



Time Warp – from Cluster to Grid

Kamil Iskra

Section Computational Science

Computing, System Architecture and Programming Laboratory

Universiteit van Amsterdam

the Netherlands

kamil@science.uva.nl



Table of Contents

- What is Time Warp
- The Aim
- Environment
- Routing processes
- Message aggregation
- Global Virtual Time algorithm
- Conclusion



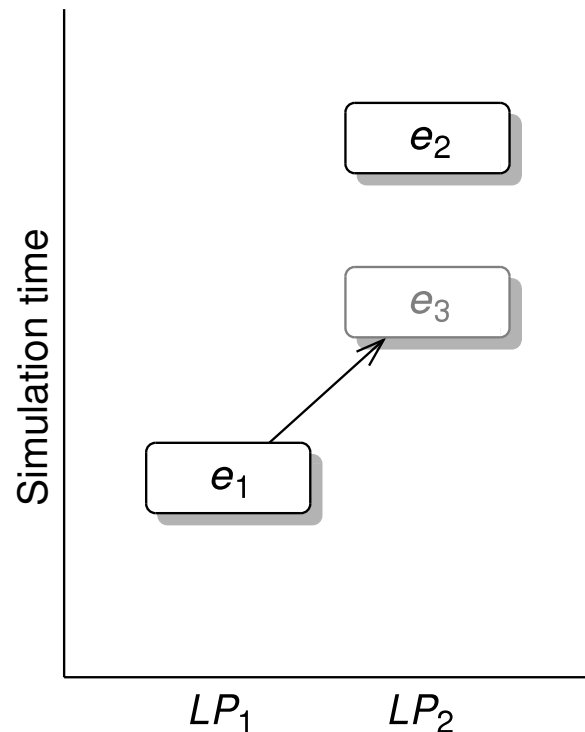
What is Time Warp?

Discrete event simulation decomposed into **Logical Processes**, each with its own:

- state,
- event list,
- clock (**Local Virtual Time**).

Communication between Logical Processes via **external events**.

Optimistic PDES **Time Warp** protocol performs **speculative computations**: events executed so long as the event list not empty.



Causality error occurs if the timestamp of arriving external event is lower than the recipient's LVT. Prematurely executed events need to be **rolled back**: old state restored, externally scheduled events annihilated with **anti-messages**.

Maintaining a log of past states and events executed and sent necessary to roll back.

Global Virtual Time periodically calculated to keep the queue sizes within limits. Together with **Virtual Time Window** can limit (over-)optimism.



The Aim

Efficient execution of parallel discrete event simulations on wide-area distributed computing resources.

Large problems need large computing power, and this can be highly distributed.

Grid community makes considerable efforts to integrate distributed resources.

Most obvious problems:

- network latency,
- inhomogeneous resources,
- resource (un)availability.

Testbed

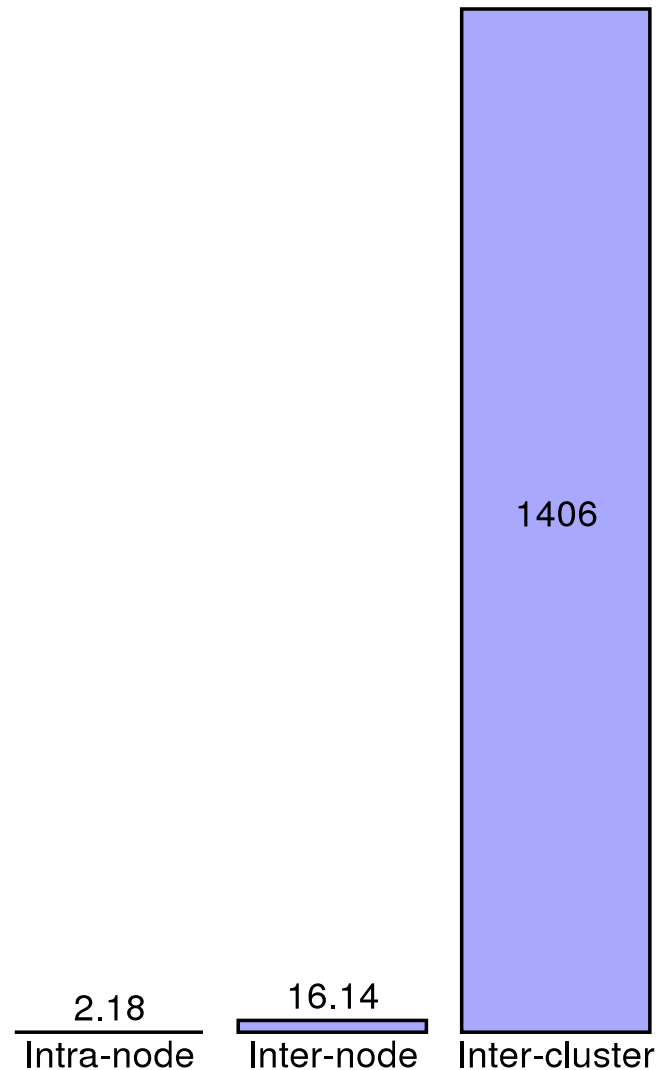


DAS-2:

- 200 nodes across 5 sites,
- Myrinet 2000 and Fast Ethernet LAN,
- Gigabit WAN,
- Dual P-III 1 GHz CPU nodes,
- Linux kernel 2.4.x,
- MPICH 1.2.5-1a / 1.2.6,
- Globus 2.2.4 / 3.2.



Communication layer



Travel times for messages of 156 bytes, in μs .

Various MPICH flavors:

GM: Myrinet,

G2: TCP/IP,

G2/GM: Myrinet inside each cluster,
TCP/IP between clusters.

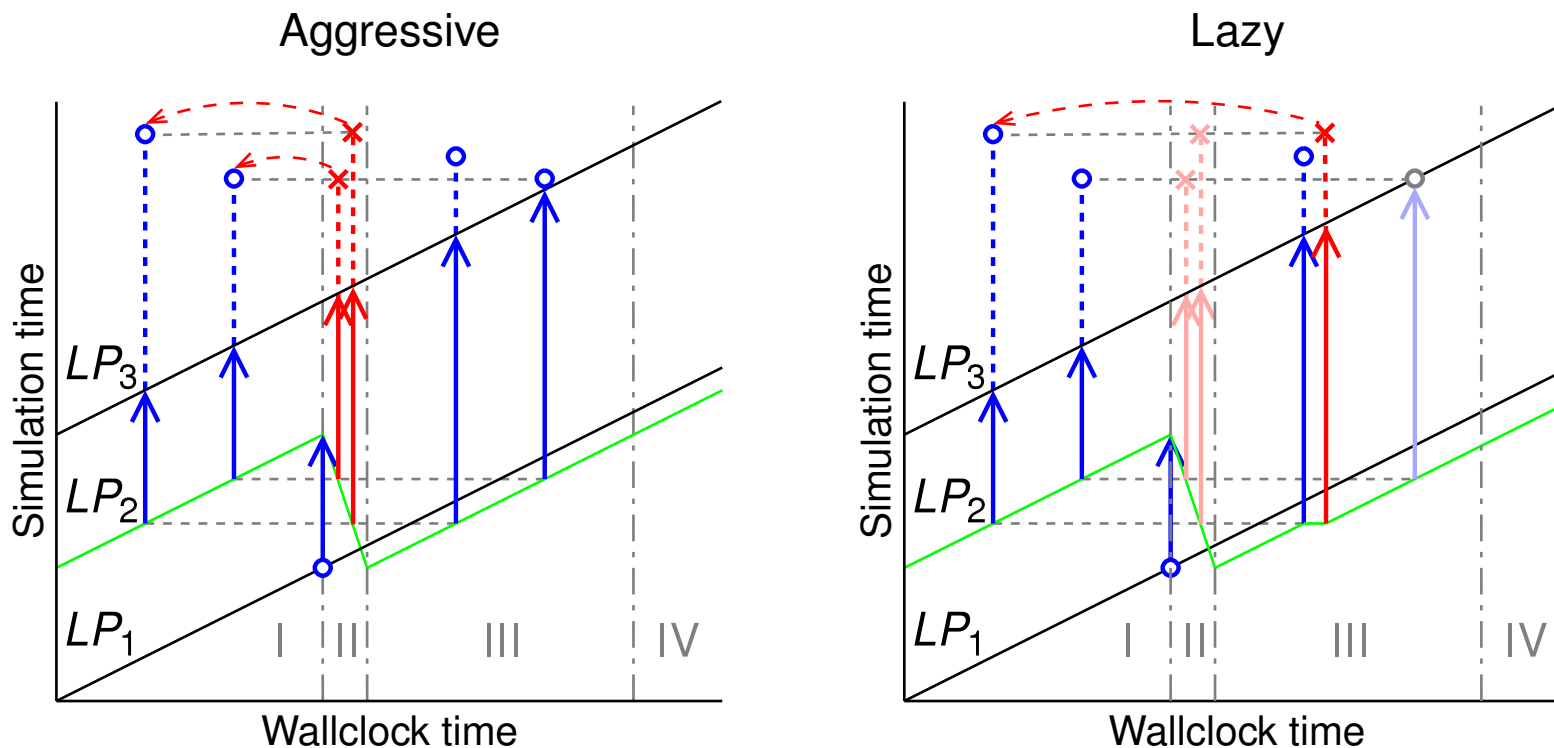


Kernel

APSYS (originally developed by B. J. Overeinder):

- optimistic PDES (Time Warp) simulation kernel,
- developed for clusters,
- 1:1 mapping between LPs, UNIX processes and processors,
- runs on top communication libraries such as MPI or PVM,
- incremental state saving,
- virtual time window bound,
- multiple cancellation strategies.

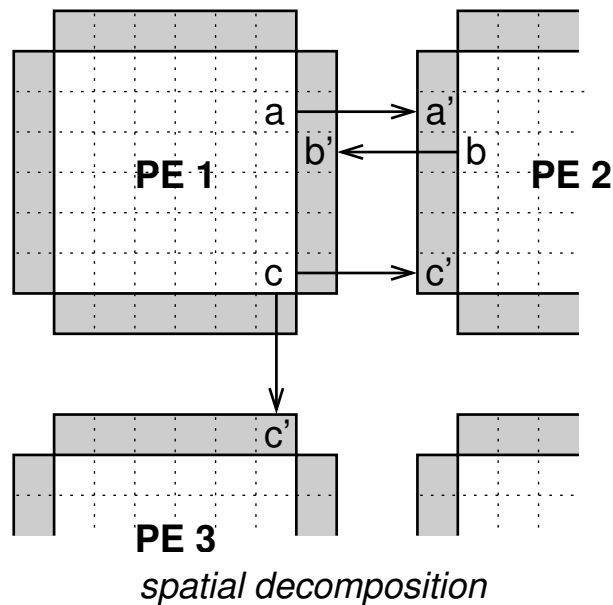
Cancellation strategies



Success of lazy cancellation depends on correlation between events.



Applications: Ising spin

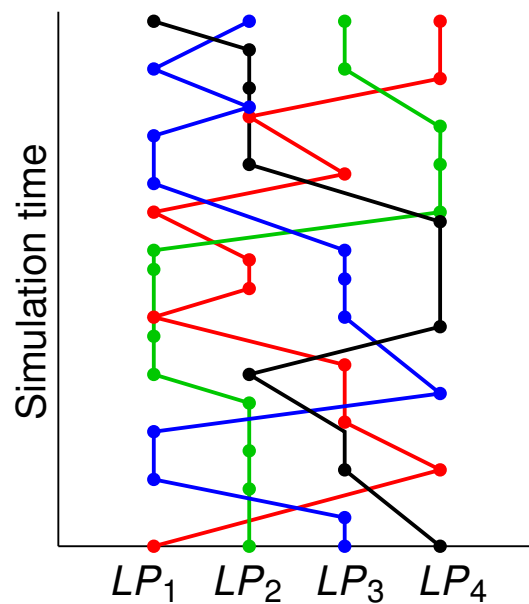


- cellular automata model of magnetization,
- torus topology,
- self-sustaining,
- external events scheduled for current simulation time,
- model temperature determines communication intensity.

Used as a benchmark.



Applications: PHOLD



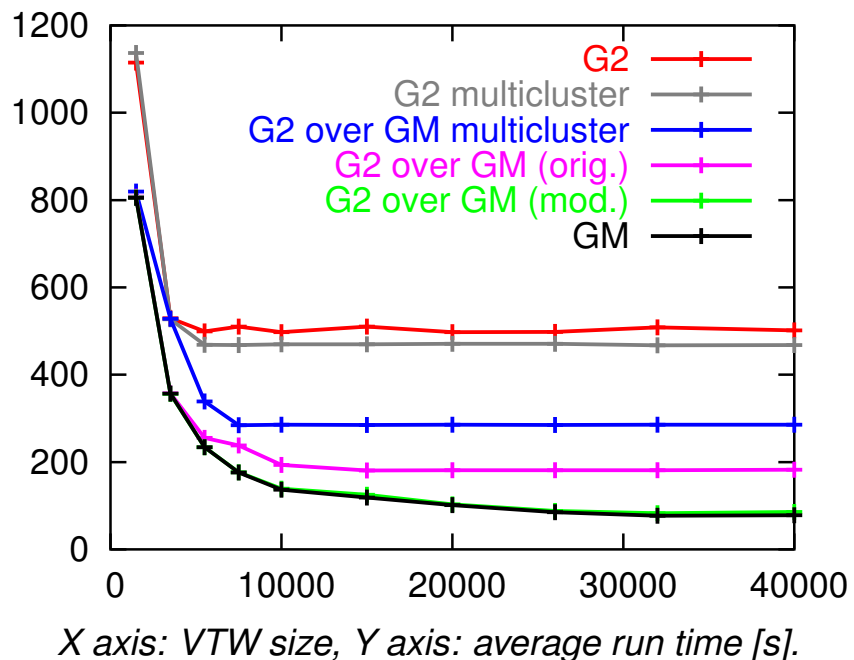
- synthetic load,
- fixed number of event threads moving through LPs,
- message-initiated model.

As used here:

- events scheduled locally or on direct neighbors,
- tunable event granularity.



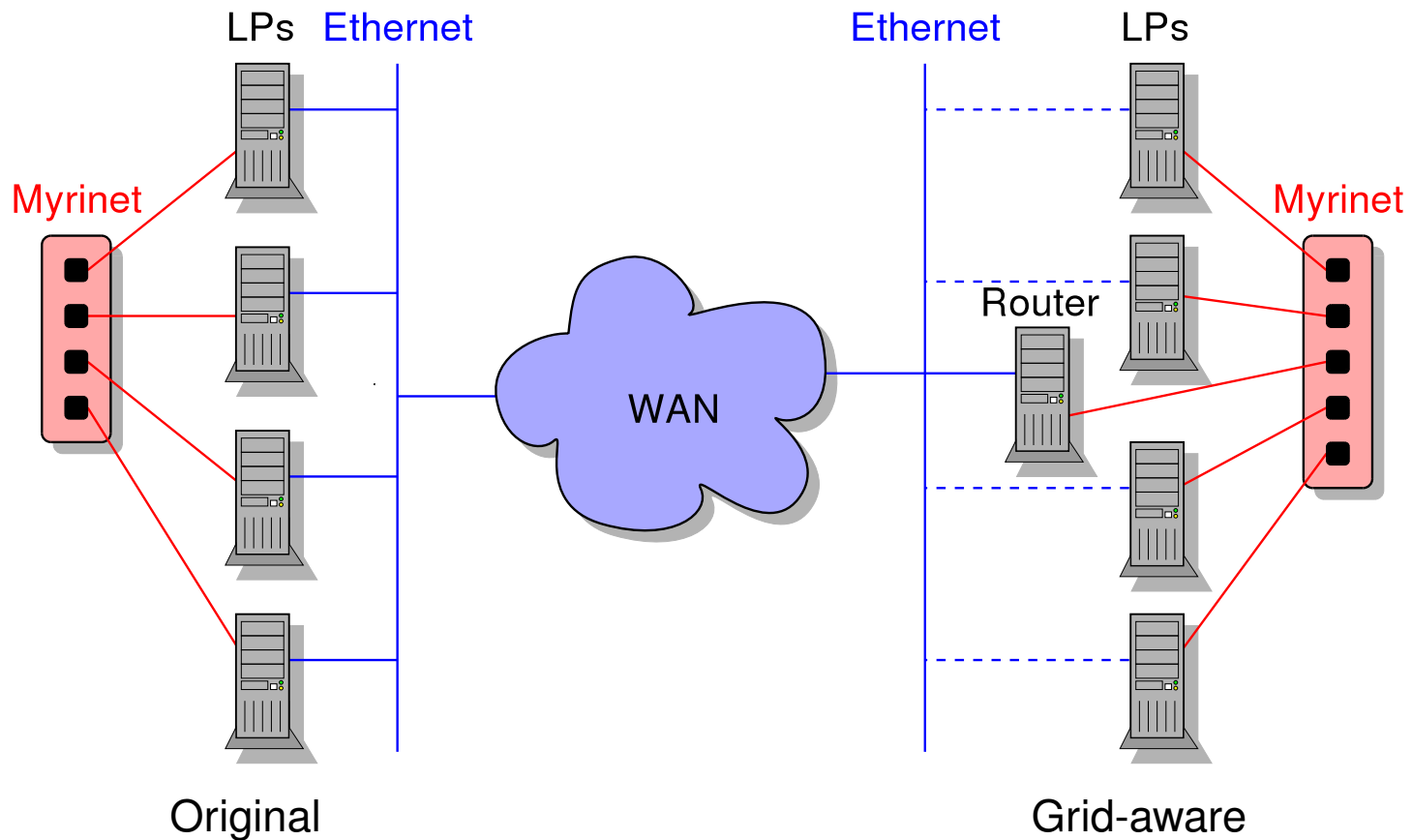
Motivation



- Ising spin,
- 24 processes,
- lazy cancellation,
- high temperature ($T = 3.5$),
- 750–5000 msg/sec per process.

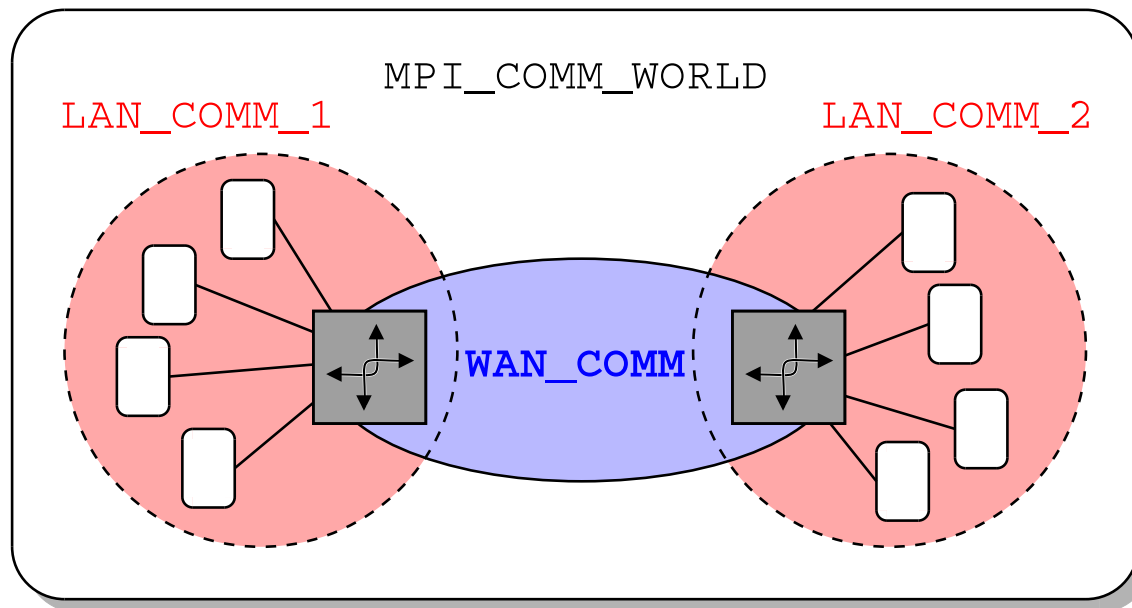
Advanced process topology can reduce communication overhead.

Grid-aware topology



Routing processes relieve LPs of WAN communication.

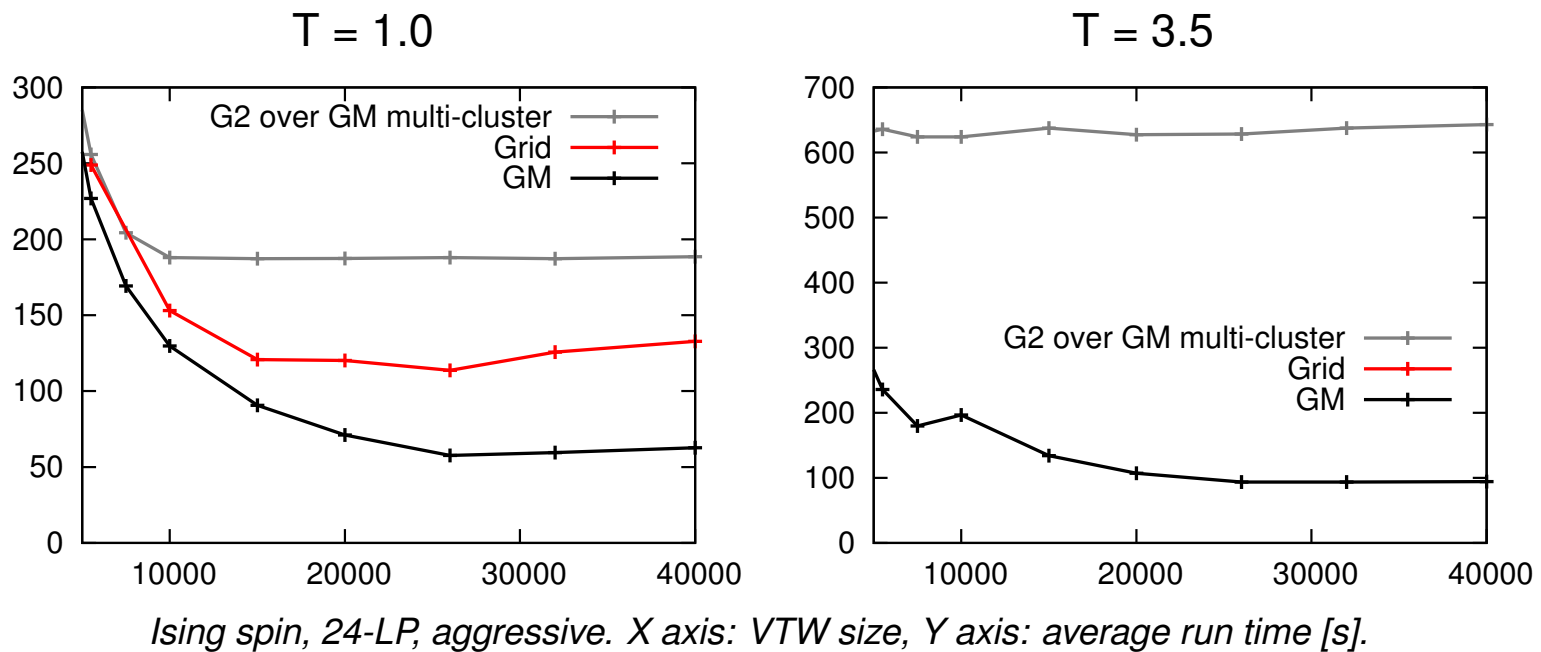
Optimized communication



Simulation processes do not need to poll TCP/IP sockets.



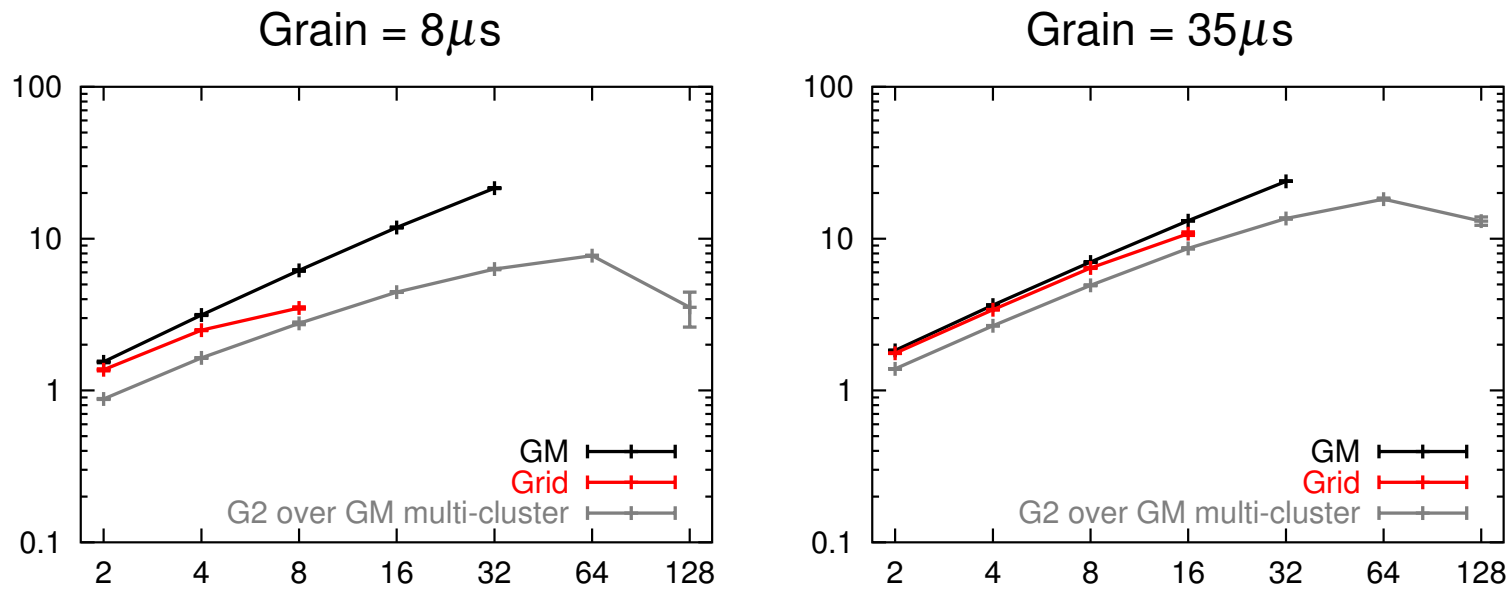
Run times with Ising spin



Routing processes improve performance, but stability can be a problem.



Speedup with PHOLD

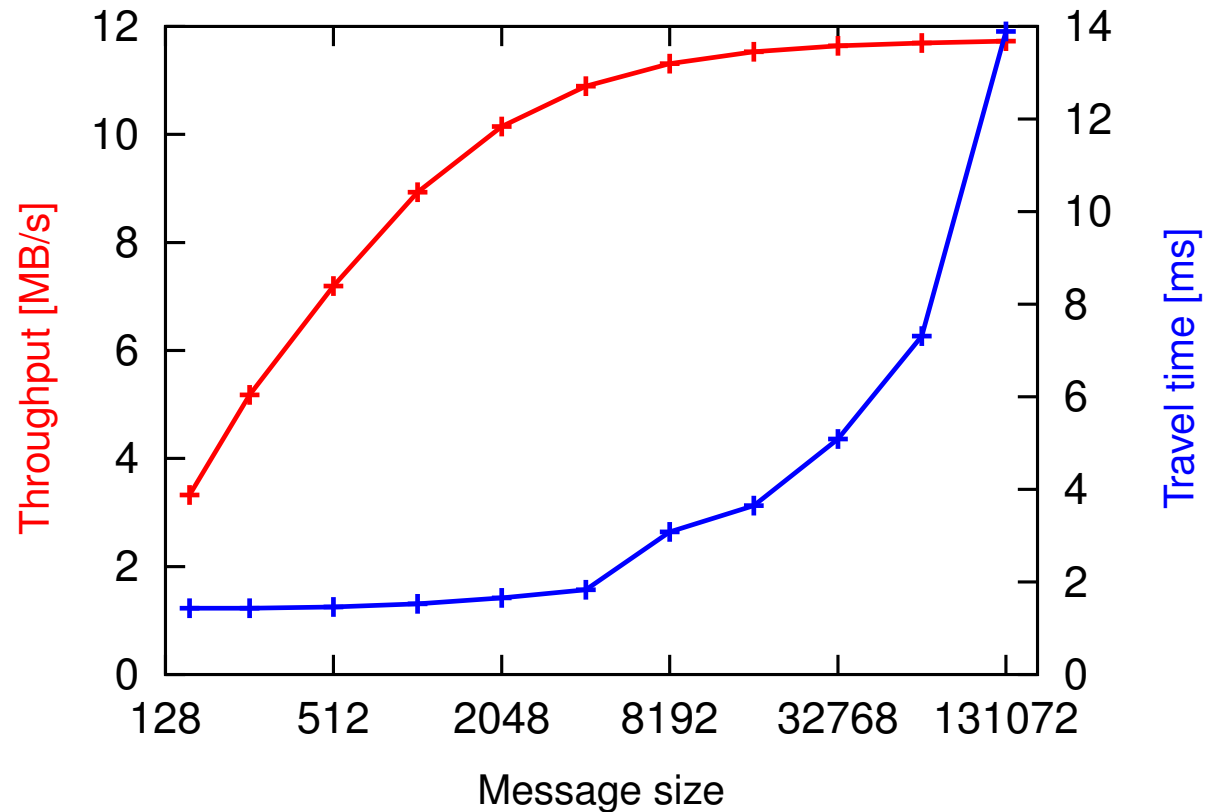


PHOLD, aggressive, no VTW. X axis: number of processes, Y axis: speedup.

- 1024 event threads,
- $P = 128$ runs performed on four clusters,
- 2.5% external events for $P = 2$, 20% for $P = 128$.



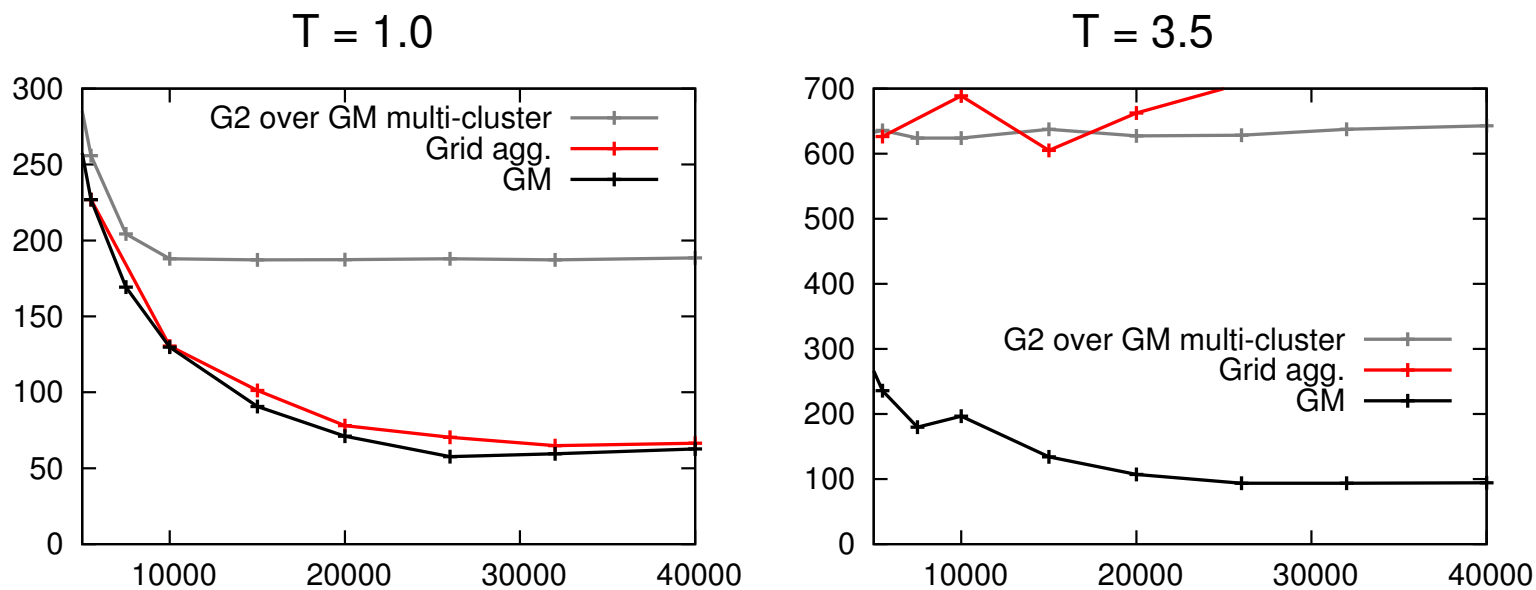
Message aggregation



- tunable aggregation buffer size and timeout,
- buffer flushing based on message type.



Run times with Ising spin

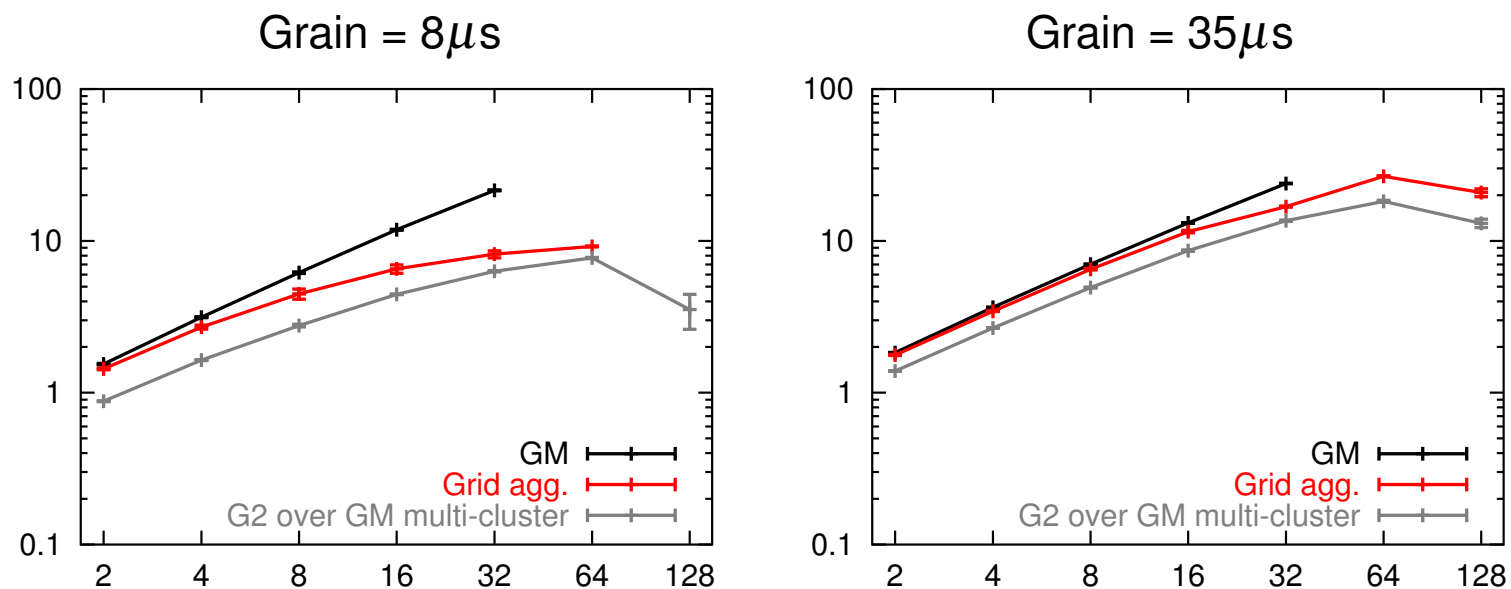


Ising spin, 24-LP, aggressive. X axis: VTW size, Y axis: average run time [s].

Aggregation improves performance and stability.



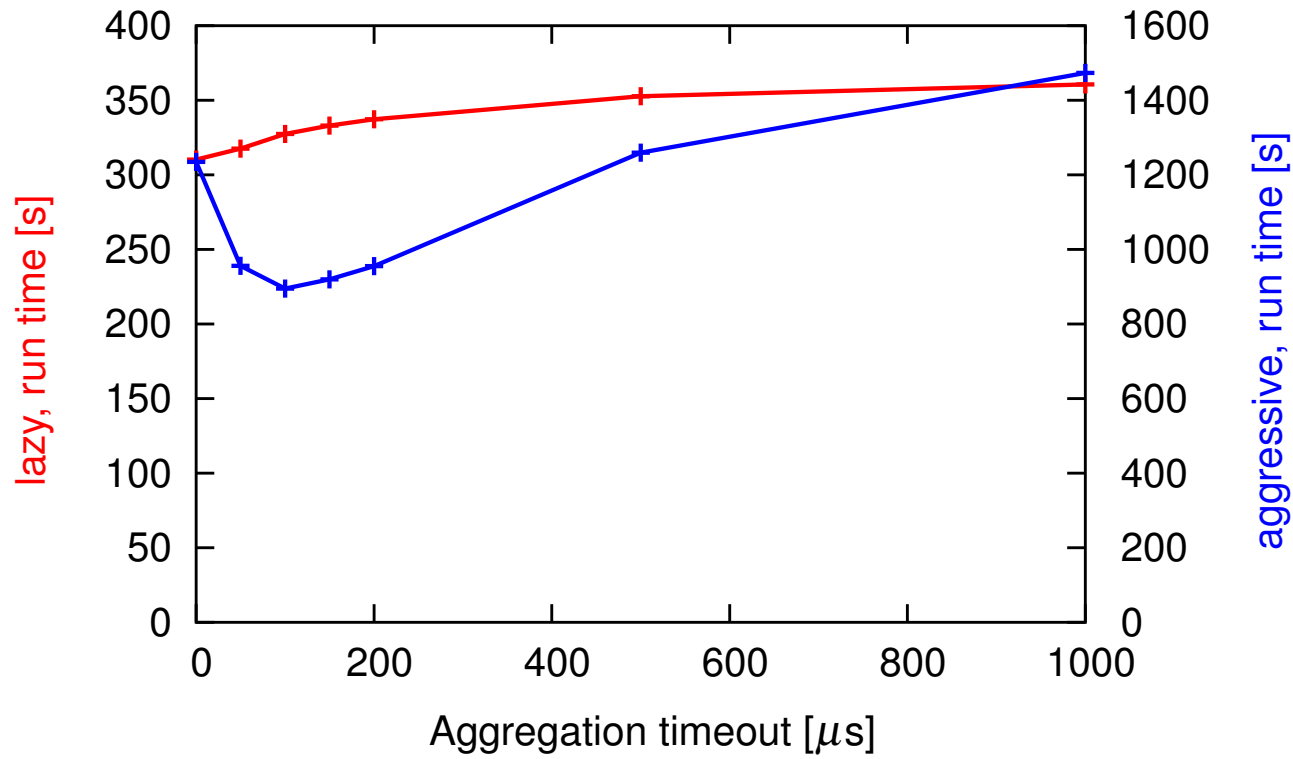
Speedup with PHOLD



PHOLD, aggressive, no VTW. X axis: number of processes, Y axis: speedup.



Aggregation timeout vs. run time

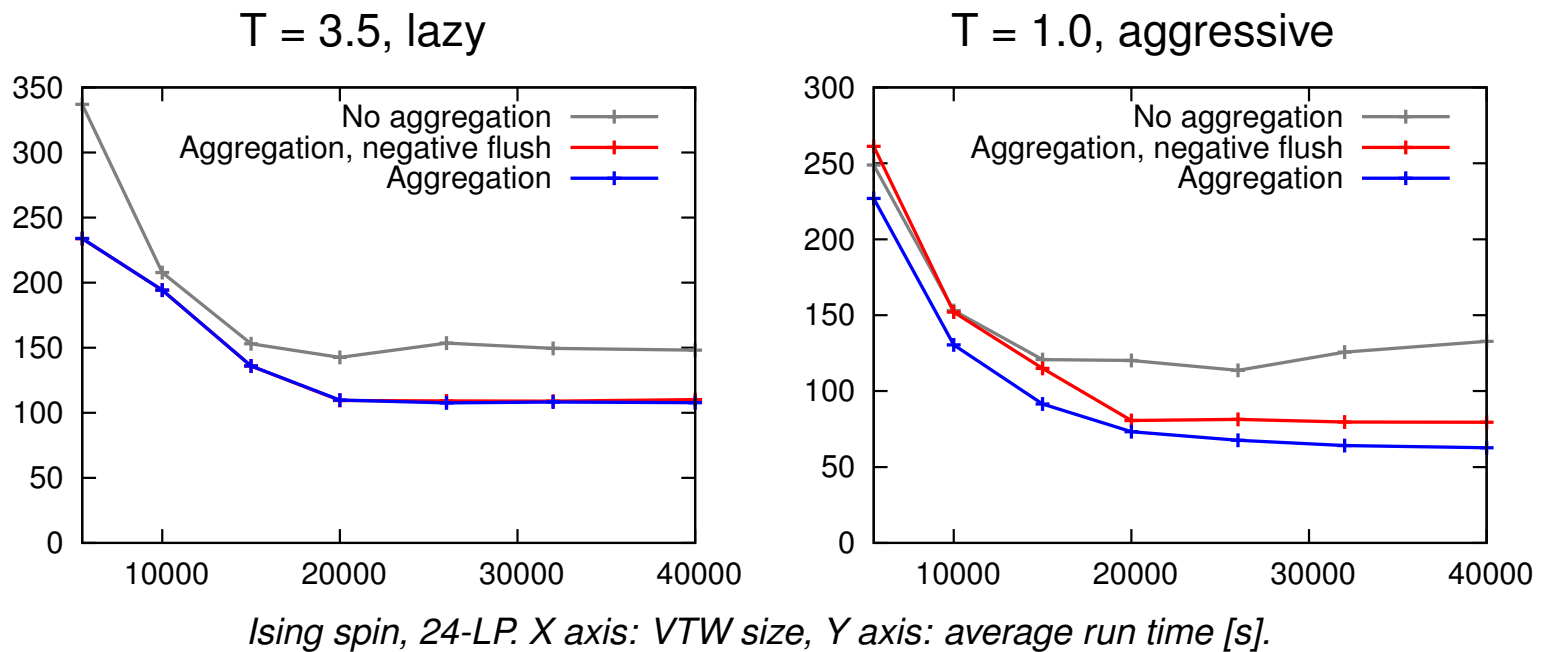


Ising spin, 8-LP, $T = 3.5$, $VTW = 40000$

Small timeouts are best!



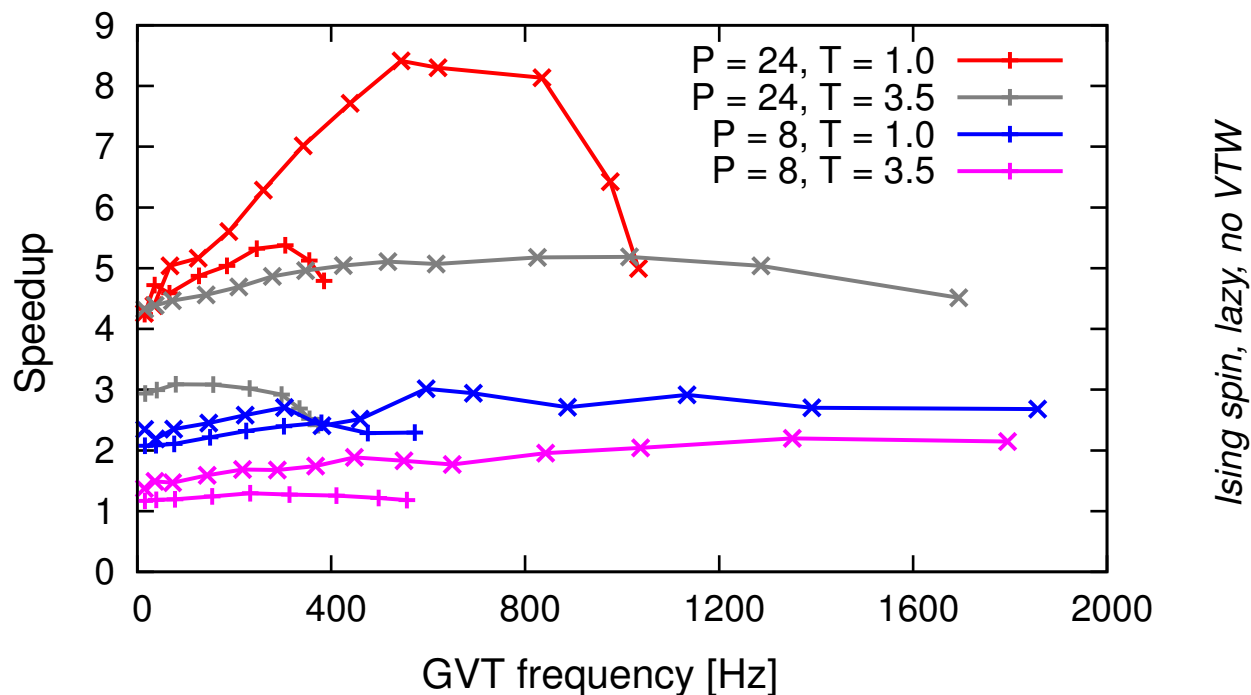
Flushing aggregation buffers



Making negative messages flush aggregation buffers degrades performance!

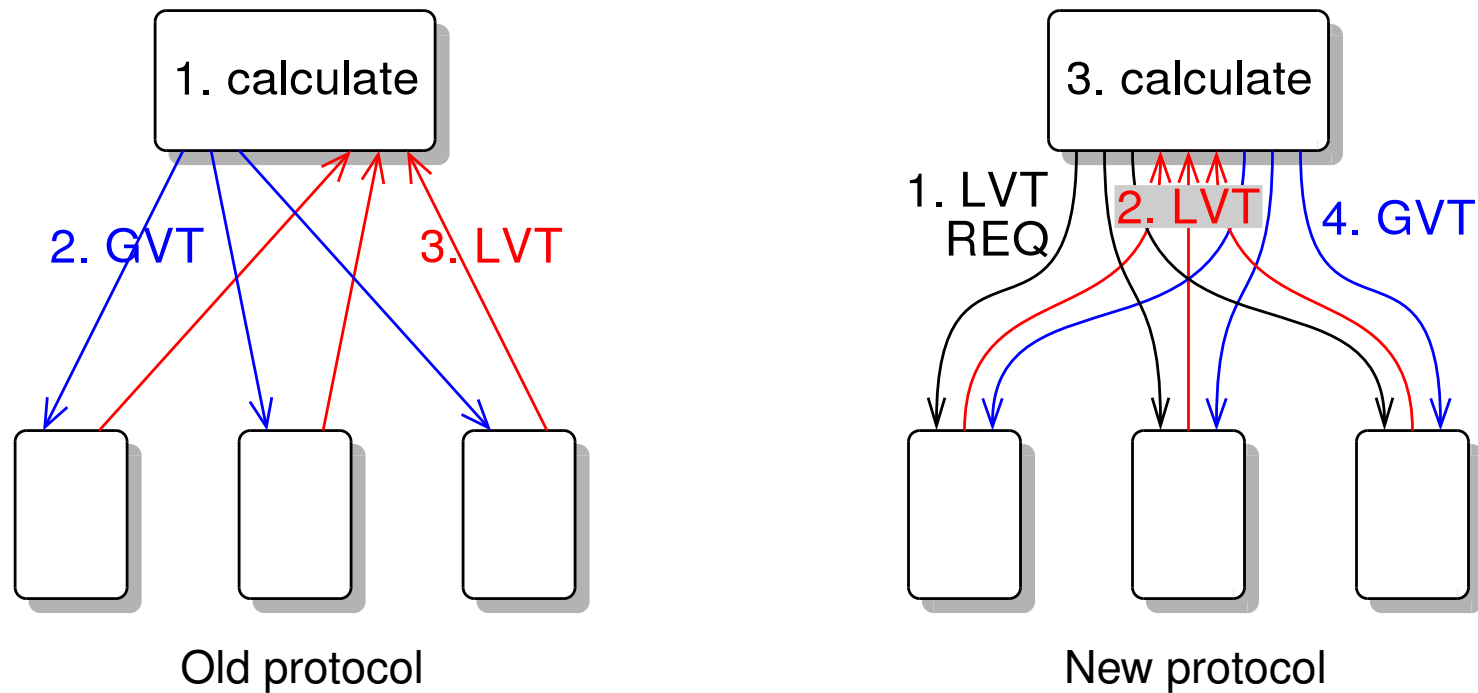


The importance of a good GVT algorithm



Maximum GVT frequency in multi-cluster runs (+) far lower than in single-cluster runs (x) due to large network latency.

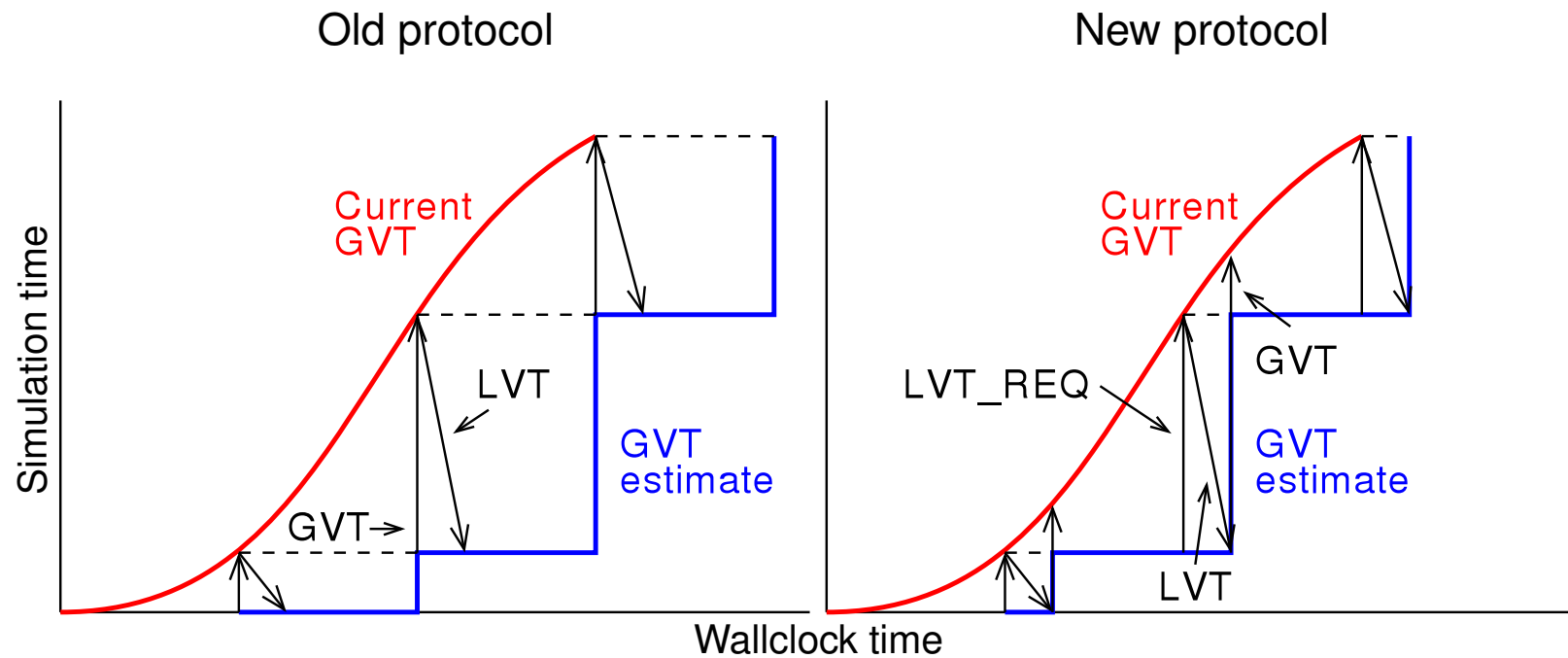
GVT algorithm: information gathering



New protocol requires 50% more messages.

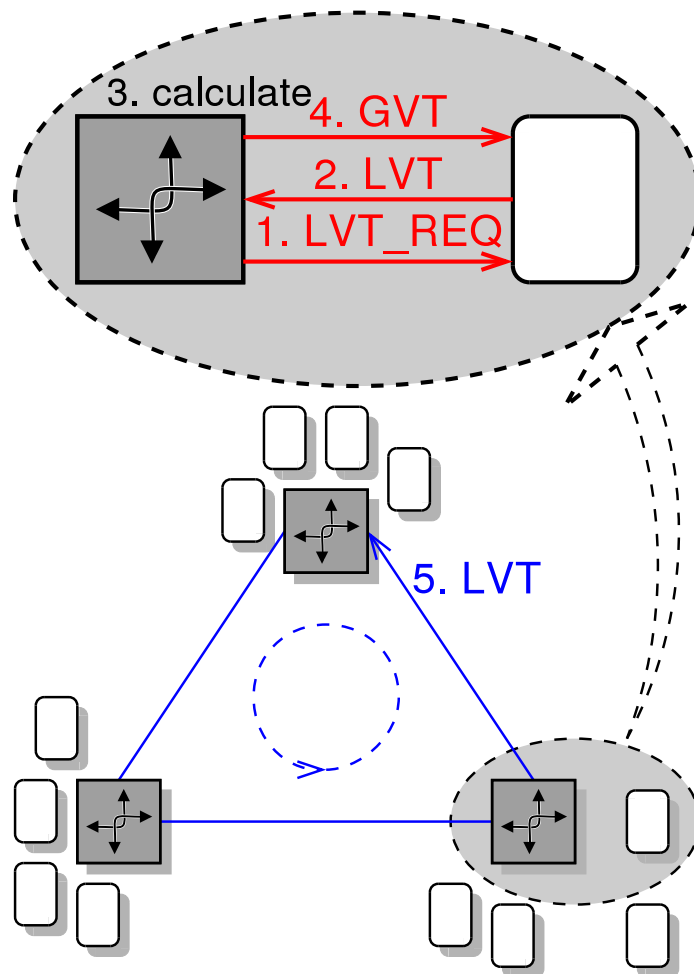


GVT algorithm: information gathering



But on average the kernel only needs half of memory.

GVT algorithm: distribution

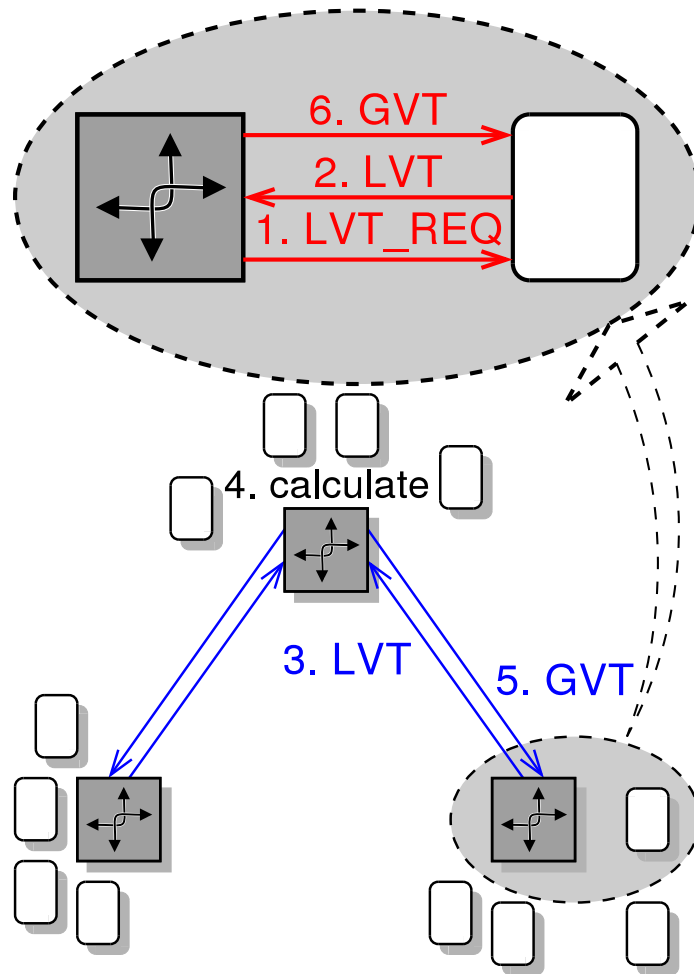


GVT calculation moved from a simulation process to routing processes.

Decentralized:

- data from remote clusters is older than local,
- protocol efficient also with high frequencies.

GVT algorithm: distribution

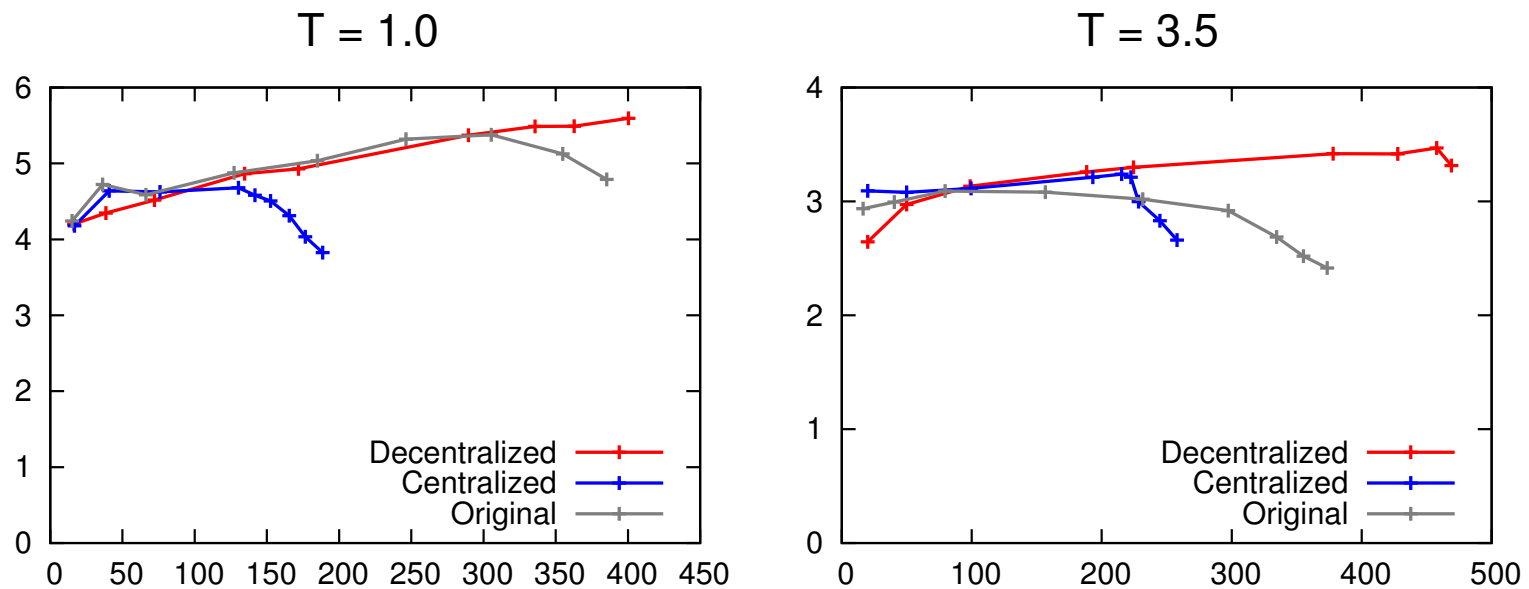


Centralized:

- all data has the same age,
- first step needs to be kept synchronized,
- more complex message exchange protocol.



Comparison of GVT algorithms



Ising spin, 24-LP, lazy, no VTW. X axis: GVT frequency [Hz], Y axis: speedup.

“Decentralized” eventually a winner, but not by much.



Conclusion

- Network latency is a major problem in successfully deploying parallel simulations on the Grid.
- Dedicated routing processes improve the performance in most cases.
- Message aggregation with a short timeout is generally a win.
- High communication rate remains a challenge, routing processes can even make things worse.
- Sufficient computational grain is required to achieve a good speedup in multi-cluster runs.
- High quality GVT estimate, helpful for Ising spin model, is difficult to achieve in multi-cluster runs.



Thank you!

Group web site:

<http://www.science.uva.nl/research/scs/>

My thesis:

<http://dare.uva.nl/en/record/159562>



Aggregation timeout vs. communication

