Comparison of Message Aggregation Strategies for Parallel Simulations on a High Performance Cluster

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

MASCOT'2000, San Francisco, Wenesday 30th, August 2000

Simulation of large-scale communication networks

- Why is it difficult?
  - at the packet level, the number of events is very large.
  - very small granularity, small computation/communication ratio.
- We simulate
  - hundreds of switches at the packet-level,
  - routing algorithms with dynamic link costs, multicast protocols,
  - connection management and admission control mechanisms.
- Problems
  - large-scale simulations are very time-consuming!
  - large-scale simulations require lots of memory capacity!

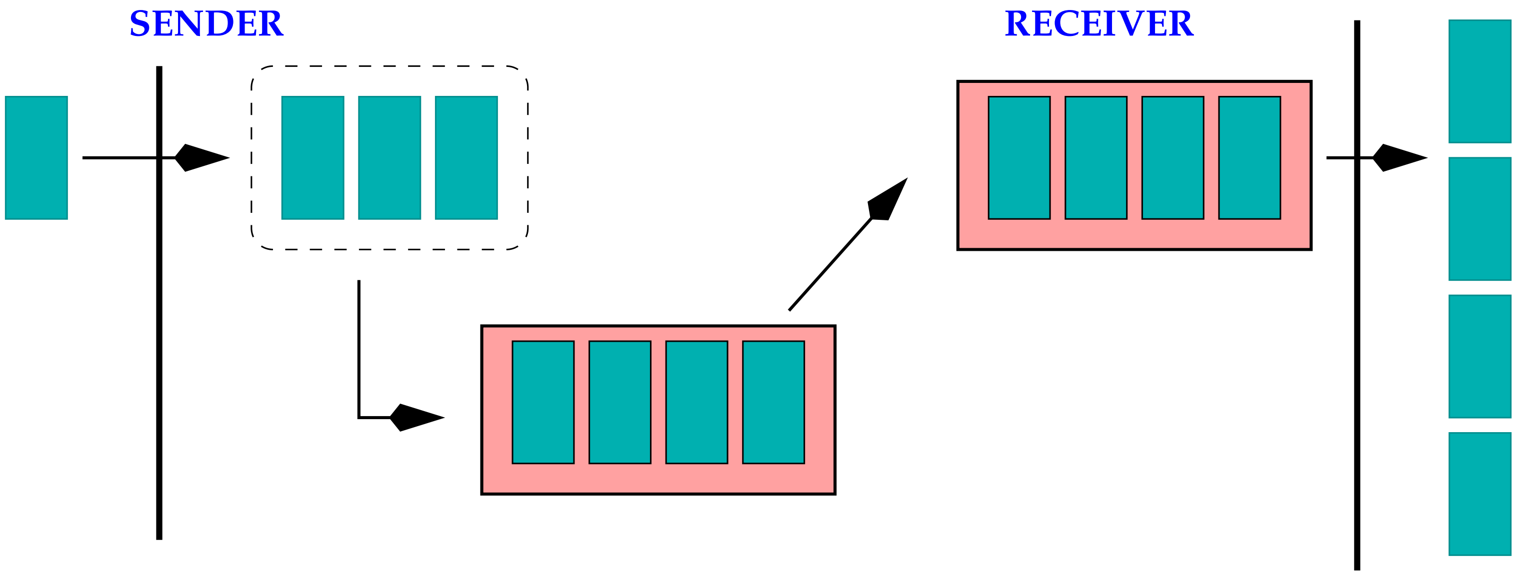
CSAM: a parallel simulator for communication network models

- Reduce the simulation time, increase the system size
  - parallel computers (CM-5, Cray T3E, Origin 2000),
  - high-performance clusters (PC-Myrinet).
- CSAM uses
  - handles causality consistency with a conservative method (good lookahead of links),
  - a window-based event processing cycle,
  - an optimized input channel management,
  - efficient message aggregation techniques (large number of small messages).

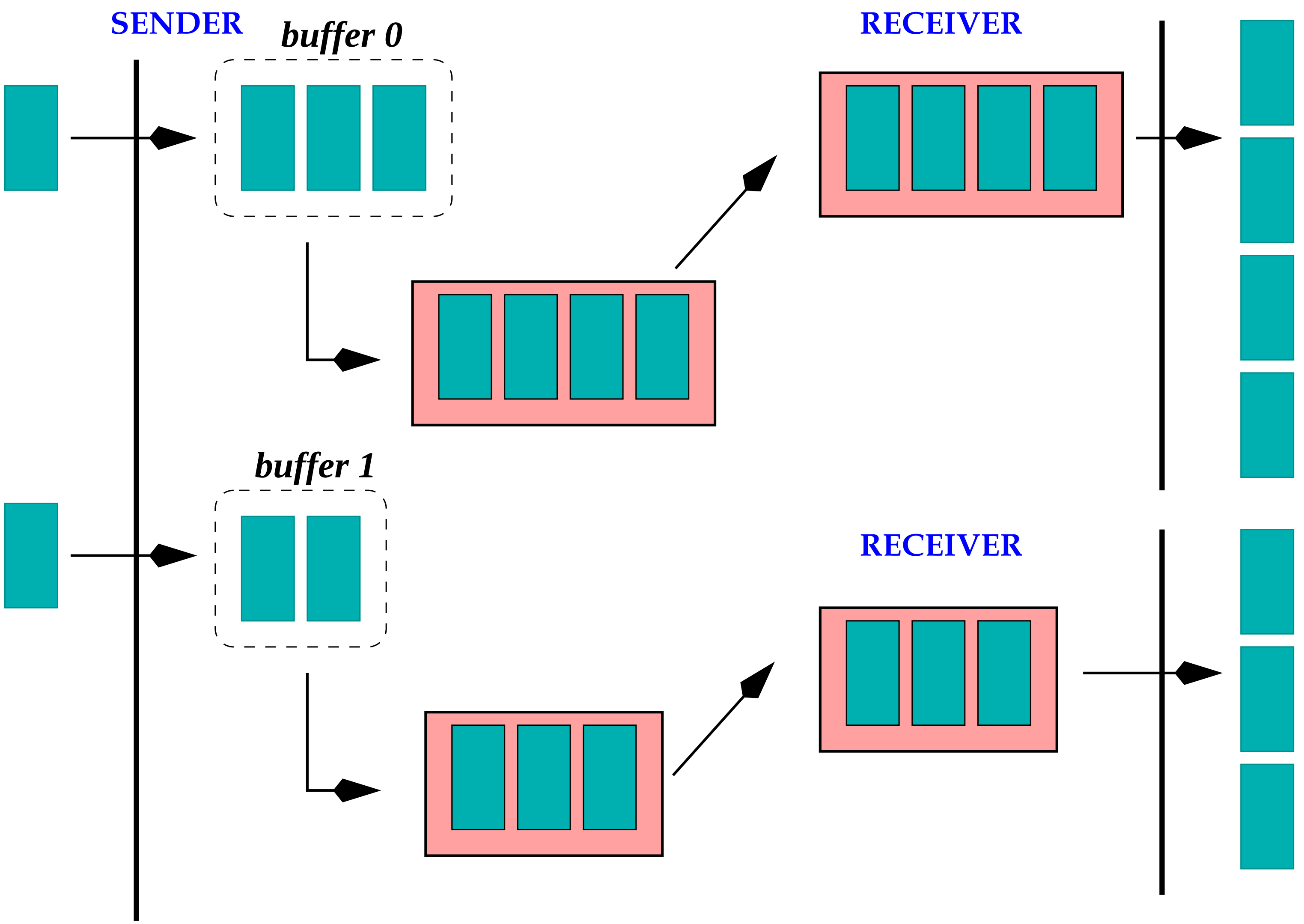
Message aggregation? For what?

- Parallel/Distributed applications are concerned about:
  - low communication latencies,
  - efficient overlapping of computation/communication.
- Well-known technique to
  - reduce the number of physical messages on the network,
  - enhance the computation/communication ratio.

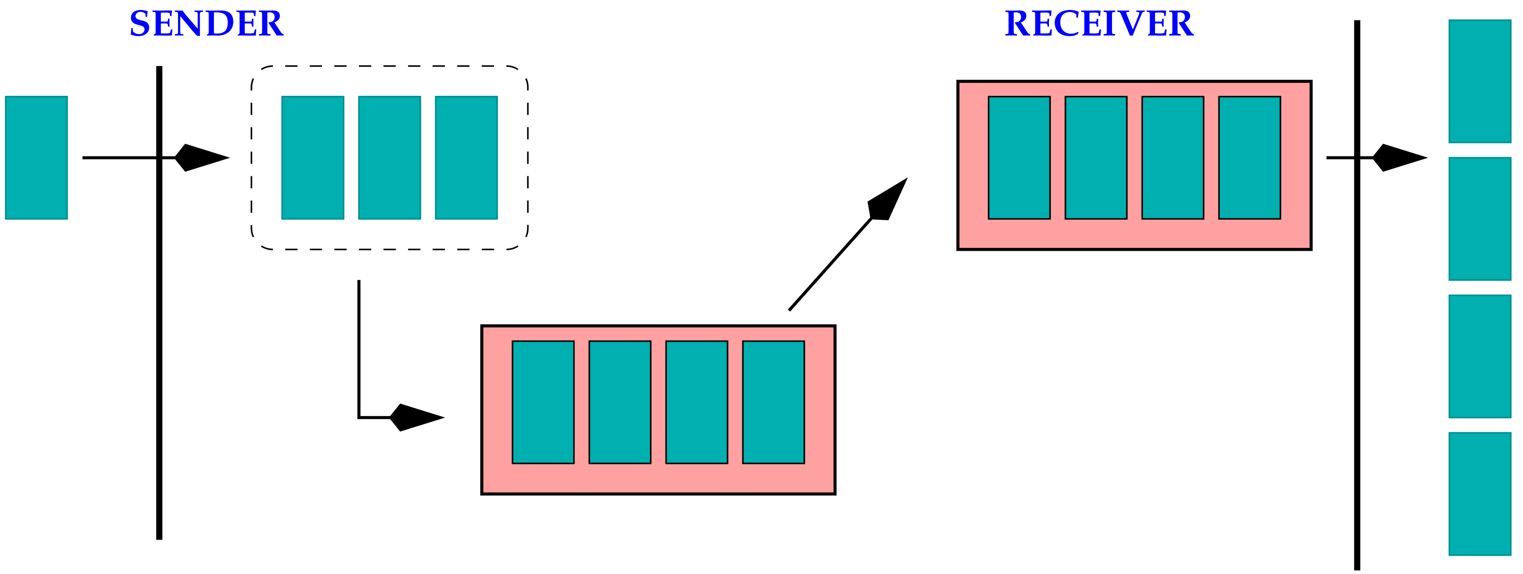
How does it work?



How does it work? (multiple buffers)



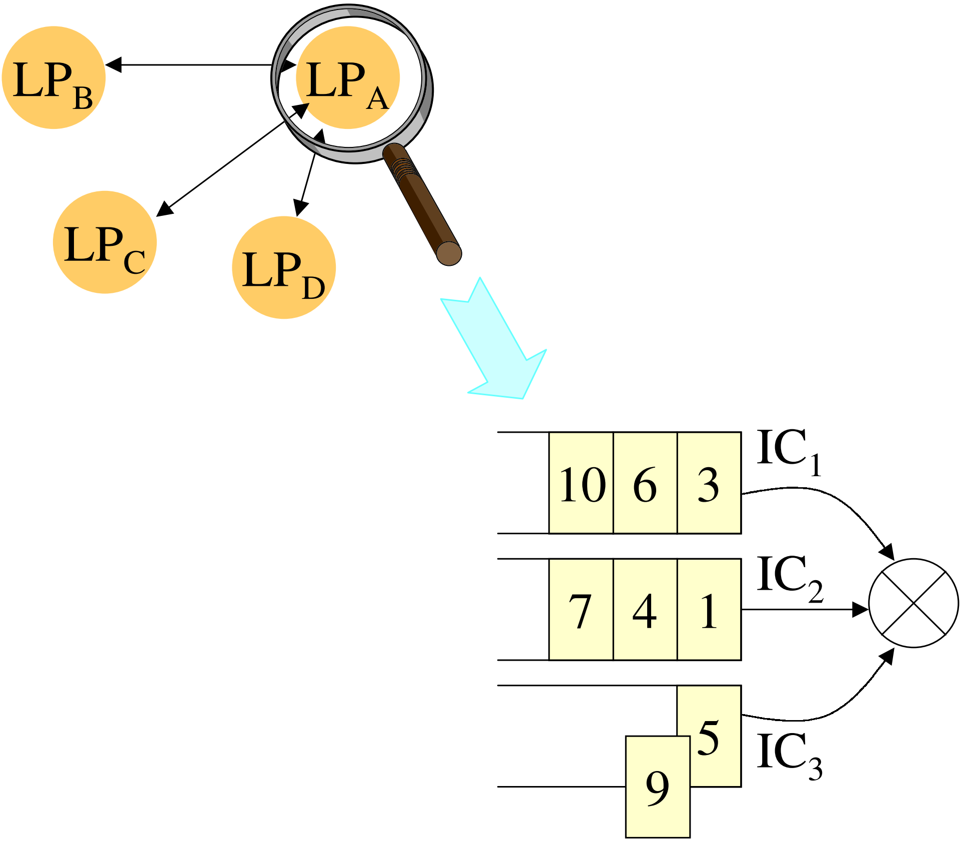
How does it works?



Very efficient if...

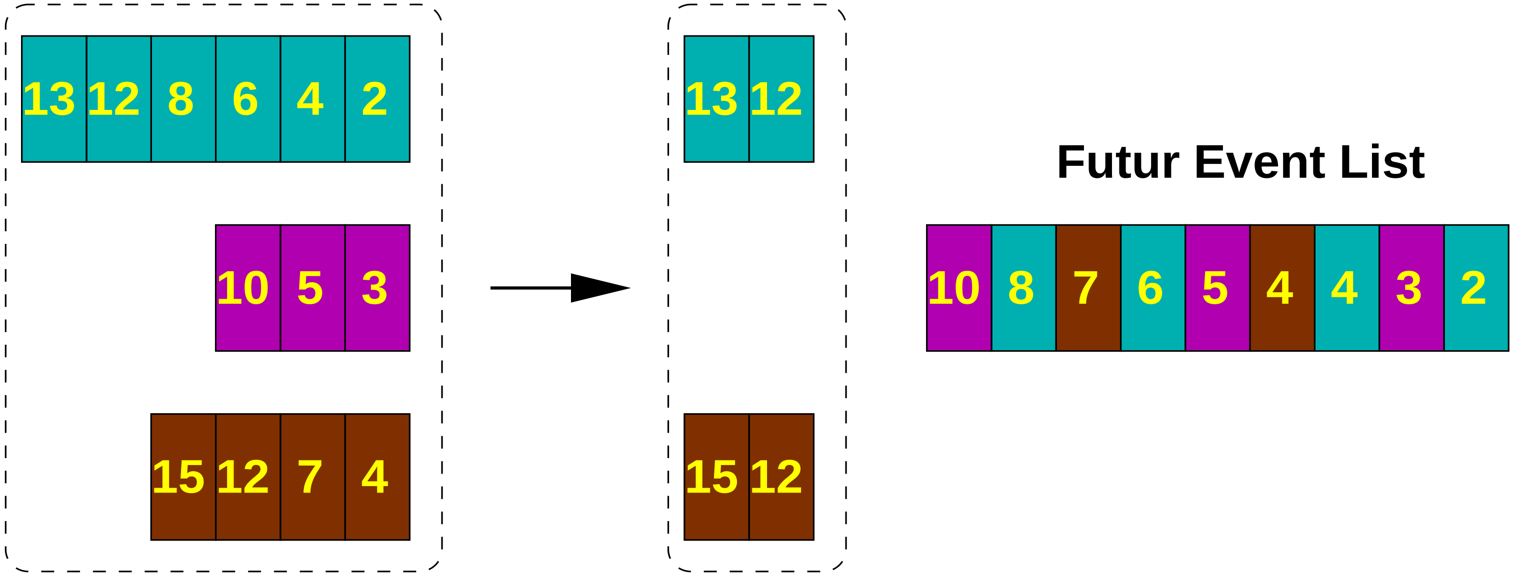
- The communication layer does not introduce higher latencies: **sending 1 large message is less costly than sending several small messages.**
- Aggregation does not introduce overwhelming additional waiting time: **the receiver is not constanly waiting (desperately) for new messages.**

The conservative approach: “safe is better”



min IC	event
2	1
1	3
2	4
3	5
3	BLOCK
1	6
2	7

The window-based processing cycle



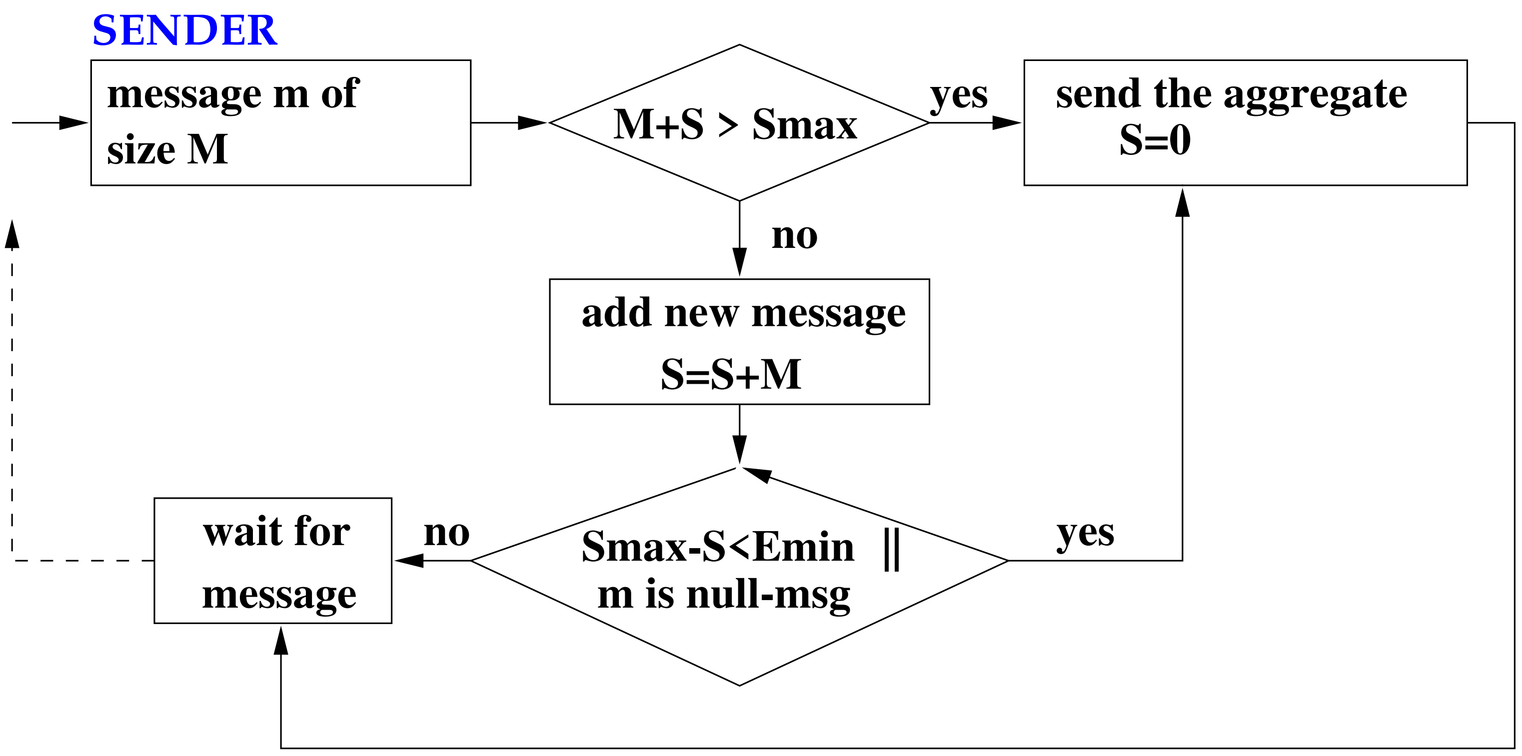
Algorithm

- get all messages from input channels (ICs) until one IC becomes empty,
- set the *safe time* to the timestamp of the last message,
- the upper bound of the time window is the *safe time*.

So...

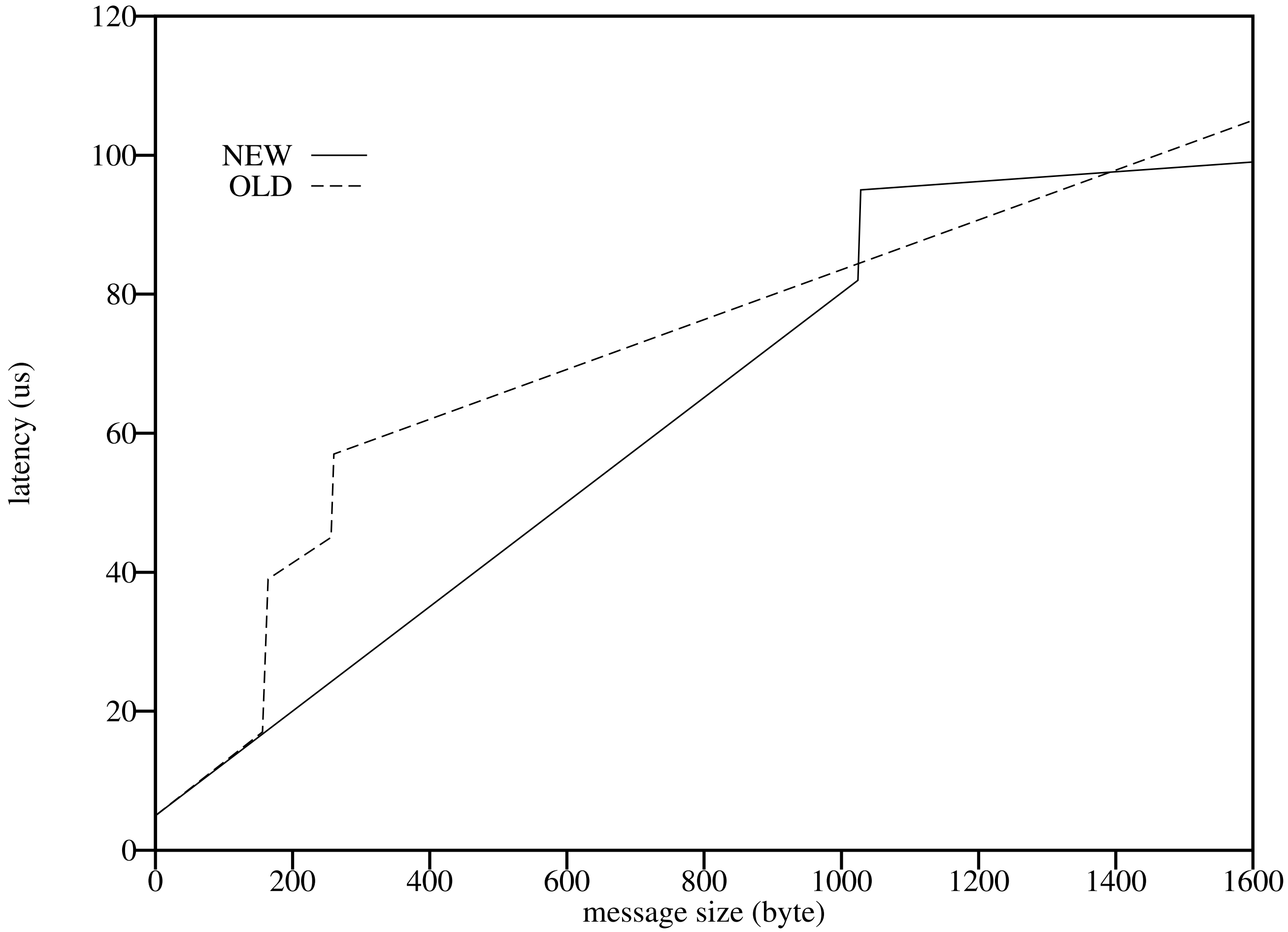
- **While in a processing cycle,**
  - no need to receive new messages too early: processed safe events first, then...
  - ... get new messages.
- **No need to send messages too early!**
  - aggregate messages at the sender side, then...
  - ... send them in one piece.

Sender-initiated aggregation strategy (AGGRxxx)



- the maximal aggregate size (or the maximal holding time) has to be defined.
- for large values of S_{max}, may not be optimal.

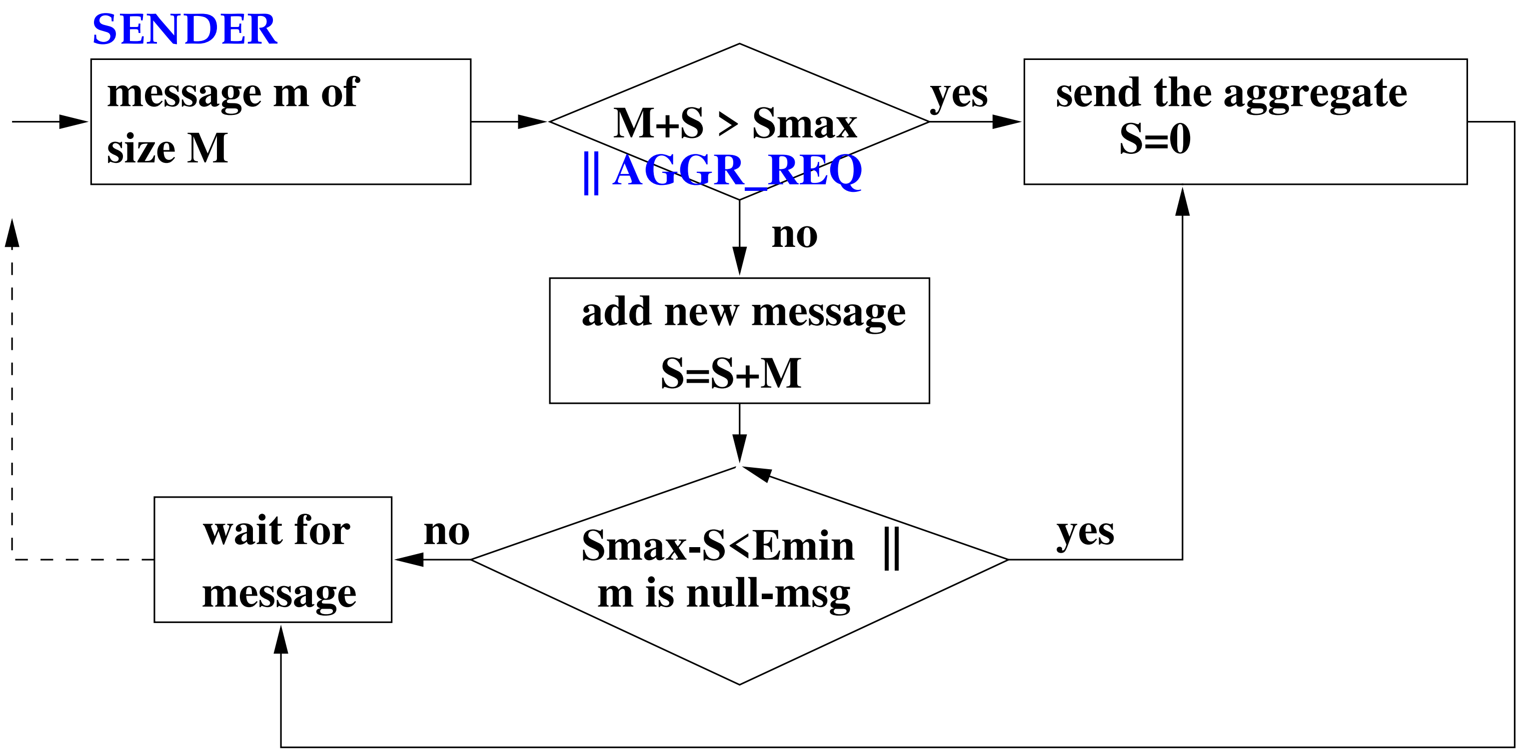
Why are large aggregates desirable?



message latency as the message size is varied

➥ why don't let the receiver determine when it needs the aggregates?

Receiver-initiated aggregation strategy



- The receiver sends an AGGR_REQ when
  - simple scheme (R_AGGRxxx): no more safe events are available.
  - advance request scheme (RA_AGGRxxx): only B safe events remain.
  - Pbs: additional overheads that have to be kept low.

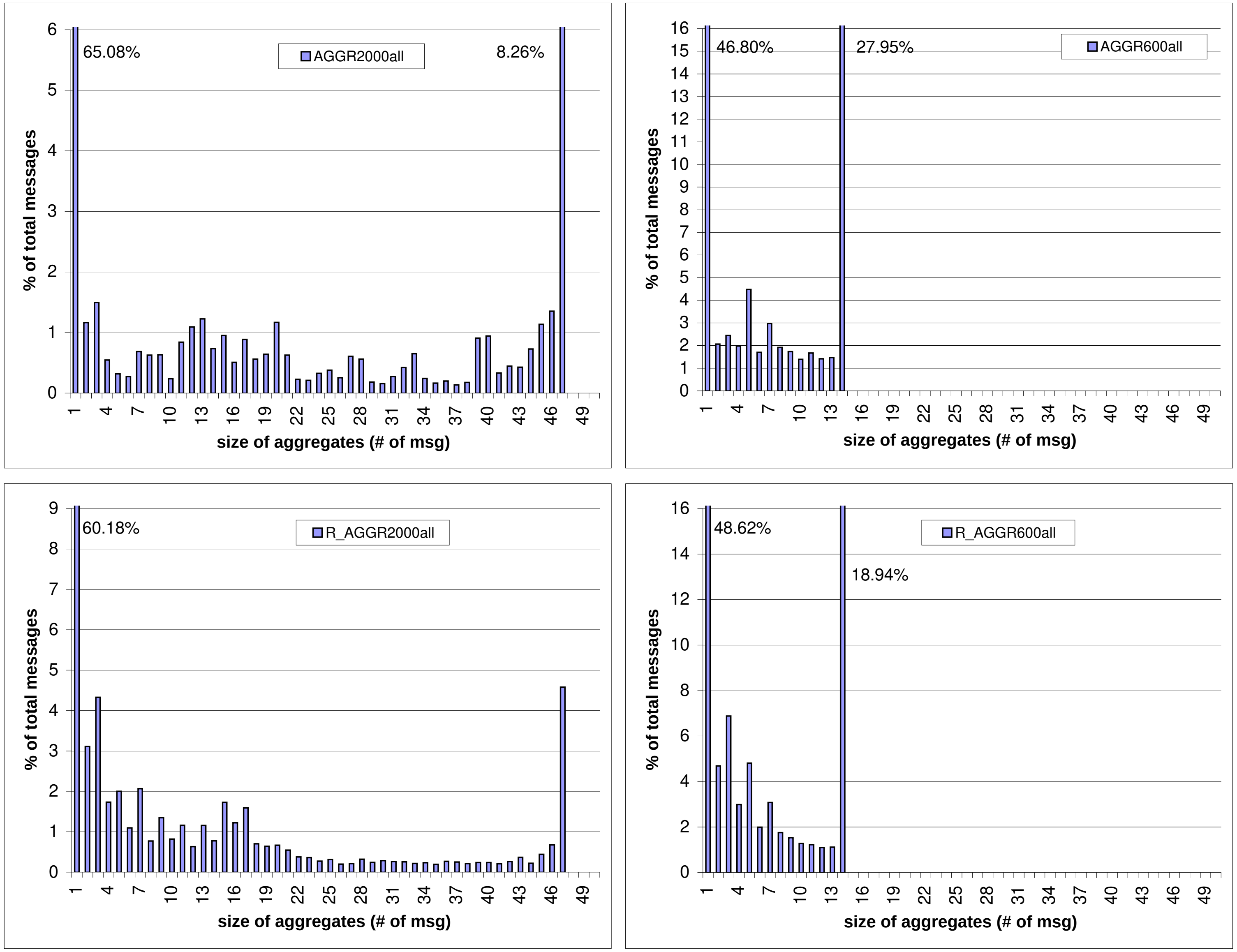
The cluster environment

Myrinet-based cluster, 4 Pentium II 200MHz

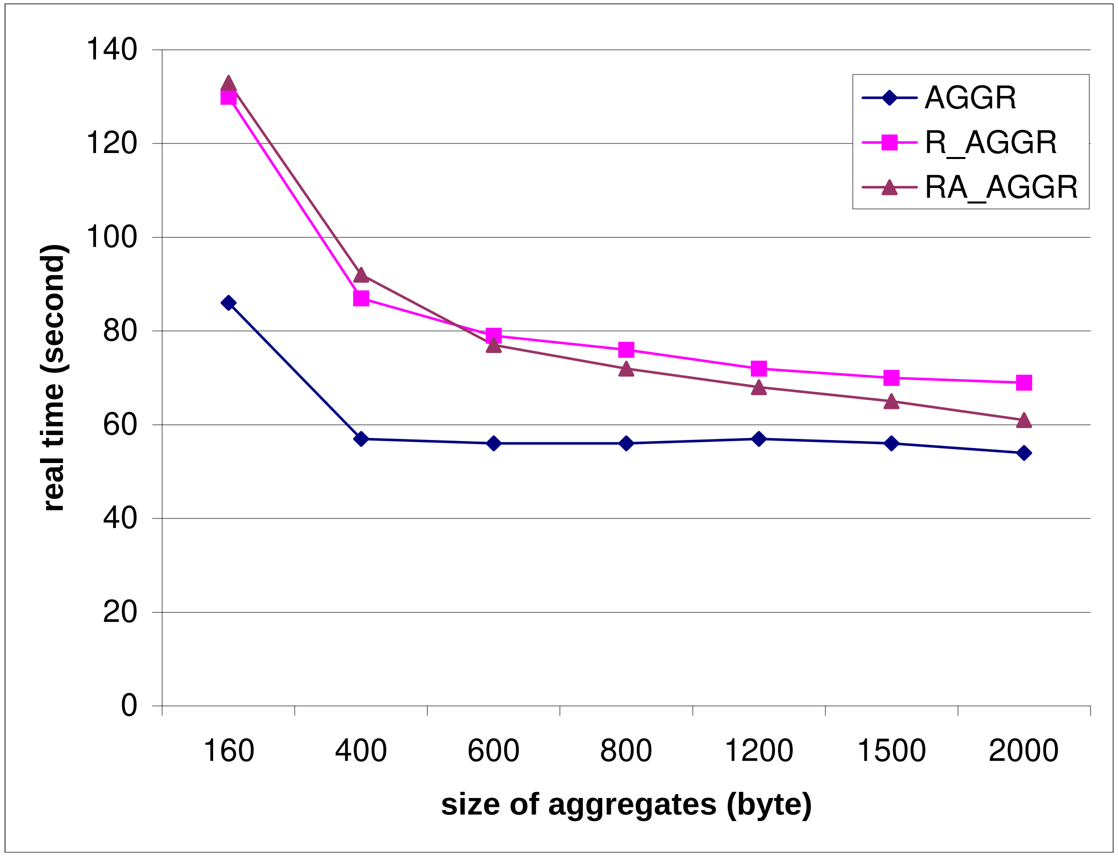
BIP, $5\mu s$ of latency,

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

Aggregate size distribution: AGGRxxx and R_AGGRxxx

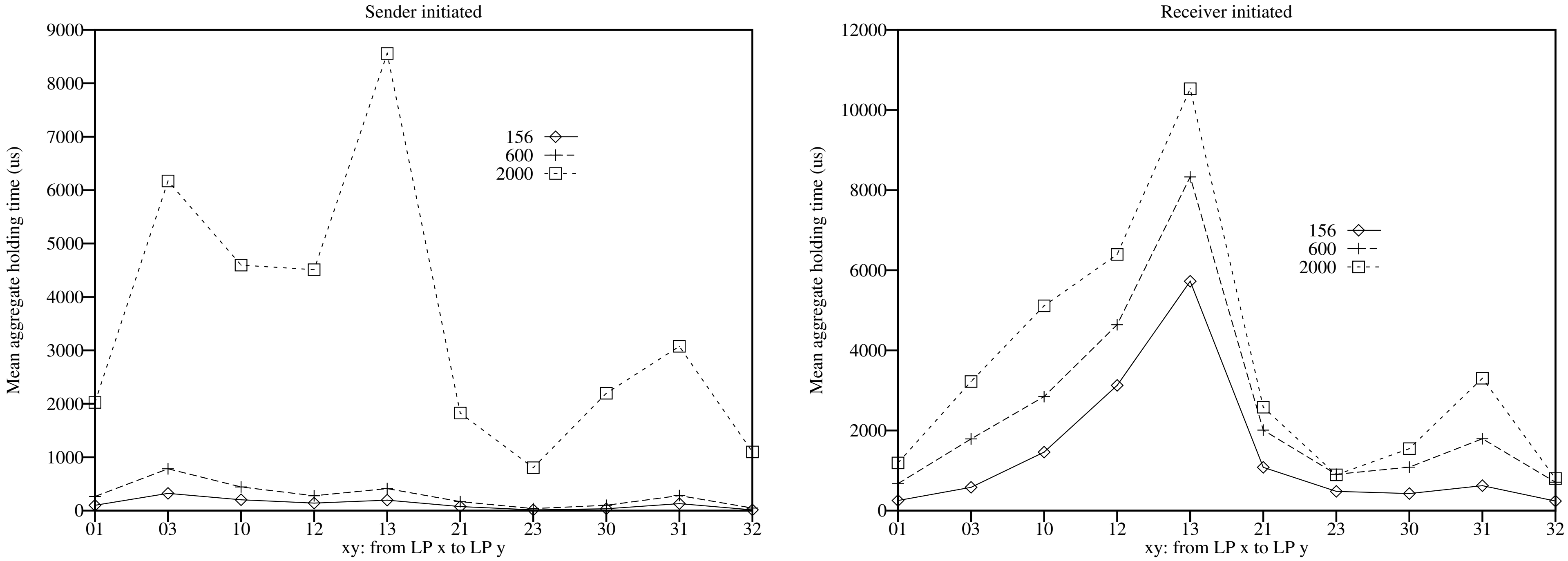


Impact on the simulation time

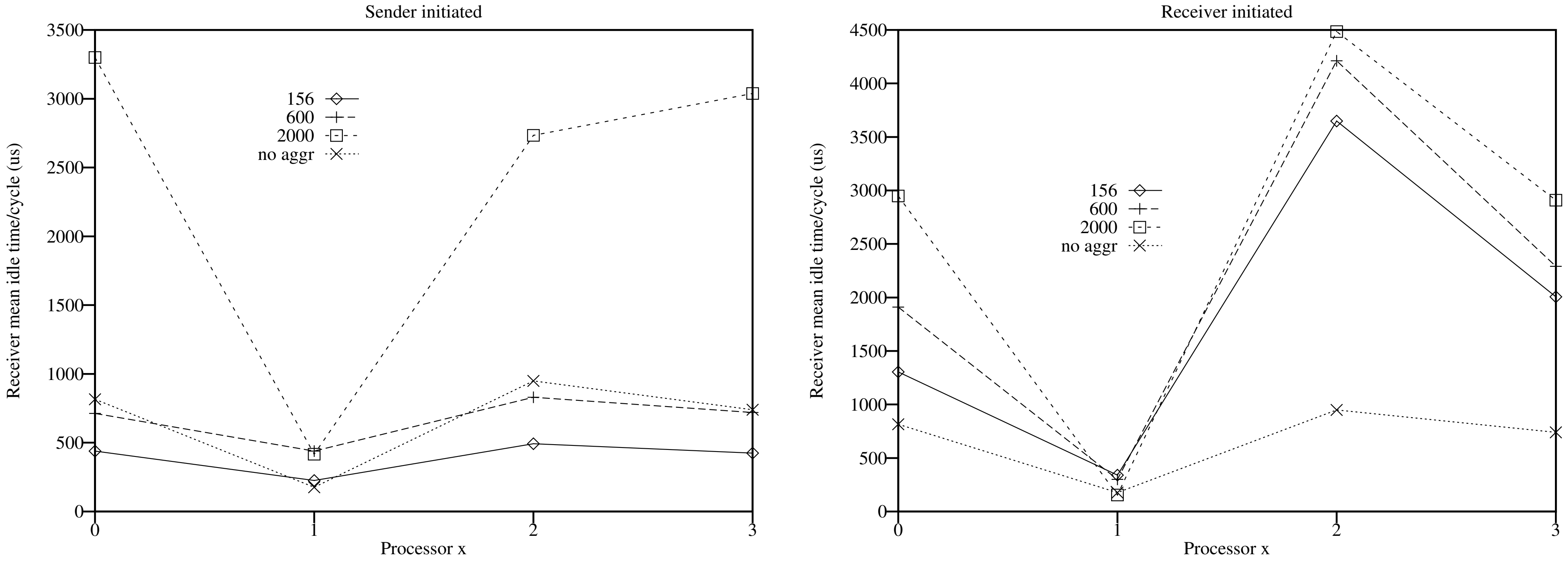


- sequential time = 141s ; unaggregated parallel time = 224s,
- sender-initiated always performs better.

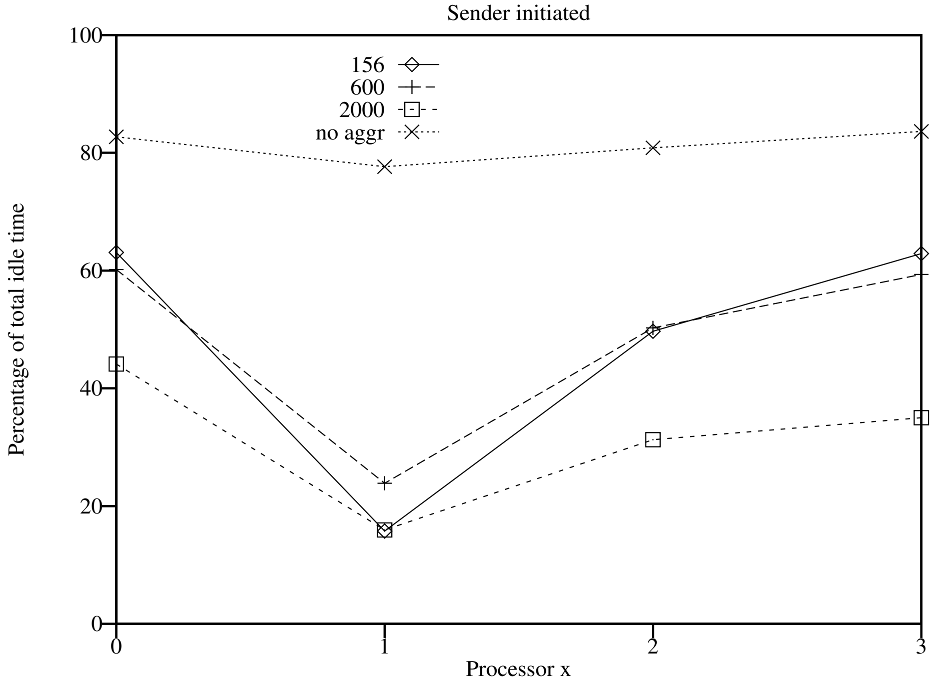
Aggregate holding time: AGGRxxx and R_AGGRxxx

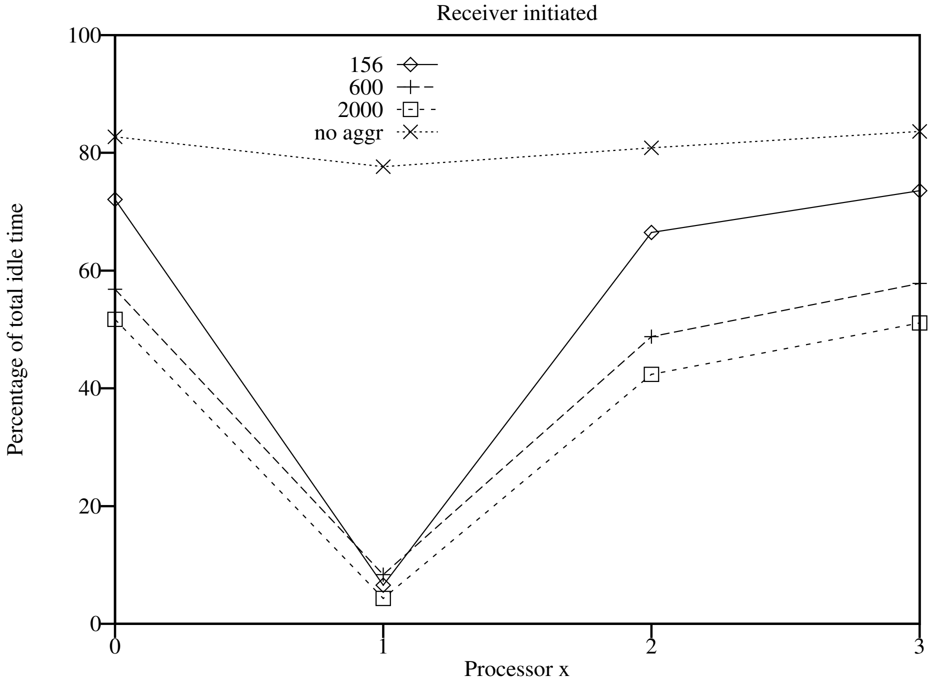


Mean idle time: AGGRxxx and R_AGGRxxx



% of total idle time: AGGRxxx and R_AGGRxxx





Conclusions

- Sender-initiated is simple and is found to always performs better.
- In the receiver-initiated scheme the number of AGGR_REQ messages is too large.
- On going implementation: use piggy-backing for AGGR_REQ.
- Adding load balancing may give better results.
- To know more on cluster-based simulation, Myrinet, BIP, MPI-BIP: `http://resam.univ-lyon1.fr`