

August 27th-31st



18th International European
Conference on Parallel and Distributed
Computing

J. Blomer, D. Piparo

In a Cloud

*Cloud built with
the titles of the
contributions.*

Topics

Conference

- **Support Tools and Environments**
- Performance Prediction and Evaluation
- Scheduling and Load Balancing
- High-Performance Architecture and Compilers
- Parallel and Distributed Data Management
- **Grid, Cluster and Cloud Computing**
- Peer to Peer Computing
- Distributed Systems and Algorithms
- **Parallel and Distributed Programming**
- Parallel Numerical Algorithms
- **Multicore and Manycore Programming**
- Theory and Algorithms for Parallel Computation
- High Performance Network and Communication
- Mobile and Ubiquitous Computing
- High Performance and Scientific Applications
- **GPU and Accelerators Computing**

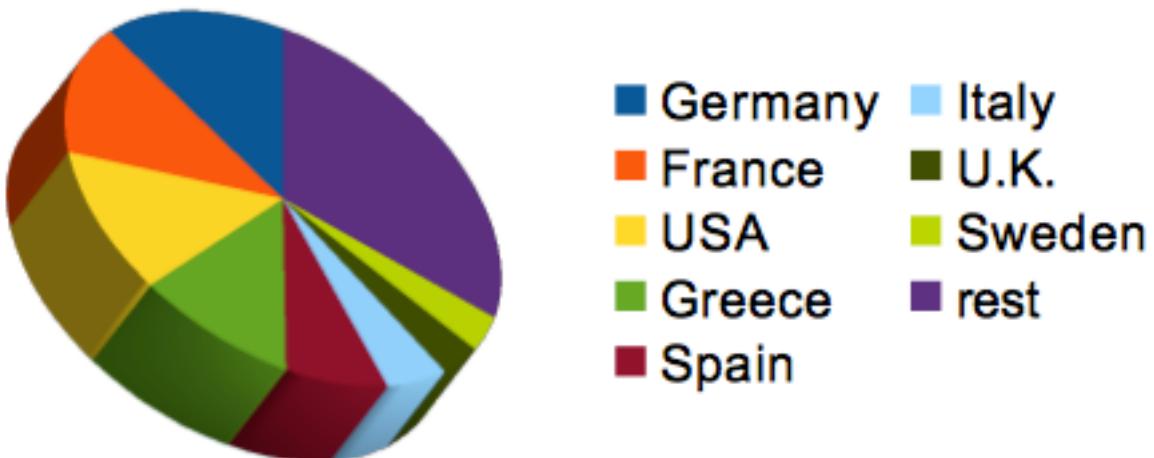
Workshops

- **10th International Workshop on Algorithms, Models and Tools for Parallel Computing on Heterogeneous Platforms**
- **CoreGRID/ERCIM Workshop on Grids, Clouds, and P2P Computing**
- 1st Workshop on Big Data Management in Clouds
- 1st Workshop on On-chip Memory Hierarchies and Interconnects: organization, management and implementation
- 5th Workshop on UnConventional High Performance Computing
- **7th Workshop on Virtualization in High Performance Cloud Computing**
- **5th Workshop on Productivity and Performance**
- 3rd Workshop on High Performance Bioinformatics and Biomedicine
- Workshop on Resiliency in High Performance Computing
- Paraphrase Workshop 2012

<http://europar2012.cti.gr>

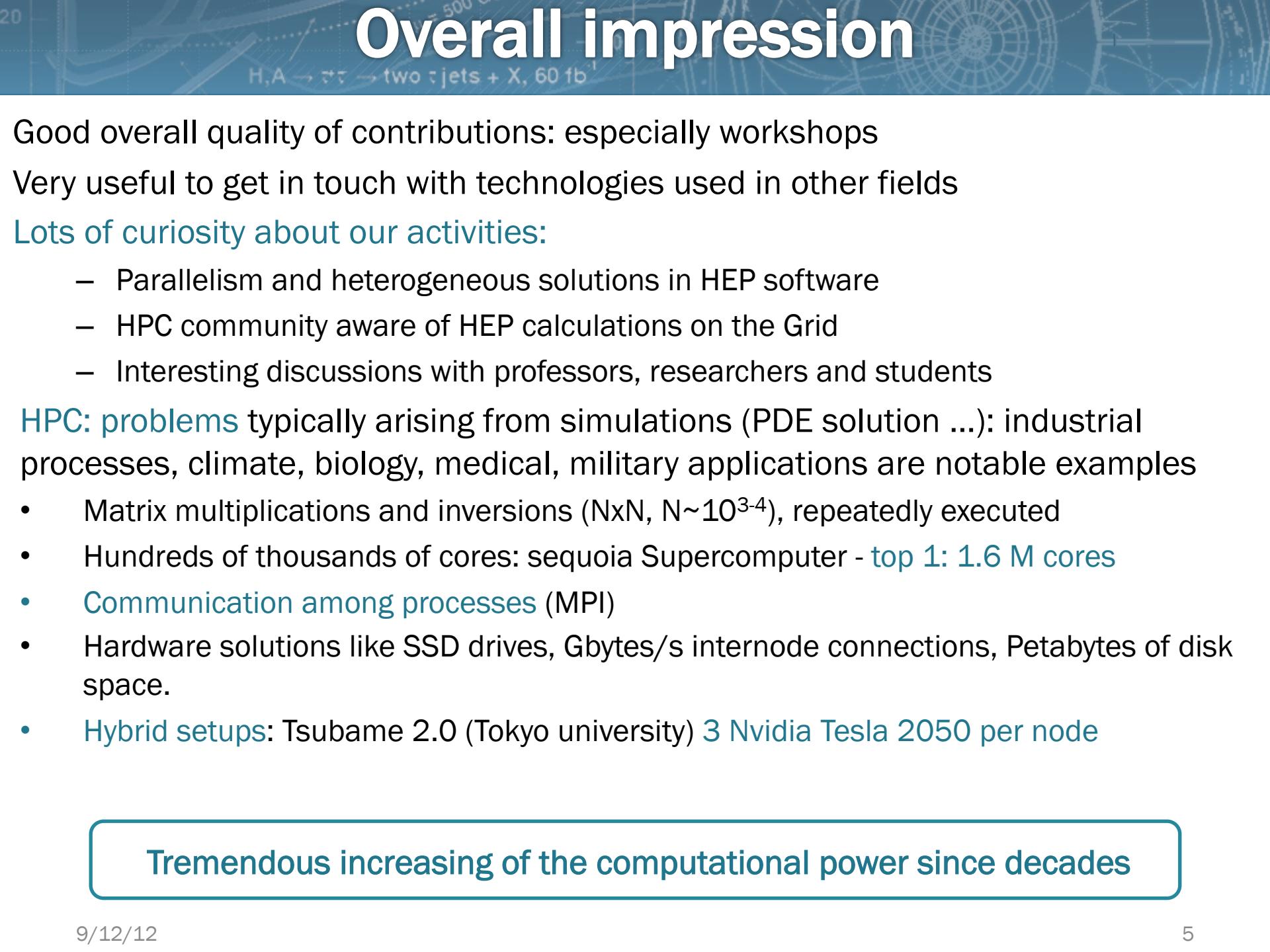
The Event

- 2 Days of workshops (10 of them) + 3 days conference
- 207 participants, 31 countries



- 92 contributions, ~10 contribution per workshop
 - Proceedings available online (see references)
 - Acceptance rate: 33%
- 2 participants from CERN

Overall impression



Good overall quality of contributions: especially workshops

Very useful to get in touch with technologies used in other fields

Lots of curiosity about our activities:

- Parallelism and heterogeneous solutions in HEP software
- HPC community aware of HEP calculations on the Grid
- Interesting discussions with professors, researchers and students

HPC: problems typically arising from simulations (PDE solution ...): industrial processes, climate, biology, medical, military applications are notable examples

- Matrix multiplications and inversions ($N \times N$, $N \sim 10^{3-4}$), repeatedly executed
- Hundreds of thousands of cores: sequoia Supercomputer - **top 1: 1.6 M cores**
- **Communication among processes** (MPI)
- Hardware solutions like SSD drives, Gbytes/s internode connections, Petabytes of disk space.
- **Hybrid setups**: Tsubame 2.0 (Tokyo university) **3 Nvidia Tesla 2050 per node**

Tremendous increasing of the computational power since decades

Virtualization in HPC

2 workshops and 1 keynote speech

Invited talk (45'+15'): *Experiences in using virtual machines for High-Throughput Computing at the CERN LHC*

Use cases

- Bio-Informatics (DNA sequencer): small collaborations with large computational needs
- Weather forecast
- Worker node co-scheduling in large data centers

Scheduling and cluster management

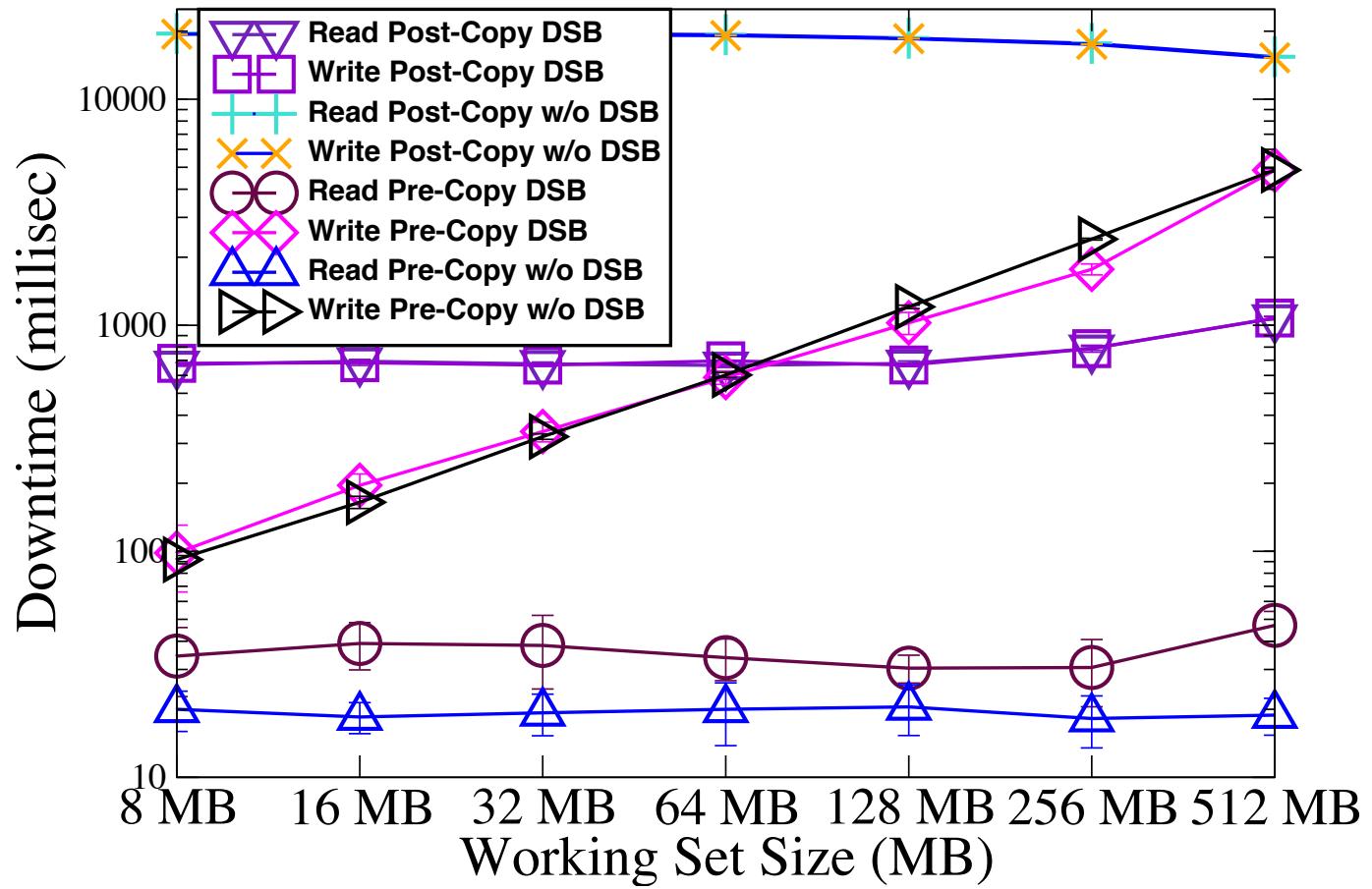
- Workflow engines mushroom (Makeflow, Taverna, ASKALON, Pegasus)
- Research areas: ensemble scheduling and consolidation scheduling
- Objective: minimize energy consumption and cost

New performance results

- Myrinet on Xen: from 44 μ s (software bridge) to 14 μ s (direct I/O: 13 μ s)
- Downtime of live migration reduced to ~20ms,
optimized for heavy memory churn
Application: instant clone + migration instead of booting (SnowFlock)

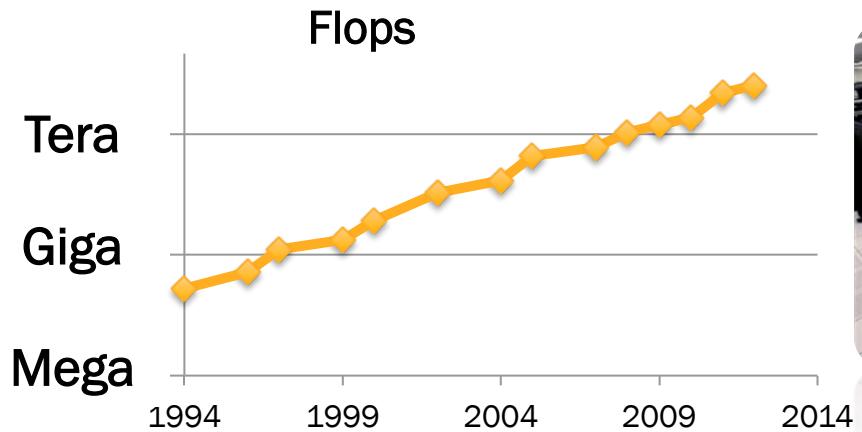
Virtualization in HPC

Downtime



Green Computing

Many flops → huge energy consumption



Rough estimates (today's values!):
Supercomputer consumption = ~10 MW
Nuclear reactor power = ~1000 MW

1% !

Unfeasible to follow the current trends for the future supercomputers. Power consumption is one of the main issues linked to the construction of Exascale facilities.

Parallel programming models

MPI widely adopted: decompose problem in processes exchanging messages

- Study message broadcasting w/o network homogeneity. BitTorrent outperforms MPI: potential application in a Cloud (system topology not known a priori) [1]

OpenMP very present: annotations in code to express parallelism

- Task model of OpenMP also exploited

→ *Massive usage of pragma directives.*



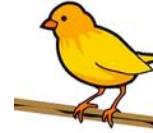
Other technologies presented:

OMPSs: extend OpenMP to support heterogeneity

OMPI: OpenMP infrastructure for C, source-to source compiler + runtime. Enhanced data locality and work stealing [2]

Threading Building Blocks barely mentioned

- Perhaps arrived too late for this community
- *FastFlow* lock-free alternative to FlowGraph



Task scheduling problem studied in presence of cpus with different frequencies, core counts/types. Assign “complexity” to tasks + estimate tasks duration trends for runtime scheduling optimisation [3]

In presence of non-uniform hardware (i.e. fat/thin cores), we might also consider to label the “weight” of our tasks/algorithms (influence of input data could be not trivial)...

Heterogeneous Computing

H,A → ττ → two τ jets + X, 60 fb⁻¹

Clear trend in the HPC community: GPU is among most frequent used terms

- Heterogeneous clusters: several GPUs per node
- Nvidia products by far the most used
 - Cuda is de facto the standard: maximum performance on Nvidia devices

Plethora of libraries ported to Cuda: CULA, CUBLAS...



Quite some programming models shown:

- OpenACC: pragma directives, offload on accelerators w/o device init, mem transfer.
Comparison with OpenCL on real problems: simpler (far less code modifications), somehow lower performance (still evolving model) [4]
- PEPPER: abstract from architectural details, C/C++ annotations, language/runtime/ compiler support, schedule different implementation variants [5]
- dOpenCL: run OpenCL application on several different cluster nodes [6]



Two examples of porting existing code to run on GPU: MeteoSwiss and NICAM atmospheric model [7]

- Impressive speedups
- Effort to port the existing code often not negligible

Transactional Memory (TM)

Awareness about the potential of this technology. Impression from speakers: easier to develop with TM than with locks.

TM: Avoid the usage of *locks* to protect critical sections but use *transactions*.

New CPUs (~2013) will support TM at hardware level

- Presently emulated at software level: “STM”

Performance of locks Vs STM compared using the StarSs programming model [9]

- In presence of *high lock contention*, for the benchmarks ran, STM can outperform traditional locking strategies

Investigation of *interaction of thread pinning and STM*, using different libraries (TinySTM, SwissSTM, TL2)

- Complex memory hierarchies (multicore CPUs): noticeable *speedups* can be achieved [8]

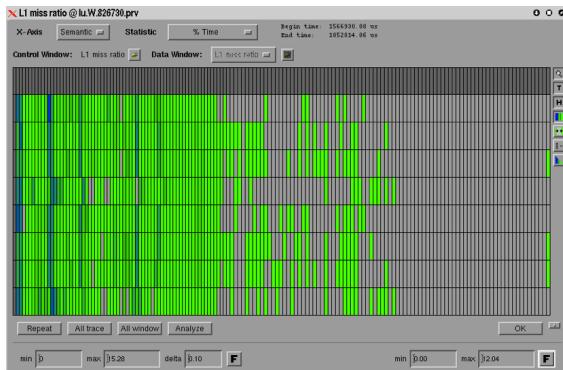
Tools – some examples

Performance patterns of multicore applications and their hardware counters signatures studied [10]

- I.e.: strided memory access, syncro overhead, false sharing, limited instruction throughput
- The LIKWID toolkit is a way to investigate them

Paraver detailed quantitative analysis of multicore program performance

- Gui
- Interface also for Pthreads



Metric	core 0	core 1
Runtime [s]	0.326242	0.32672
CPI	4.84647	4.14891
DP MFlops/s (DP assumed)	245.399	189.108
Packed MUOPS/s	122.698	94.554
Scalar MUOPS/s	0.00270351	0
SP MUOPS/s	0	0
DP MUOPS/s	122.701	94.554

Tulipse^[11]: visualisation framework to help programmers in the parallelisation process

- Eclipse Plugin, loop procedure/data dependency views, basic profiling
- Maybe not silver bullet, but might be useful for a didactical approach

Our Conclusions

There is room for contributions from within the HEP community

In comparison with previous experiences of SuperComputing:

- Less tradeshow, no posters/hands-on, more academically oriented, workshops had better quality

Parallel Computing:

- In HPC parallelism is everyday business since years
- HPC problems: seldom data intensive
- Code annotations (OpenMP): the main way to express parallelism
 - Task-centric model being studied and adopted as well
- Useful input for HEP in terms of tools, ideas and discussions

Virtualization:

- Performance overhead shifts away
- Allows small participants to handle large computational need

Attending the next conference would be certainly useful.

References

Europar2012 Proceedings:

<http://www.springerlink.com/content/978-3-642-32819-0/>

- [1] MPI vs BitTorrent: switching between large-message broadcast algorithms in the presence of bottleneck links
- [2] Speeding up OpenMP tasking [here](#)
- [3] Exploring heterogeneous scheduling using the task-centric programming model
- [4] OpenACC – First experiences with Real-World applications [here](#)
- [6] dOpenCL – Supporting distributed heterogeneous computing on HPC clusters
- [7] High-Level Support for pipeline parallelism on many-core parallelism [here](#)
- [8] Dynamic thread mapping based on machine learning for transactional memory applications [here](#)
- [9] Transactional access to shared memory in StarSs, a task based programming model [here](#)
- [10] Performance patterns and hardware metrics on modern multicore processors: best practices for performance engineering
- [11] Tulipse - A visualisation framework for user-guided parallelisation [here](#)
- [12] On the Instrumentation of OpenMP and OmpSs Tasking Constructs

The proceedings of the workshops are yet to be published. Some links are therefore missing.