HIGH PERFORMANCE COMPUTING

Exercise 1

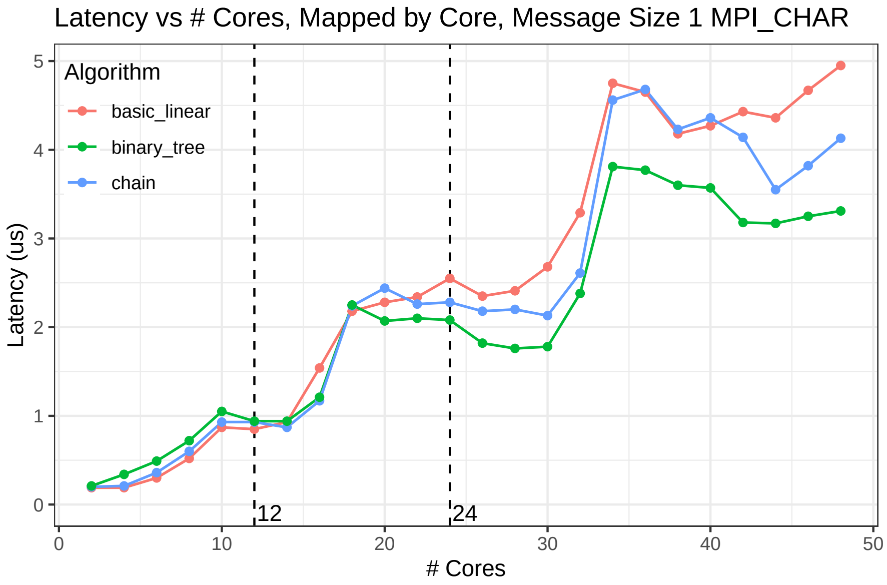
Barrasso Marco

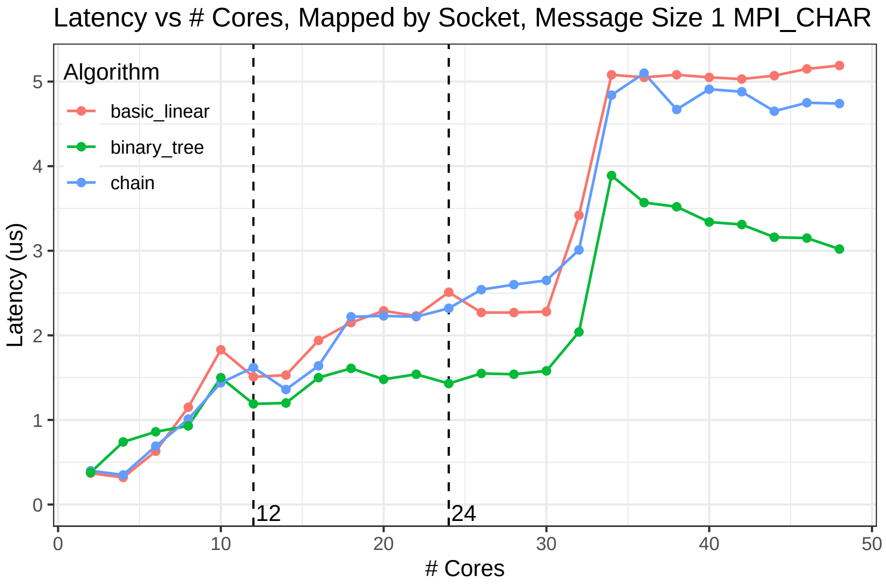
May 13, 2024

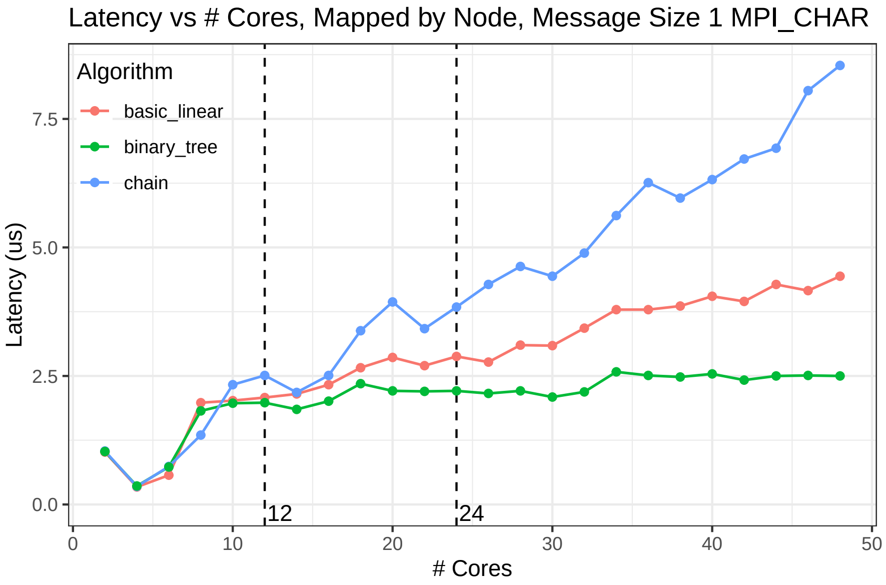
Overview

- Aim of the project
  - Comparing different OpenMPI algorithms for collective operations
- Collective operations and algorithms
  - Broadcast operation: linear, chain, binary tree algorithms
  - Gather operation:linear, binomial tree
- Analysis
  - Algorithms comparison
  - Allocation type comparison
  - Performance model for the latency
- Metodology
  - OSU Benchmark to estimate the latency
  - Bash script to obtain data using 2 THIN nodes on ORFEO cluster
  - R for data analysis

Algorithms Comparison

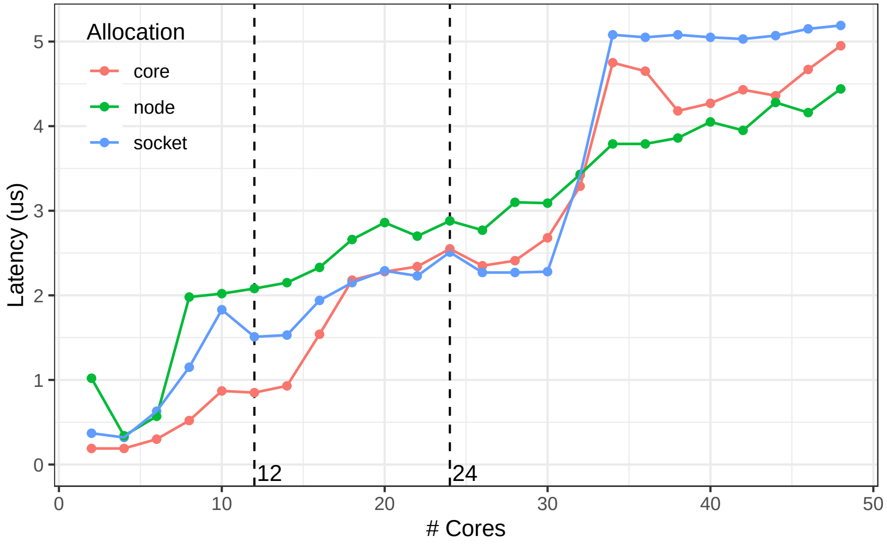




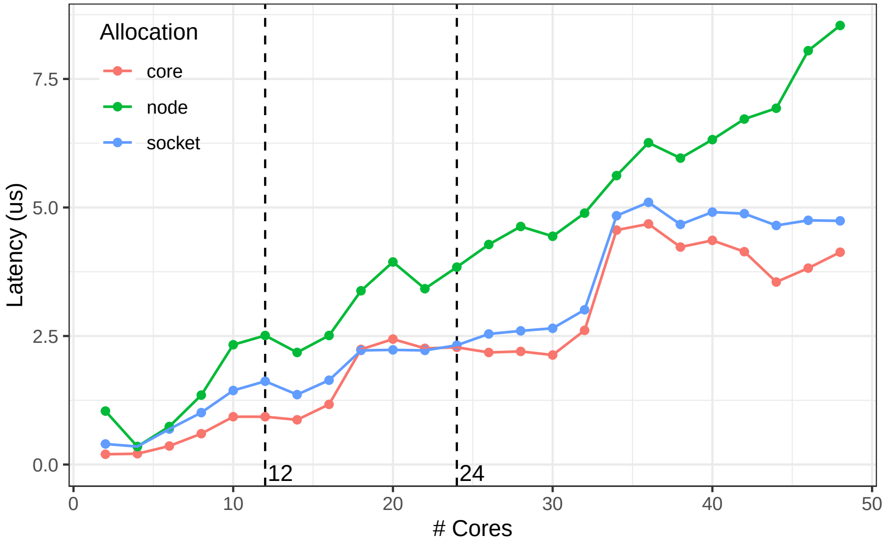


Allocation Type Comparison

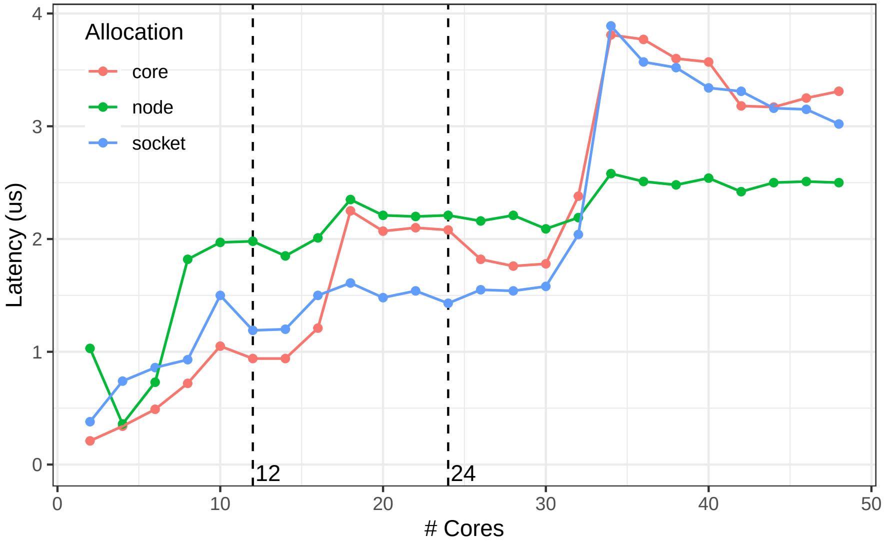
Linear Algorithm, Latency vs # Cores



Chain Tree Algorithm, Latency vs # Cores

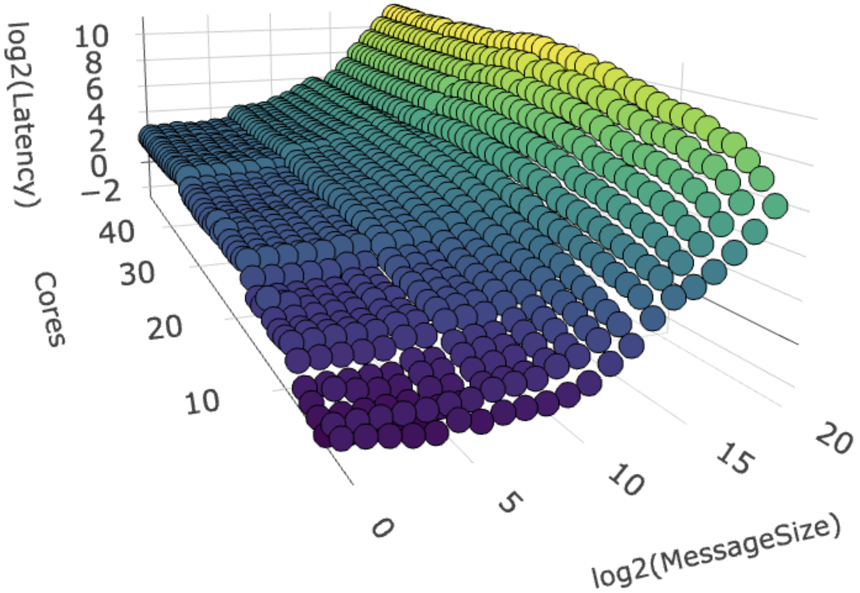


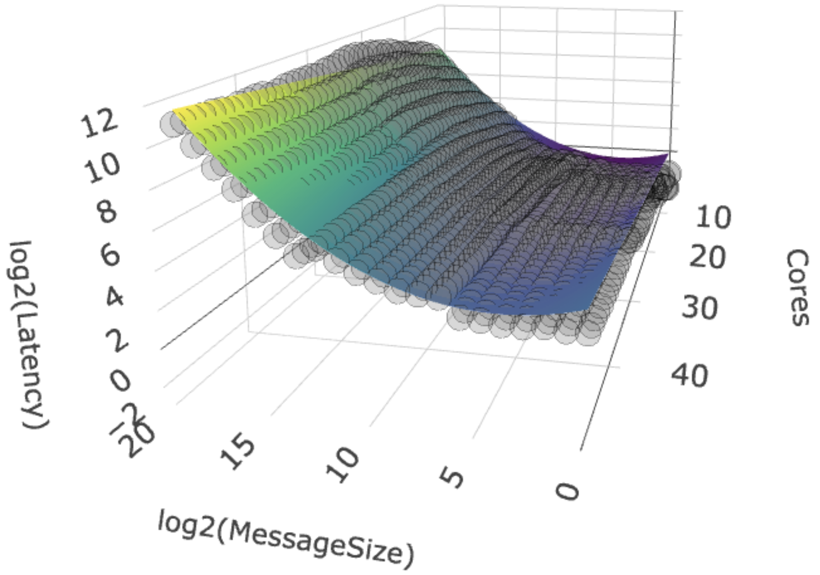
Binary Tree Algorithm, Latency vs # Cores



Broadcast Performance Model

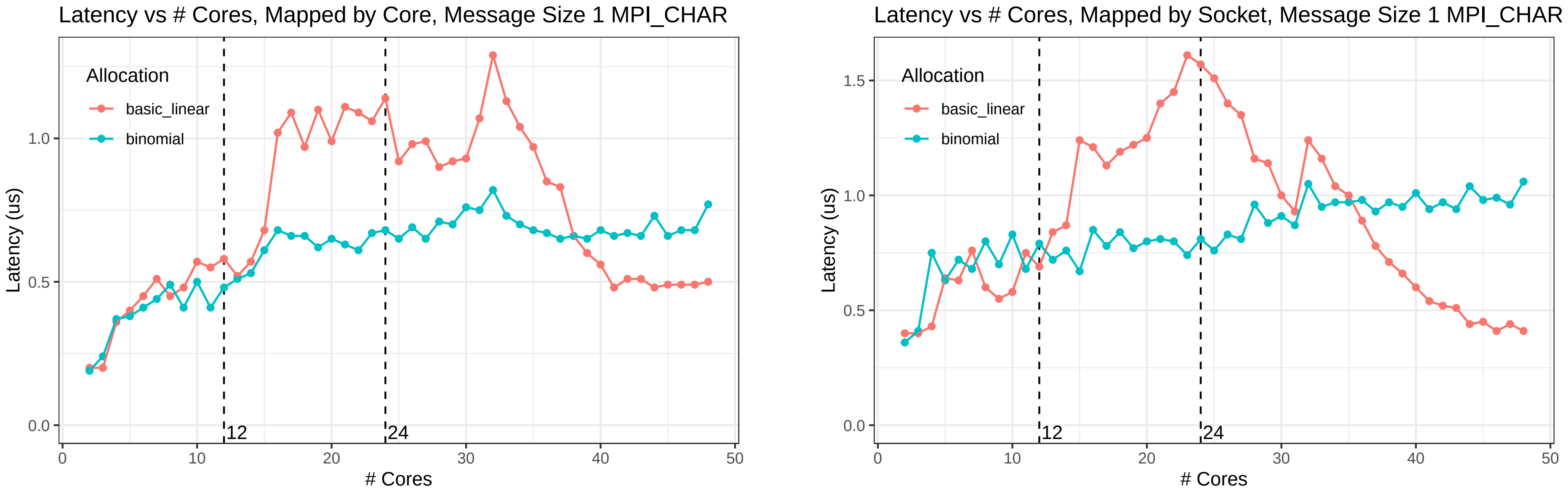
$$\log_2(\text{Latency}) = \beta_1 \cdot \text{Number of Processes} + \beta_2 \cdot \log_2(\text{Message Size}) + \beta_3 \cdot \log_2(\text{Message Size})^2$$

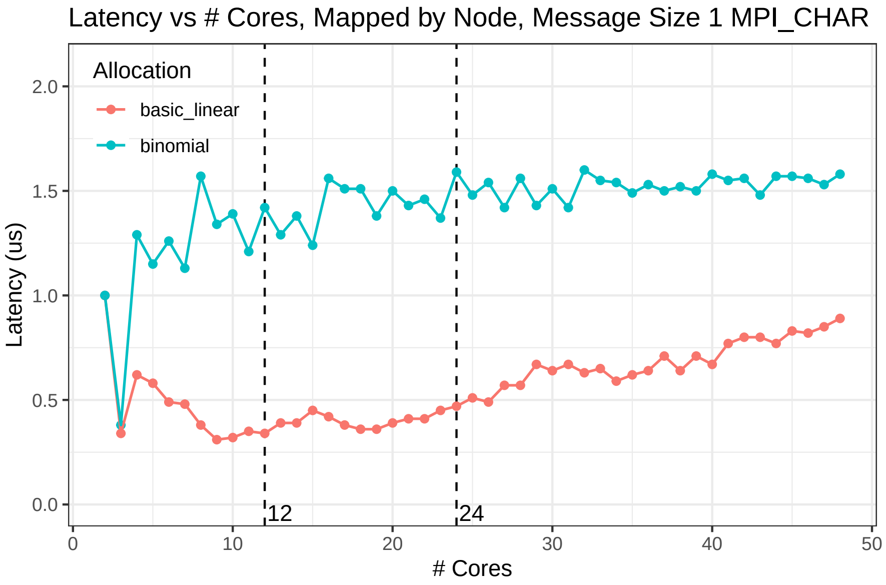




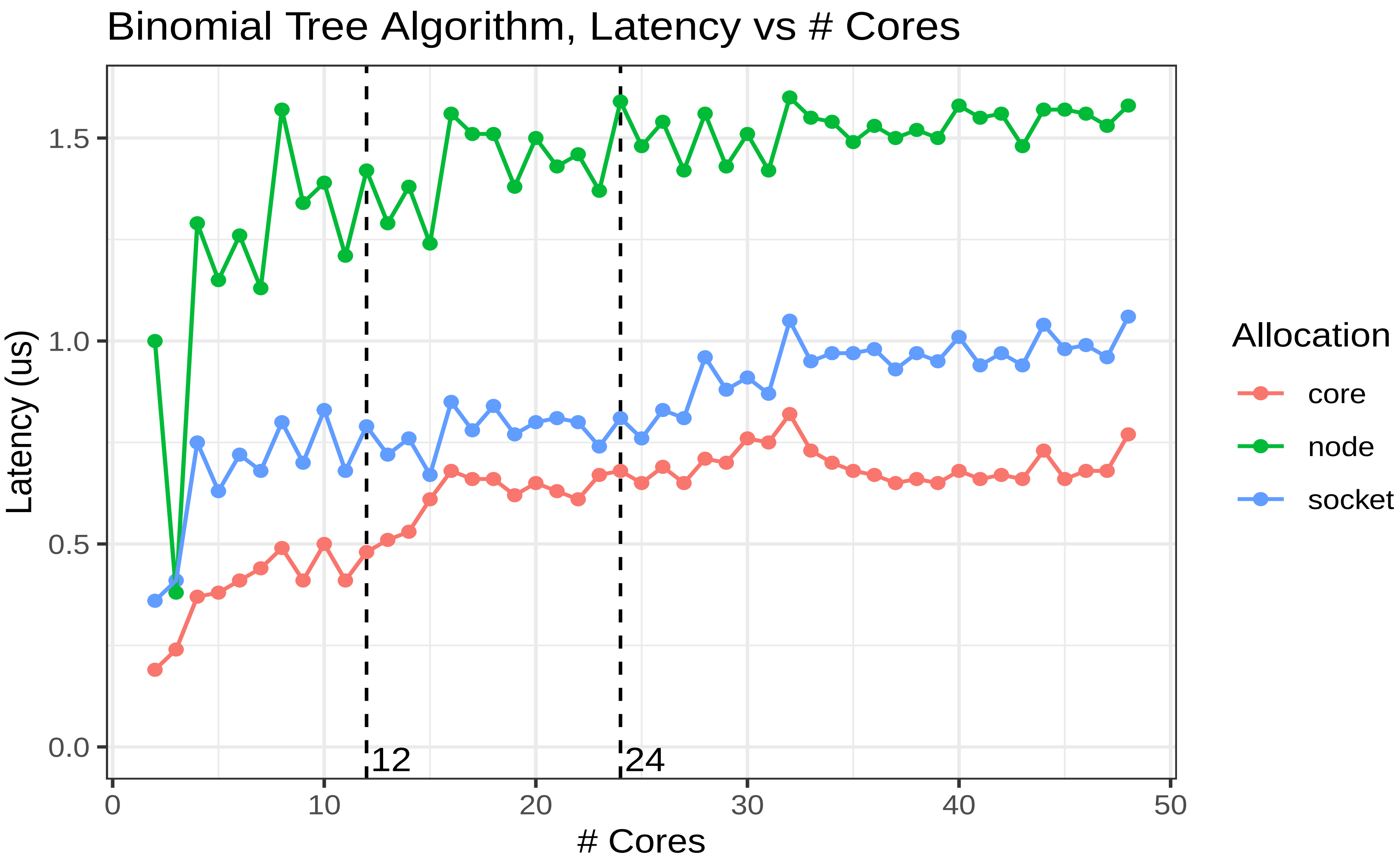
Algorithms	β_1	β_2	β_3	R^2_{adj}
Linear	0.07 ± 0.001	-0.289 ± 0.009	0.035 ± 0.001	97.94%
Chain	0.063 ± 0.001	-0.314 ± 0.008	0.035 ± 0.0004	98,32%
Binary Tree	0.059 ± 0.001	-0.322 ± 0.007	0.036 ± 0.0003	98,71%

Algorithms Comparison



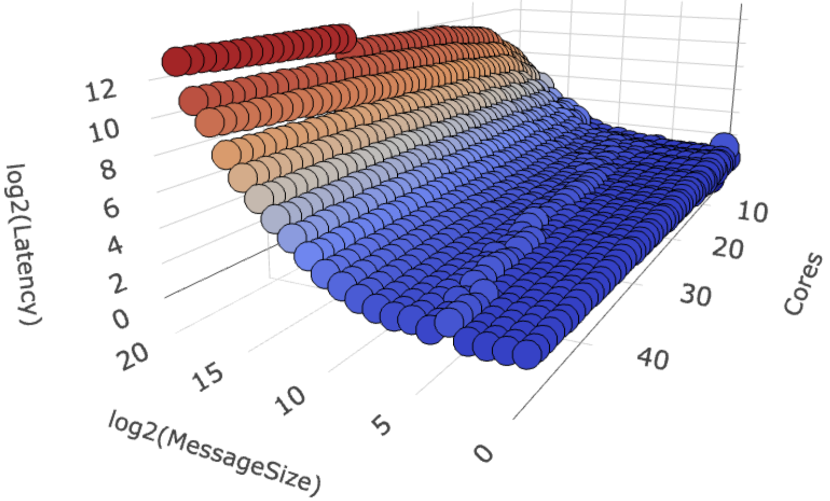


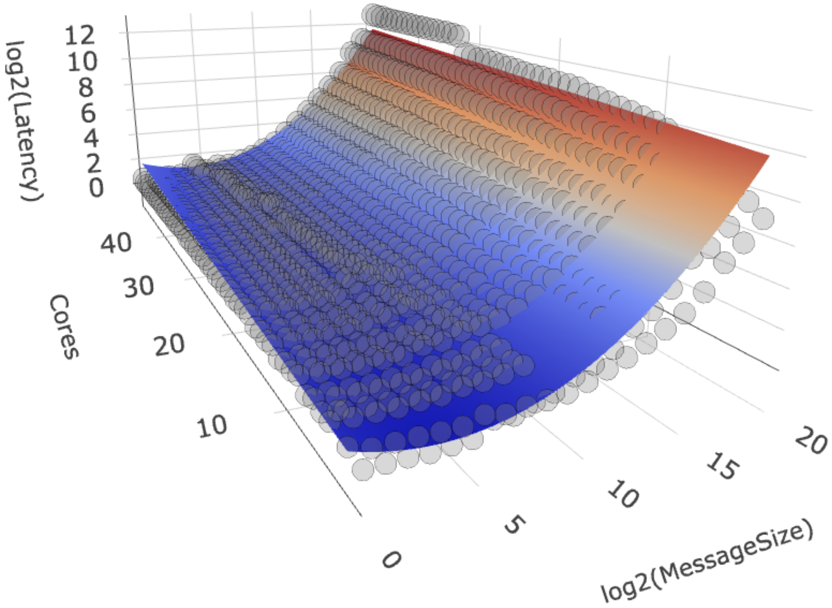
Allocation Type Comparison



Gather Performance Model

$$\log_2(\text{Latency}) = \beta_1 \cdot \text{Number of Processes} + \beta_2 \cdot \log_2(\text{Message Size}) + \beta_3 \cdot \log_2(\text{Message Size})^2$$





Binomial Tree	Estimate	Std. Error	P-value	R^2_{adj}
Processes	0.0394438	0.0011324	$< 2e-16$	
$\log_2$(Message Size)	-0.3175077	0.0084104	$< 2e-16$	
$\log_2$(Message Size)^2	0.0402835	0.0004586	$< 2e-16$	98,47%