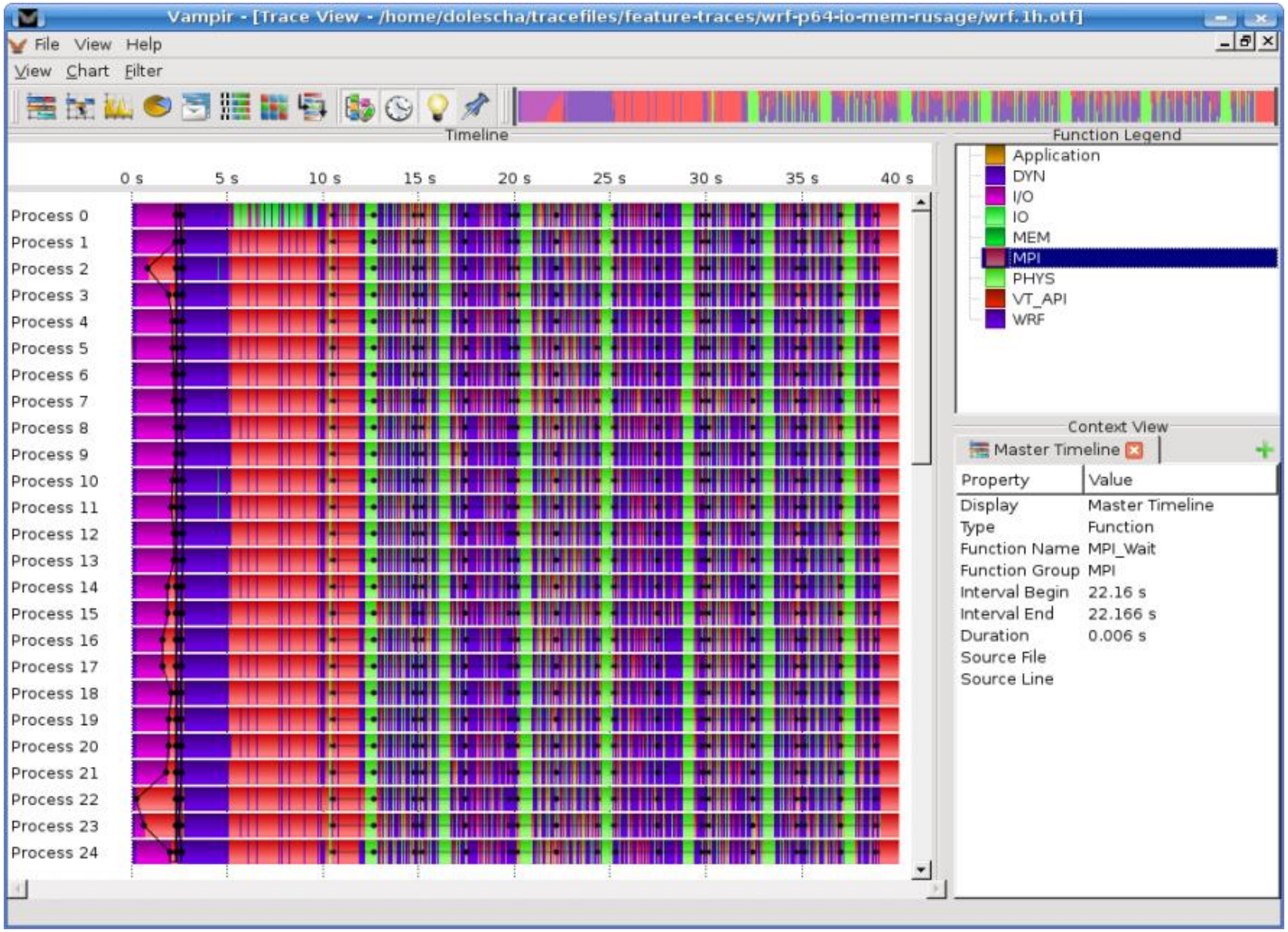
The Scalasca Performance Analysis Toolset

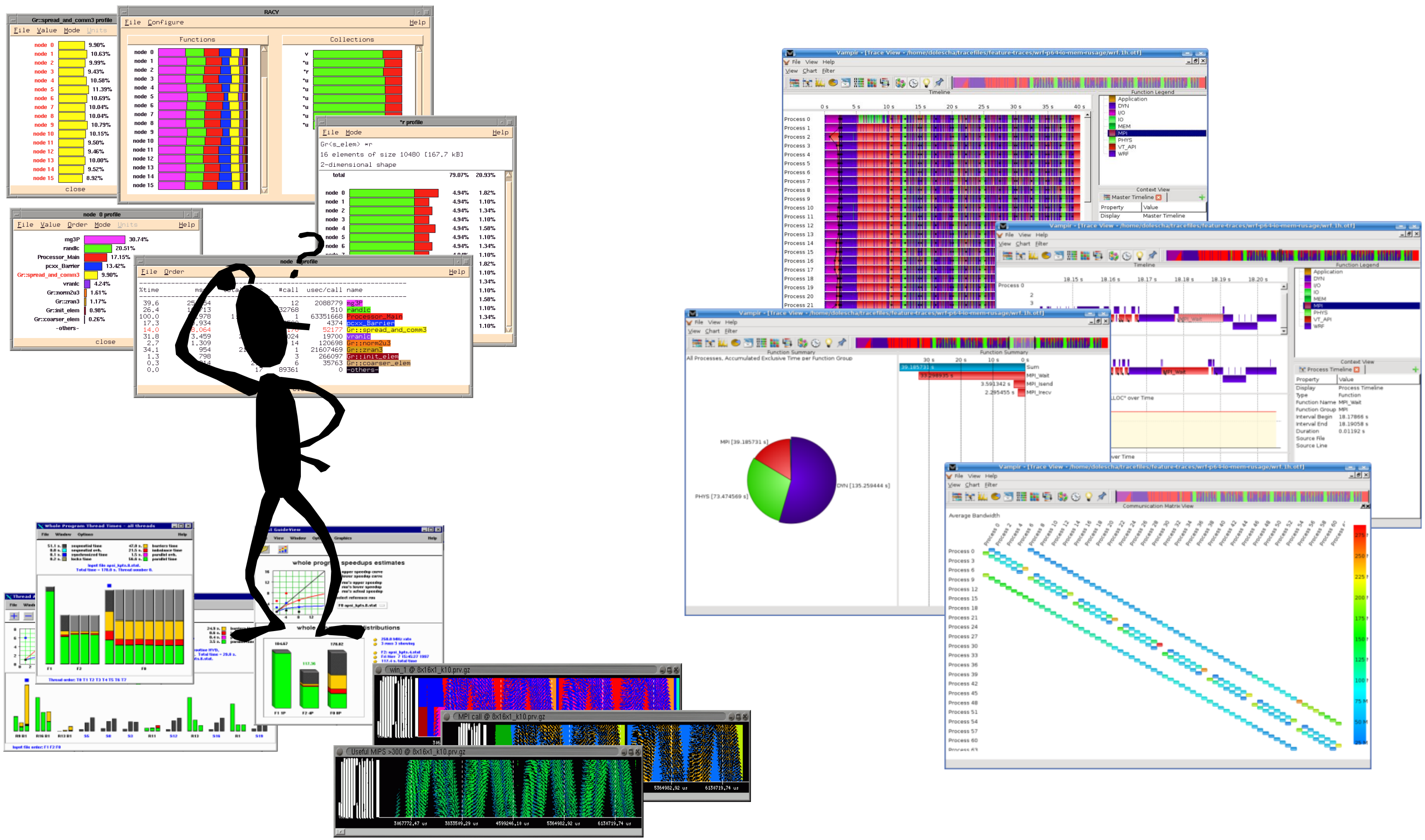
Markus Geimer
Jülich Supercomputing Centre
m.geimer@fz-juelich.de
April 3, 2013

scalasca

“A picture is worth 1000 words…”

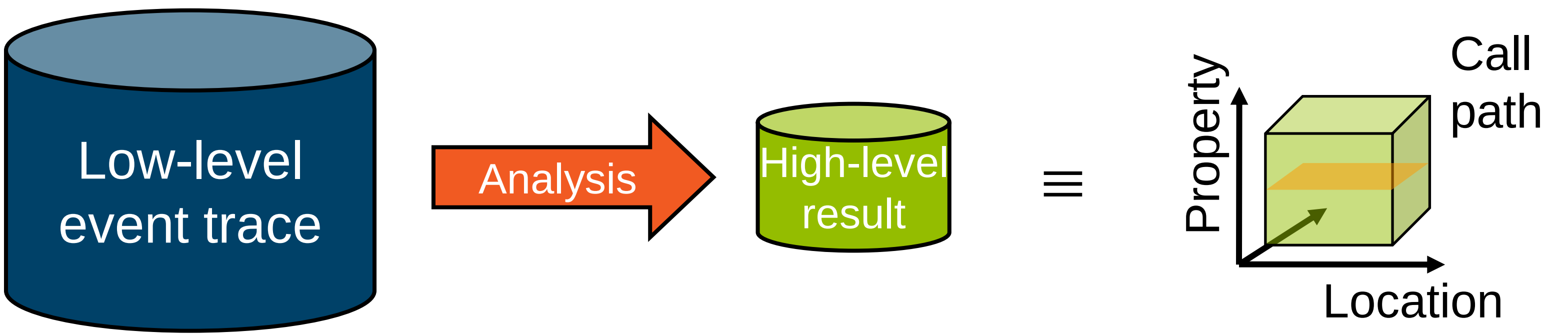


“But what about 1000’s of pictures?”



Automatic trace analysis

- Idea
  - Automatic search for patterns of inefficient behavior
  - Classification of behavior & quantification of significance



- Advantages
  - Guaranteed to cover the entire event trace
  - Quicker than manual/visual trace analysis
  - Helps to identify hot-spots for in-depth manual analysis
    - Complements the functionality of other tools

scalasca

Higher degrees of parallelism

TOP500 SUPERCOMPUTER SITES

Number of Cores share for TOP 500 November 2012

NCore	Count	Share	∑Rmax	Share	∑NCore
1025-2048	1	0.2%	122 TF	0.1%	1,280
2049-4096	2	0.4%	155 TF	0.1%	7,104
4097-8192	81	16.2%	8,579 TF	5.3%	551,624
8193-16384	206	41.2%	24,543 TF	15.1%	2,617,986
> 16384	210	42.0%	128,574 TF	79.4%	11,707,806
Total	500	100%	161,973 TF	100%	14,885,800

Average system size: **29,772 cores**
Median system size: **15,360 cores**

Higher degrees of parallelism (II)

- Also new demands on scalability of software tools
  - Familiar tools cease to work in a satisfactory manner for large processor/core counts
- Optimization of applications more difficult
  - Increasing machine complexity
  - Every doubling of scale reveals a new bottleneck
- Need for scalable performance tools
  - Efficient to meet performance expectations
  - Effective to use so that programmer productivity is maximized

The Scalasca project

- Project started in 2006
  - Follow-up to pioneering KOJAK project (started 1998)
    - Automatic pattern-based trace analysis
  - Initial funding by Helmholtz Initiative & Networking Fund
  - Many follow-up projects
- Objective:
  - Development of a **scalable** performance analysis toolset
  - Specifically targeting **large-scale** parallel applications
    - such as those running on IBM Blue Gene or Cray XT with 10,000s or 100,000s of processes
- Now joint project of
  - Jülich Supercomputing Centre
  - German Research School for Simulation Sciences

The Scalasca toolset

Scalasca 1.4.3

- Custom instrumentation & measurement system
- Scalasca trace analysis components based on custom trace format EPILOG
- Analysis report explorer & algebra utilities CUBE v3
- New BSD license

Scalasca 2.0β

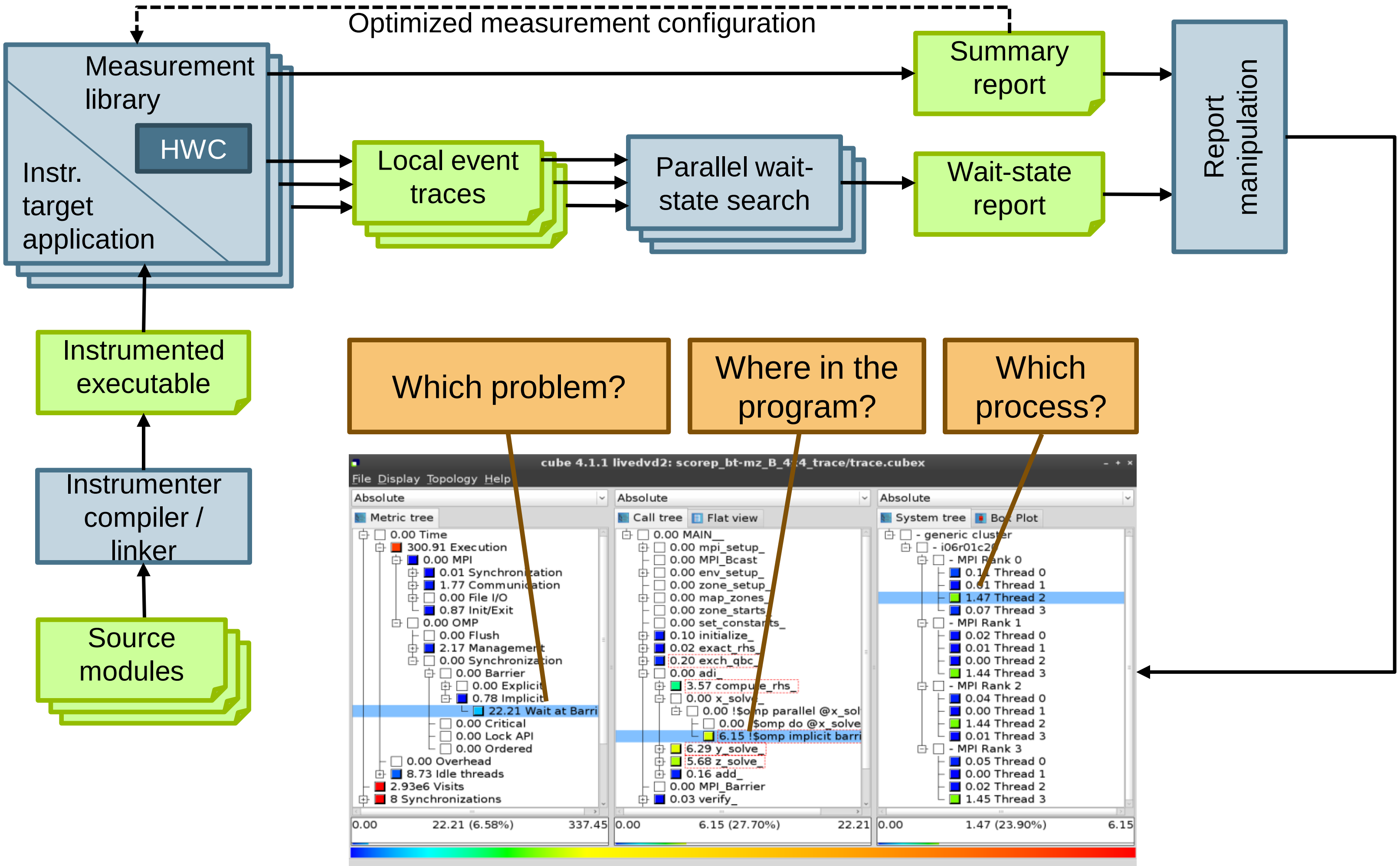
- Community instrumentation & measurement system **Score-P**
- Scalasca trace analysis components based on community trace format **OTF2**
- Analysis report explorer & algebra utilities **CUBE v4**
- New BSD license

http://www.scalasca.org

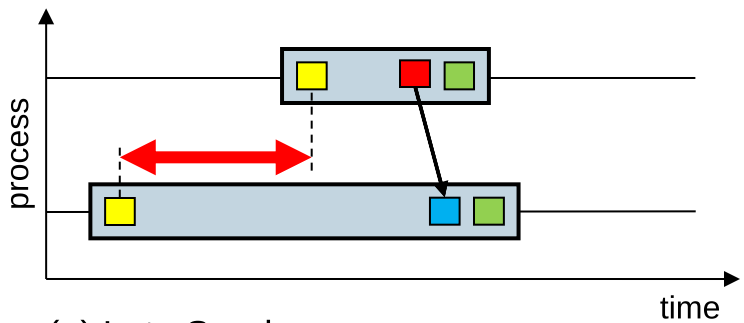
Scalasca features

- Open source
- Portable
  - Blue Gene/Q, Blue Gene/P, IBM SP & blade clusters, SGI Altix, Solaris & Linux clusters
  - Scalasca 1.4.3 only: Cray XT, NEC SX, K Computer, Fujitsu FX10
- Supports parallel programming paradigms & languages
  - MPI, OpenMP & hybrid MPI+OpenMP
  - Fortran, C, C++
- Scalable trace analysis
  - Automatic wait-state search
  - Parallel replay exploits memory & processors to deliver scalability

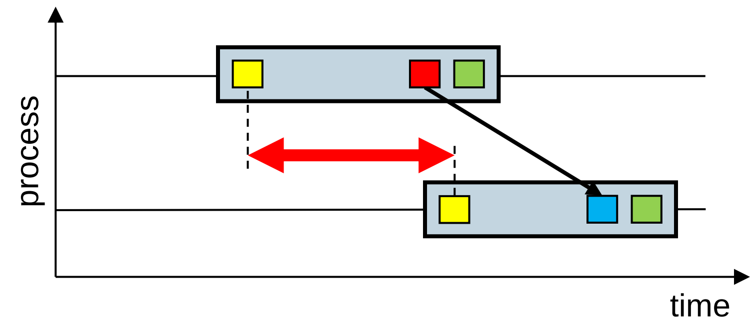
Scalasca workflow



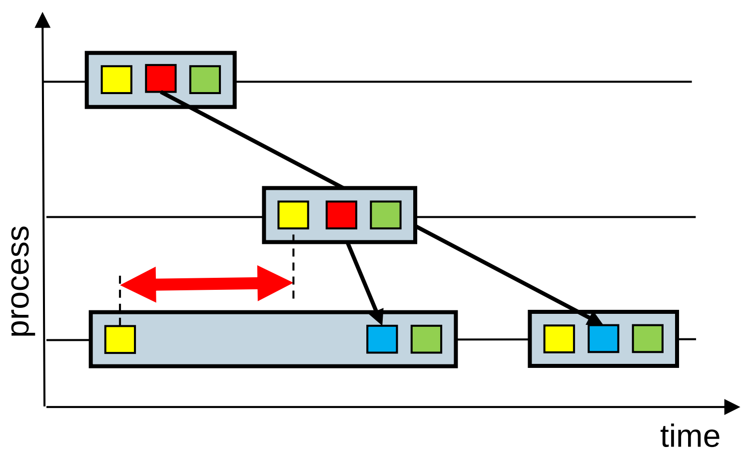
Example: MPI patterns



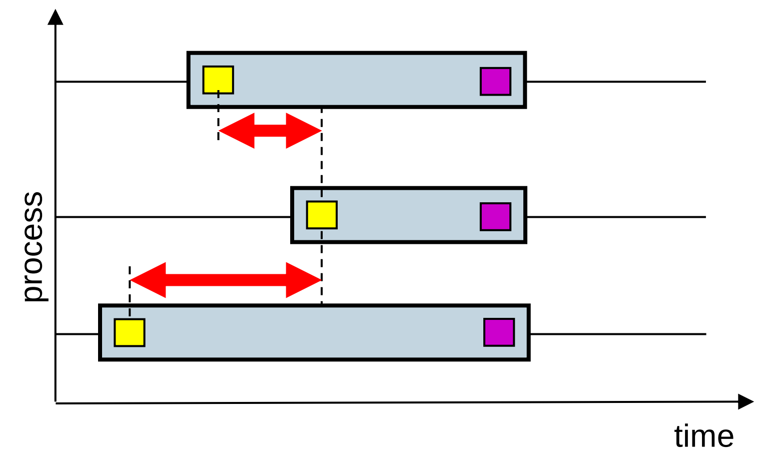
(a) Late Sender



(b) Late Receiver



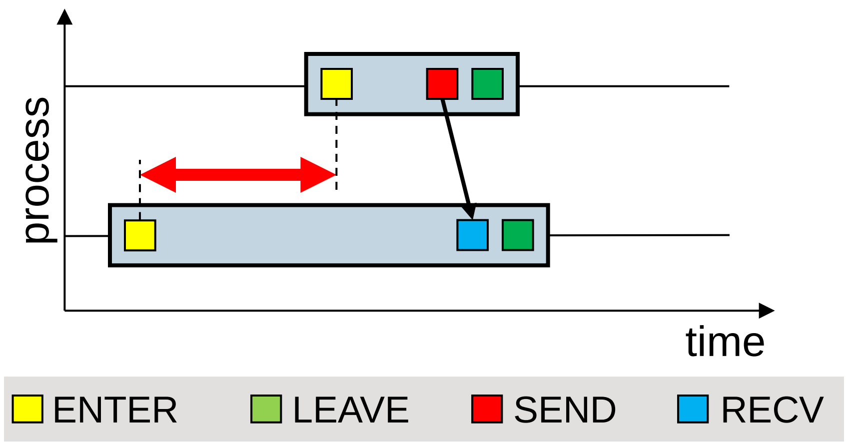
(c) Late Sender / Wrong Order

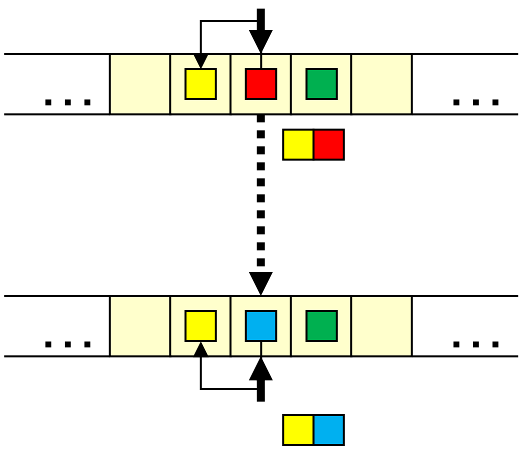


(d) Wait at N x N

ENTER LEAVE SEND RECV COLLECTIVE

Example: Late Sender





Sender:

Triggered by send event

Determine enter event

Send both events to receiver

Receiver:

Triggered by receive event

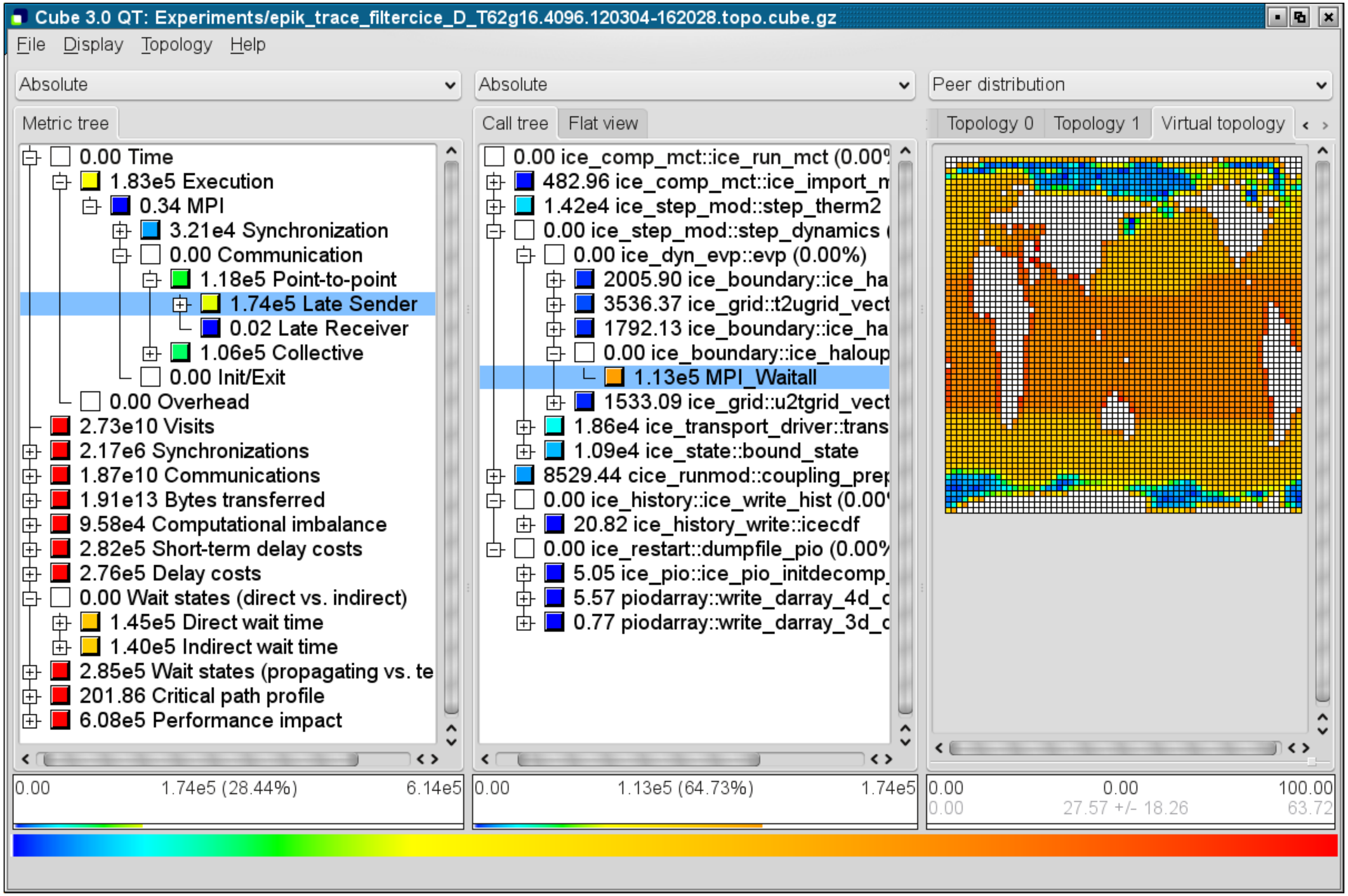
Determine enter event

Receive remote events

Detect *Late Sender* situation

Calculate & store waiting time

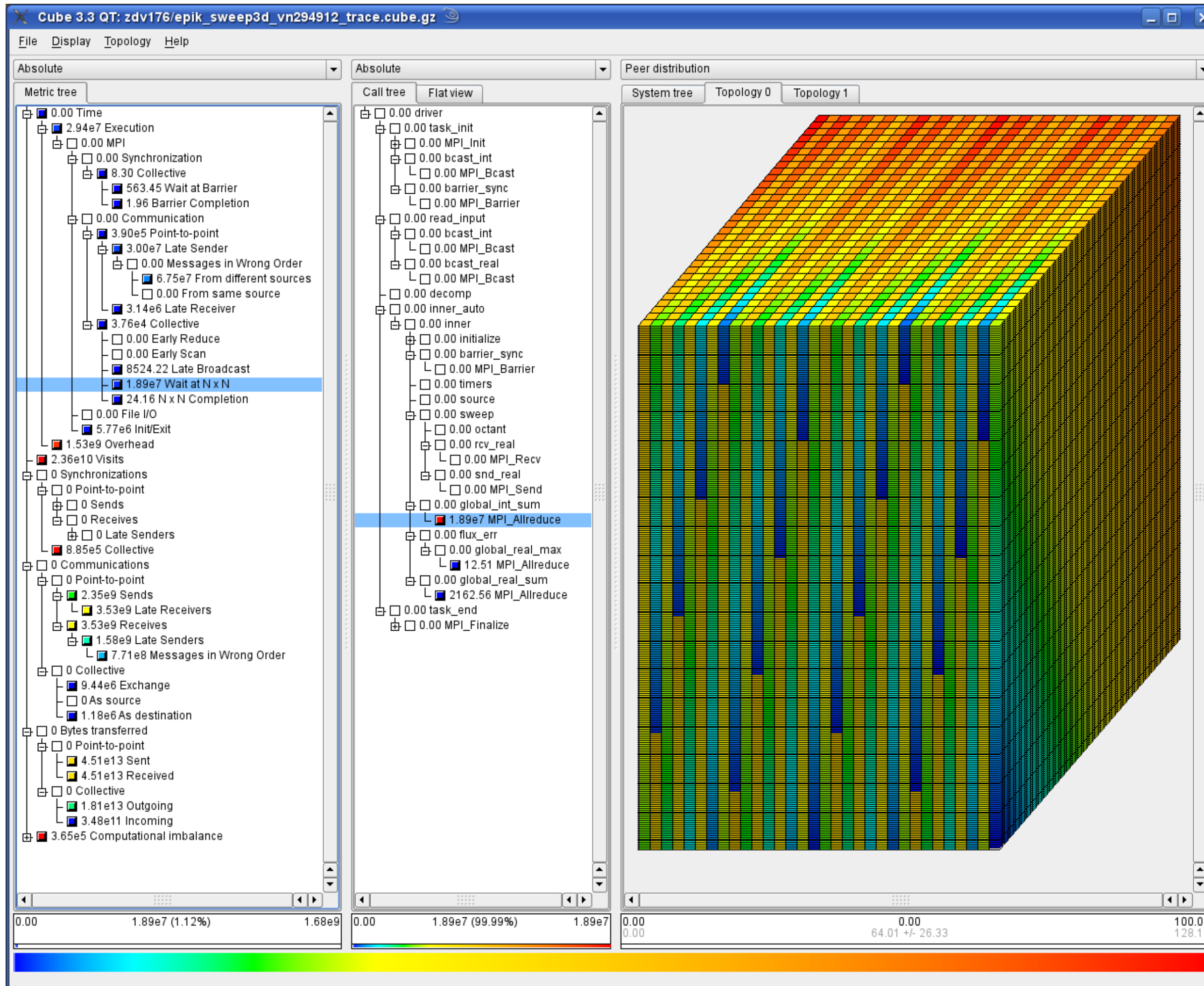
CESM Sea Ice Module – Late Sender Analysis



scalasca

Scalasca trace analysis sweep3D@294,912 BG/P

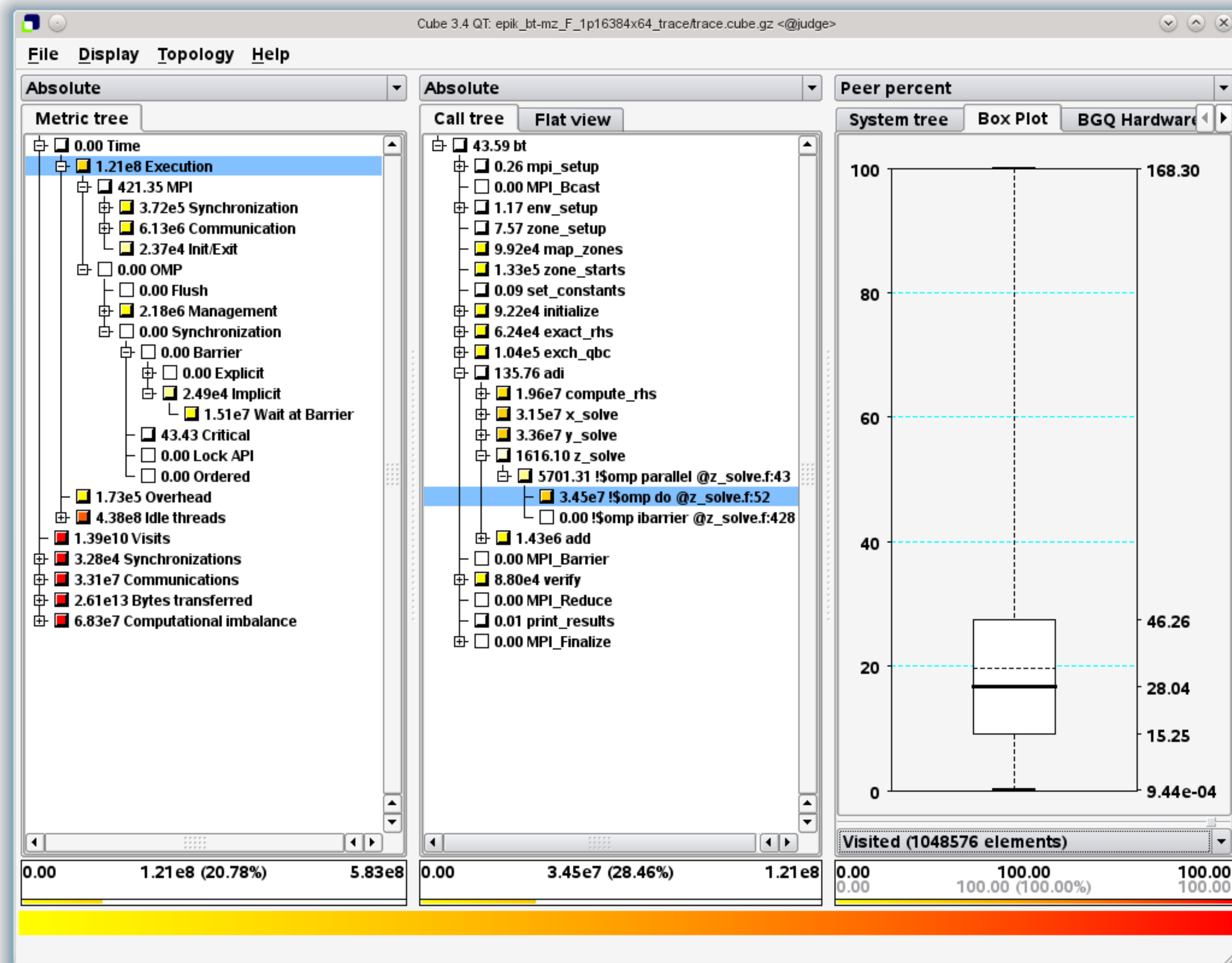
- 10 min sweep3D runtime
- 11 sec analysis
- 4 min trace data write/read (576 files)
- 7.6 TB buffered trace data
- 510 billion events



B. J. N. Wylie, M. Geimer, B. Mohr, D. Böhme, Z.Szebenyi, F. Wolf: Large-scale performance analysis of Sweep3D with the Scalasca toolset. Parallel Processing Letters, 20(4):397-414, 2010.

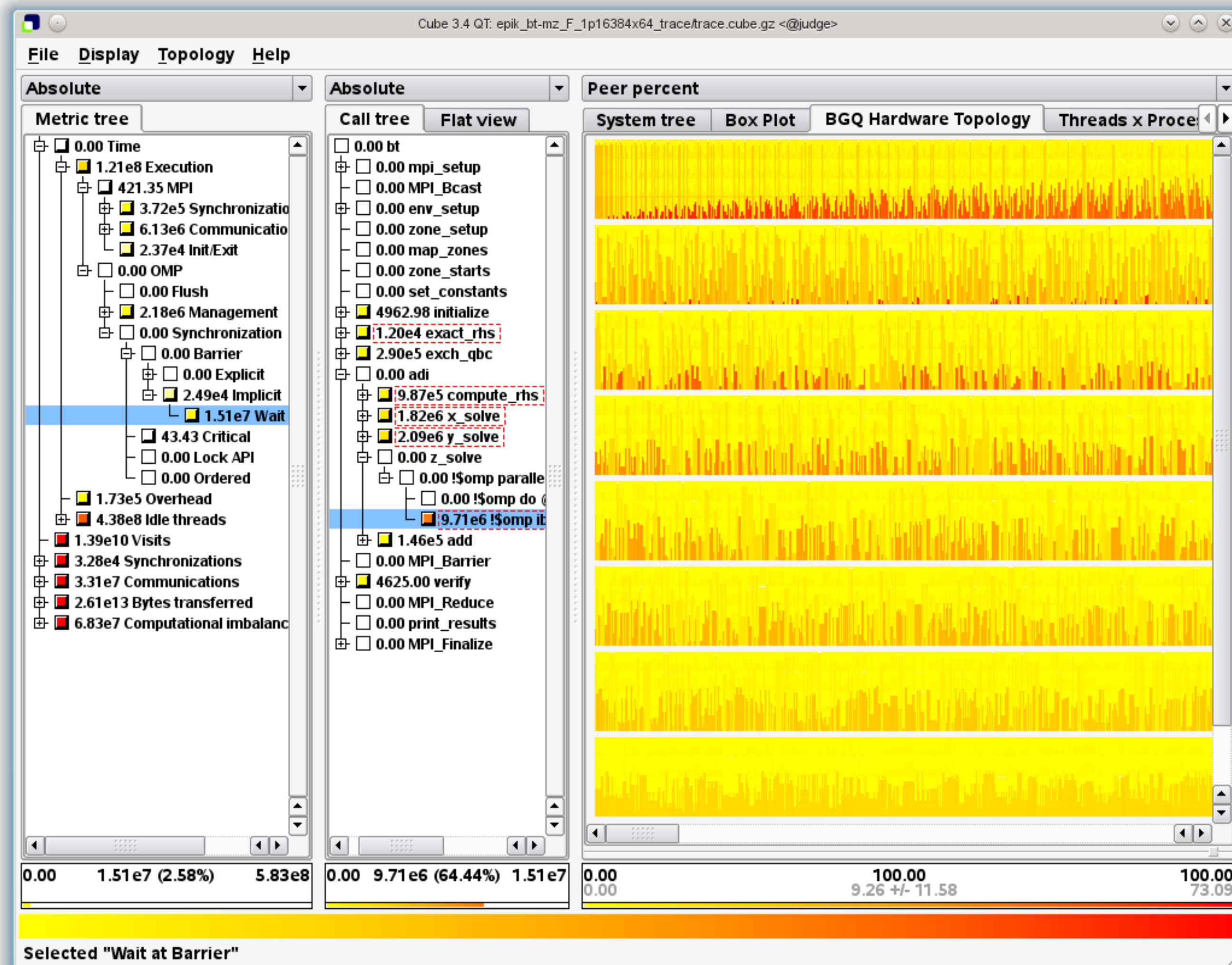
Scalasca trace analysis bt-mz@1,048,704 BG/Q

Execution imbalance "z_solve"



Scalasca trace analysis bt-mz@1,048,704 BG/Q

Wait at implicit barrier "z_solve"



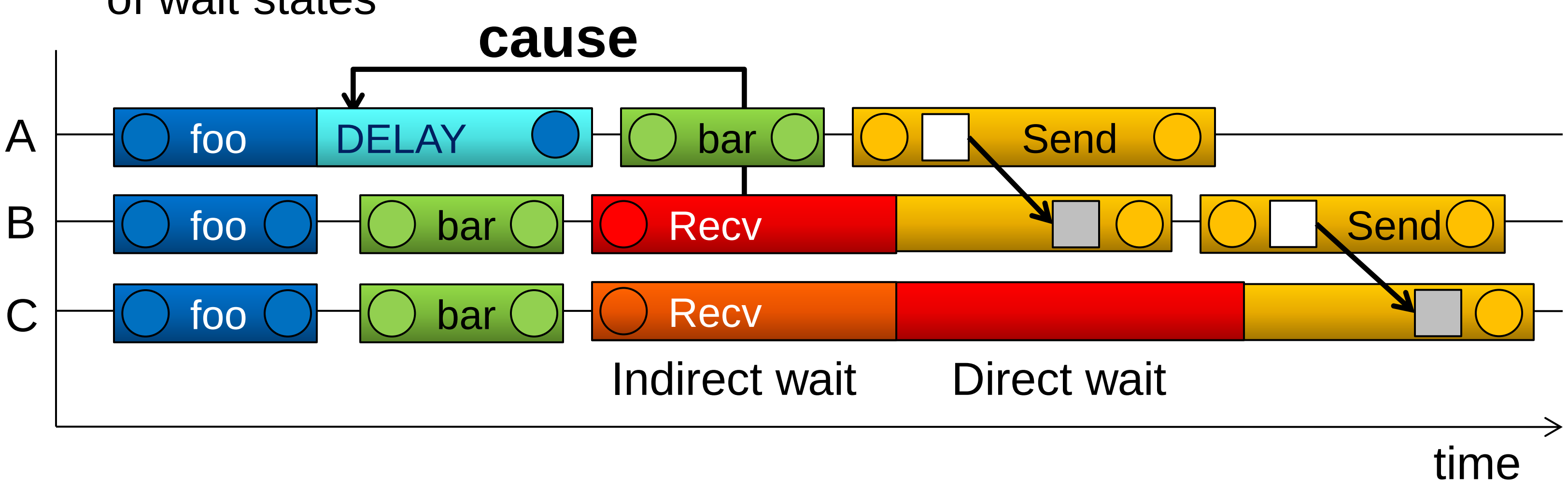
Research: Root Cause Analysis

Root-cause analysis

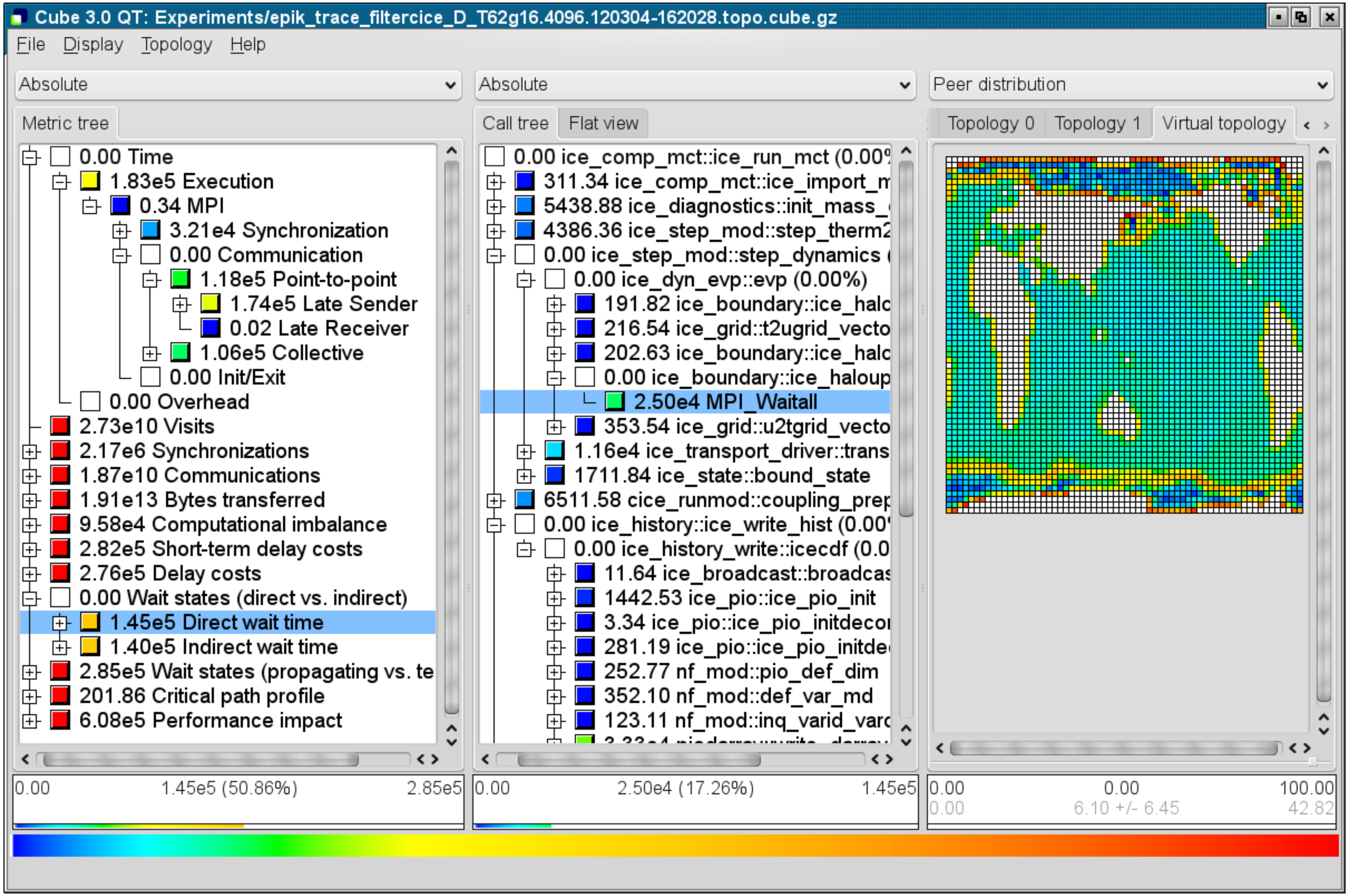
- Wait states typically caused by load or communication imbalances earlier in the program
- Waiting time can also propagate (e.g., indirect waiting time)
- Goal: Enhance performance analysis to find the root cause of wait states

Approach

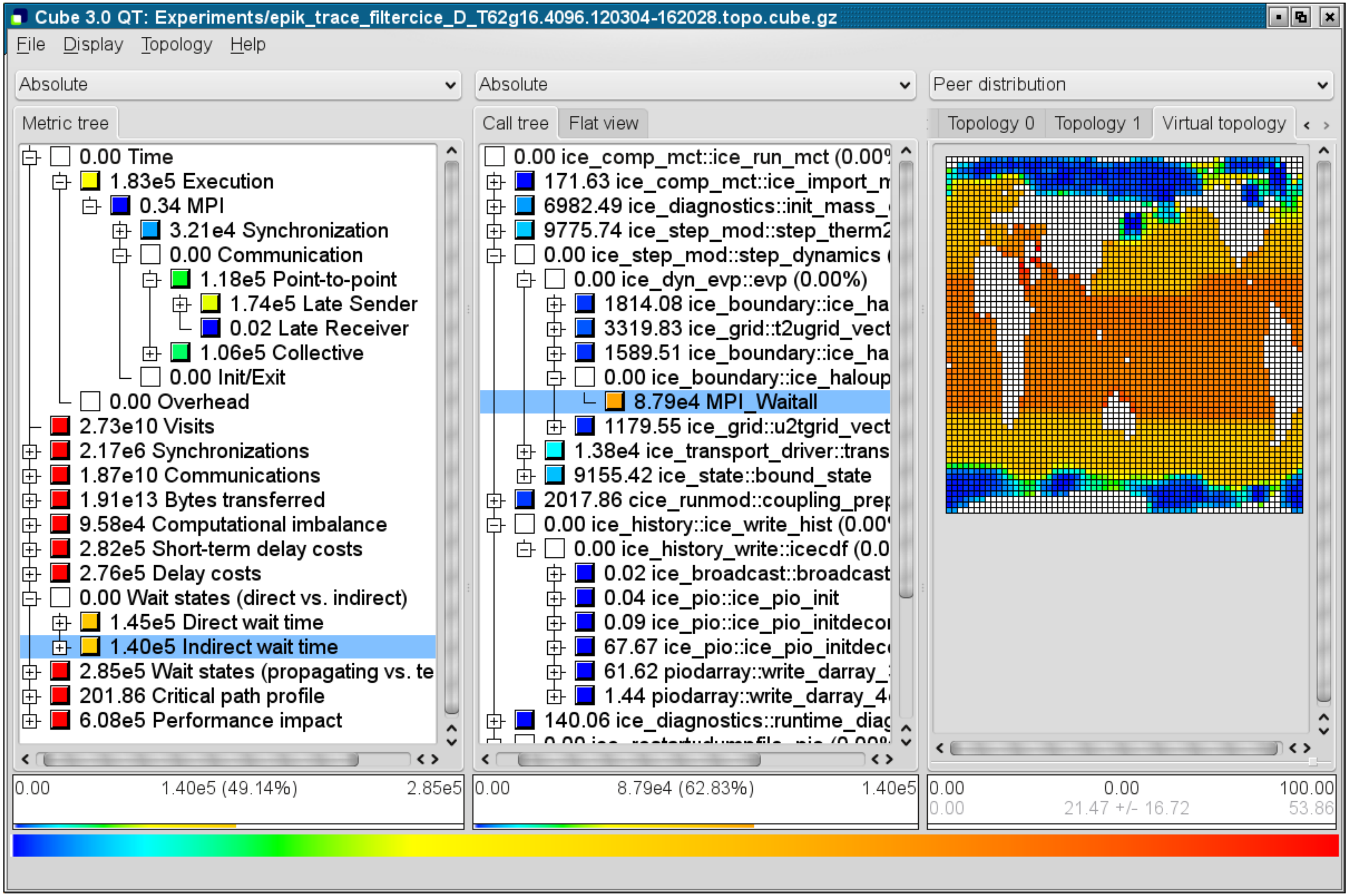
- Distinguish between direct and indirect waiting time
- Identify call path/process combinations delaying other processes and causing first order waiting time
- Identify original **delay**



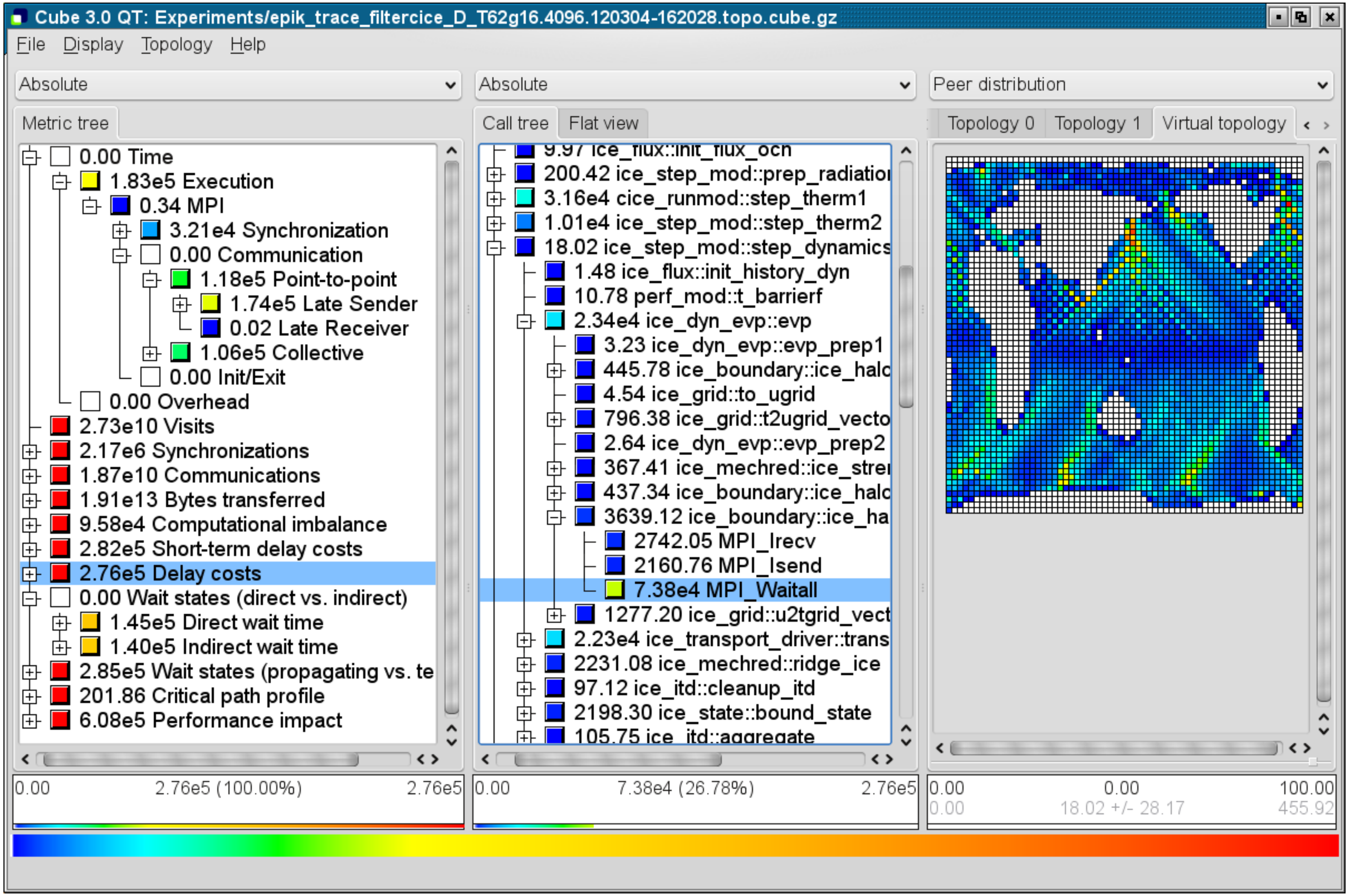
CESM Sea Ice Module – Direct Wait Time



CESM Sea Ice Module – Indirect Wait Time



CESM Sea Ice Module – Delay Costs



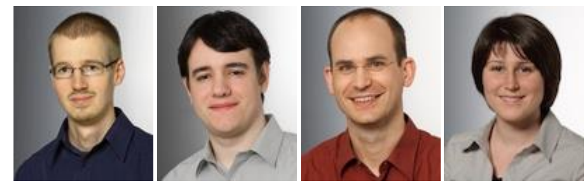
Acknowledgements

Scalasca team (JSC)

Markus Geimer | Jie Jiang | Michael Knobloch | Daniel Lorenz | Bernd Mohr | Peter Philippen | Christian Rössel

Pavel Saviankou | Marc Schlütter | Alexandre Strube | Anke Visser | Brian Wylie | Ilja Zhukov

(GRS)



David Böhme | Alexandru Calotoiu | Marc-André Hermanns | Monika Lücke

Guoyong Mao | Aamer Shah | Sergei Shudler | Felix Wolf

Sponsors

HELMHOLTZ ASSOCIATION

Deutsche Forschungsgemeinschaft DFG

ITEA 2 INFORMATION TECHNOLOGY FOR EUROPEAN ADVANCEMENT

Bundesministerium für Bildung und Forschung

SEVENTH FRAMEWORK PROGRAMME

DEPARTMENT OF ENERGY • UNITED STATES OF AMERICA

Thank you!

http://www.scalasca.org