Hybrid Parallelization: Performance from SMP Building Blocks

Christian Terboven
terboven@rz.rwth-aachen.de

Center for Computing and Communication
RWTH Aachen University, Germany

September 29th, Leogang, Austria

Agenda

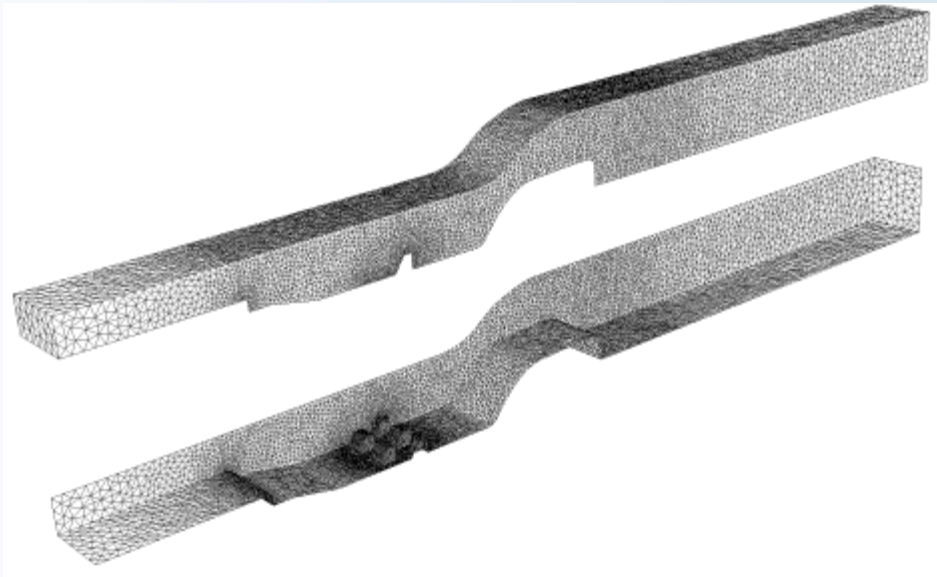
- Hybrid Applications
  - XNS: Hybrid Parallelization improves efficiency
  - FLOWer: Using OpenMP for Load Balancing
- OpenMP Activities
  - NestedCP: Nested OpenMP for Load Balancing
  - Items currently in discussion (for OpenMP 3.1 / 4.0)
- HPC off the beaten track
  - GraS: Grassland Succession (HPC on Windows with Delphi)
- Conclusion and Outlook

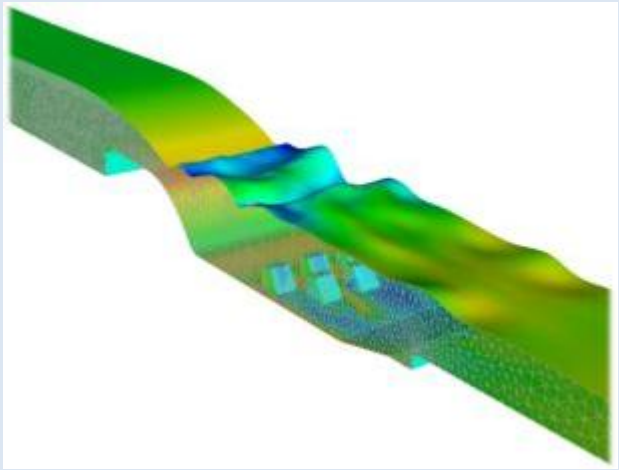
Agenda

- Hybrid Applications
  - XNS: Hybrid Parallelization improves efficiency
  - FLOWer: Using OpenMP for Load Balancing
- OpenMP Activities
  - NestedCP: Nested OpenMP for Load Balancing
  - Items currently in discussion (for OpenMP 3.1 / 4.0)
- HPC off the beaten track
  - GraS: Grassland Succession (HPC on Windows with Delphi)
- Conclusion and Outlook

XNS: Simulation Free-Surface Flows

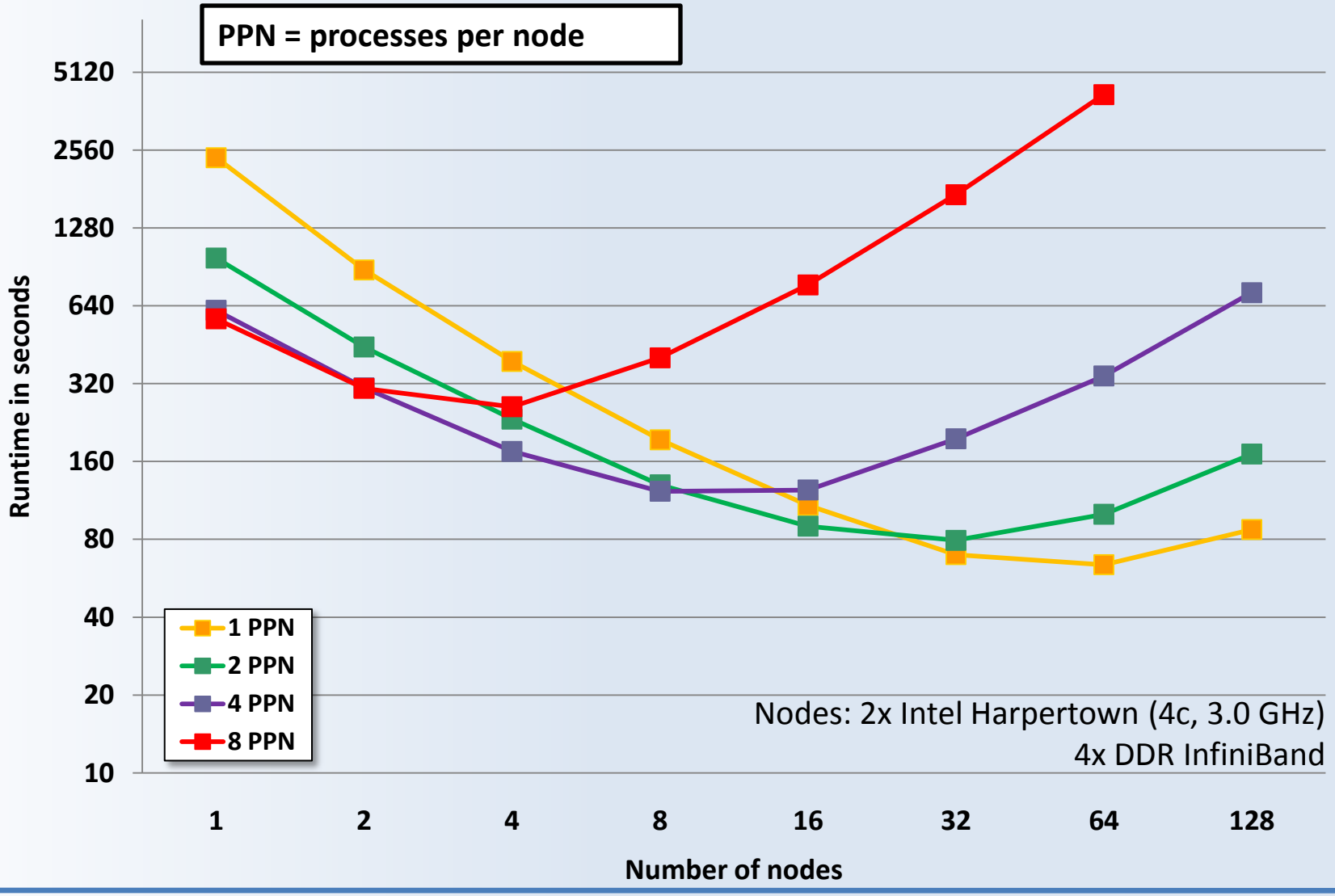
- XNS: Developed by the Chair for Computational Analysis of Technical Systems (CATS) at RWTH Aachen University
  - Finite Element CFD solver written in Fortran and C (48k loc)
  - viscuous, incompressible Navier-Stokes
- Data set: Olmsted dam of the Ohio river
  - 418,249 tetrahedral elements



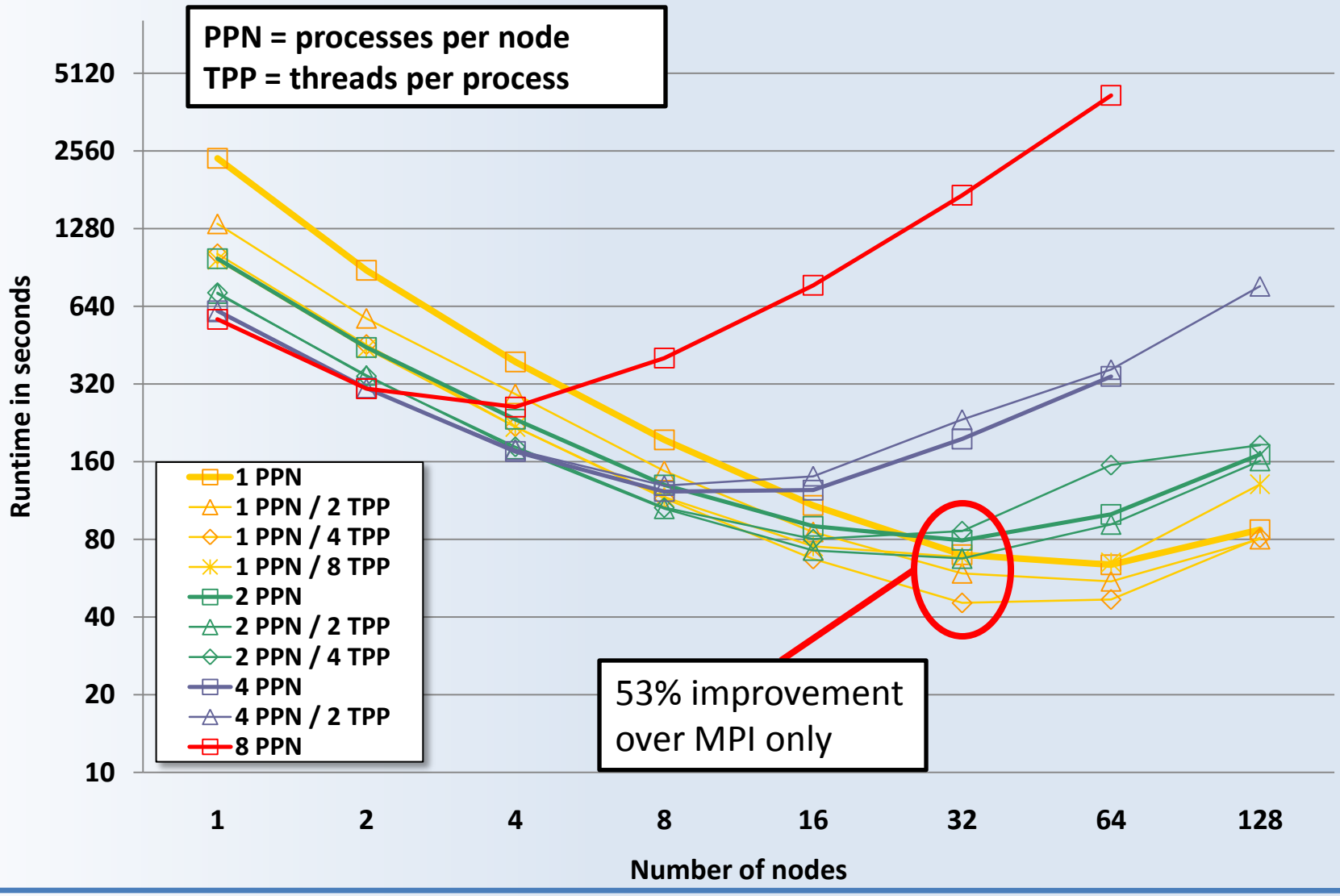


Marek Behr, CATS,
RWTH Aachen University

Scalability of XNS: MPI only

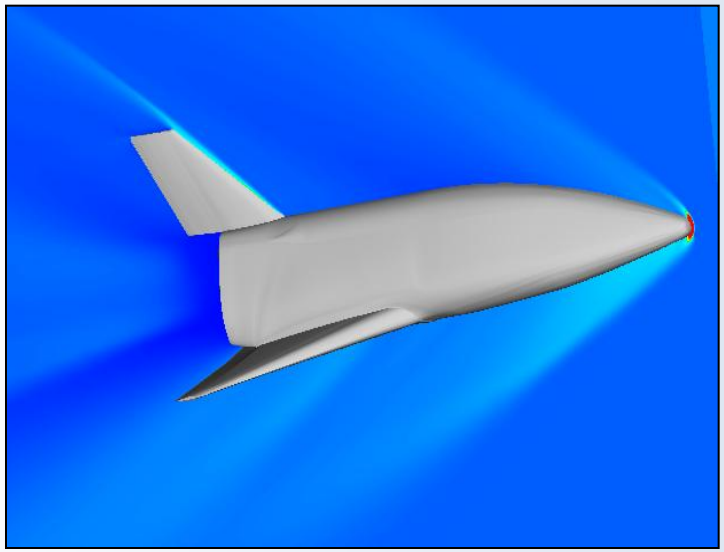


Scalability of XNS: Hybrid (= MPI + OpenMP)



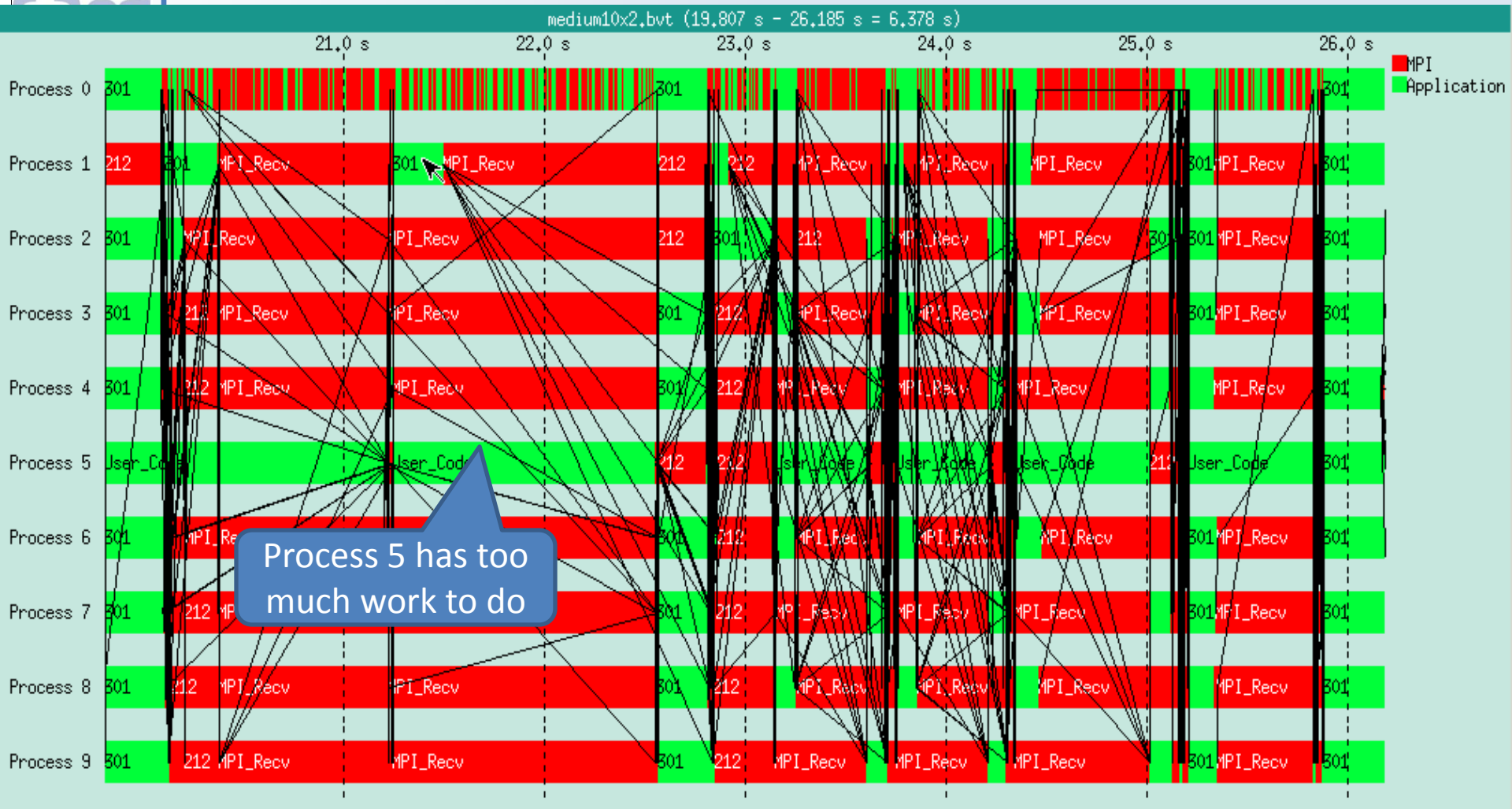
FLOWer: A Navier-Stokes solver

- FLOWer: Navier-Stockes solver, German Aerospace Center
- PHOENIX: a small scale prototype of the space launch vehicle HOPPER (take off horizontally, place cargo in orbit, glide back to earth)
  - MPI + OpenMP / autoparallelization → hybrid parallel program
  - DTB library used to automatically adjust number of threads to improve load balance of MPI

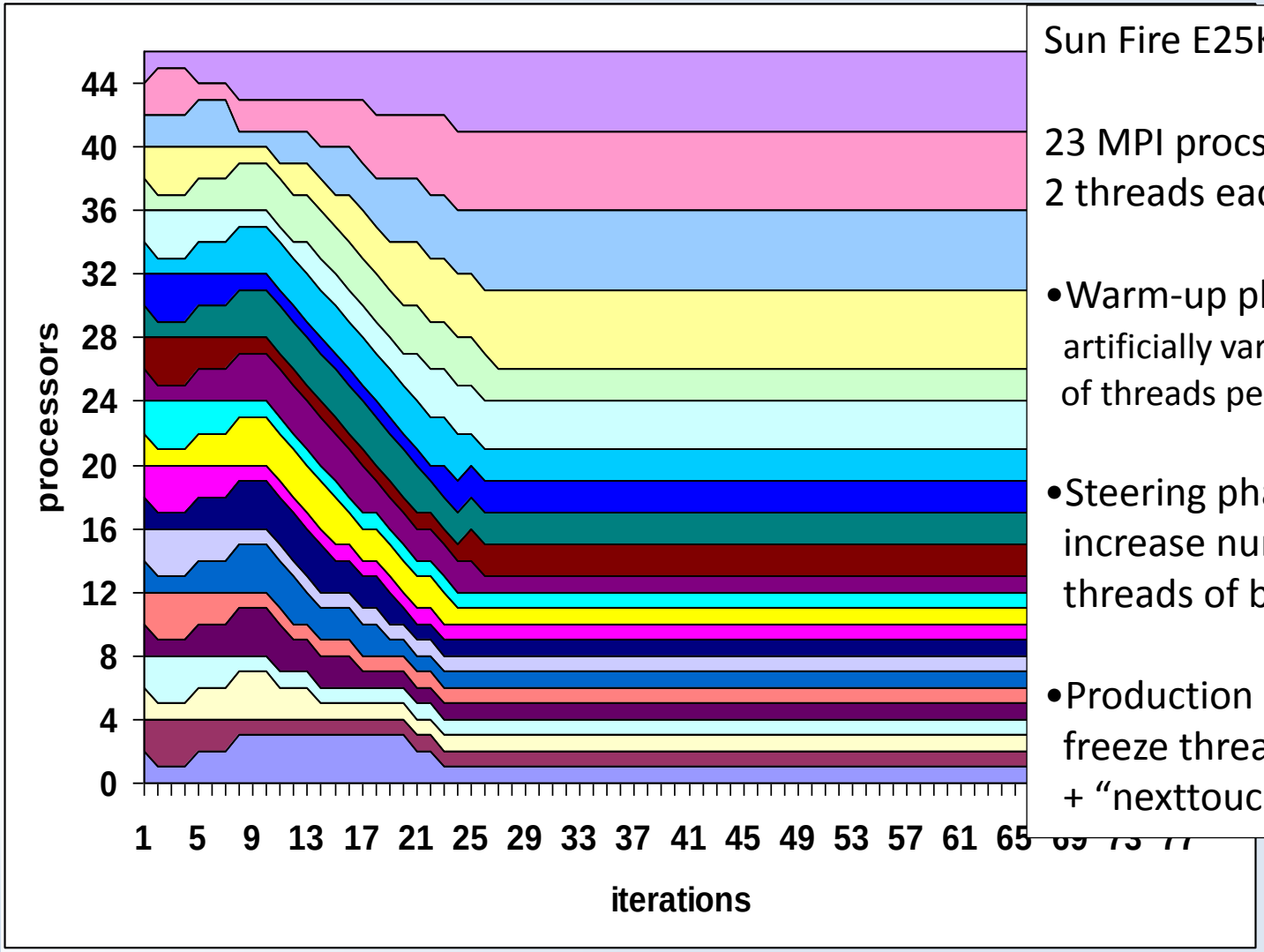


Birgit Reinartz and Michael Hesse, Laboratory of Mechanics, RWTH Aachen University

FLOWer: MPI parallelization is not balanced



FLOWer: Dynamic Thread Balancing



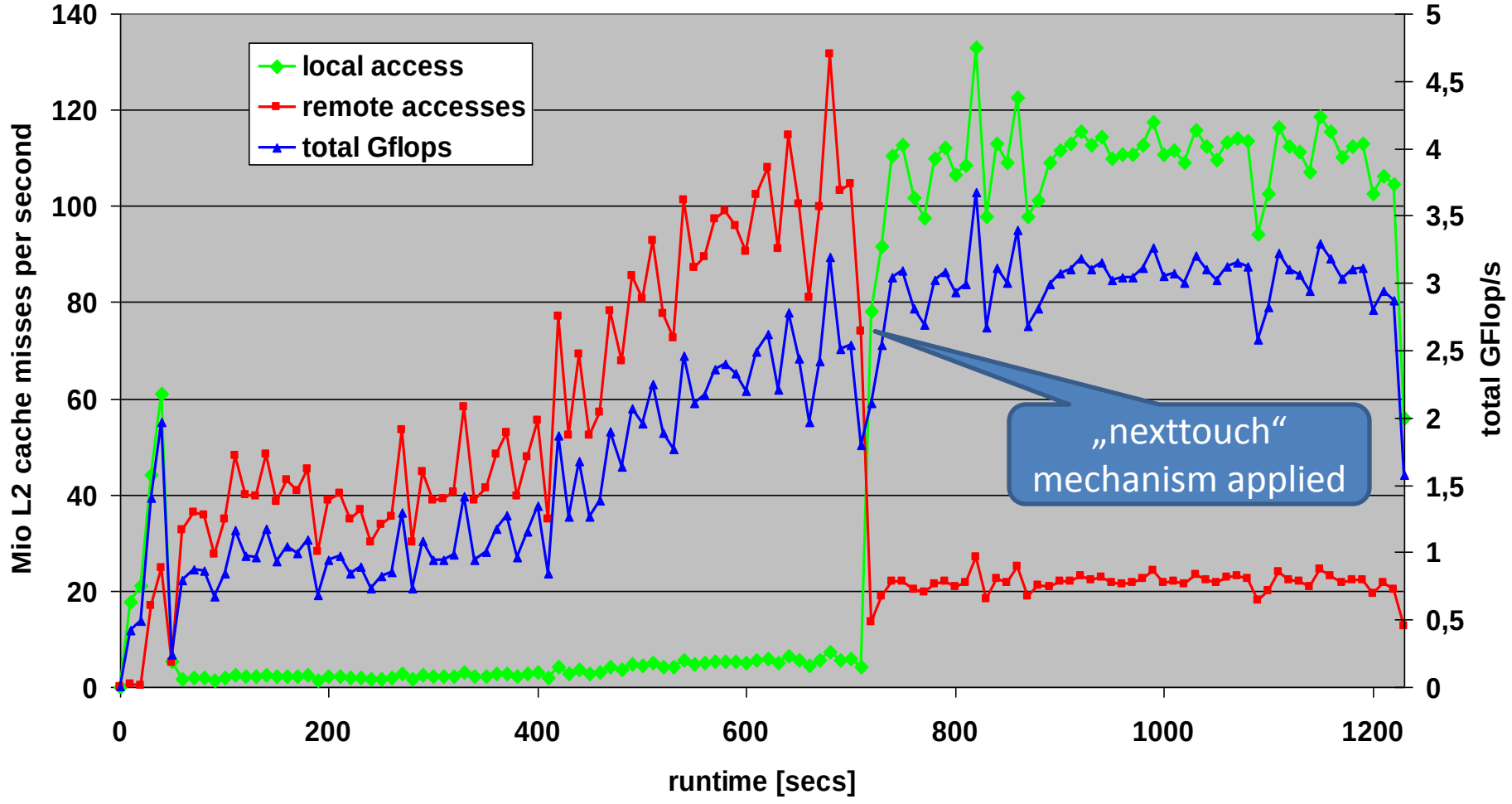
Sun Fire E25K

23 MPI procs start with 2 threads each

- Warm-up phase (1-12)
  artificially vary number of threads per process
- Steering phase (13-30)
  increase number of threads of busy procs
- Production phase (31-)
  freeze thread numbers + “nexttouch”

FLOWer: Data Migration after Thread Balancing

Sun Fire 25K, ~ 65 Mflop/s per thread = 3% of peak performance
(high MPI communication overhead)

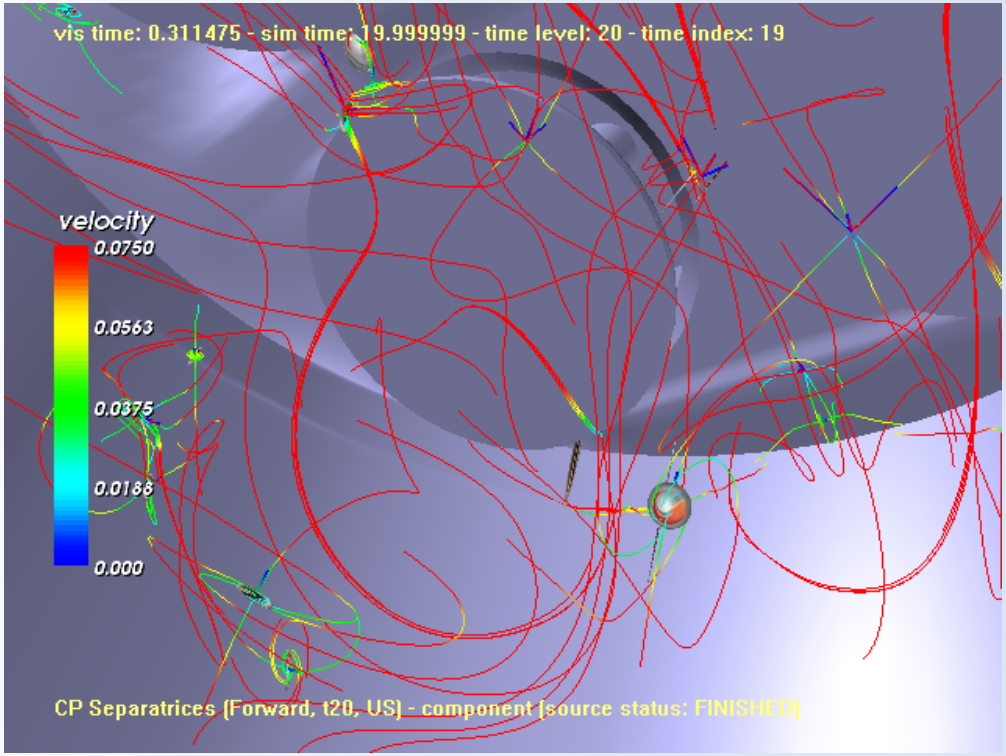


Agenda

- Hybrid Applications
  - XNS: Hybrid Parallelization improves efficiency
  - FLOWer: Using OpenMP for Load Balancing
- OpenMP Activities
  - NestedCP: Nested OpenMP for Load Balancing
  - Items currently in discussion (for OpenMP 3.1 / 4.0)
- HPC off the beaten track
  - GraS: Grassland Succession (HPC on Windows with Delphi)
- Conclusion and Outlook

NestedCP: Parallel Critical Point Extraction

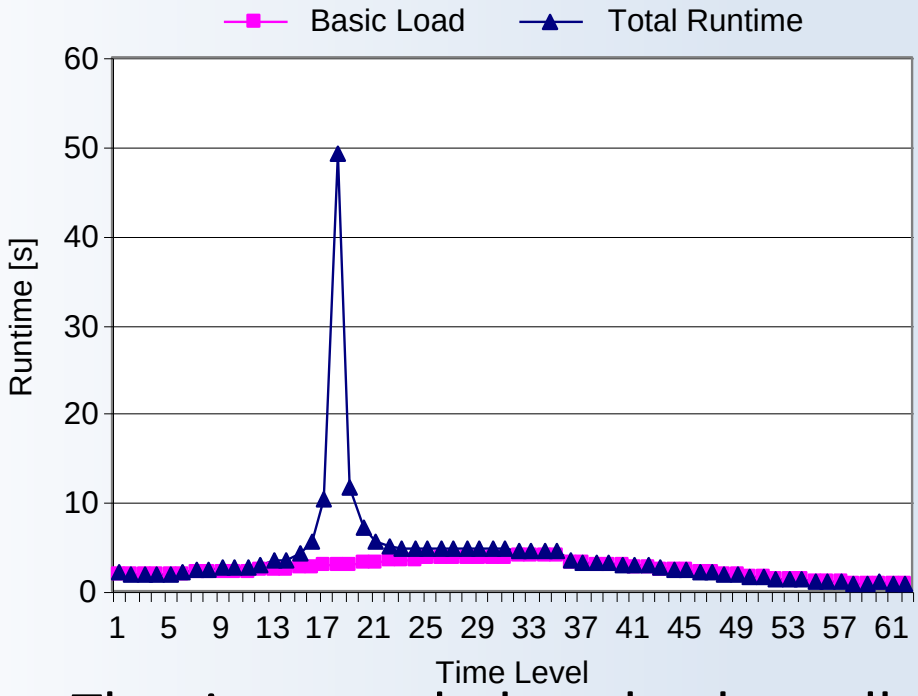
- VR in Aachen: Analysis of large-scale flow simulations
  - Feature extraction from raw data
  - Interactive analysis in virtual environment (e.g. a cave)
- Critical Point: Point in the vector field with zero velocity

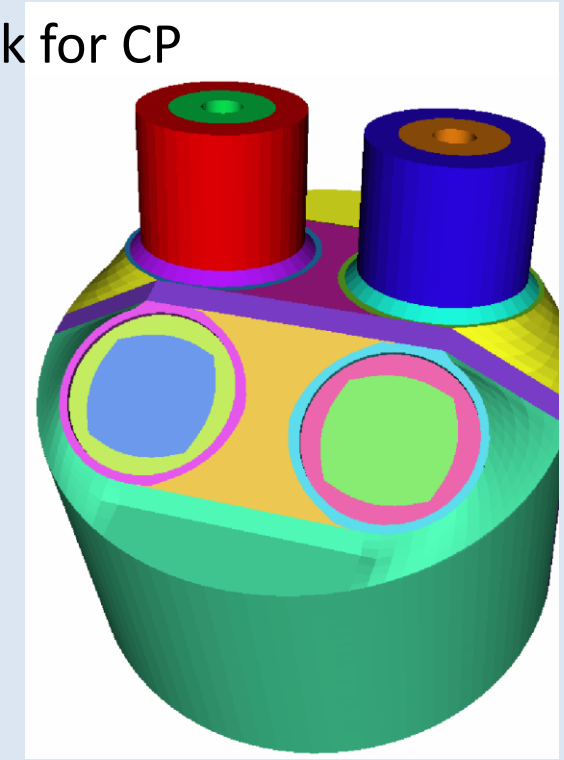


Andreas Gerndt, Virtual Reality Center, RWTH Aachen

NestedCP: Addressing Load Imbalance

- Algorithmic sketch of Critical Point extraction:
  - Loop over the time steps of unsteady datasets
  - Loop over the blocks of multi-block datasets
  - Loop checking the cells withing the block for CP





- The time needed to check a cell may vary considerably!

NestedCP: Addressing Load Imbalance

- Solution in OpenMP is rather simple:

```
#pragma omp parallel for num_threads(nTimeThreads) \
                         schedule(dynamic,1)
for (cutT = 1; curT <= maxT; ++curT)
{
#pragma omp parallel for num_threads(nBlockThreads) \
                         schedule(dynamic,1)
   for (curB = 1; curB <= maxB; ++curB)
   {
#pragma omp parallel for num_threads(nCellThreads) \
                         schedule(guided)
      for (curC = 1; curC <= maxC; ++curC)
      {
         findCriticalPoints(curT, curB, curC);
} } }
```

You may also think of this as Tasking-style parallelization.

Future development of OpenMP (1/3)

- Revised committee rules:
  - New committee chair: Bronis R. de Supinski (LLNL)
  - Three face-2-face meetings per year + weekly concall(s)
  - Attendance requirements for voting rights
- 11 major topics (by vote), 5 subcommittees:
  - Development of an OpenMP Error Model
    - Tim Mattson (Intel) and Michael Wong (IBM)
    - Status: Trend towards callback model (with C++0x support)
  - Interoperability and Composability
    - Christian Terboven (RWTH)
    - Status: Focus on interoperability w/ Native Threading packages
  - Incorporating Tools Support into the OpenMP Specification
    - Status: No progress yet

Future development of OpenMP (2/3)

- 11 major topics, 5 subcommittees (cont'd):
  - Associating Computation or Memory across Workshares
    - Locality and Affinity: Dieter an Mey (RWTH)
    - Status: Focus on standardization of existing vendor extensions
    - Thread Team Control: Barbara Chapman (UH)
    - Status: No consensus yet
  - Accelerators, GPUs and More
    - Status: No clear direction, several proposals on the table
  - Refinements to the OpenMP Tasking Model
    - Federico Massaioli (Caspur) and Grant Haab (Intel) and Alex Duran (BSC)
    - Status: Proposal for Task Dependencies on the table, nothing on Task Reductions

Future development of OpenMP (3/3)

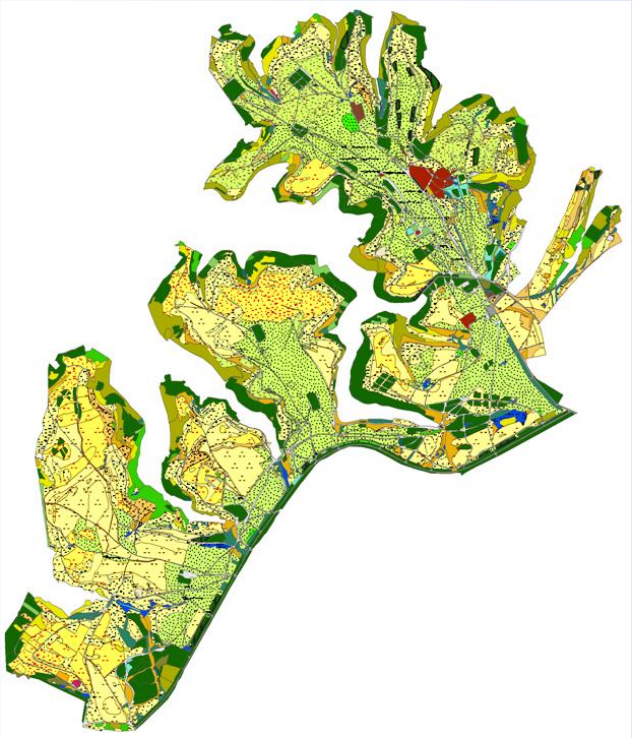
- 11 major topics, 5 subcommittees (cont'd):
  - Extending OpenMP to Fortran 2003 and C++0x
    - C++0x: Michael Wong (IBM) and Christian Terboven (RWTH)
    - Status: No work on Fortran 2003 yet, work on C++0x became obsolete without Concepts
  - Clarifications to the Existing Specification (3.1 Targets)
    - Status: Proposal for user-defined reductions on the table (discussion: Duran, Massaioli, Klemm (Intel), Terboven)
    - Status: Proposal for Atomic Extensions on the table (Grant Haab)
    - Status: Several clarification for the current spec done already
  - Miscellaneous Extensions (4.0 Targets)
  - Additional Task / Thread Synchronization Mechanisms
    - Status: No progress yet
- What do you think is important?

Agenda

- Hybrid Applications
  - XNS: Hybrid Parallelization improves efficiency
  - FLOWer: Using OpenMP for Load Balancing
- OpenMP Activities
  - NestedCP: Nested OpenMP for Load Balancing
  - Items currently in discussion (for OpenMP 3.1 / 4.0)
- HPC off the beaten track
  - GraS: Grassland Succession (HPC on Windows with Delphi)
- Conclusion and Outlook

GraS: Simulation of Grassland Succession

- Institute for Environmental Research (Bio5) and Research Institute for Ecosystem Analysis and Assessment (GAIAC)
  - Simulation of grassland development under var. land use forms
    → decision support for grassland mgmt. on the landscape scale

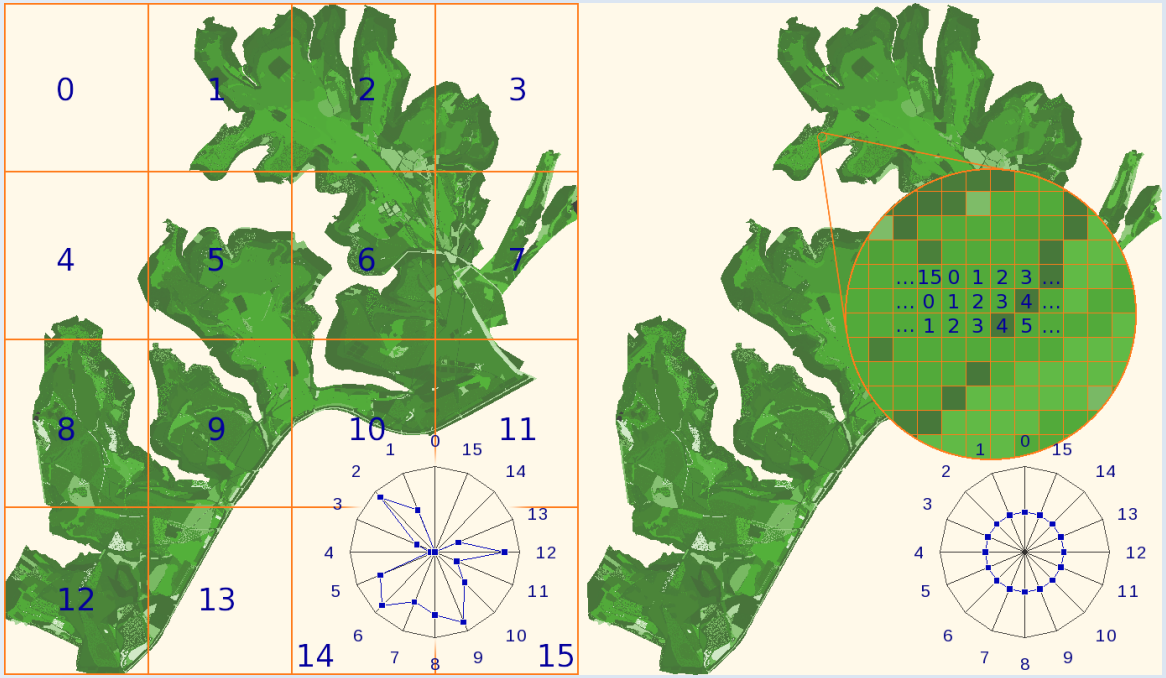


Dataset: Eifel National Park established in 2004, includes open grasslands of former military training site Vogelsang

Simulation tool developed at Bio5 / GAIAC, developed in Delphi on and for Windows.

GraS: Delphi for HPC

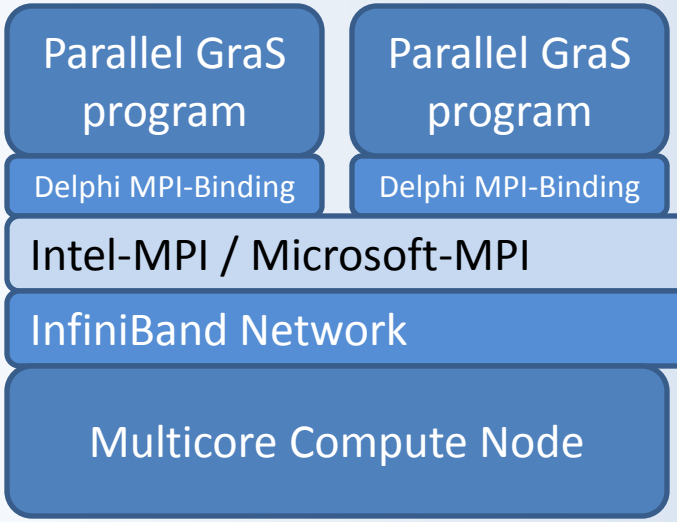
- The simulation code was not suitable for the Eifel dataset:
  - Delphi currently supports only 32bit (not enough memory)
    → Dataset was manually partitioned to conduct simulation
    → No means of interaction between the partitions
- Work done so far: Creation of an MPI application including seed exchange for the whole Eifel dataset.
- Seed-funded by JARA-SIM.



Left: bad Right: good load balancing.

GraS: An HPC-Environment for Delphi

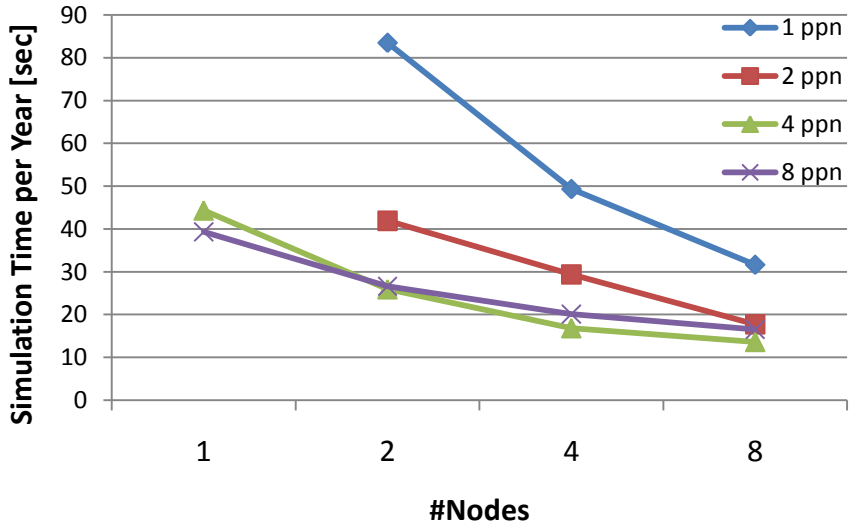
- Delphi comes with virtually no support for parallel programming, so we created / modified a suitable environment:



MPI Performance Analysis:

- I-MPI: Intel Trace Analyzer and Collector
- MS-MPI (w/ Tracing): Vampir on Windows

- No support for Shared-Memory parallelization, we are thinking of re-writing some components in C++ ...



Agenda

- Hybrid Applications
  - XNS: Hybrid Parallelization improves efficiency
  - FLOWer: Using OpenMP for Load Balancing
- OpenMP Activities
  - NestedCP: Nested OpenMP for Load Balancing
  - Items currently in discussion (for OpenMP 3.1 / 4.0)
- HPC off the beaten track
  - GraS: Grassland Succession (HPC on Windows with Delphi)
- Conclusion and Outlook

Conclusion (and Outlook)

- MPI is clearly the dominant parallelization paradigm in HPC, but Hybrid Parallelization can improve efficiency:
  - A domain decomposition may only allow so much parallelism
  - A growing number of cores per node leads to IB bottlenecks
  - Adding more levels of parallelism may be less work in OpenMP
  - 64-bit GPGPUs may add another dimension of parallelism ...
- Lessons learned on big SMPs have to be applied to cc-NUMA x86-based multicore systems today.
- HPC on Windows has become a viable solution, user base at Aachen is growing steadily.

The End

Thank you for

your attention!