Swedish e-Science Research Centre

SESSI – SeRC EXASCALE SIMULATION SOFTWARE INITIATIVE

Berk Hess[1], Stefano Markidis[2], Philipp Schlatter[3]

[1]Theoretical Physics, [2]Computational Science and Technology, [3]Mechanics

KTH Royal Institute of Technology

ExaFLOP, Exascale, Exascience

- **ExaFLOP** = 10^{18} Floating Point Operations with HPL
  - Machine delivering an ExaFLOP is expected in 2024
- **Exascale** machine = usable ExaFLOP machine to make science.
- ExaFLOP ≠ **Exascience** = problems that can be solved on exascale machine and couldn't be solved on Petascale machine (= today machines). When does the strong scaling stop?

The Curse of Exascale

- Predicted to occur in 2018, now people (President Obama) talk about 2024.
- No major technological shift other than deeper memory hierarchy → This makes programming these machines more difficult.
- **Software and Simulation Technology lagging behind hardware development**:
  - No algorithmic disruption
  - No new theory on how to deal with billion processes
  - Still MPI and waiting for + X

Simulation Software Challenges at Exascale

- **Billion-way hierarchical parallelism** for irregular communication (unstructured mesh and multigrid solvers, i.e. CFD) → improving scalability of **communication**.
- **Impact of load imbalance → task-based** approaches.
- Use all the parallelisms available! → **SIMD** and **accelerator** parallelism.
- Can't optimize if you don't measure but:
  - How much invasive are the tools? Overhead!
  - where do you store the trace of million processes?

  → We need low-overhead **tools** to **monitor code performance providing an amount of data we can manage** .

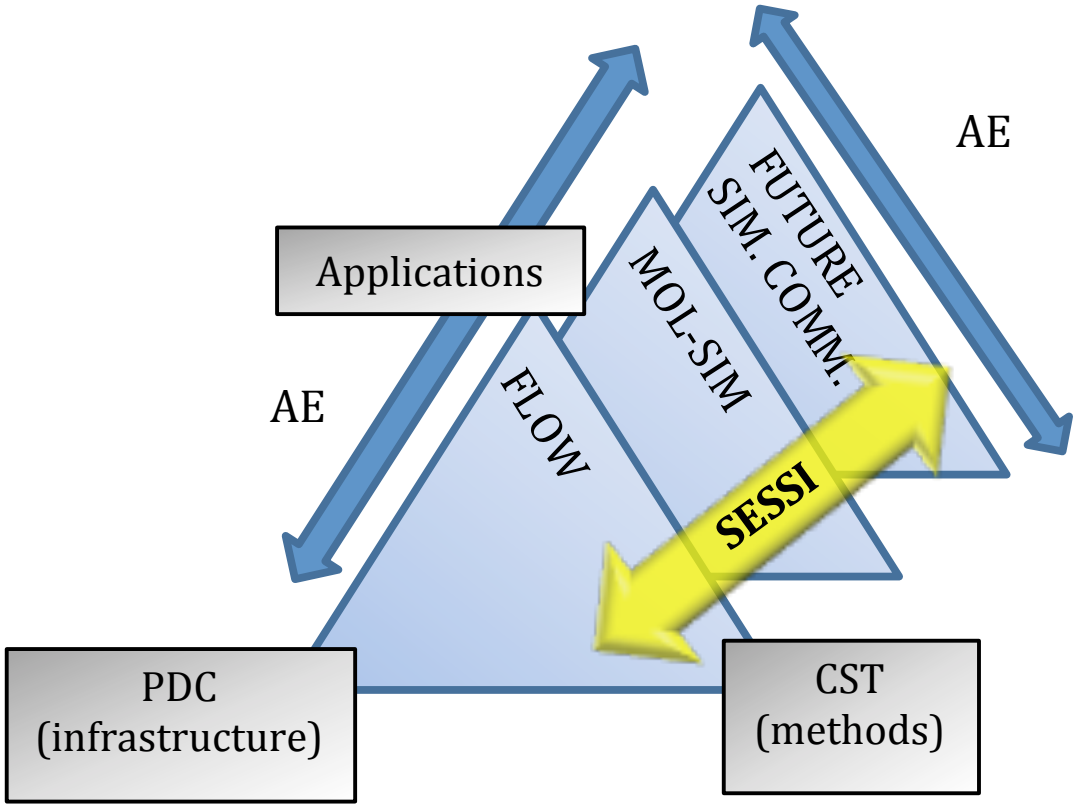
Why SESSI ?

- Software and Simulation Technology **IS** the real problem at exascale.
- To tackle software and simulation challenges at exascale needs different competences from different domains:

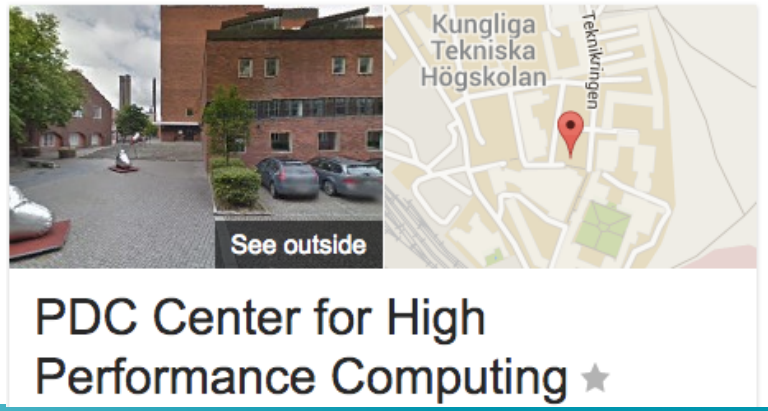
application algorithms → software engineering → parallel computing → low-level optimizations

We need to put together people with different competences and research background to tackle this overwhelming challenge!

SESSI IMPLEMENTATION

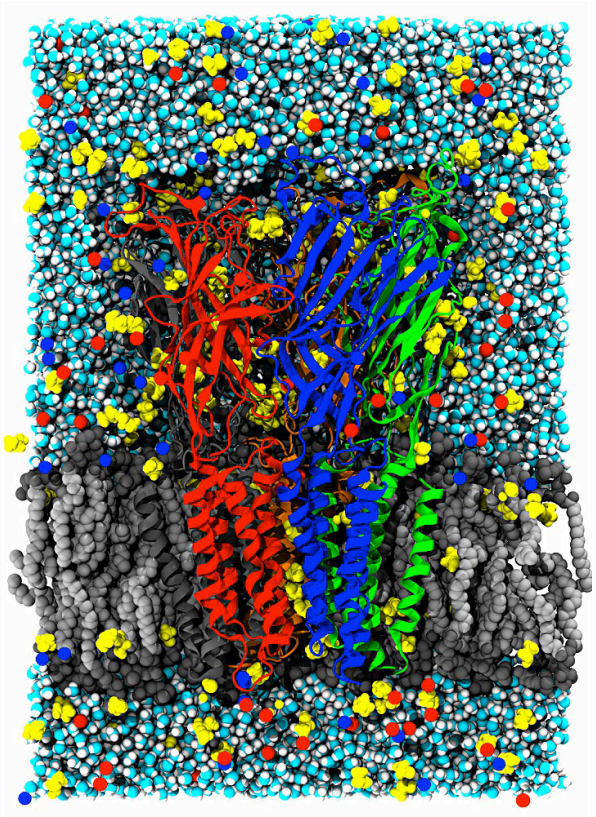


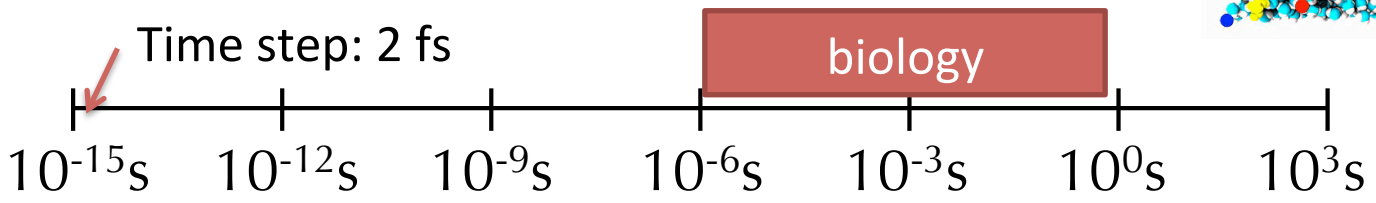
- Common open workspace at PDC to spur interaction
- AE permanently in the workspace
- PhD students and researchers have shared desks for close collaboration and interaction with AE



Why Bio-molecular MD simulations?

- Simulations reveal atomistic detail and dynamics (experiments are limited)
- Typical bio-molecular simulation:
  - One or more bio-molecules solvated in water + ions + e.g. lipid bilayer membrane
  - Fixed size: 100 000 – 200 000 atoms
- Use ensemble simulations to parallelize when possible
- But we still need > µs simulations **strong scaling needed!**





GROMACS

- Atomistic (+course-grained) classical molecular dynamics
- Written in C++, CUDA, OpenCL
- LGPL license
- Thousands of users in bio-molecular, polymer, … fields

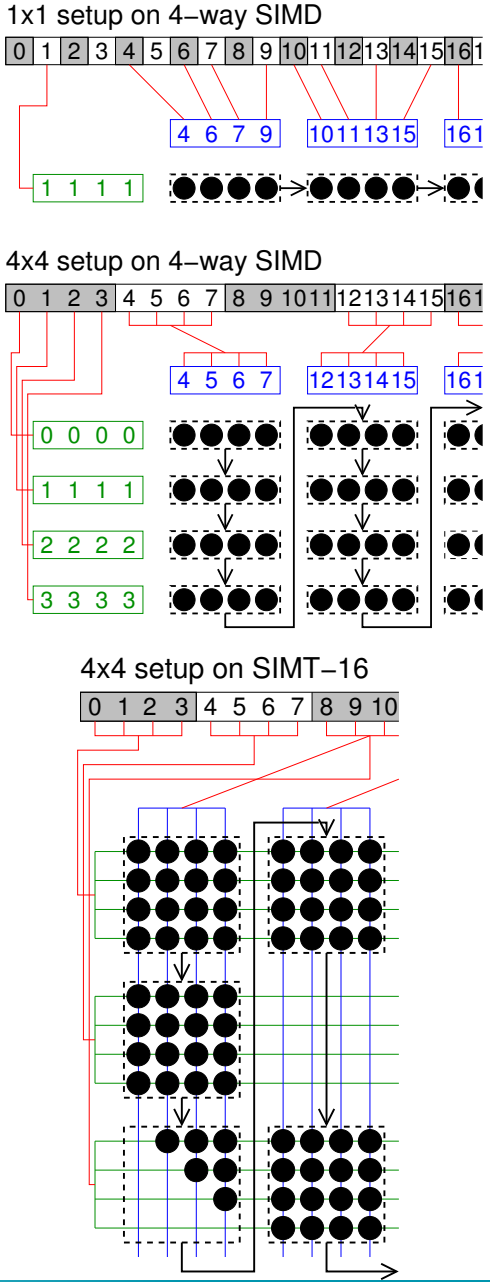
- Very good absolute performance
- Good strong and weak scaling

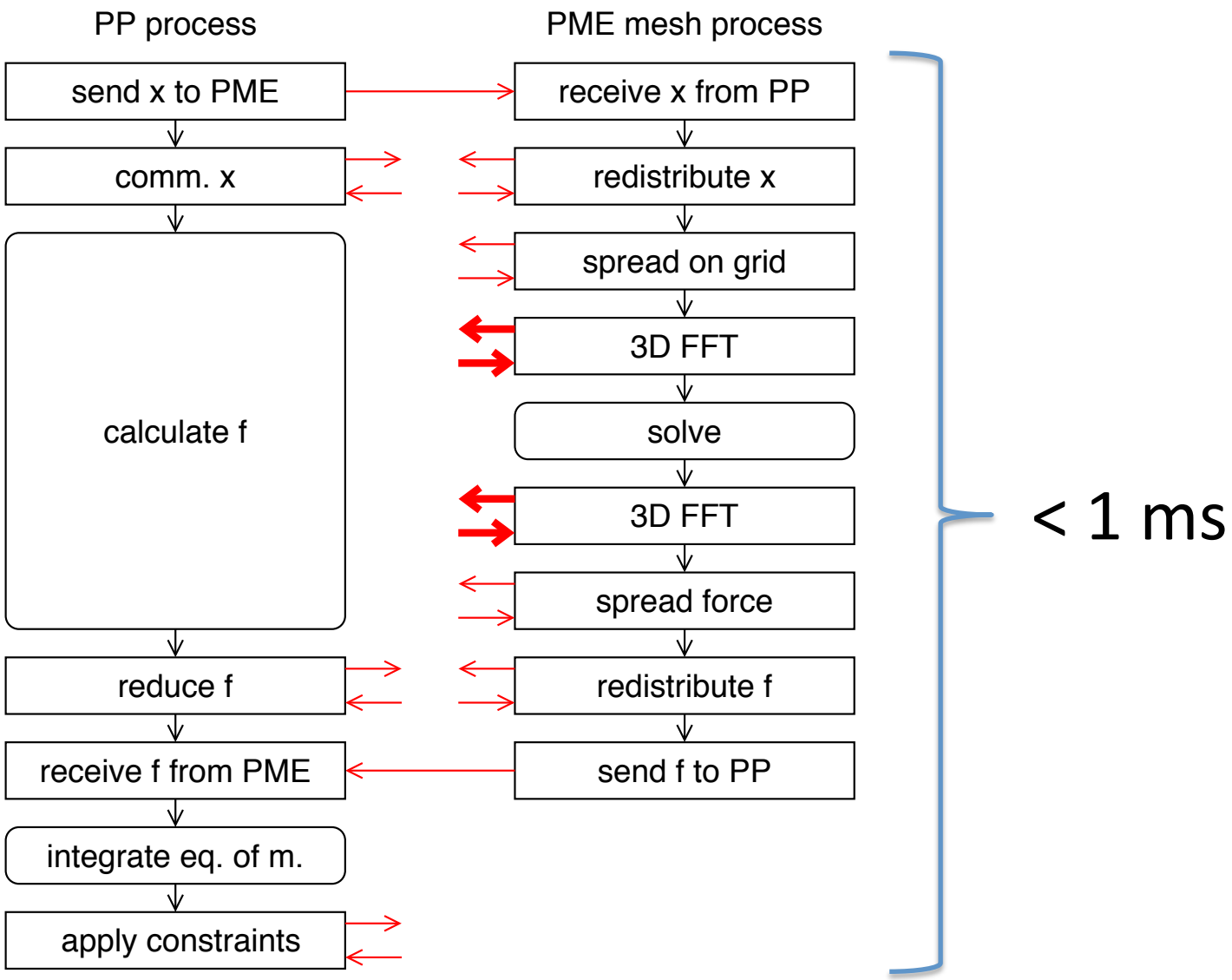
main computational cost

$$m_i \frac{d^2 \mathbf{x}_i}{d\,t^2} = -\nabla_i V(\mathbf{x}) \quad , \qquad i = 1, \ldots, N$$

Force computation

- Coulomb+LJ pair interactions, 70%
  - Uncoupled, but high reduction cost
  - Efficient SIMD & GPU code
- Bonded interactions, 5%
  - Simple SIMD, but inhomogeneous
- Particle-Mesh Ewald electrostatics, 25%
  - Several, coupled tasks
  - 3D FFT, fast, but global communication
  - SIMD, GPU coming

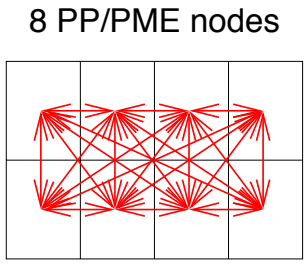


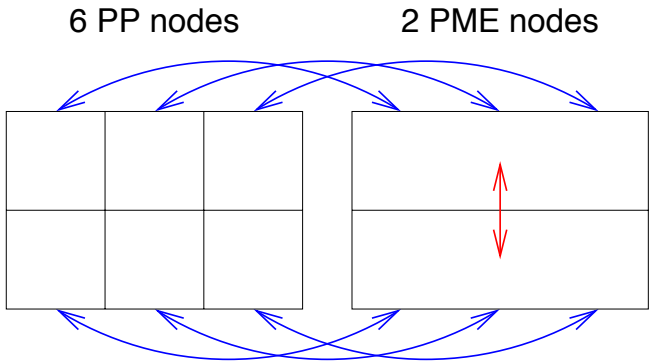

PP process
PME mesh process
send x to PME
receive x from PP
comm. x
redistribute x
spread on grid
3D FFT
calculate f
solve
3D FFT
spread force
reduce f
redistribute f
receive f from PME
send f to PP
integrate eq. of m.
apply constraints
< 1 ms

Parallelization bottleneck

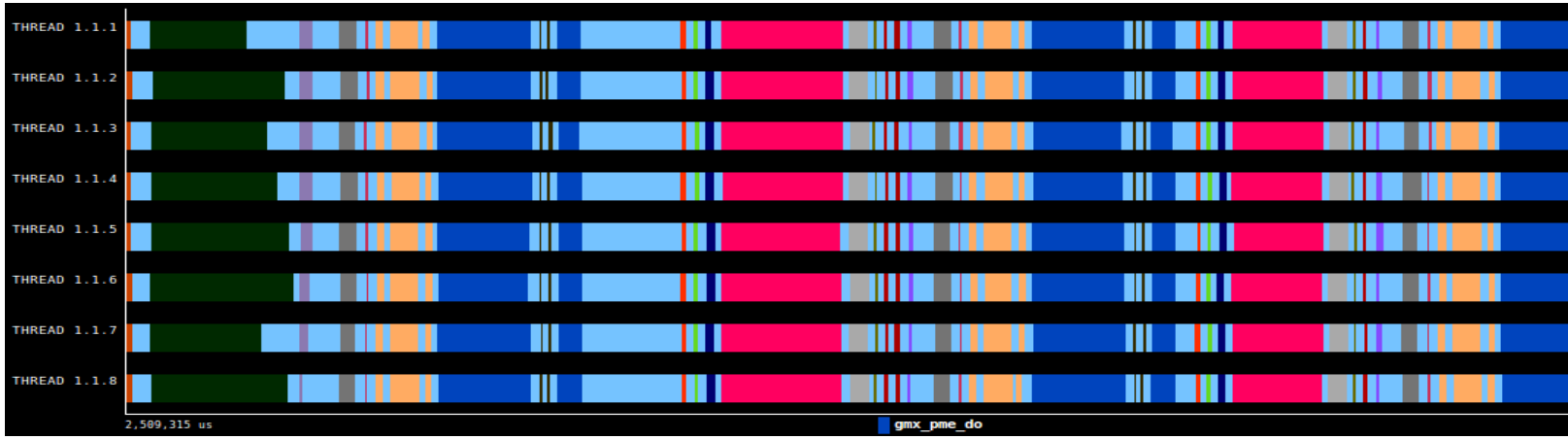
- In MD electrostatics is the parallelization bottleneck; Coulomb 1/r: every particle sees every other particle
  - PME uses many MPI_Alltoall
- Solutions Stefano is looking into:
  - Overlapping calculation & comm.
  - Better communication patterns
- Better scaling solution

GROMACS supports PME MPI task parallelization



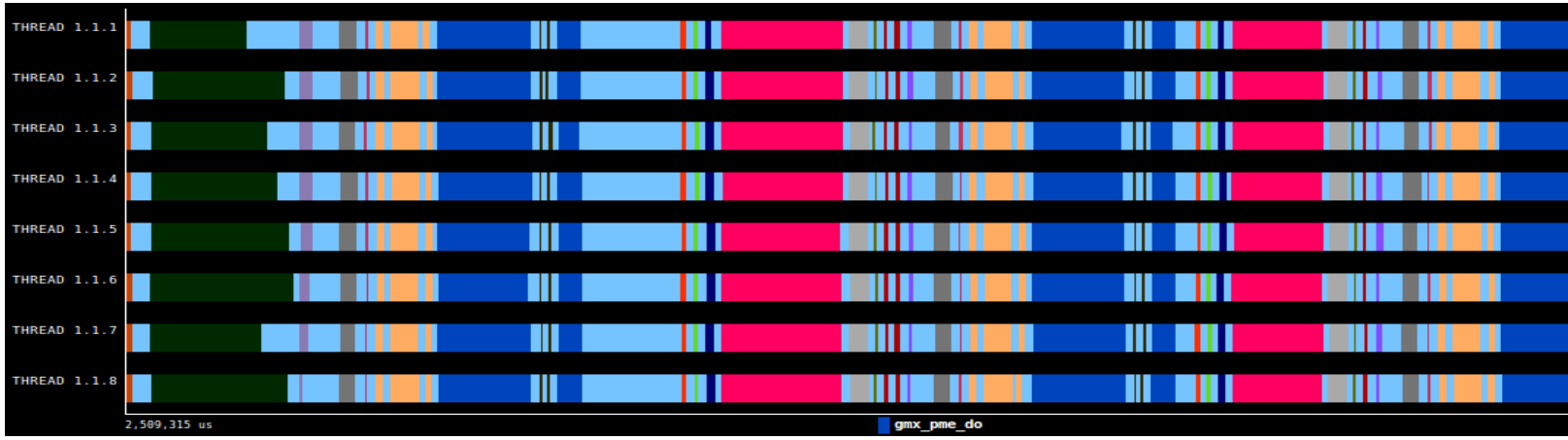


Task-parallelization challenge



- Currently we use OpenMP
  - Parallel for overhead: **2 ms**
  - Many thread-parallel regions
- Solution: run multiple tasks in parallel
  - We need a better tasking framework

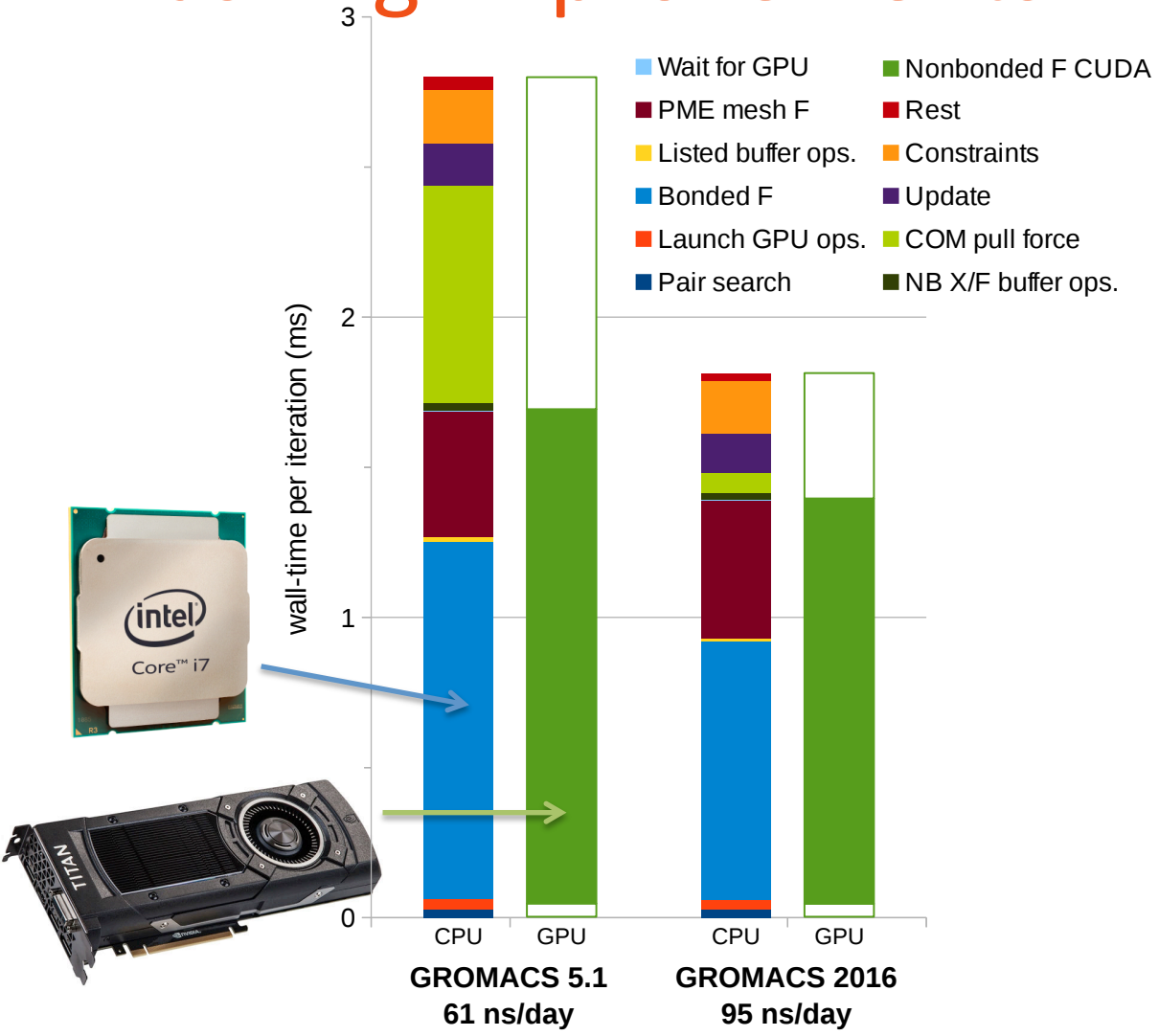
Profiling/tracing challenge

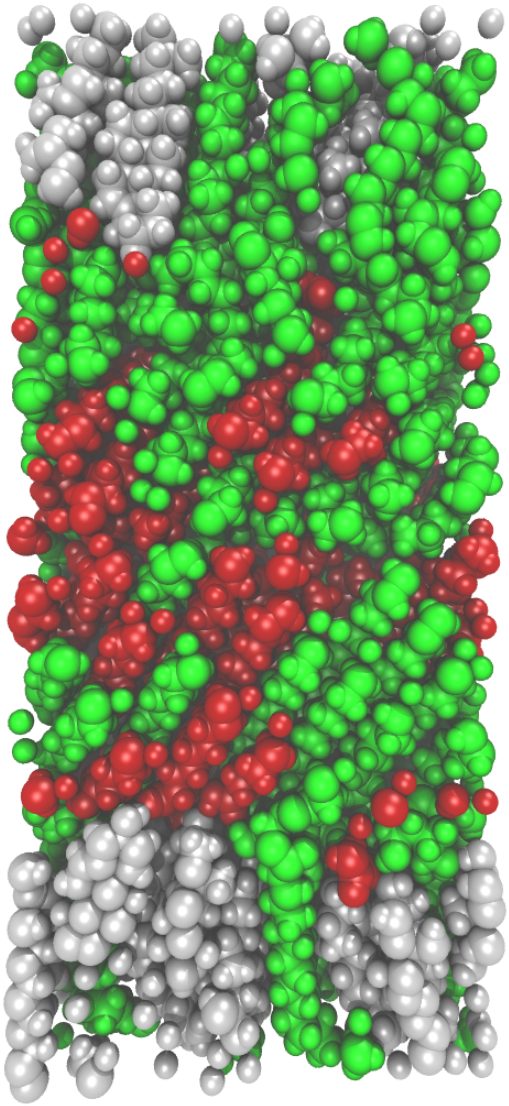


1-step: 500 ms

- Sub millisecond iteration times
- Many tasks & OpenMP regions
  - Overhead of most tools is too high
  - Generates a large amount of data in very short time
- PDC has a new, better profiling/tracing approach

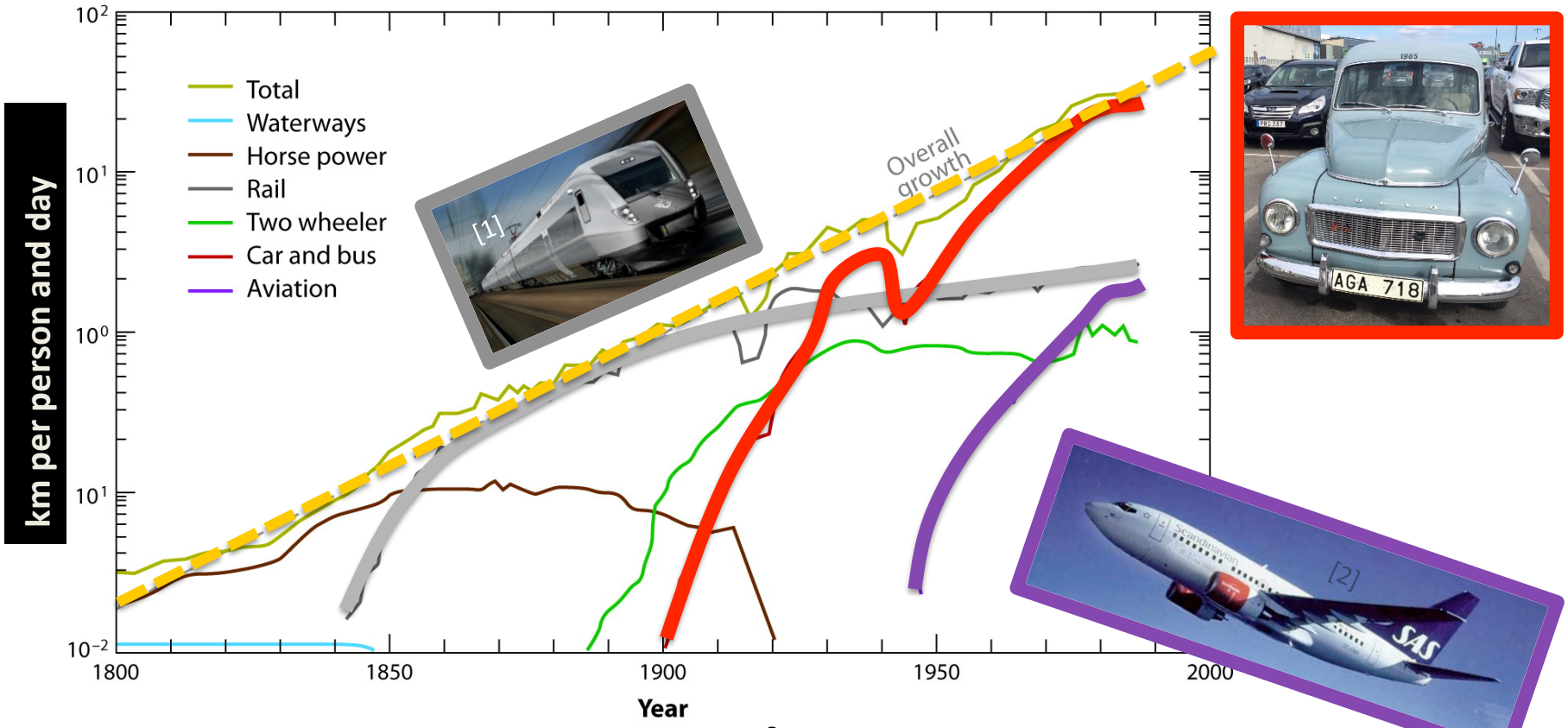
Tasking improvements





Why CFD?

Skin friction/drag reduction is the key for economically and ecologically more efficient transport

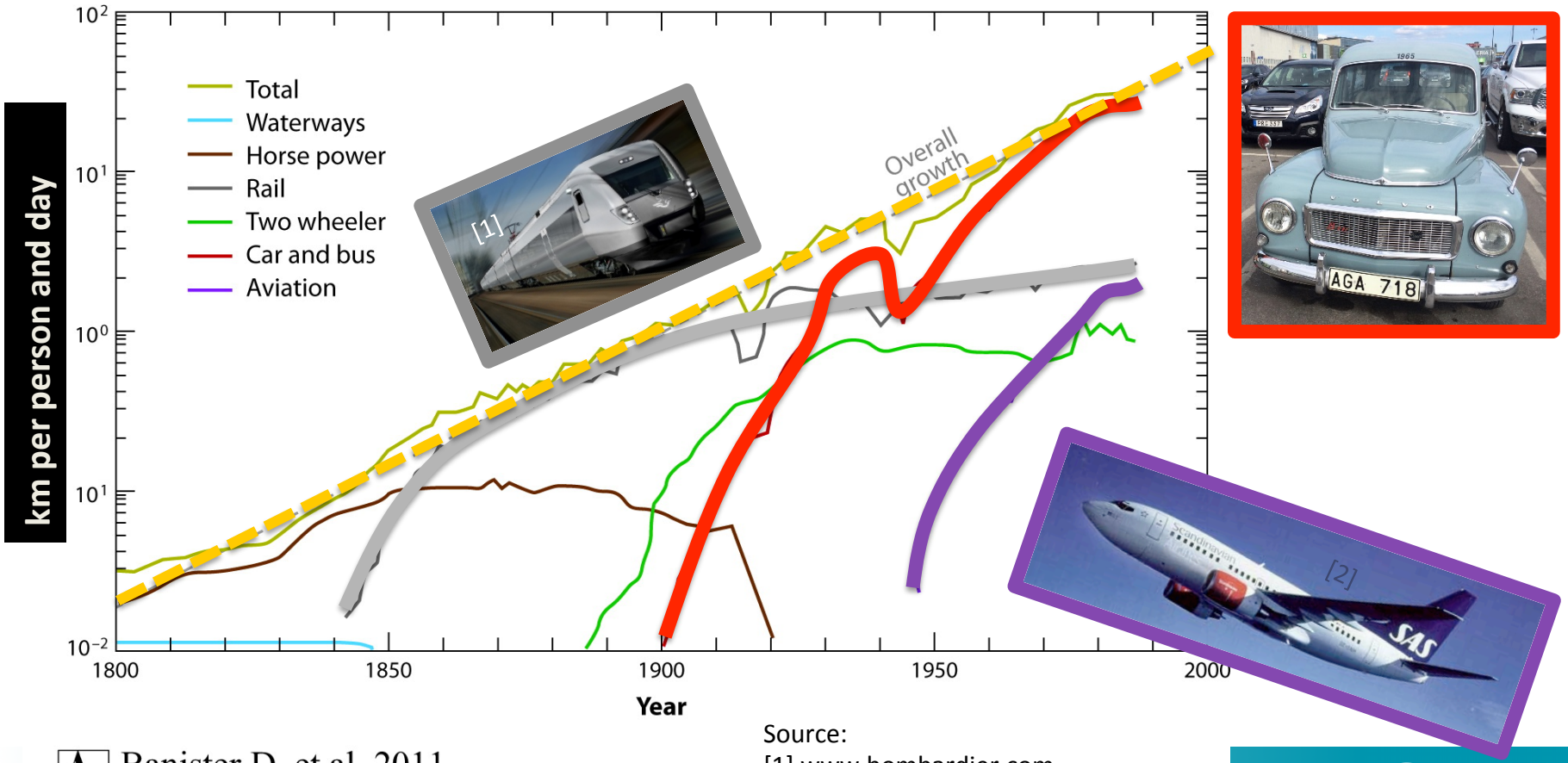


Banister D, et al. 2011.
Annu Rev. Environ. Resour. 36:247–70

Source:
[1] www.bombardier.com
[2] www.flysas.de

Why Computational Fluid Dynamics?

Skin friction/drag reduction is the key for economically and ecologically more efficient transport



Banister D, et al. 2011.
Annu Rev. Environ. Resour. 36:247–70

Source:
[1] www.bombardier.com
[2] www.flysas.de

SeRC
Swedish e-Science Research Centre

Why CFD?

An Airbus 310 cruising at 250 m/s at 10000m…

- Fuselage about 50 m, wing span 44 m, chord 5 m
- **For 1 second simulated flight:** 10^{19} ops / $4 \cdot 10^{-7} = 2.5 \cdot 10^{25}$ ops
- **Teraflops machine (10^{12} Flops):** $8 \cdot 10^{5}$ years
- **To have the result in one week:** $4 \cdot 10^{19}$ flop machine (40 EFlops)

Data from Mira (2013), million core hours

– Engineering/CFD	525	19%
– Subsurface flow & reactive transport	80	3%
– Combustion	100	4%
– Climate	280	10%
– Astrophysics	28	1%
– Supernovae	105	4%
	1118	**40%**

(fraction of Navier-Stokes based simulation on current supercomputer)

APS-DFD Gallery of Fluid Motion 2015

Entry #: V0078

APS Gallery of Fluid Motion 2015

Turbulent flow around a wing profile, a direct numerical simulation

Mohammad Hosseini, Ricardo Vinuesa, Ardeshir Hanifi
Dan Henningson, and Philipp Schlatter

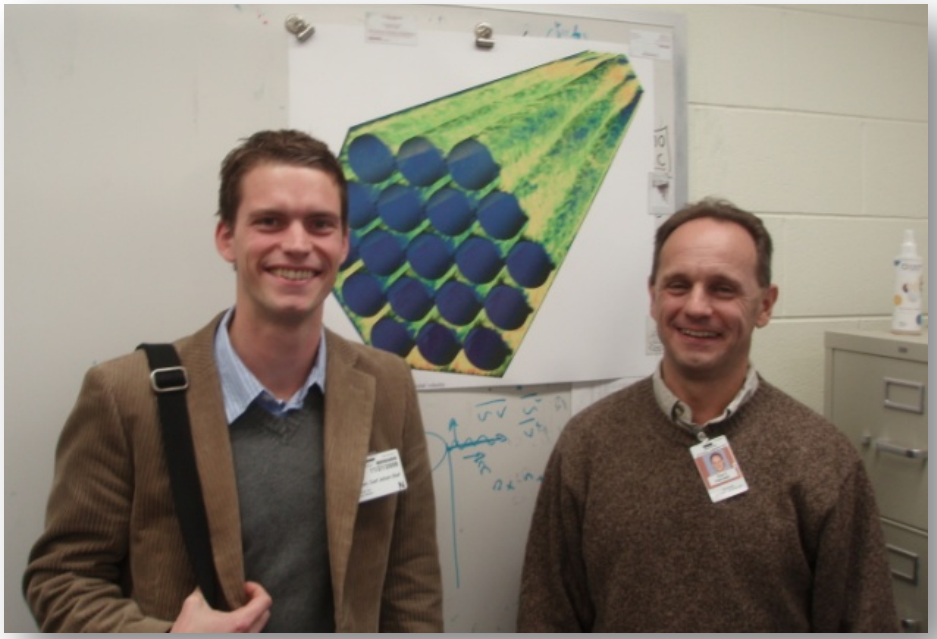
Linné FLOW Centre
and
Swedish e-Science Research Centre (SeRC)
KTH Mechanics, Stockholm, Sweden

https://www.youtube.com/watch?v=hz7UjN_vYuw

Swedish e-Science Research Centre

Nek5000

- Open source code by **Paul F. Fischer**, Argonne National Laboratory and University of Illinois Urbana-Champaign (UIUC), USA
- General purpose DNS/LES code for fluid dynamics, heat transfer, MHD, combustion,…
  - moving meshes,
  - direct and adjoint linear solvers
  - …
- Fortran 77 & C code
  - F77 (70K) & C (30K)
  - MPI parallelization
  - Interfaces VisIt & MOAB
- ”Keep it simple” – world’s most powerful computers have very weak operating systems
- Present scaling up to 1 M cores



Can we go to exa-scale with Nek5000?

- Number of grid points N per processor important, local work has to outweigh cost for communication
- For Nek5000 on BG/P: (N / P) ~ 1000 - 10,000 sufficient

➔ **~10^{12} = minimum number of points to scale to $P = 10^8$**

- We must increase problem size for efficient usage of exa-scale, no problem for higher Reynolds numbers
- More work per grid point advantage
  - HOM (Higher Order Methods) such as SEM
  - Multi-physics (magneto-hydrodynamics, combustion, heat transfer)
  - Accelerators (GPU) require more points per processor
- Major bottleneck: (global) pressure calculation!

CFD for exascale – What we intend to do in SESSI

- Parallel implementation of the setup of an algebraic multigrid solver
- Investigation of communication kernel in Nek5000

- Runtime profiling and automatisation of projections

- Highly tuned small dense matrix-matrix multiplications using SIMD and LIBXSMM
- Refactoring of Nek5000

CFD for exascale – What we intend to do in SESSI

- Parallel implementation of the setup of an algebraic multigrid solver → PRESSURE PRECONDITIONER
- Investigation of communication kernel in Nek5000
  → COARSE GRID COMMUNICATION
- Runtime profiling and automatisation of projections
  → DECREASE OF PRESSURE ITERATIONS
- Highly tuned small dense matrix-matrix multiplications using SIMD and LIBXSMM → FASTER
- Refactoring of Nek5000

Parallel AMG setup

- Pressure preconditioning based on additive Schwarz method → solve Poisson eq. on the whole domain
- Coarse grid solver strategies for the linear system

  – XXT: direct (Cholesky) solver using projection onto space spanned by A-conjugate vectors

    typically for N_{cores}<50k

  – Algebraic Multigrid: multigrid solver depending on the coefficients in the underlying matrix . Used for N_{cores}>50k. Required operators:

    - Coarsening, smoothing, interpolation
    - Parallelisation together with Stefano

Conclusions

- Software and simulation technologies are the real problem at exascale
- SESSI addresses this challenge by bringing together application experts and researchers from MOL-SIM, FLOW, CST and PDC in a common open workspace at PDC. AE permanently at the open workspace.
- SESSI research current tasks:
  - Communication bottleneck in FFT and linear solvers
  - Task parallelism
  - Low overhead profiling
  - SIMD parallelism in critical computational parts of the code

Additional Slides

ExaFLOP ≠ ExaScience

- ExaFLOP will be reality for compute-intense apps? Yes maybe, what about other apps?
- Hungry for strong scaling → communication cost becomes larger than computation
- If we can't use ExaFLOP machine, why do we need one?