Optimizing Message Aggregation for Parallel Simulation on High Performance Clusters

C.D. Pham

Université Claude Bernard Lyon 1, Bât. ISTIL
cpham@lhpca.univ-lyon1.fr

C. Albrecht
Institut für Technische Informatik
Medizinische Universität zu Lübeck
albrecht@iti.mu-luebeck.de

Some complex problems

- PARALLEL PROGRAMS, large and complex code
  - Performances, Correctness,
  - Verification, Scalability.
- MOBILE COMMUNICATION, thousands of mobiles
  - Handoff probability,
  - Frequency assignment.
- ATM NETWORKS, hundreds of nodes
  - Routing overhead,
  - End-to-End delays.

Performance Evaluation

How to handle these large systems?

- Analytical methods are too limited.
- Simulations are very time-consuming.

Reduce the simulation time

- Faster sequential computer,
- Parallel architectures.

Parallel and Distributed Simulation

- On parallel computers, networks of workstations, clusters of workstations,
- Focus mainly on synchronization problems.

Parallel simulation

Spatial partitioning and Logical Process (LP)

⬇

Distribution of event lists and virtual clocks.

⬇

Causality constraints and time errors.

⬇

A synchronization algorithm is needed to respect causality.

⬇

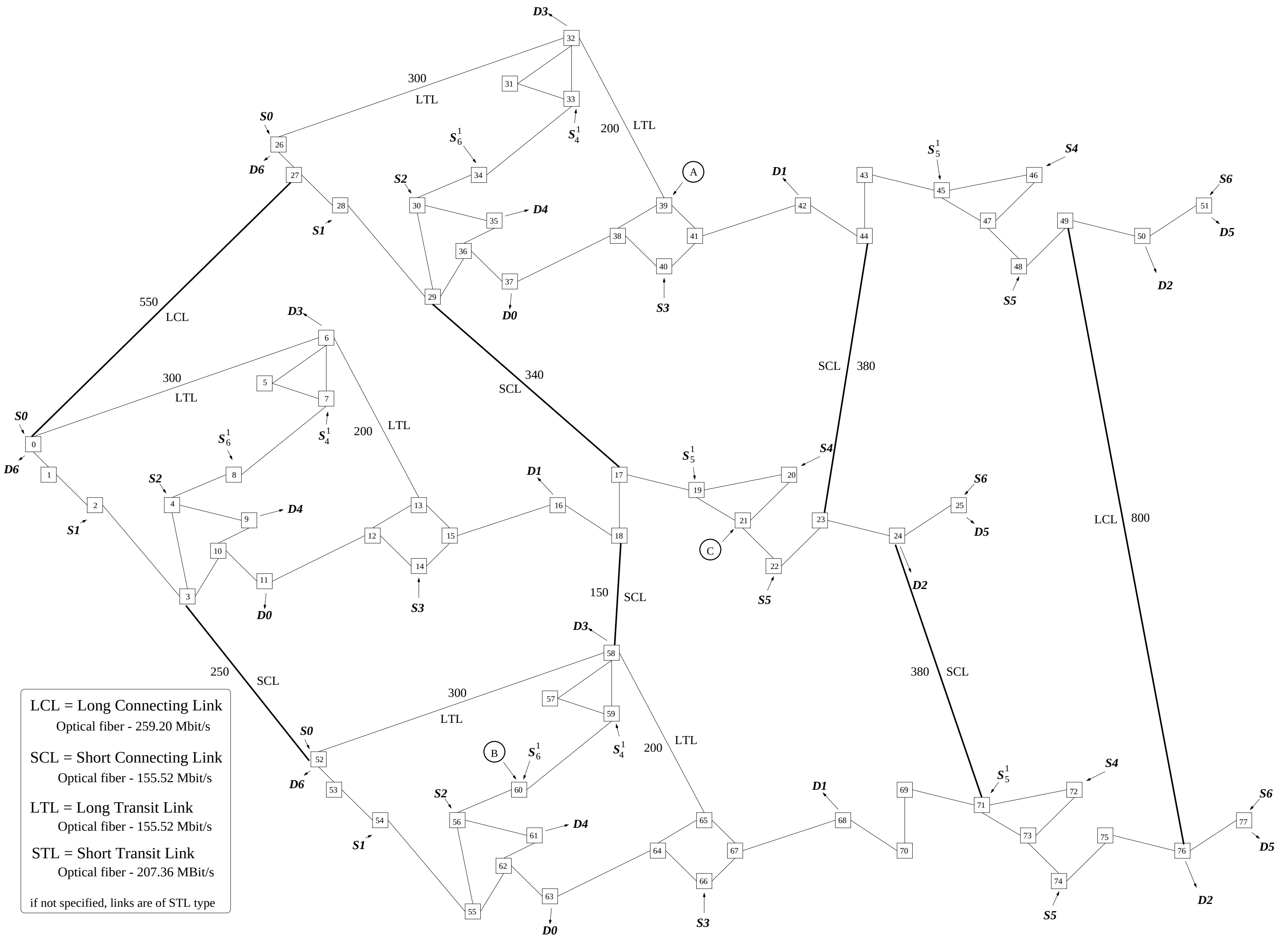
These synchronization mechanisms fall in two categories:

conservative and **optimistic**.

Parallel simulation of ATM networks

- Why is it difficult?
  - at the cell-level, the number of event is very large.
  - very small granularity, small computation/communication ratio.
- We are simulating:
  - a 78-switch network at the cell-level,
  - a routing algorithm with dynamic link costs,
  - connection management and admission control.
- We are using:
  - a conservative method (good lookahead of links),
  - message aggregation techniques (large number of small messages),

The ATM network model



The cluster environment

Parallel simulations have traditionally been performed on parallel computers to keep the communication overhead low

* * *

Network technology for clusters of workstations have dramatically improved making clusters very efficient with a high performance/cost ratio

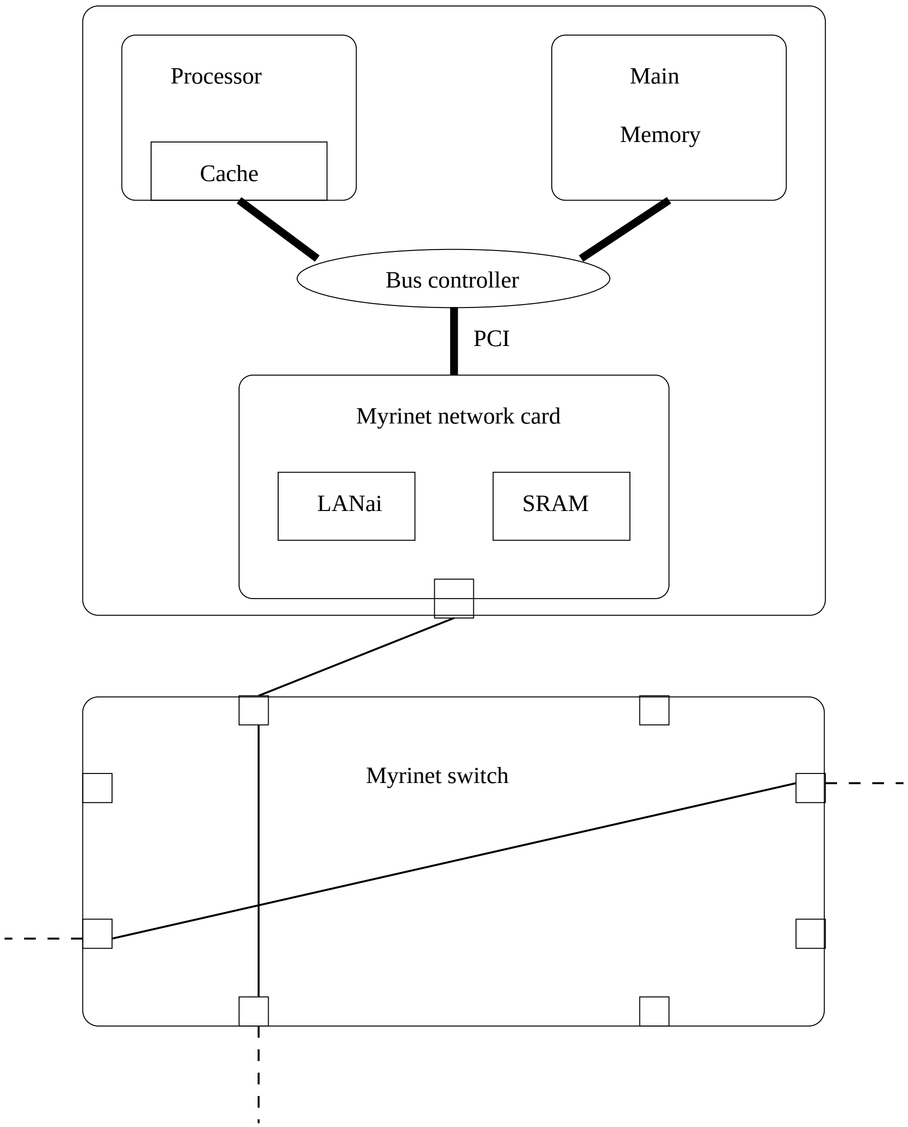
⬇

Myrinet-based cluster, 12 Pentium II 200MHz

BIP, $5\mu s$ of latency,

MPI-BIP communication library, adds about $4\mu s$ (CPU).

The myrinet technology



Message aggregation? For what?

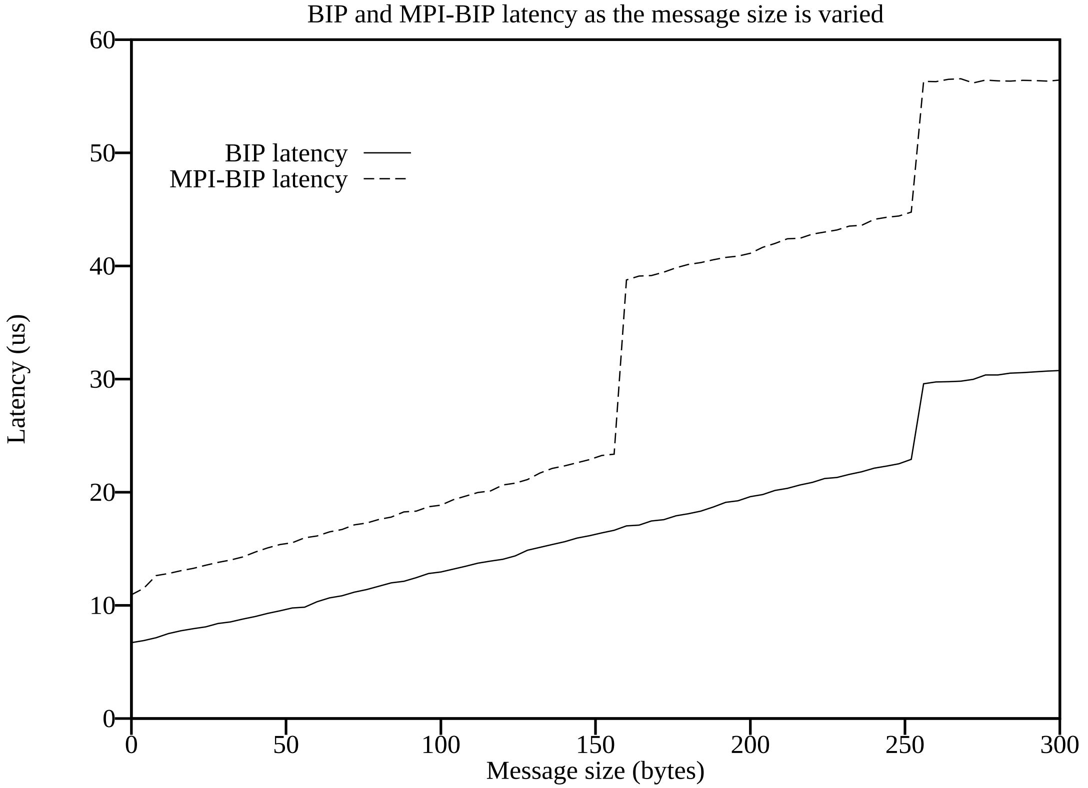
- Well-known technique to
  - increase the granularity of the application,
  - reduce the number of physical messages on the network,
  - enhance the computation/communication ratio.
- Parallel/Distributed applications have to worried about:
  - high communication latencies,
  - efficient overlapping of communication/computation.

Message aggregation: rational

- ONE LARGE MESSAGE VS SEVERAL SMALL MESSAGES
  - sending a message involves 2 kinds of overhead: a constant part and a variable part.
  - the constant part is generally high: it is more efficient to send an M bytes message than to send n times an m bytes message with $n.m = M$
- ONLY EFFICIENT IF...
  - aggregation does not introduce extra waiting time,
  - the communication layer does not introduce higher latencies.
- First, check the communication layer!

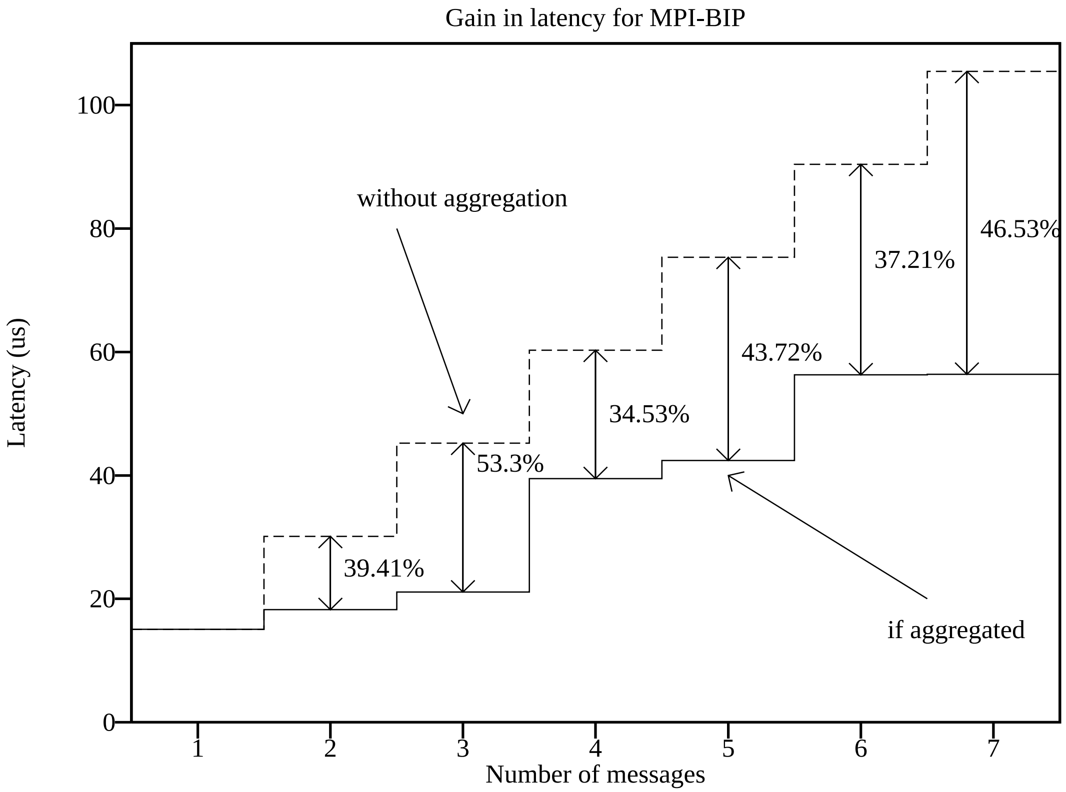
Checking the communication layer behavior

On a myrinet-based cluster, BIP and MPI-BIP are used for the communication layers



BIP and MPI-BIP latency as the message size is varied

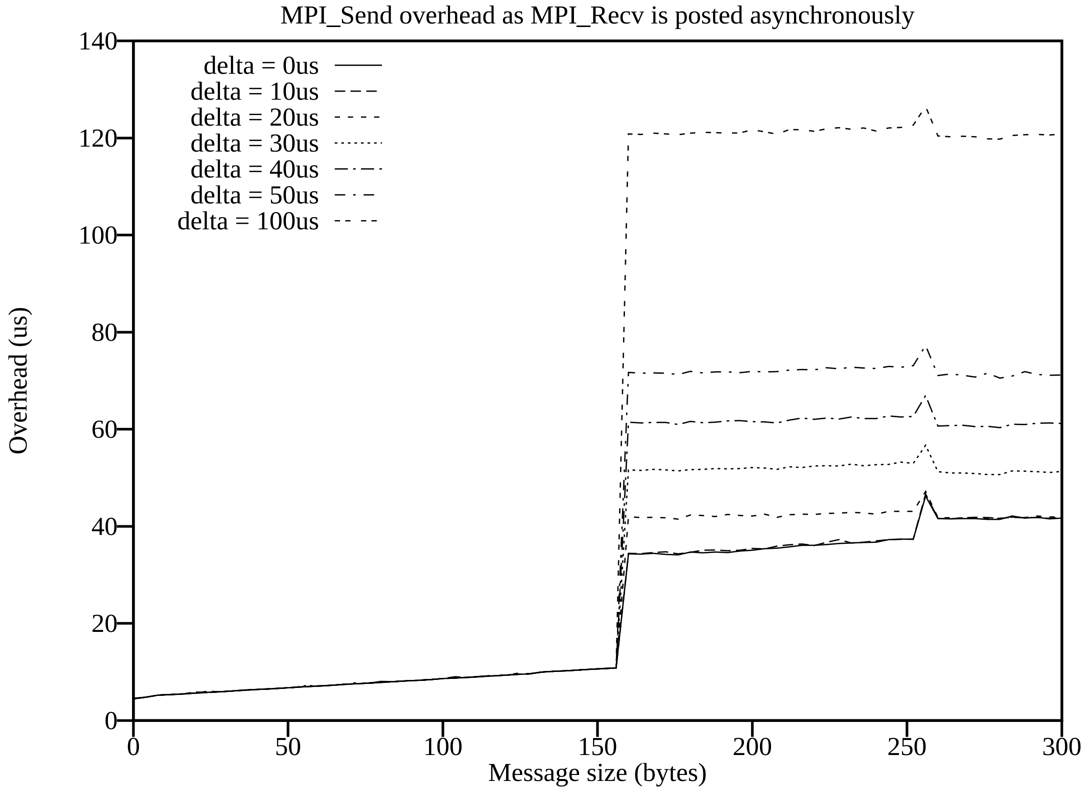
The benefit of message aggregation



Gain in latency for MPI-BIP

- aggregation can decrease the communication overhead,
- aggregation can provide additional buffering.

The impact of asynchronous send and receive



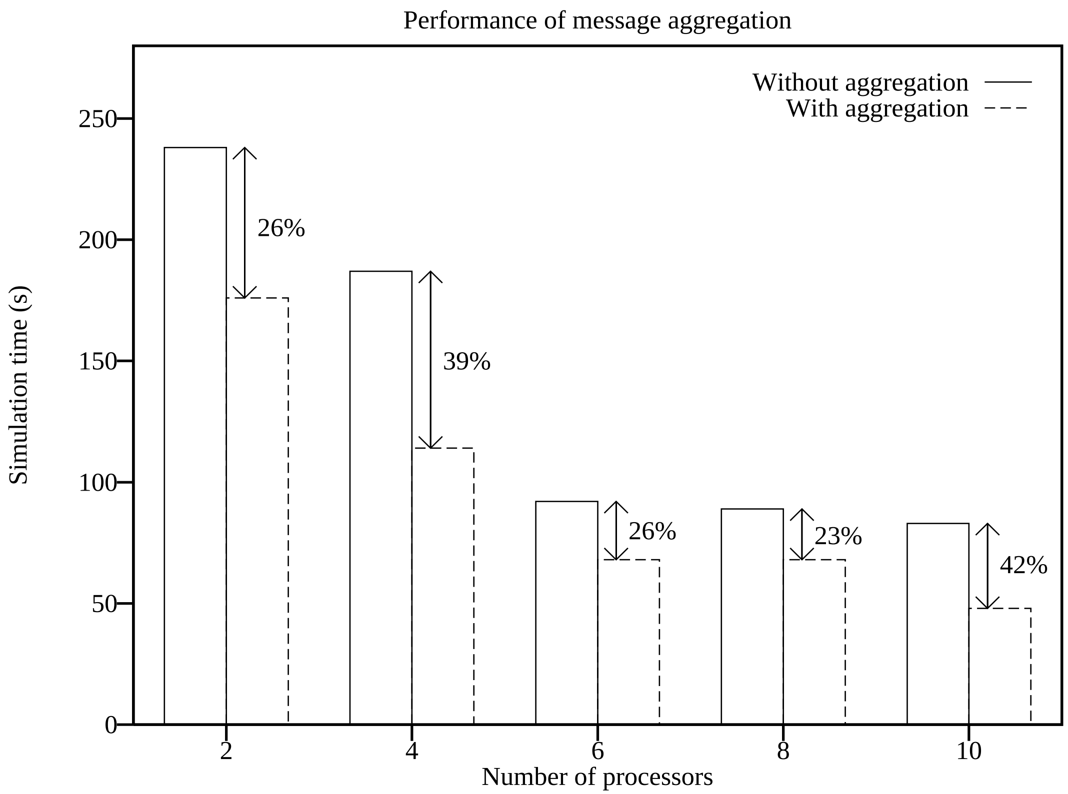
MPI_Send overhead as δ is varied

- it is essential to remain below the critical message size,
- the more asynchronism is added, the less computation and communication are overlapped.

The fixed-size aggregation strategy

- Large aggregates are costly
  - use only aggregate size smaller than the critical message size S_{max},
  - no need of dymanic scheme.
- An aggregate is sent if:
  - aggregating the new message will produce an aggregate of size $S > S_{max}$,
  - after aggregating the new message, the space left is less than the minimum event size,
  - the new aggregated message is a null-message.

Impact on the simulation time



Performance of message aggregation as the number of processors is varied.

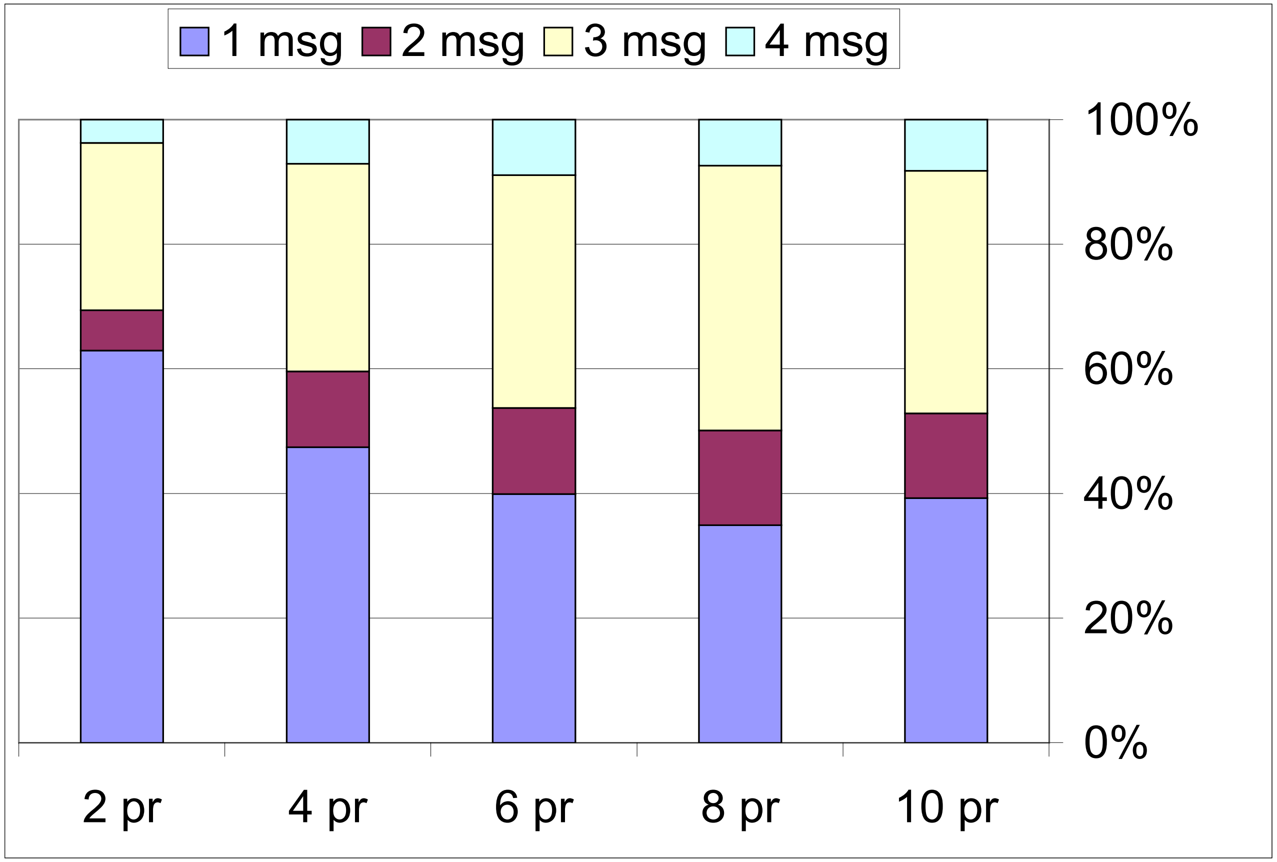
➡ message aggregation can save up to 42% of the simulation time.

Message statistics

	2 pr.	4 pr.	6 pr.	8 pr.	10 pr.
total sent un.	1300337	3643078	3420847	5101153	5378893
total sent w/A	758595	1820562	1588672	2293524	2488476
efficiency	41.6%	50%	54%	55%	54%
null-msg w/A	568620	1344526	1153182	1520551	1751671
null w/0	83.95%	64.18%	54.97%	52.63%	56.90%
null w/1	5.55%	14.86%	16.80%	20.08%	17.18%
null w/2	5.50%	11.35%	15.91%	16.10%	13.99%
null w/3	4.98%	9.59%	12.29%	11.16%	11.92%

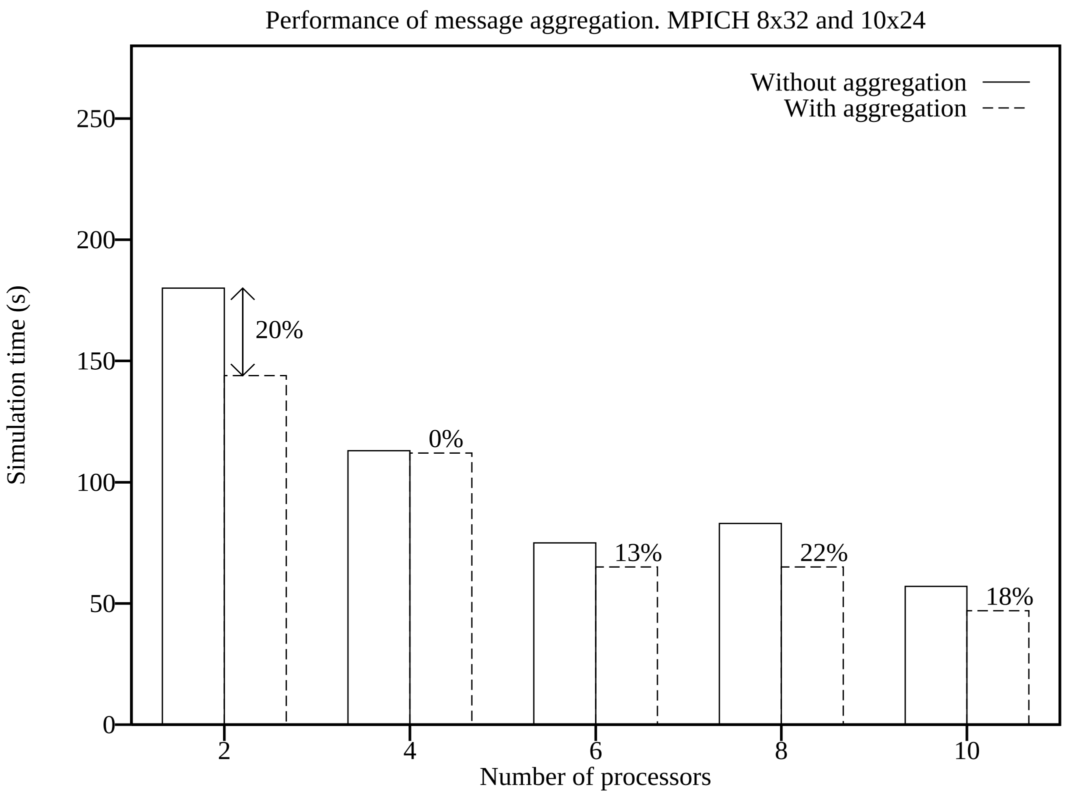
- message aggregation can save up to 54% of the number of sent messages,
- on 2 processors, the number of null-messages sent alone is high because there are less events per cycle.

Message size repartition



Repartition of aggregate size in number of messages

Increasing the number of reception buffers



Performance of message aggregation - MPICH 8x32 and 10x24

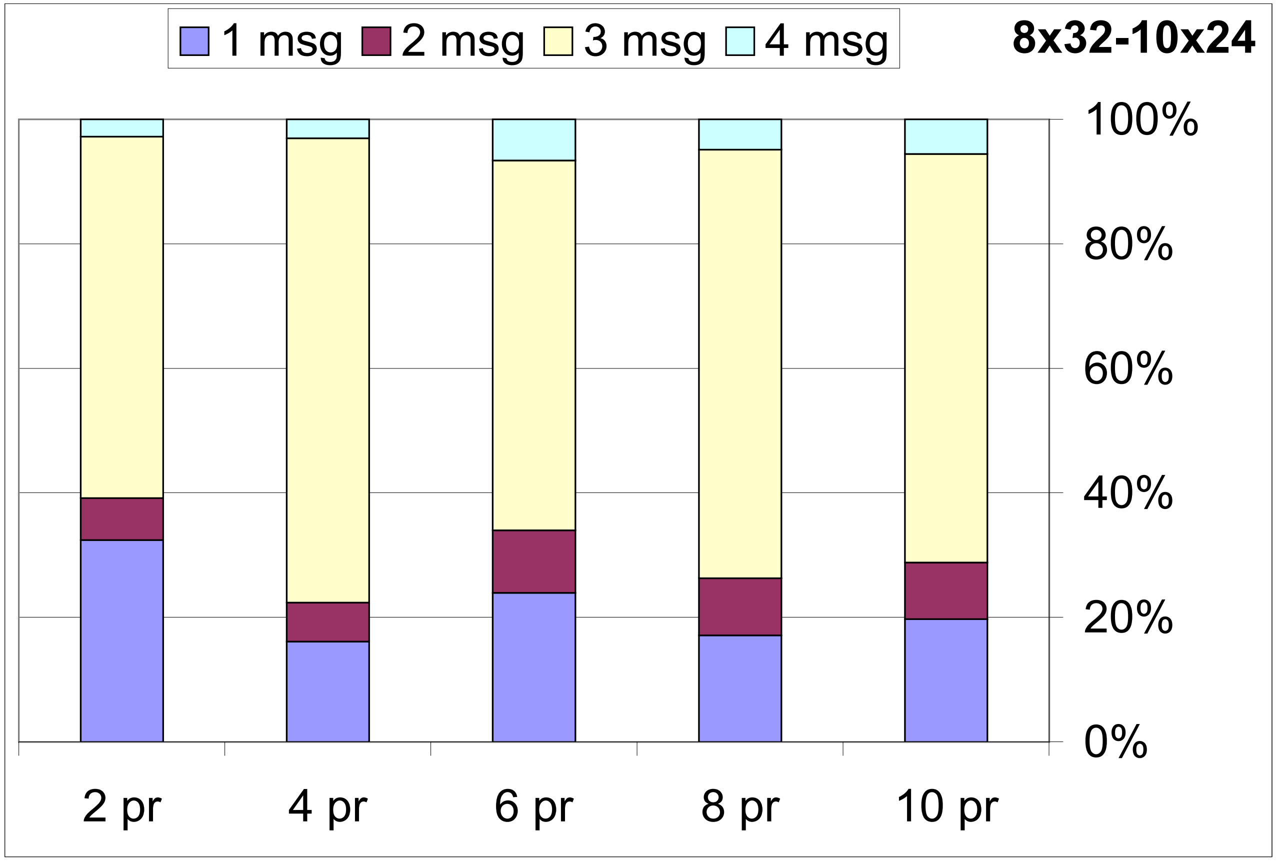
- increasing the number of reception buffers improves the performance of the unaggregated version, but it is less scalable,
- aggregation is less sensitive to the number of available buffer.

Increasing the number of reception buffers - con't

	2 pr.	4 pr.	6 pr.	8 pr.	10 pr.
total sent un.	852890	2549018	2761616	3940195	4257933
total sent w/A	368800	963202	1110277	1506972	1656210
efficiency	57%	62%	(59%)	(62%)	(61%)
null-msg w/A	142980	248727	493663	516347	619830
null w/0	83.57%	62.25%	53.83%	49.94%	52.66%
null w/1	5.02%	13.34%	17.04%	17.99%	16.79%
null w/2	4.16%	12.52%	14.23%	17.90%	15.64%
null w/3	7.23%	11.87%	14.88%	14.15%	14.89%

- efficiency is higher,
- the number of null-messages sent alone is a bit lower because there are more events per cycle, except on 2 processors.

Increasing the number of reception buffers - con't



Repartition of aggregate size in number of messages

- less than 10% of 3-message aggregates contain a null-message.

Conclusions

- Message aggregation is efficient on high performance cluster but the behavior of the communication layers must be taken into account.
- On Myrinet-based clusters with MPI-BIP, optimizing the message aggregation strategy can reduce the simulation time and increase the scalability.
- The preliminary process of checking the communication layer can be reused for other architectures in order to automatically tune the message aggregation strategy.
- To know more on cluster-based computing, BIP, MPI-BIP: `http://lhpca.univ-lyon1.fr`