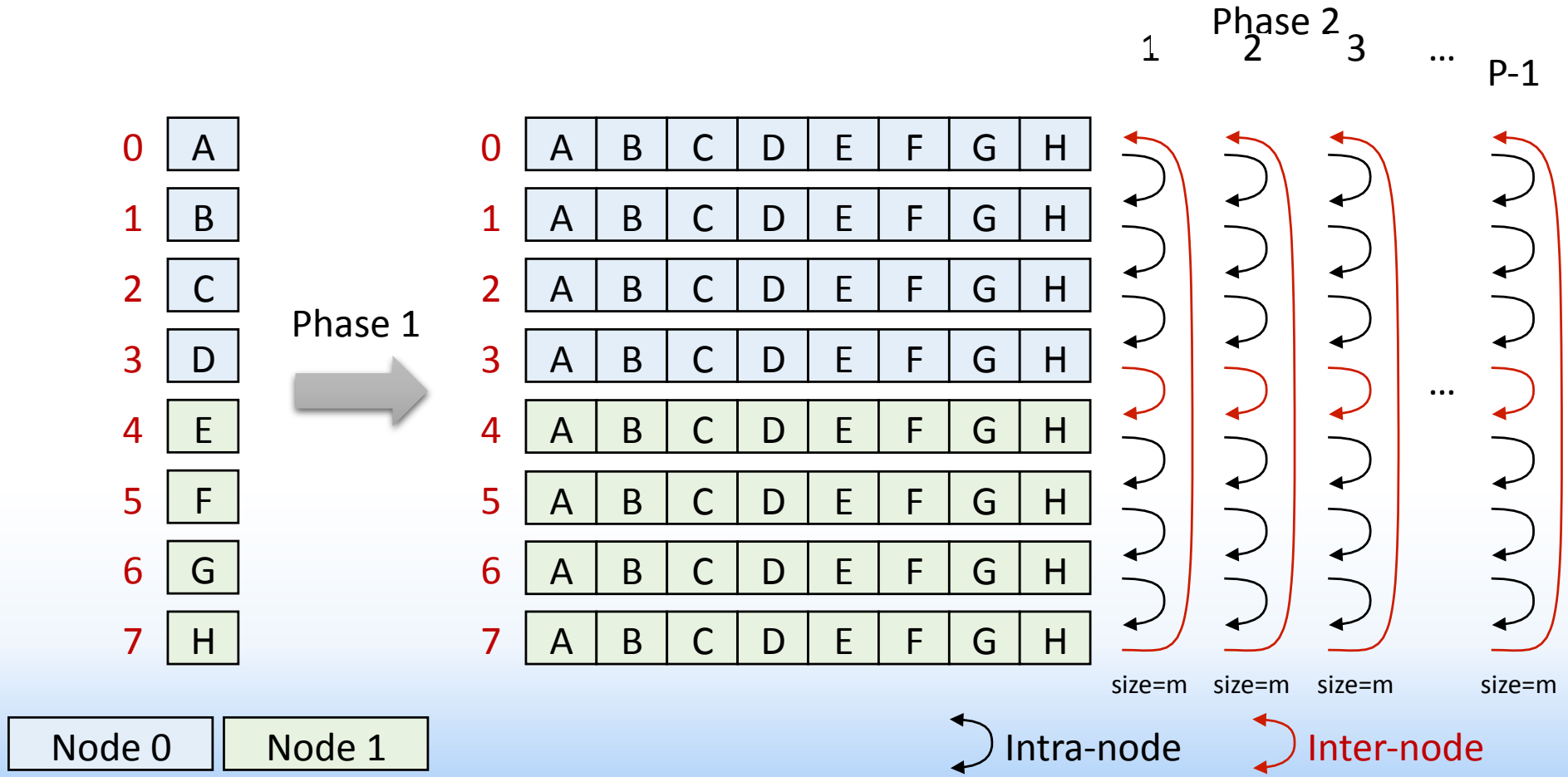
Modeling the influence of mapping

- *MPI_Allgather* with Ring: multi node with **RR mapping**



Modeling the influence of mapping

- *MPI_Allgather* with Ring: multi node with **SEQ mapping**



Modeling the influence of mapping

- Visually we realize:
  - *MPI_Allgather - RDA*: better with RR mapping
  - *MPI_Allgather - Ring*: better with SEQ mapping

- Formally demonstrated with LogGP:
  - *P processes, M nodes, Q processes/node*
  - *m message size large enough.*
  - *G is time interval between two bytes (G_0=Sh.M., G_1=Net.).*

LogGP cost	Overall intra-node	Overall inter-node
RDA SEQ	$(Q-1)\ m\ G_0$	$(M-1)\ Q\ m\ G_1$
RDA RR	$(Q-1)\ M\ m\ G_0$	$(M-1)\ m\ G_1$

Run-time reordering

- Change mapping from RR to SEQ or vice versa when advantageous.

  - No migration: each rank **behaves** as another in collective algorithm (send/recv. pattern).

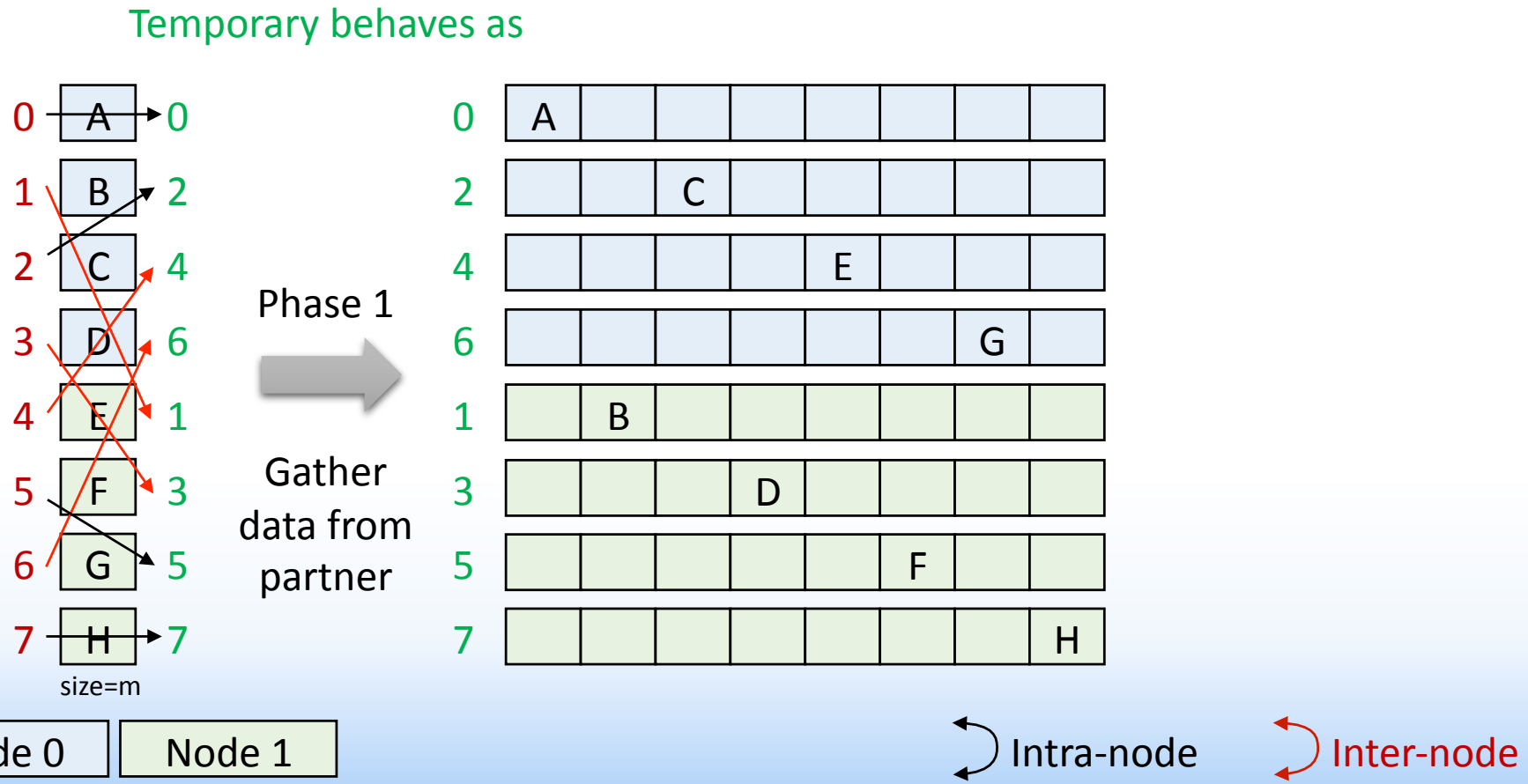
  - For a given location (node,core) transform *p* from RR rank to SEQ rank, and backwards:
    - $f_{SEQ \rightarrow RR}(p) = ((p \times M) \% P) + \lfloor p/Q \rfloor$
    - $f_{RR \rightarrow SEQ}(p) = ((p \times Q) \% P) + \lfloor p/M \rfloor$
    - $f_{RR \rightarrow SEQ}(f_{RR \rightarrow SEQ}(p)) = p$

    - *What is my new rank to behave as?* *What is the new rank of **p**?*
      Which rank is behaving as me? *What is the original rank of **p**?*

  - Not free: each rank needs its new rank's data to start algorithm.

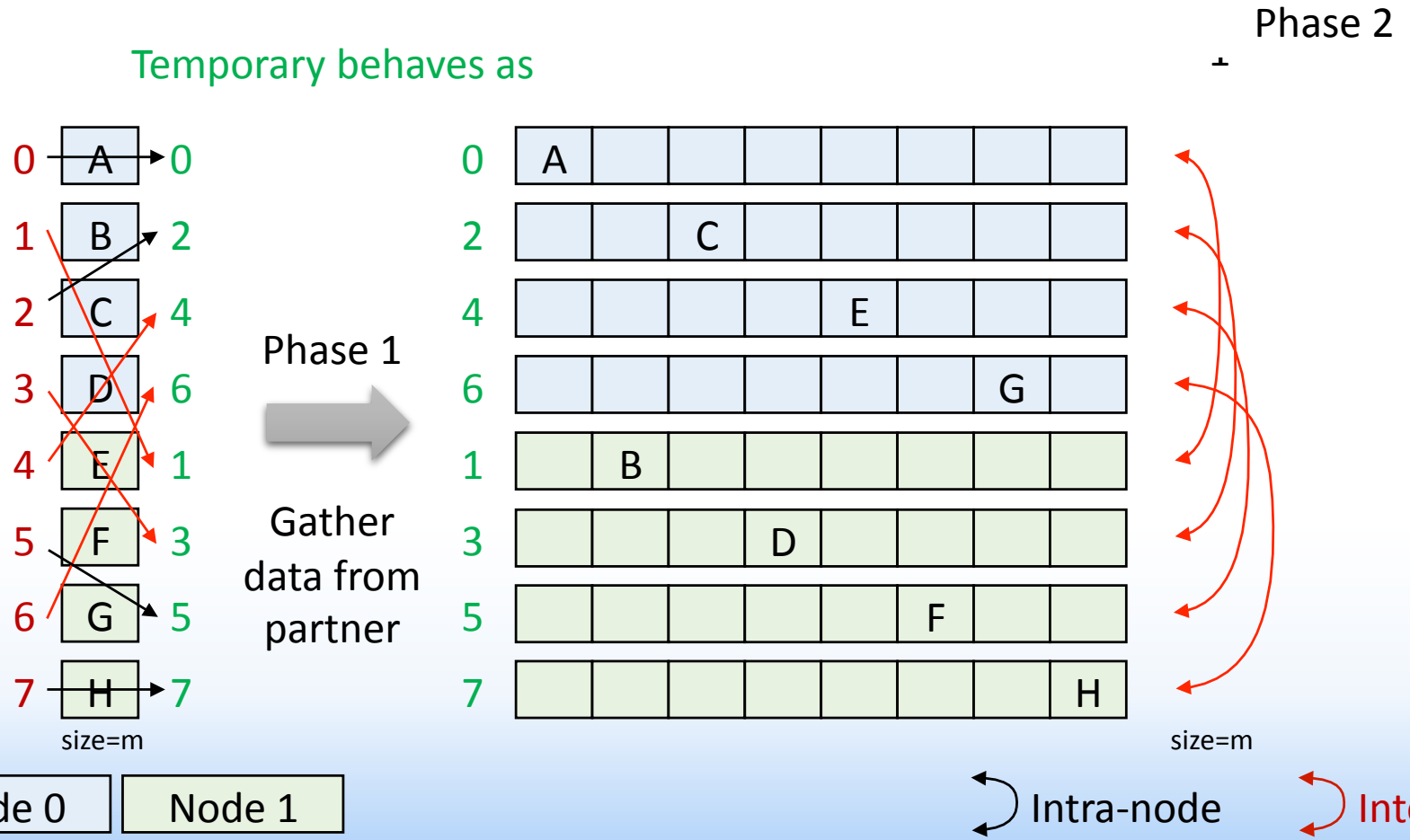
Run-time reordering

- *MPI_Allgather* with RDA and SEQ (+remapping)



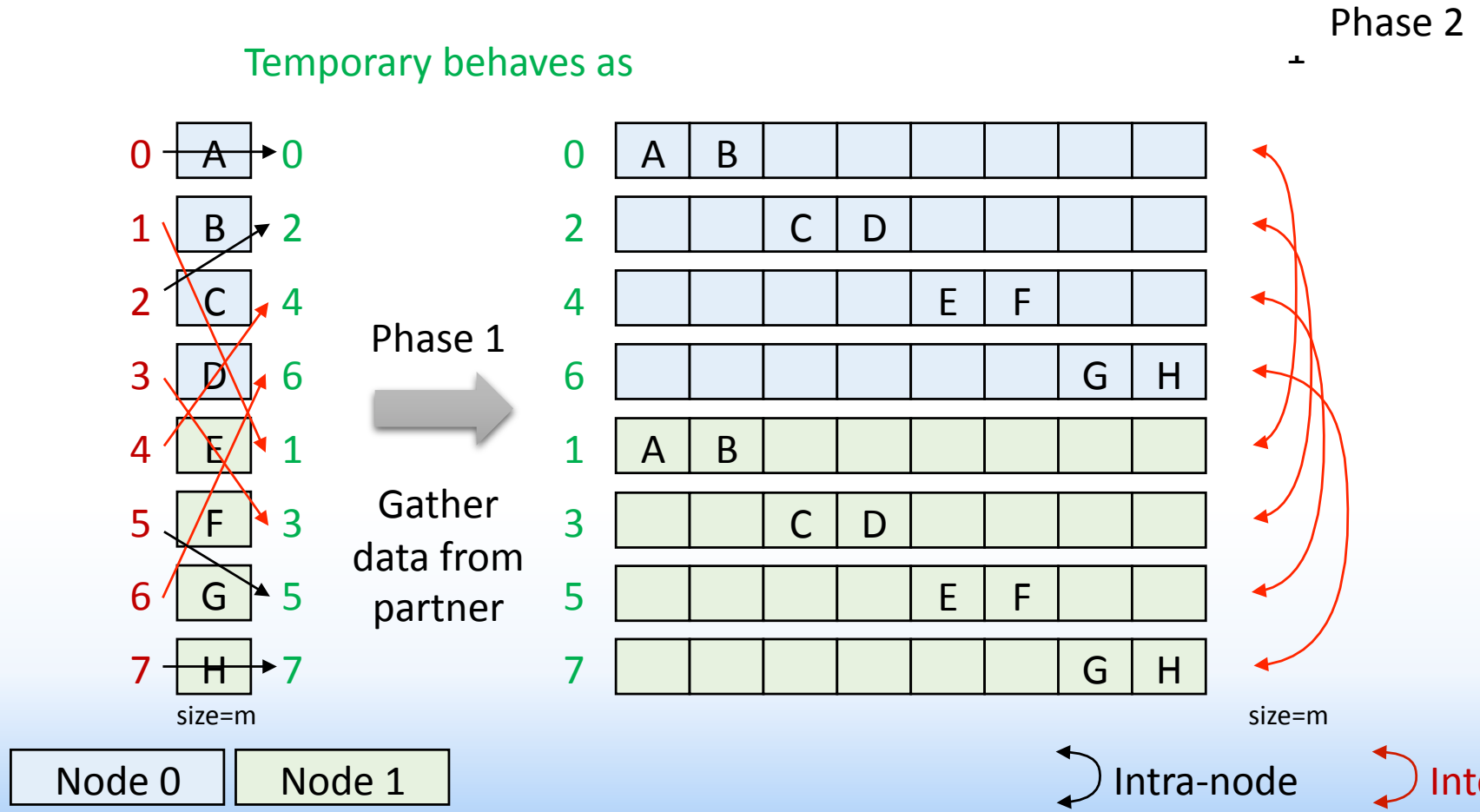
Run-time reordering

- *MPI_Allgather* with RDA when SEQ (+remapping)



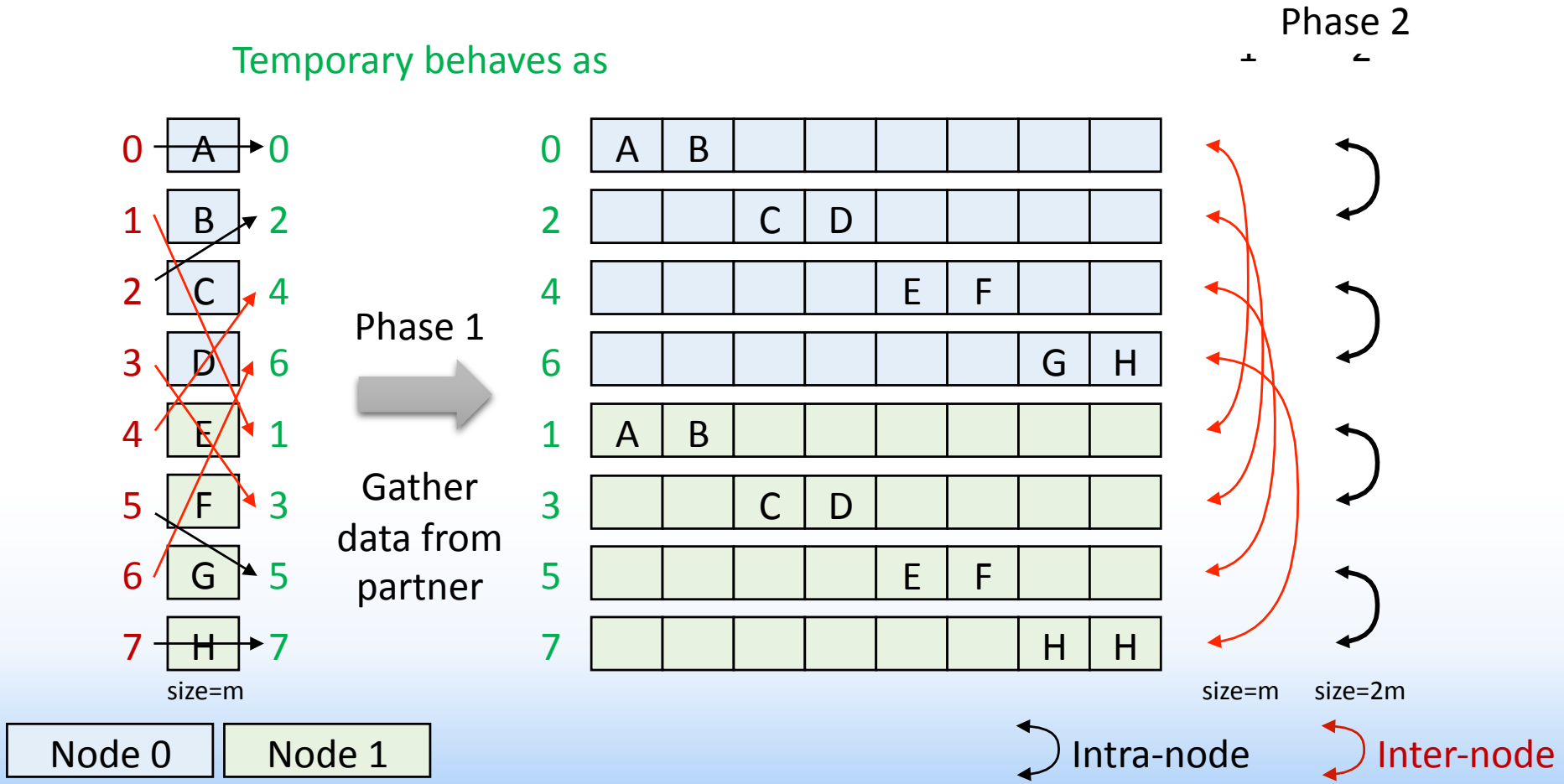
Run-time reordering

- *MPI_Allgather* with RDA when SEQ (+remapping)



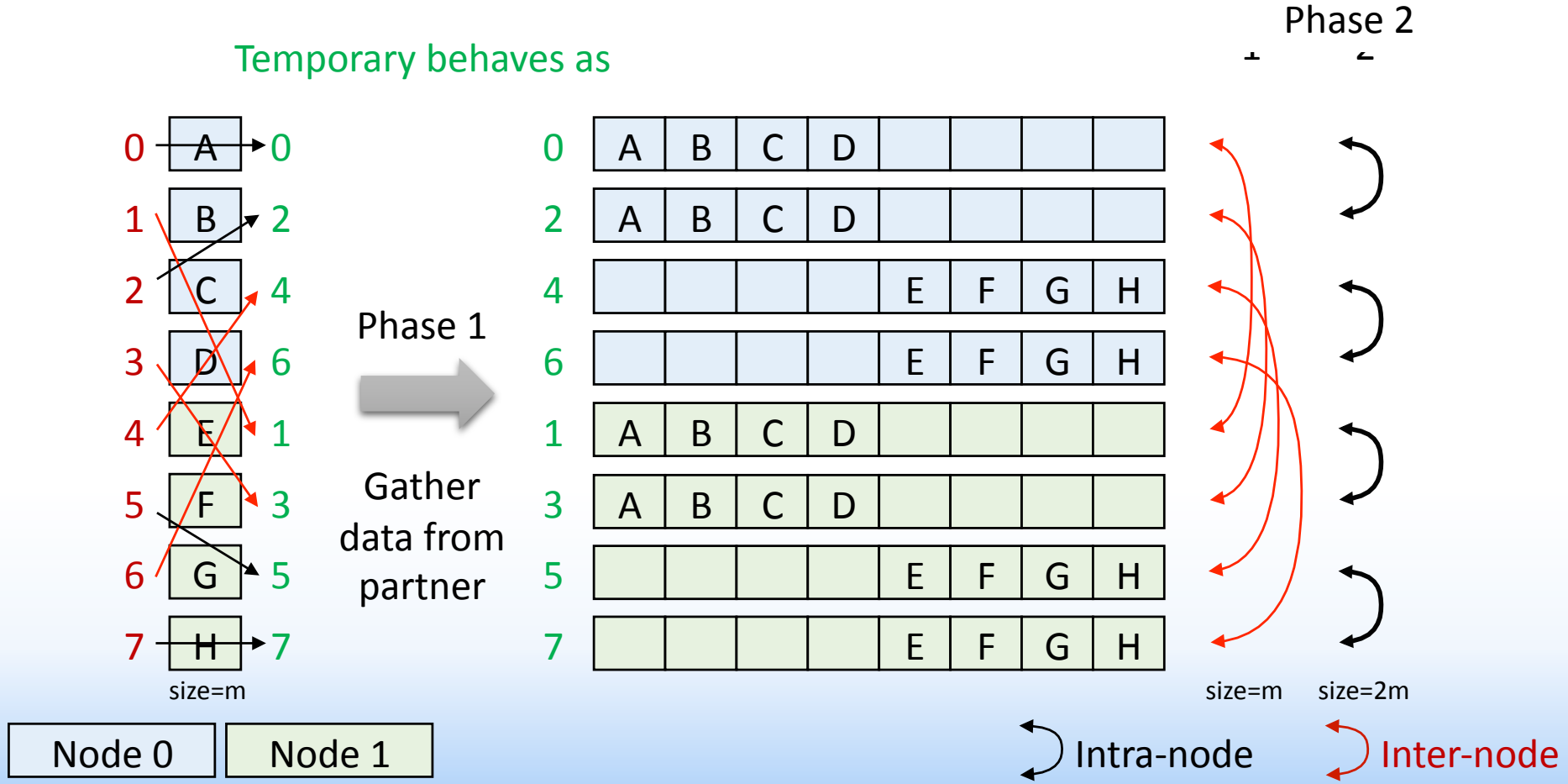
Run-time reordering

- *MPI_Allgather* with RDA when SEQ (+remapping)



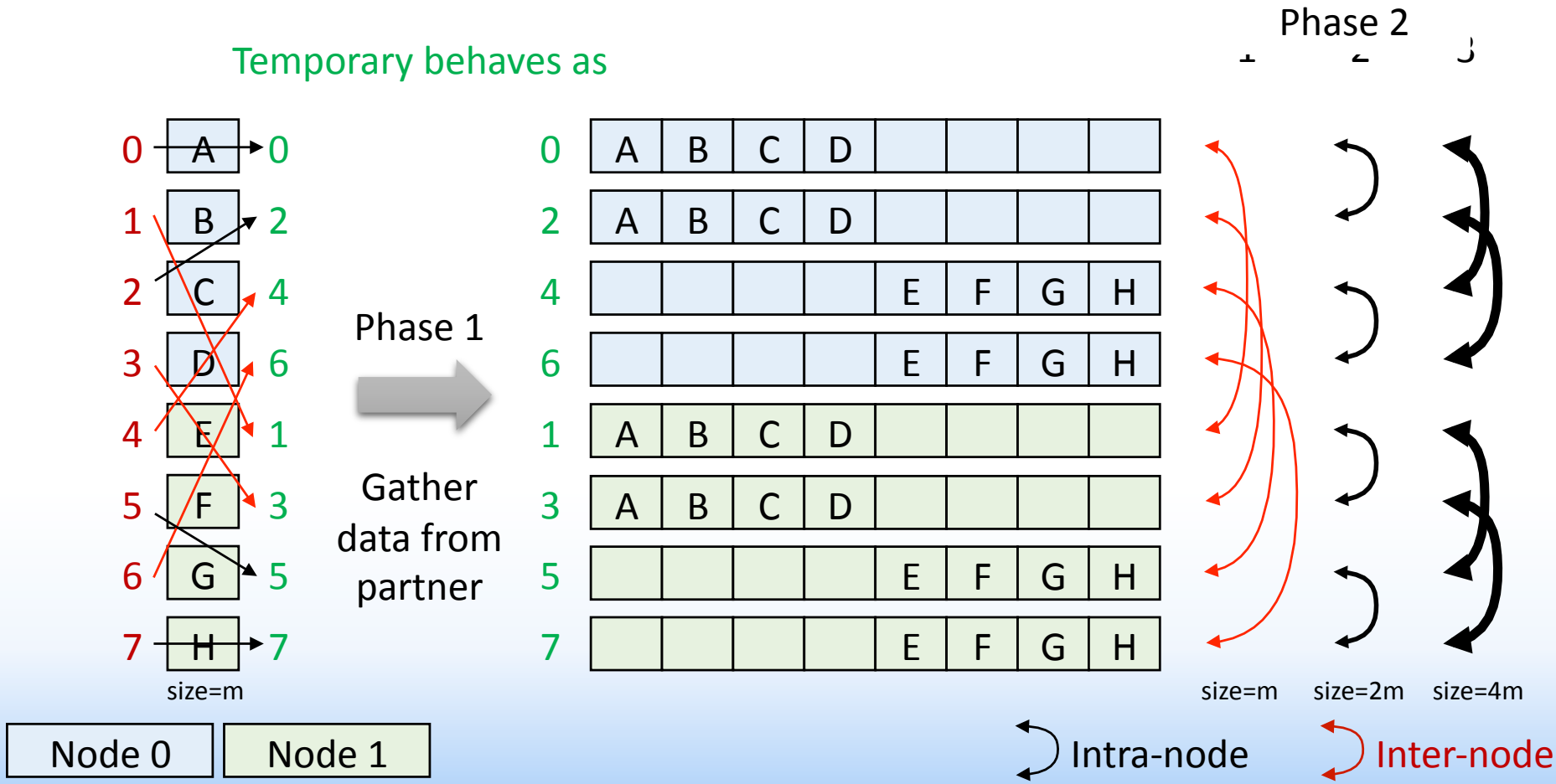
Run-time reordering

- *MPI_Allgather* with RDA when SEQ (+remapping)



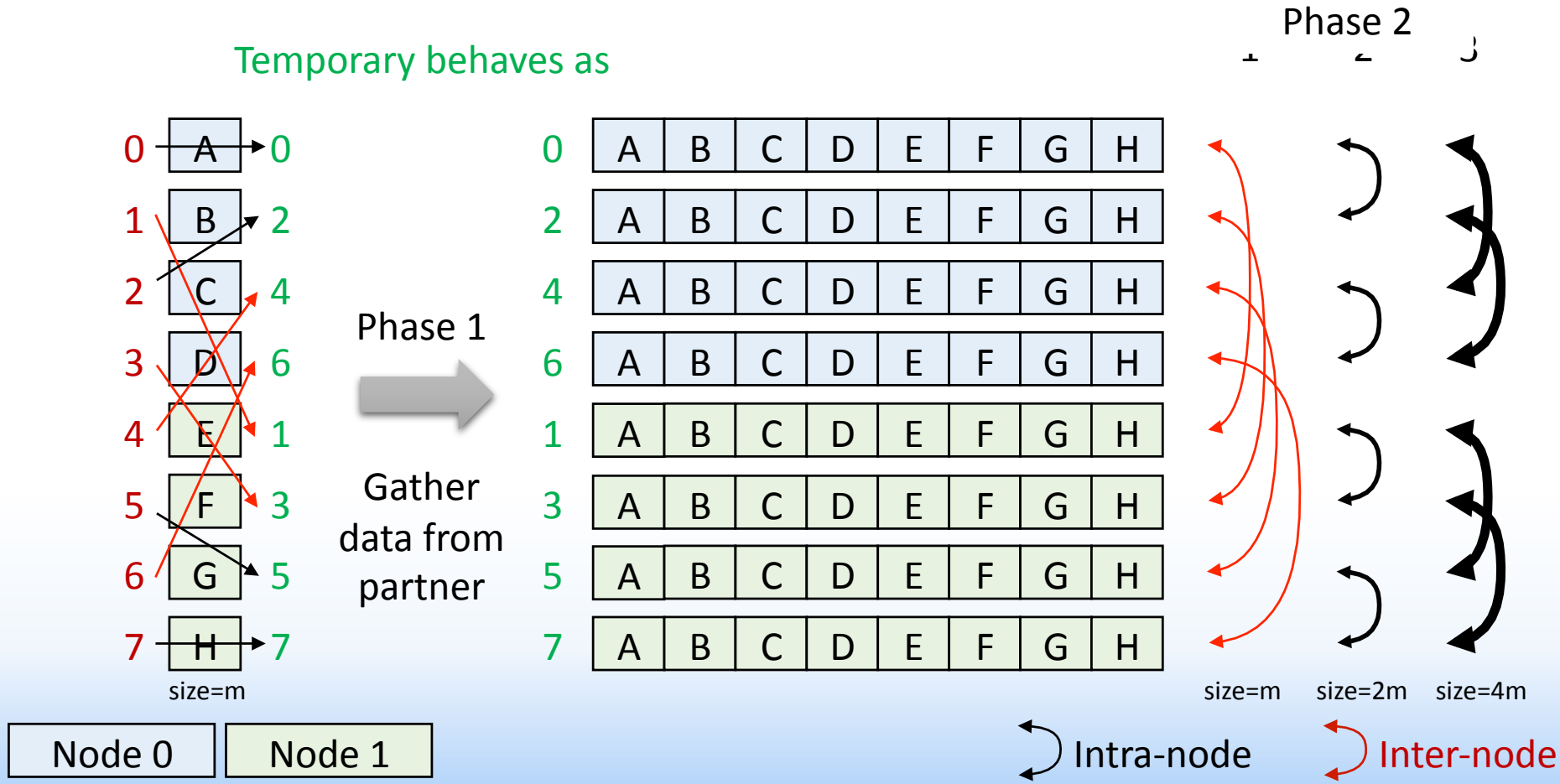
Run-time reordering

- *MPI_Allgather* with RDA when SEQ (+remapping)



Run-time reordering

- *MPI_Allgather* with RDA when SEQ (+remapping)



Run-time reordering

- Visually we realize:
  - *Phase 2 is optimal (works as RR)*
  - *Little penalty in Phase 1*

- Formally modelled with LogGP:
  - *P processes, M nodes, Q processes/node*
  - *m message size large enough.*
  - *G is time interval between two bytes (0=Sh.M., 1=Net.).*

LogGP cost	Overall intra-node	Overall inter-node
RDA SEQ	$(Q-1)\ m\ G_0$	$(M-1)\ Q\ m\ G_1$
RDA RR	$(Q-1)\ M\ m\ G_0$	$(M-1)\ m\ G_1$
RDA*	$(Q-1)\ M\ m\ G_0$	$M\ m\ G_1$

Performance evaluation

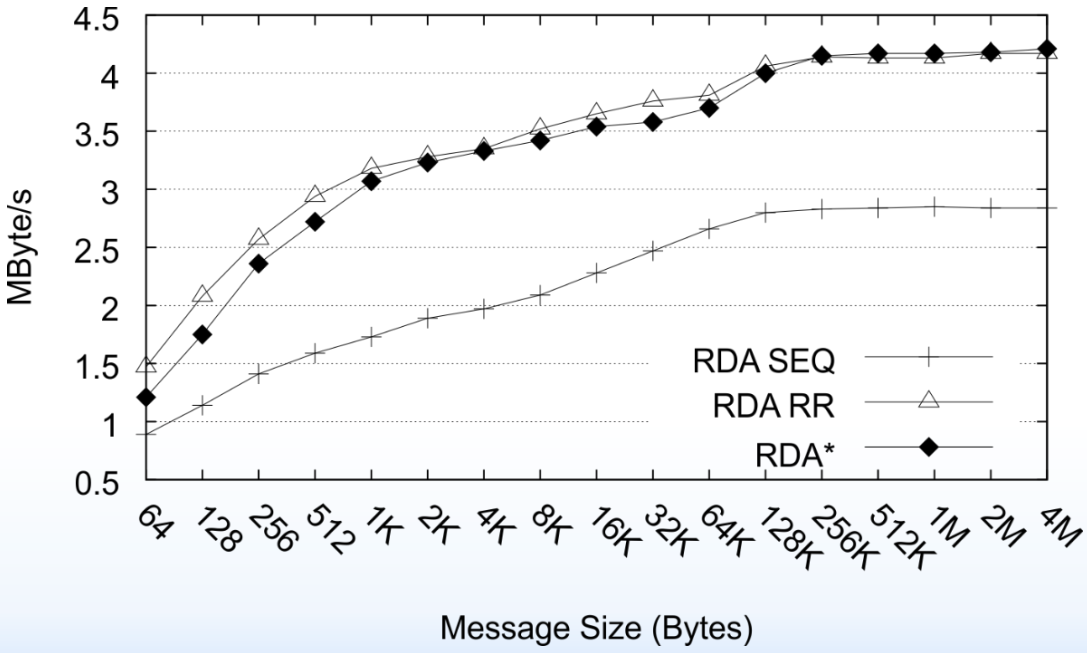
- IMB (Intel MPI Benchmark) running:
  - MPI_Allgather with RDA algorithm.
  - MPI_Allgather with Ring algorithm.
  - MPI_Allgather with Neighbor Exchange algorithm.
  - MPI_Broadcast with Binomial Tree algorithm.
  - MPI_Allreduce with Ring algorithm.

- Mappings SEQ, RR and remapping

- Cluster of 16 nodes × (2×6) cores + QDR infiniband (P=192, M=16, Q=12)

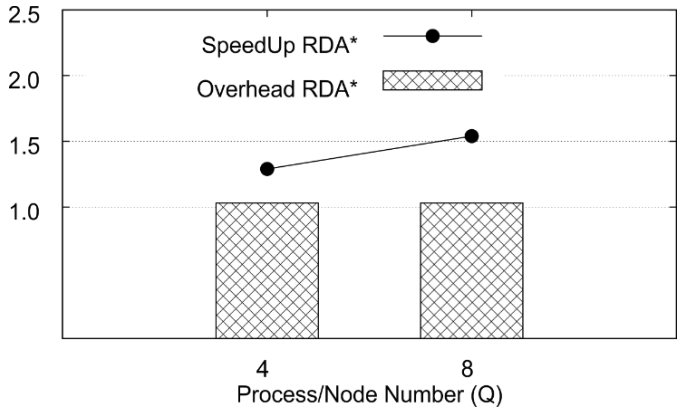
Performance evaluation

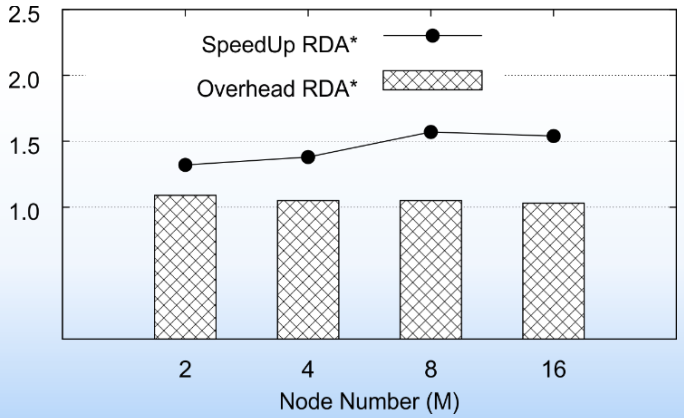
- MPI_Allgather with RDA
  (P=128, M=16, Q=8)



Bandwidth performance by message size

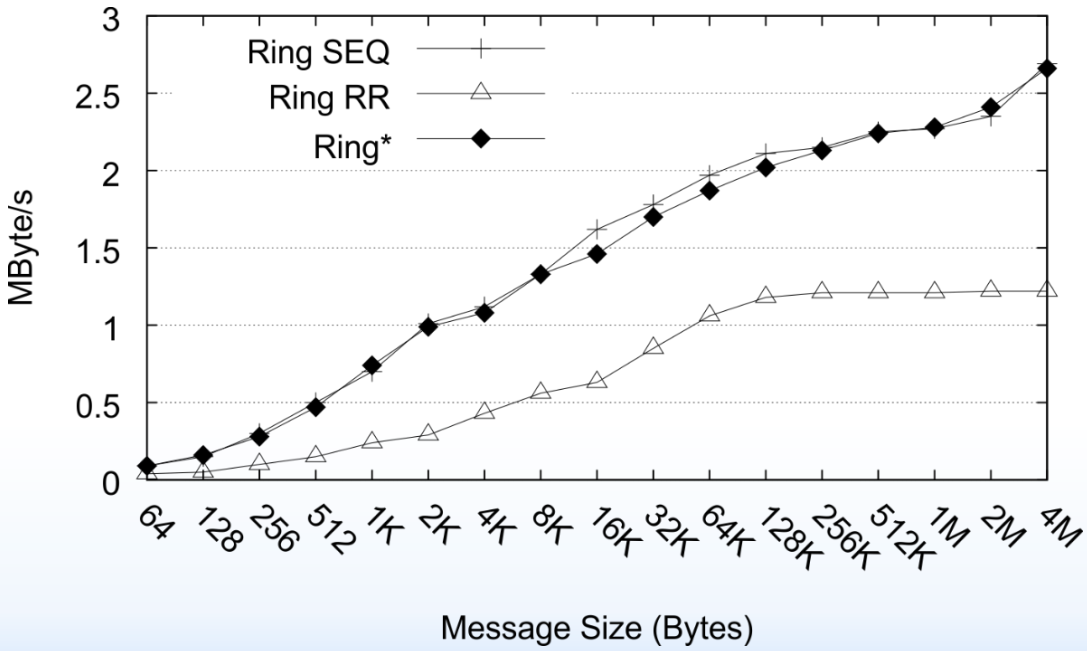
Speedup and overhead





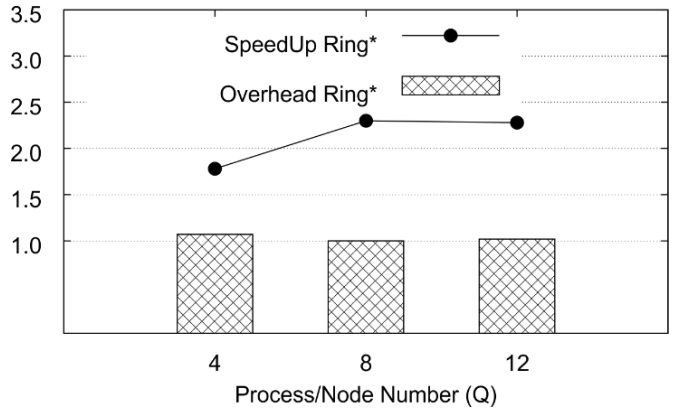
Performance evaluation

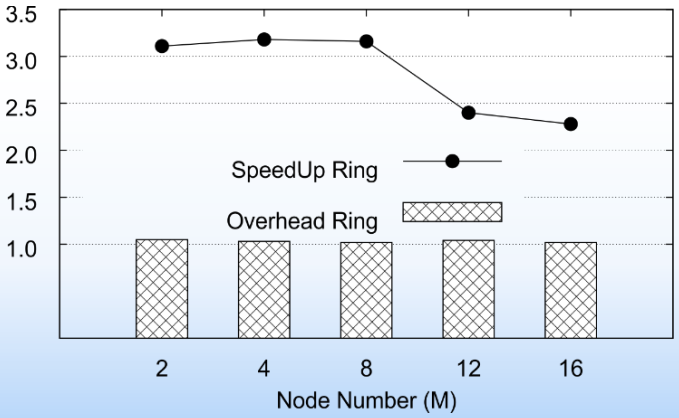
- MPI_Allgather with Ring
  (P=128, M=16, Q=12)



Bandwidth performance by message size

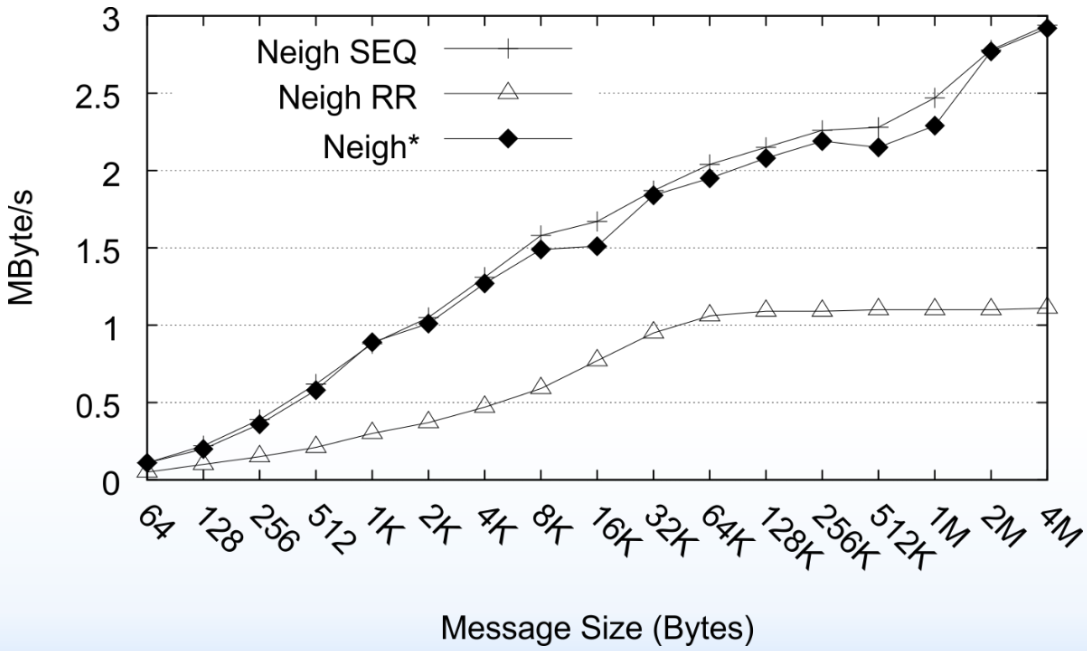
Speedup and overhead





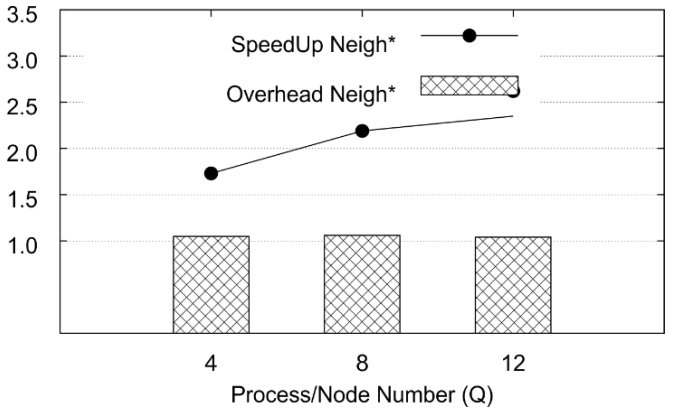
Performance evaluation

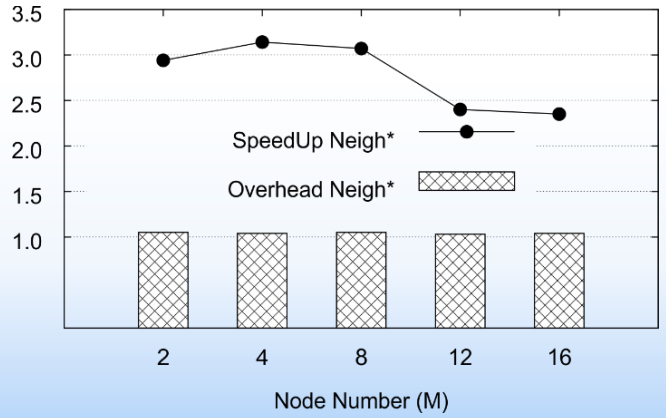
- MPI_Allgather with Neighbor Exchange
  (P=128, M=16, Q=12)



Bandwidth performance by message size

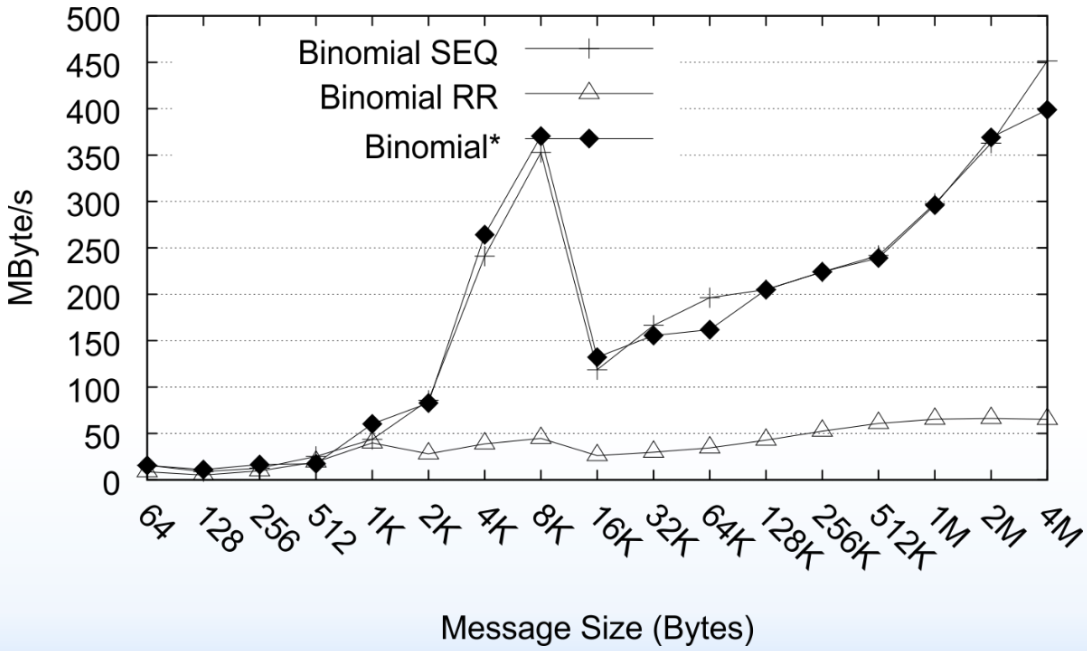
Speedup and overhead





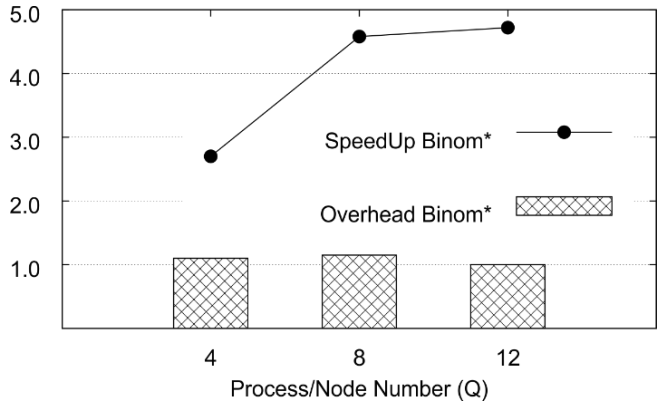
Performance evaluation

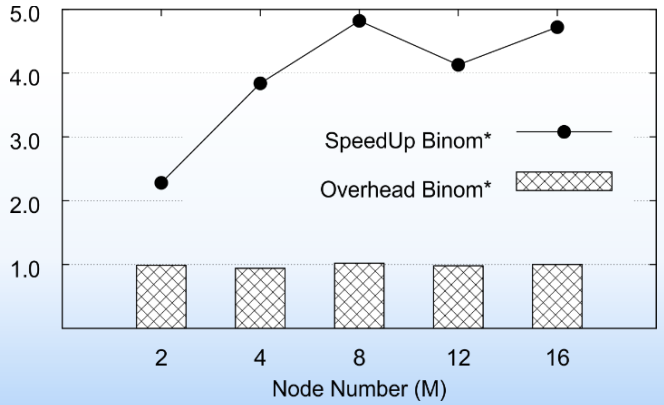
- MPI_Broadcast with Binomial Tree
  (P=128, M=16, Q=12)



Bandwidth performance by message size

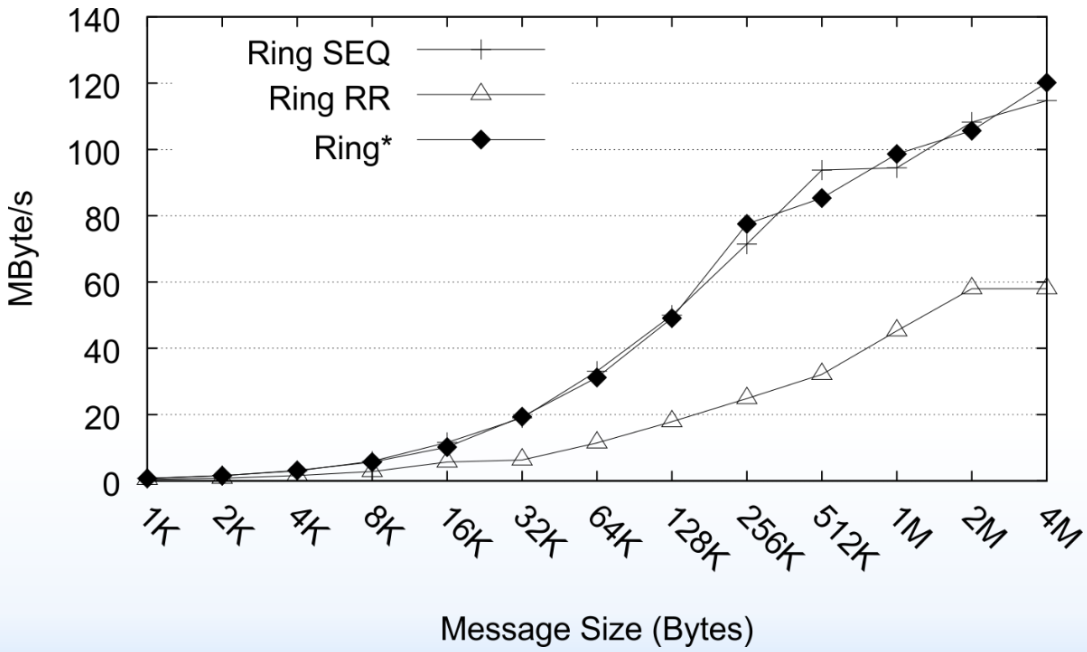
Speedup and overhead





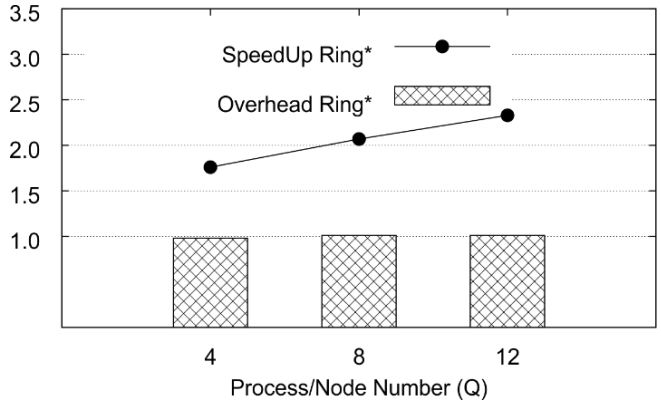
Performance evaluation

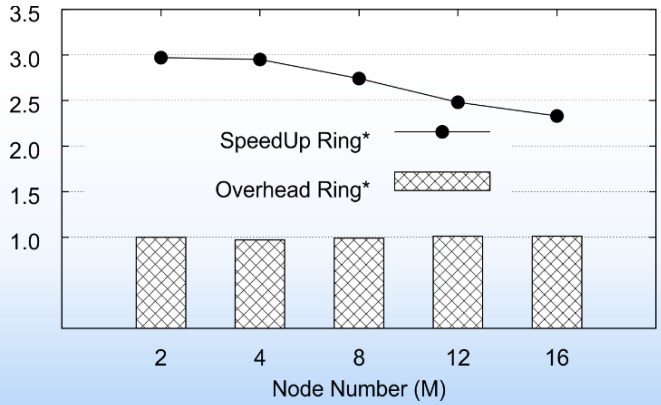
- MPI_Allreduce with Ring
  (P=128, M=16, Q=12)



Bandwidth performance by message size

Speedup and overhead





Conclusions

- Proved that rank-to-processor mapping has influence in performance.
  - Ring, Neighbour Exchange and Binomial Tree better with SEQ
  - RDA better with RR
- Switching between RR and SEQ when suitable reduces network traffic. Our rank reordering method:
  - Temporary reordering
  - On run time
  - Without migration
  - **Speedup 1.5× – 5×, negligible overhead**

Future work

- Current limitation: assumption of regular mapping (SEQ/RR)
  - Use of custom mapping

Node 0	0	1	5	7

Node 1	2	3	4	6

  - New communicators using a subset of processes

Node 0	0			

Node 1	1	2		3

- Find (near-)optimal mapping for non-regular mappings
  - Analytic models to evaluate performance of remapping.
  - Each rank needs to know the whole remapping (scalability?)

Thank you!

Jesús M. Álvarez (llorente@unex.es)

Este trabajo está financiado por la Junta de Extremadura y por la Unión Europea (fondos FEDER) a través de los fondos de ayuda a grupos de investigación GR1517

This work is financed by the Junta de Extremadura and the European Union (ERDF funds) through the support funds to research groups GR1517