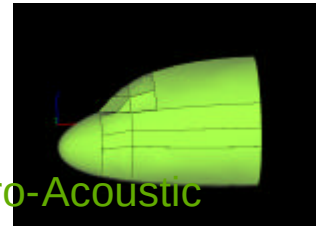
MPI Development Tools and Applications for the Grid

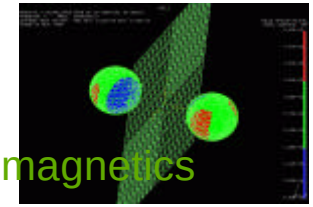
Rainer Keller, Bettina Krammer, **Matthias Mueller**, Michael Resch
High Performance Computing Center Stuttgart
mueller@hlrs.de

Edgar Gabriel
Innovative Computing Laboratory
UTK

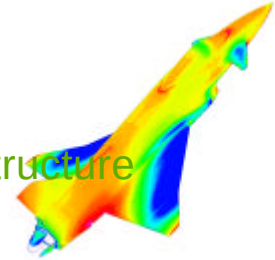
Motivation – Complex, compute-intensive Applications



Vibro-Acoustic



Electromagnetics



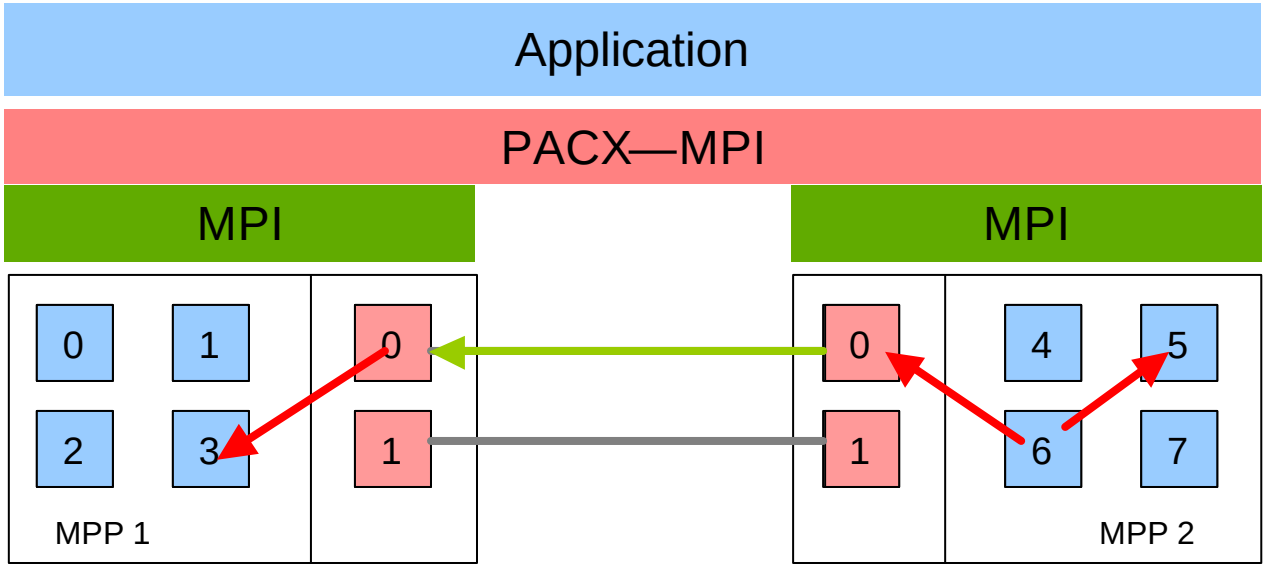
Fluid-Structure

Application

PACX—MPI

NEC-MPI | IBM-MPI | Cray-MPI

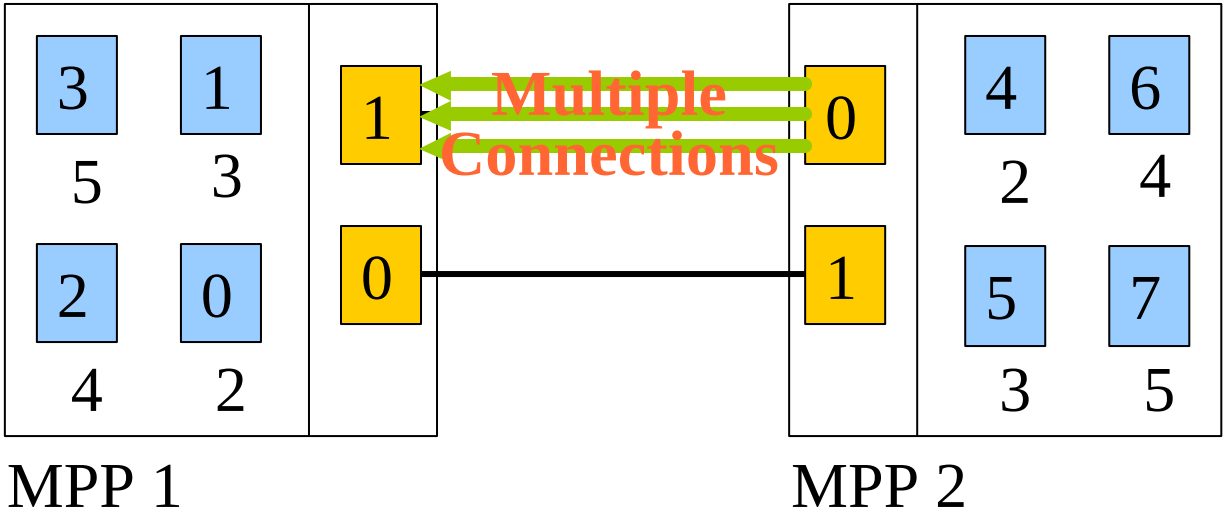
PACX-MPI: A Grid enabled MPI library



- Implementation of MPI optimized for Grid-environments
- Application just needs to be recompiled.
- Current functionality:
  - full MPI 1.2
  - several parts of the MPI-2 standard

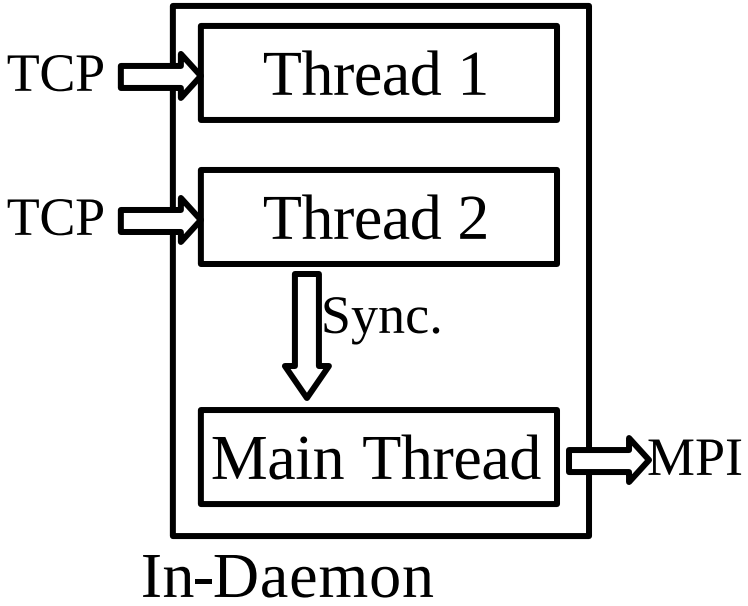
PACX-MPI: Network Optimization

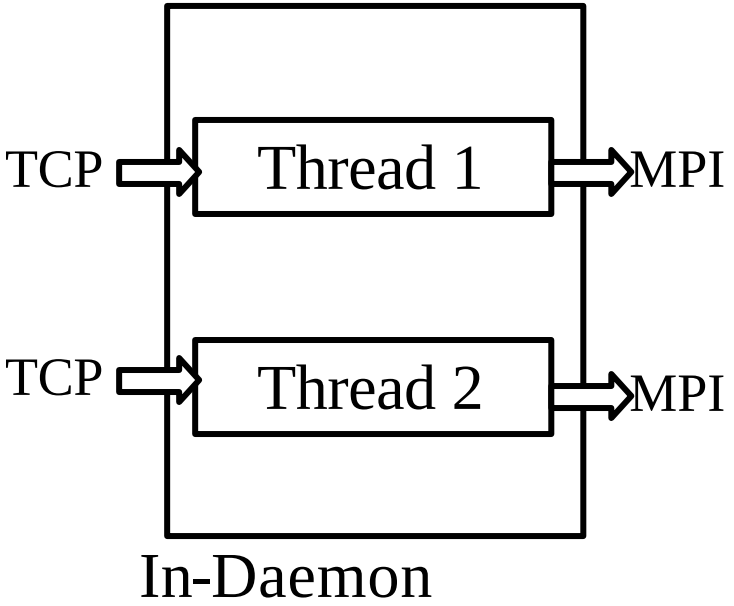
- External Communication is the bottle-neck.
- Bandwidth & Latency varies a lot, e.g.:
  - Bandwidth: 550kB/sec down to 10kB/sec.
  - Latency: 60ms up to 600ms.
- Strive to improve latency / bandwidth.



Integration of Threads in PACX-MPI 1/2

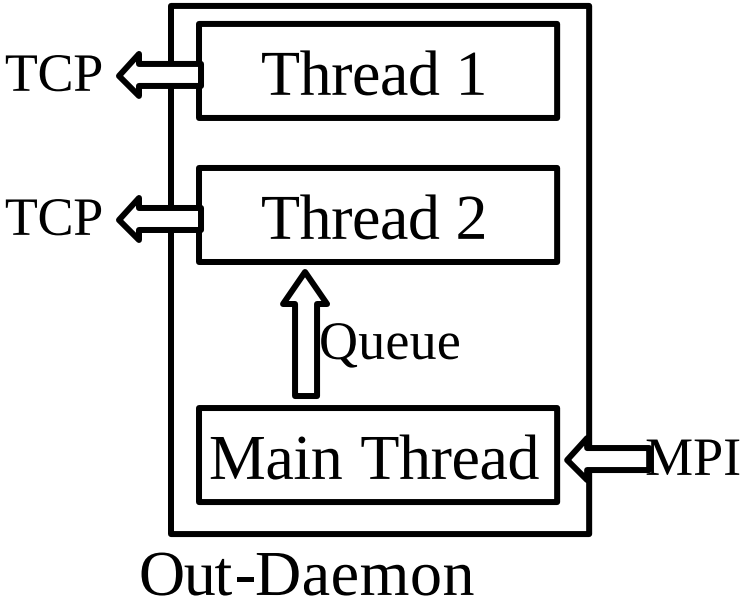
- Threaded versions of two daemons implemented.
- Two versions of the `in_daemon`: For thread-safe and non-thread-safe native MPI-implementations.
  (still we have 2xtimes the TCP-Buffer for non-thread-safe).





Integration of Threads in PACX-MPI 2/2

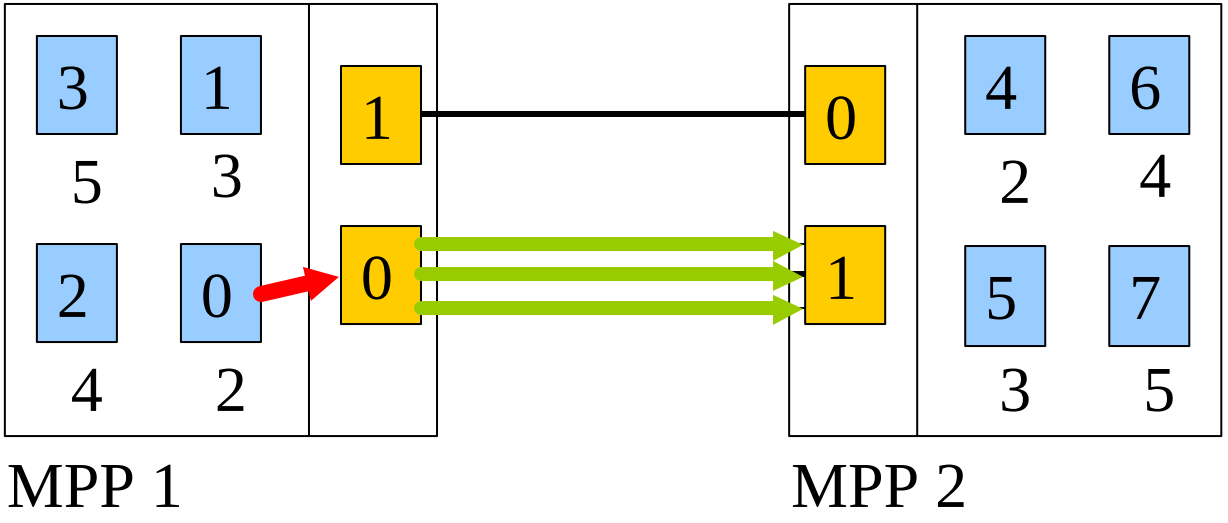
- Only one version of multi-threaded `out_deamon`.
- User processes can't map destination processes to network connections, i.e. threads!



Dynamic open of multiple network connections

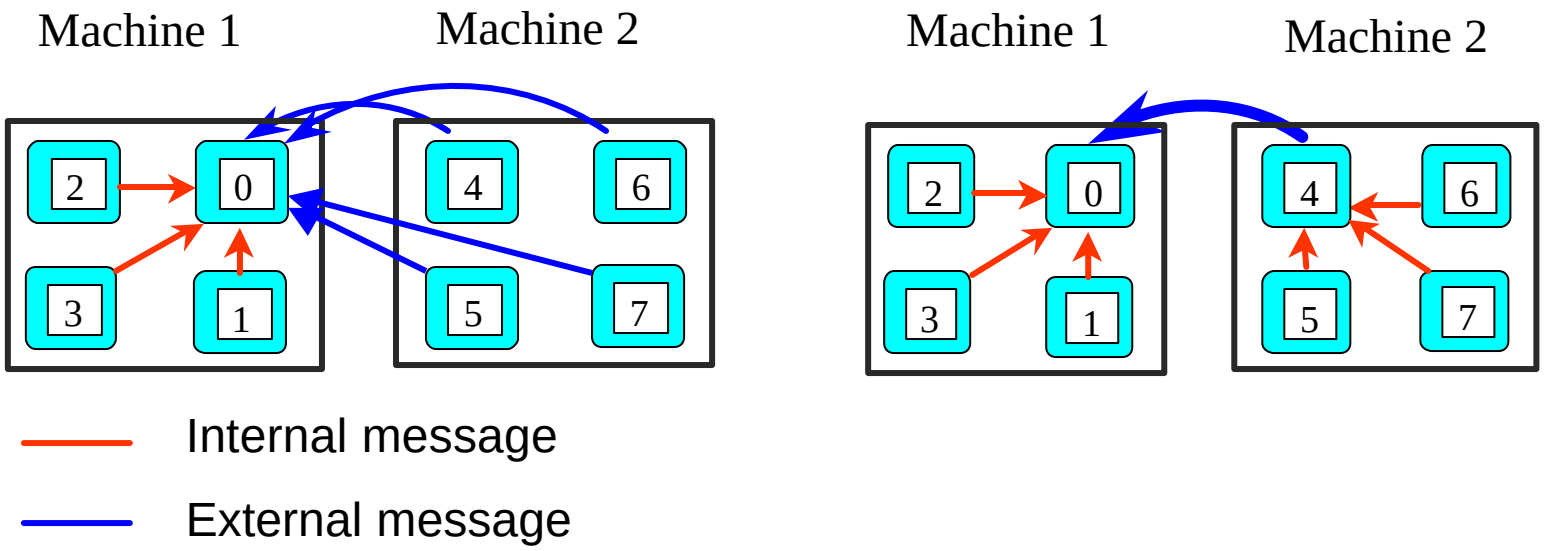
- Multiple network connections may be opened dynamically:

```
new_connections = 2;
PACX_Attr_put(MPI_COMM_WORLD,
PACX_USE_NEWCONN,
&new_connections);
```



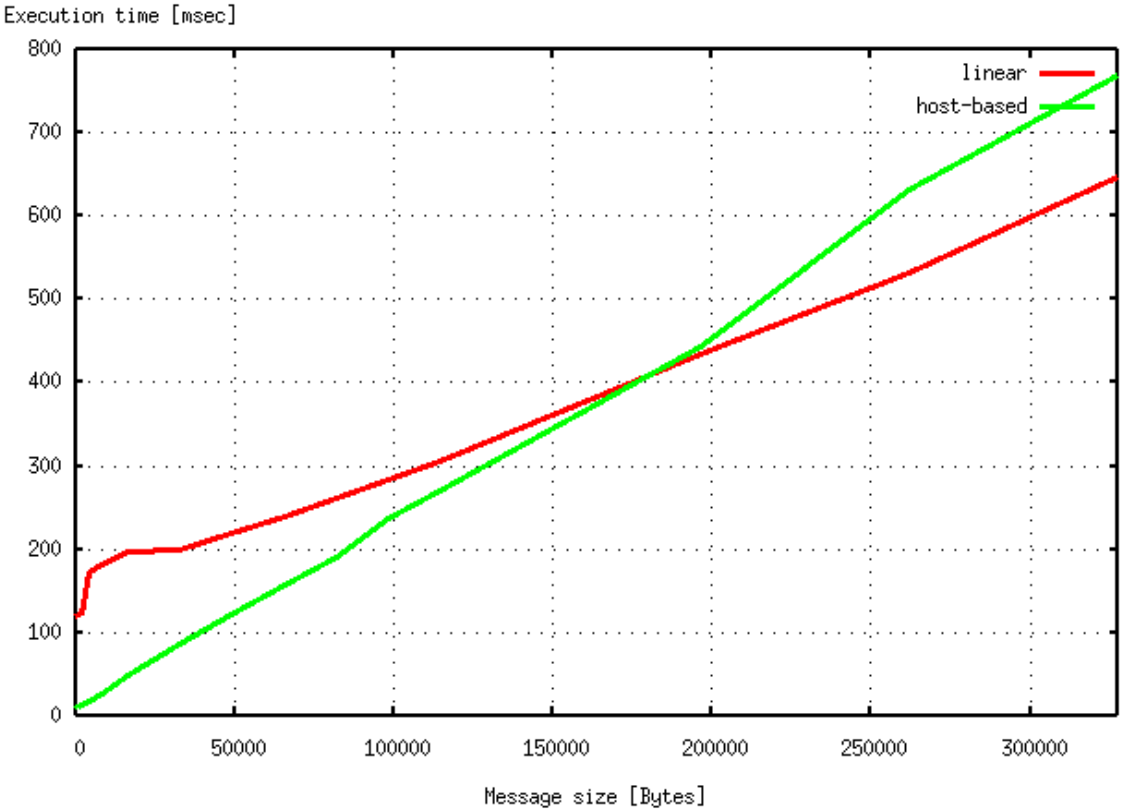
Topology aware collective operations (I)

- Topology aware collective operations presented by Kielmann[1999-2002], Gabriel[1999, 2001], Karonis[2000] and Fagg[2000, 2001]
- Linear vs. topology aware operation: example MPI_Gather(v)



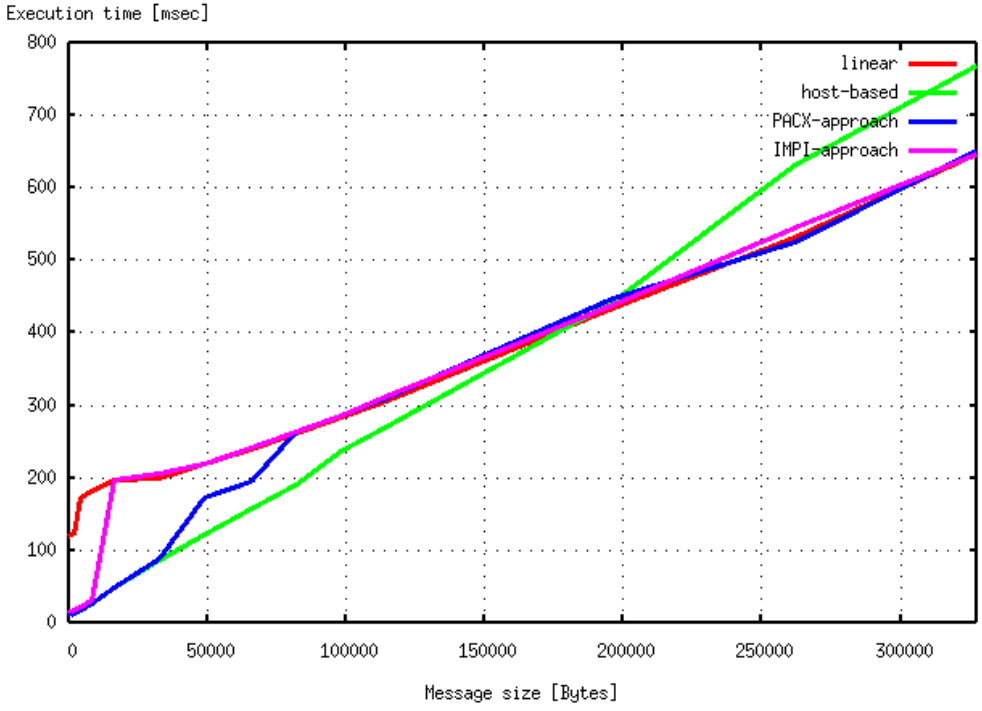
Topology aware collective operations (II)

- Example: MPI_Gatherv on 3 machines using 32-32-16 nodes



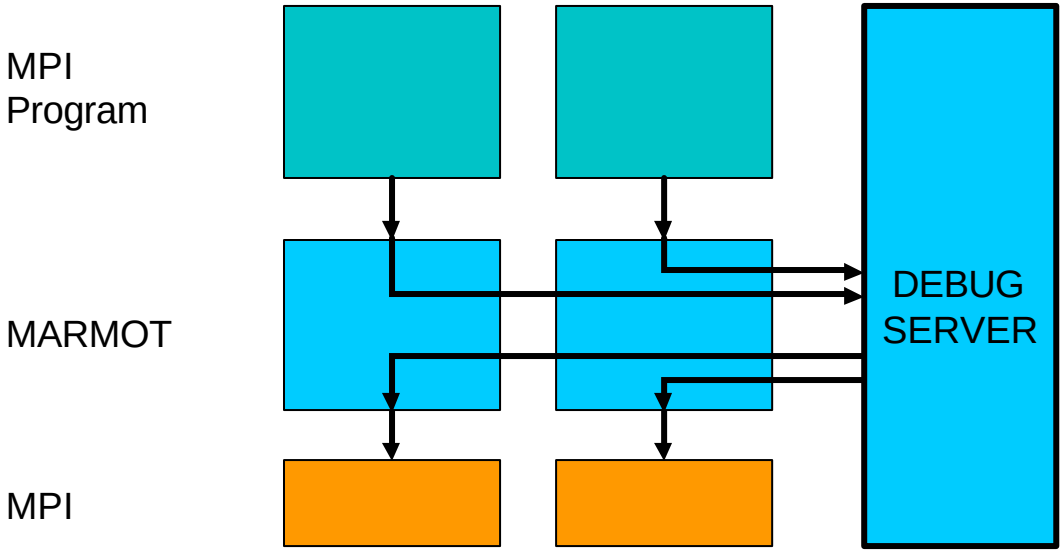
Topology aware collective operations (III)

- Decision which algorithm to use is determined between each pair of hosts
- Smoother approach to the cross-over point



MPI analysis tool: MARMOT

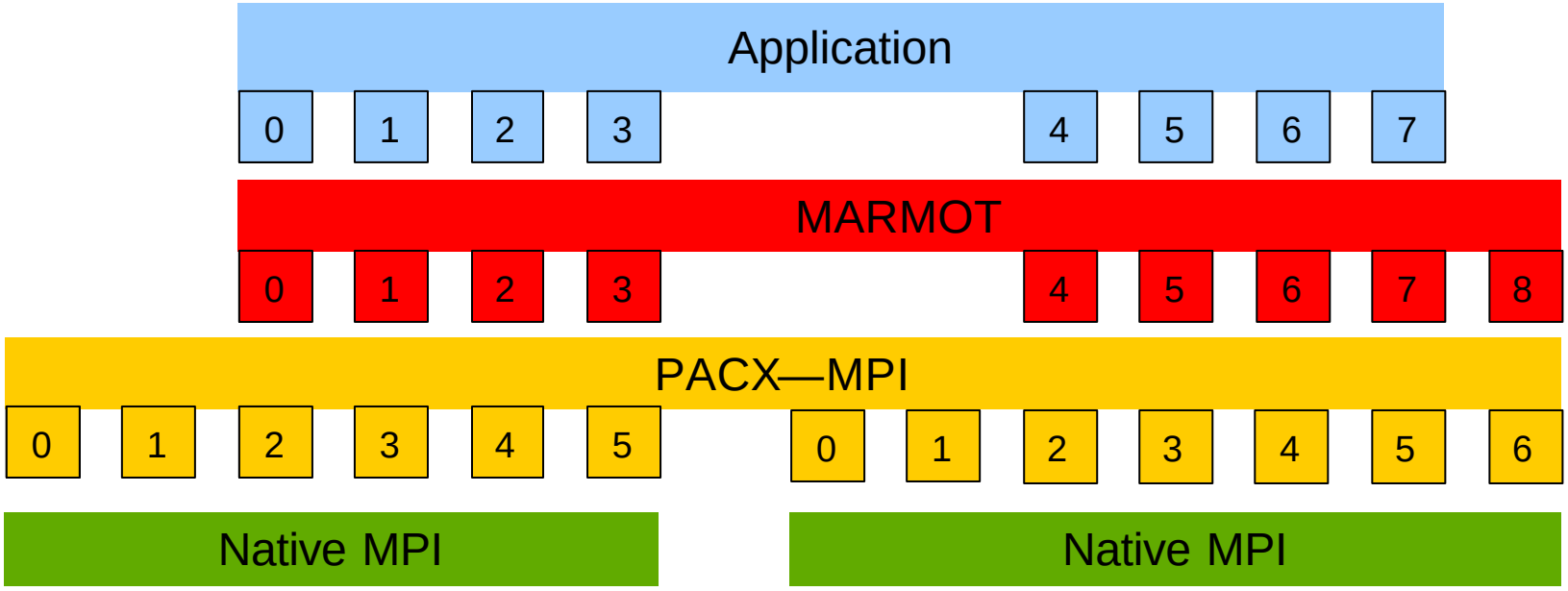
- Many applications have problems with MPI on the Grid
- MPI developers spend a lot of time debugging applications instead of debugging their MPI implementation.
- HLRS develops MPI debugging and verification tool: MARMOT
- Goal: increase reliability and portability of MPI applications



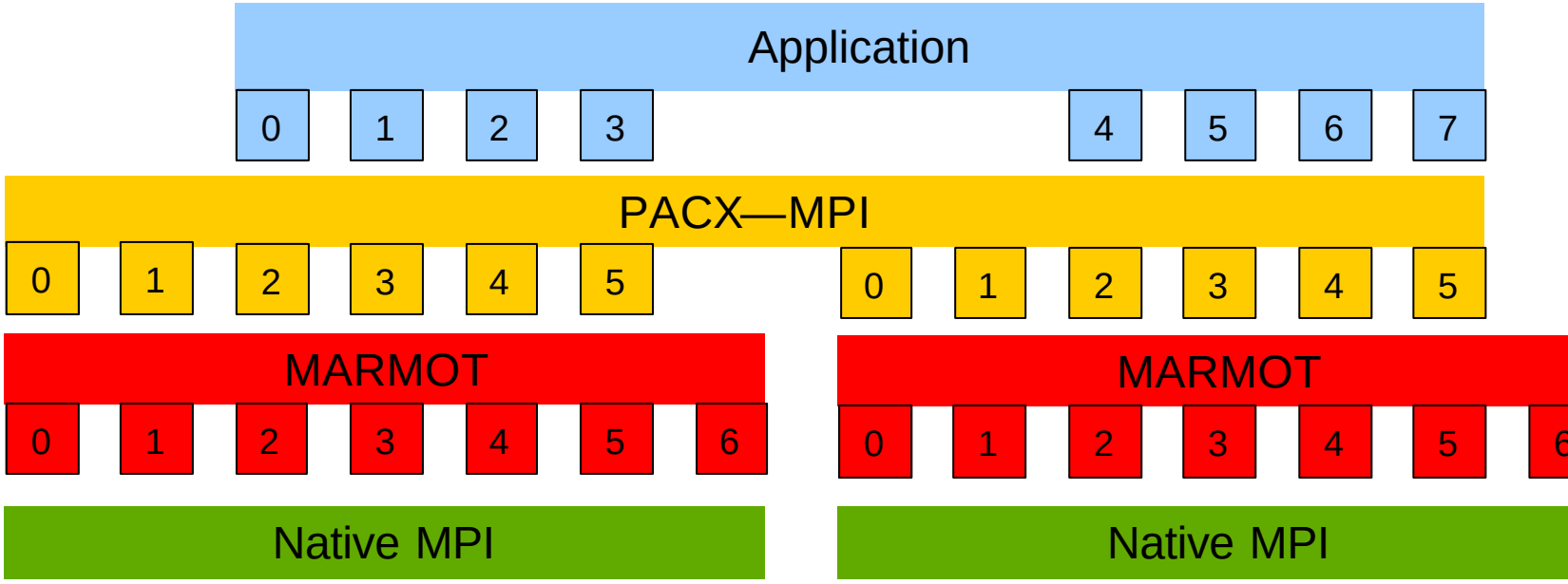
Examples for Checks performed by MARMOT

- Verification of MPI_Request usage
  - invalid recycling of active request
  - invalid use of unregistered request
  - warning if number of requests is zero
  - warning if all requests are MPI_REQUEST_NULL
- Verification of tag range
- Verification if requested Cartesian communicator has correct size
- Verification of communicator in Cartesian Calls
- Verification of groups in Group Calls
- Verification of sizes in calls that create groups or communicators
- Verification if ranges are valid (e.g. in group constructor calls)
- Verification if ranges are distinct (e.g. MPI_Group_incl, -excl)
- Check for pending messages and active requests in MPI_Finalize

MARMOT used to debug a Grid application



MARMOT used to debug PACX-MPI



Applications Level Issues

Matthias Müller
Höchstleistungsrechenzentrum Stuttgart

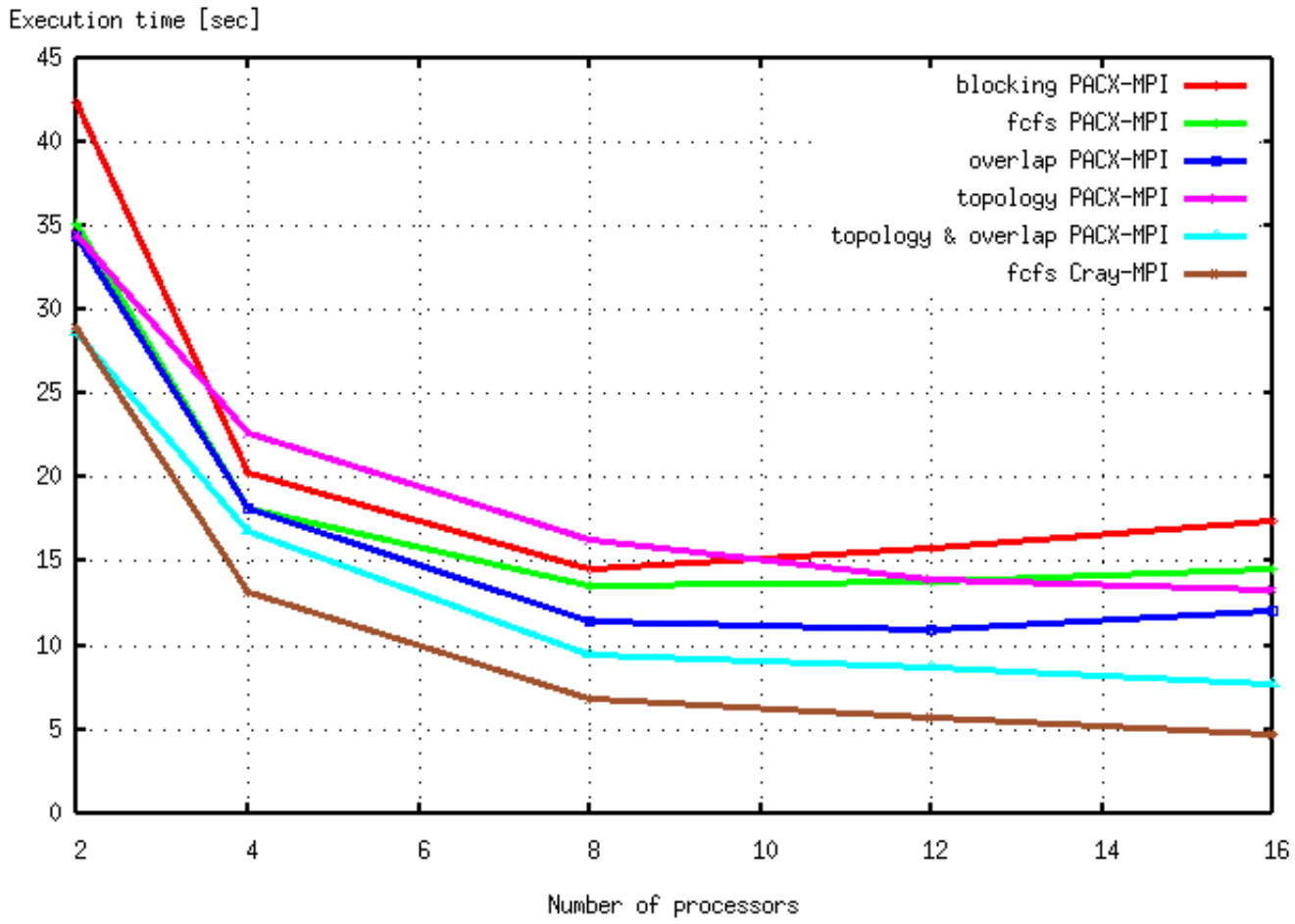
Example: parallel equation solver

Comparison of execution time between

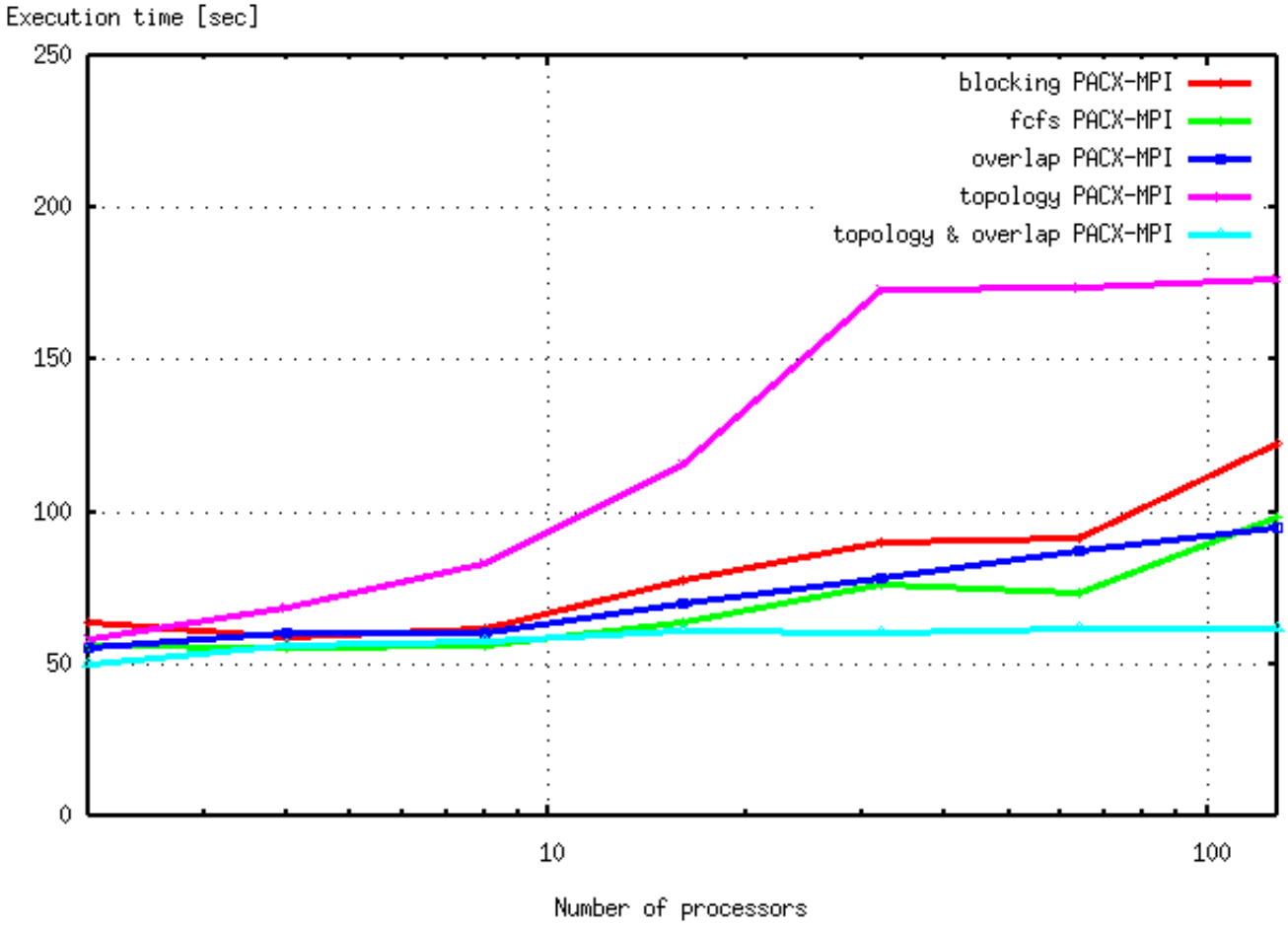
- Executing on a single Cray T3E using Cray MPI:
  - **7 microseconds latency**
  - **300 MB/s bandwidth**

- Executing with PACX-MPI on 2 Cray T3E's and different algorithms:
  - **4 milliseconds latency**
  - **10 MB/s bandwidth**

Performance evaluation: speedup

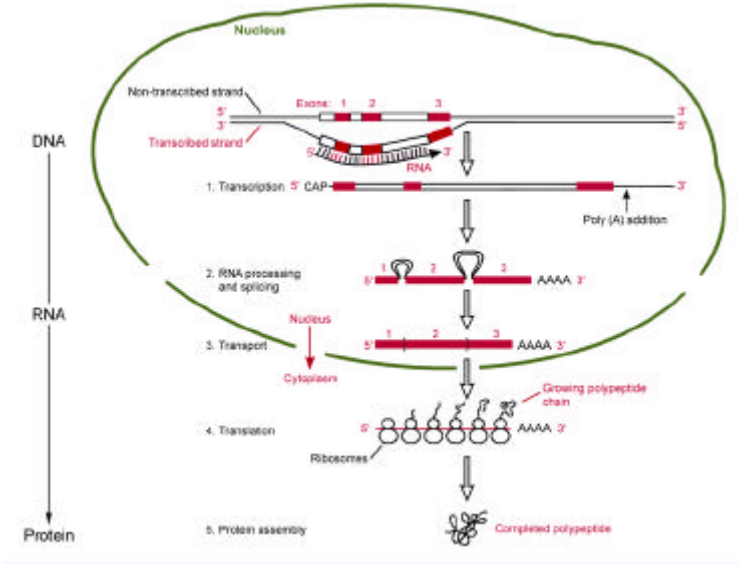


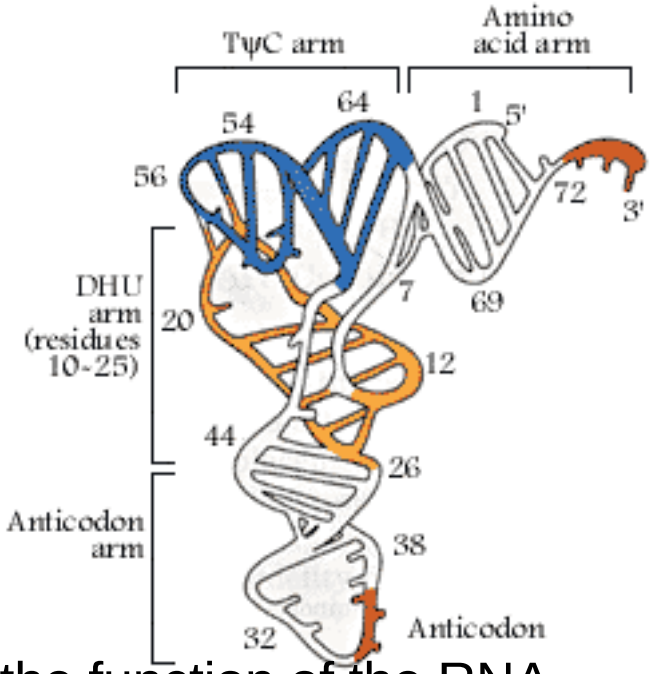
Performance evaluation: scaleup



Application – RNAfold

- RNA plays a major role in expression of genetic code





- The 3-d tertiary structure defines the function of the RNA
- The computation of tertiary structure is computationally expensive, but may be predicted out of the secondary structure

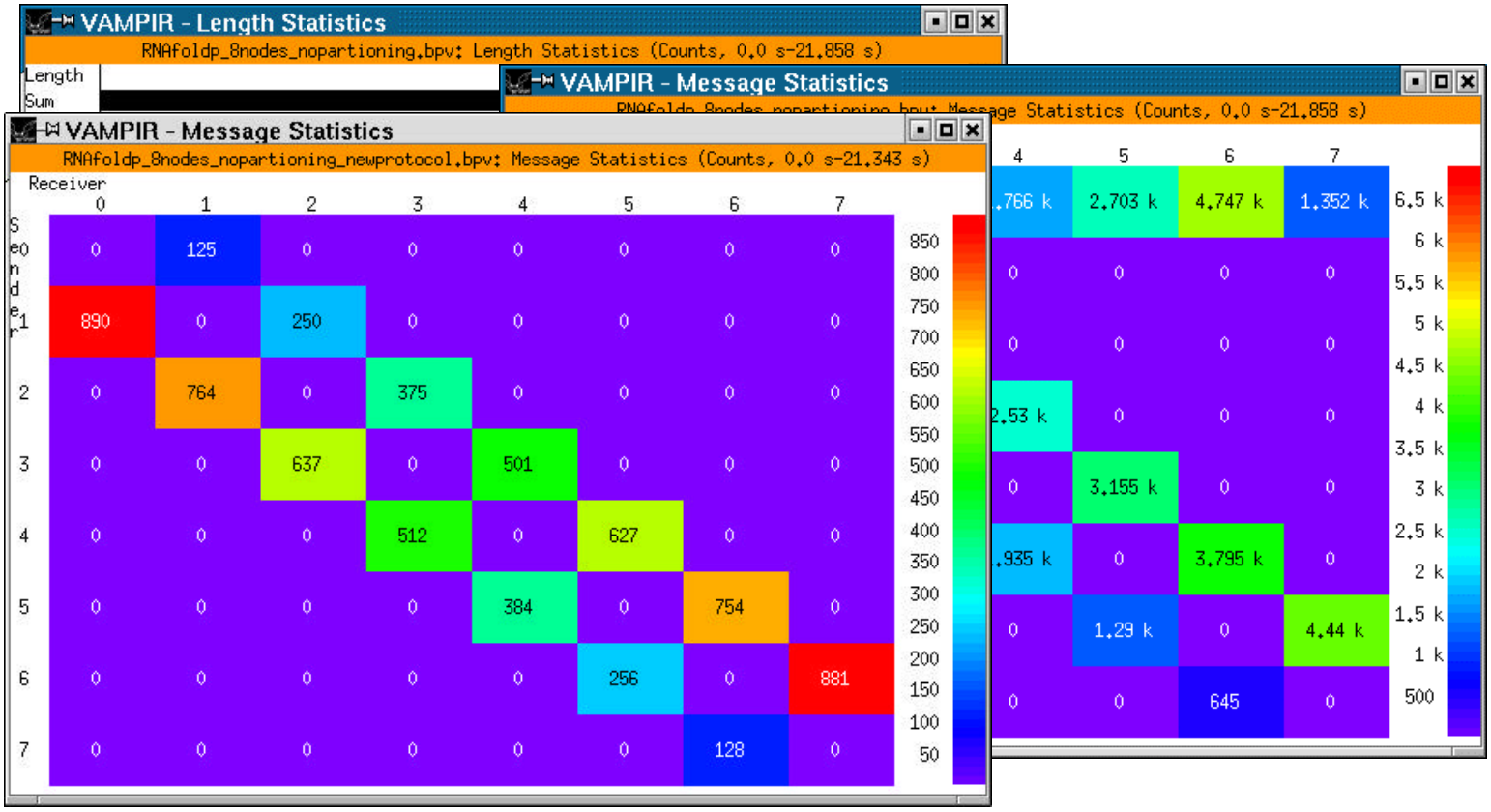
Application – RNAfold

- RNAfold computes the secondary structure of minimal free energy of long RNA sequences.
- Derived out of the Vienna-RNA package of Ivo Hofäcker.
- Tightly coupled MPI-parallelized version.

- Version has been improved to:
  - Include the newest free energy parameters.
  - Better communication pattern:
    - **Improve efficiency by sending bigger packets,**
    - **Get rid of redundant communication,**
    - **Better hide communication.**
  - Integration into Virtual Environment:
    - **Interactive startup of calculation on a Computational Grid**
    - **Visualization**
    - **Collaboration**

Better communication pattern 1/2

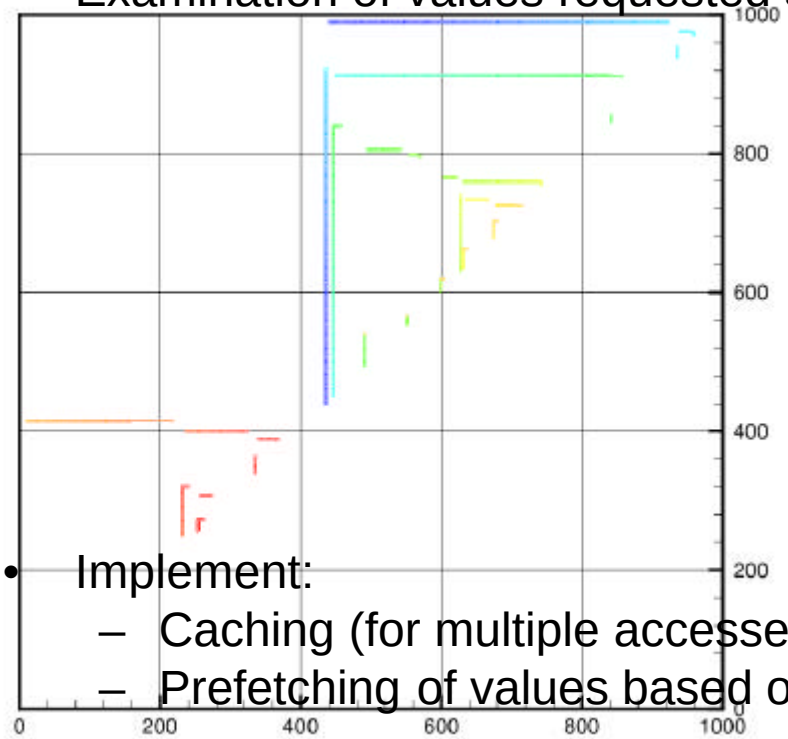
- Consecutive small messages being sent (vis.: Vampir).

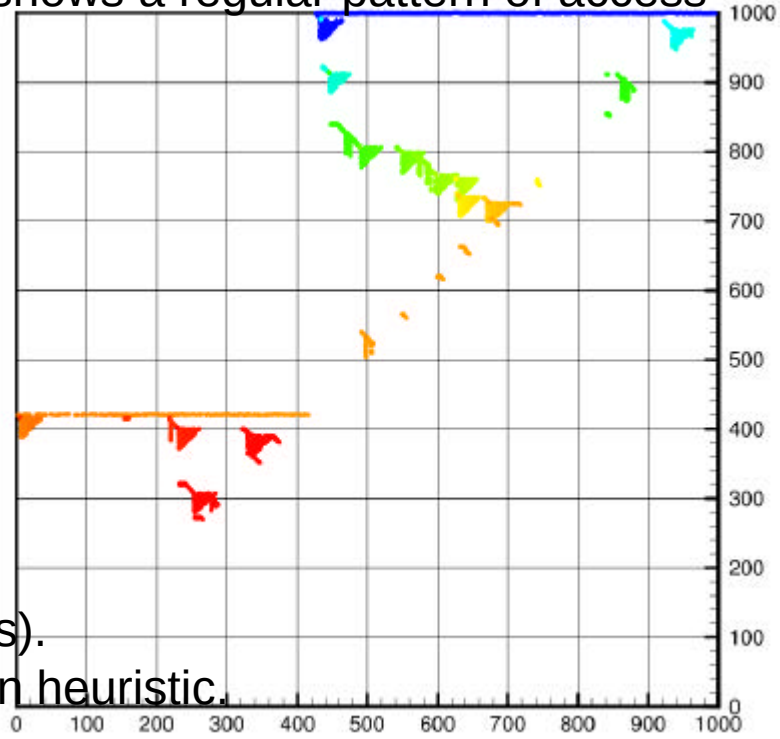


- Six messages integrated into one message.

Better communication pattern 2/2

- Especially a gathering communication step was expensive:
  - Process 0 requests values of matrices stored on other processes: sends 3 Integer request.
  - Reply of one Integer long message.
- Examination of values requested shows a regular pattern of access





- Implement:
  - Caching (for multiple accesses).
  - Prefetching of values based on heuristic.

Summary: Grid Applications – a layered approach

Application	• Latency hiding • Topology aware algorithms • Caching, prefetching • Coalescing of messages
Middleware: -PACX-MPI -MARMOT	• Optimized collective operations • Tools to support applications • Sufficiently high abstraction level
Network	• Support for different protocols • Multiple network connections • Multi-threading

Questions and Answers - General

1. **What grid-related problems did you run into? How did you solve them?**
   Portability problems, firewalls, routing, QoS
2. **Are you using Globus directly? If yes, why did you choose to use the globus toolkit?**
   No Globus, but Unicore support. But we are working on Globus support.
3. **Does your app/tool run in a heterogeneous environment? Did you run into problems due to the heterogeneity? Could grid standardization help with those problems? How?**
   PACX-MPI:Yes. Marmot: Currently not. Problems exist. Difficult to see how GGF can help.

Questions and Answers - Grid-Enabling an Application

1. **How does your application *use* the grid? (What grid features does it use that improves the app?)**
   Larger capacity, more flexibility (coupled multi-physics).
2. **Did you use any useful tools in grid-enabling your app? What do they do?**
   PACX-MPI, Dimemas (see next talk), Vampir, MpCCI
3. **What aspects of the grid-enabling process could be simplified by a tool? What would the tool need to do?**
   Portability problems. Performance prediction.
4. **What standards (if any) would help the grid-enabling process?**
   Tools and libraries providing a sufficient abstraction level.

Q & A - Tools for Grid-Enabling Applications

1. **What problem are you solving for the user? How do you make Grid-enabling the user's application easier? How do you help him?**
   Focus on Compute intensive applications. Providing a Grid enabled MPI library for a smooth migration. MARMOT to attack portability problems.
2. **How difficult is it to use your tool? Does the user need to read a lot of stuff before being able to use it or is the tool intuitive to use?**
   It should be intuitive. But performance issues can be tricky.
3. **What was your most challenging issue/problem you had to solve as part of creating your tool? How did you solve it?**
   Portability of applications. Network performance problems.
4. **What would make it easier for you to create tools for the Grid? Can the UPDT RG help you achieve that? How? (creating standards, etc.)**
   Standard ways for start-up: resource discovery.
   Co-scheduling.