The evolving marriage of hardware and software, as seen from the openlab perspective

Andrzej Nowak, CERN openlab CTO office

CERN IT Technical Forum, Feb 21 2014

Hardware

Problem

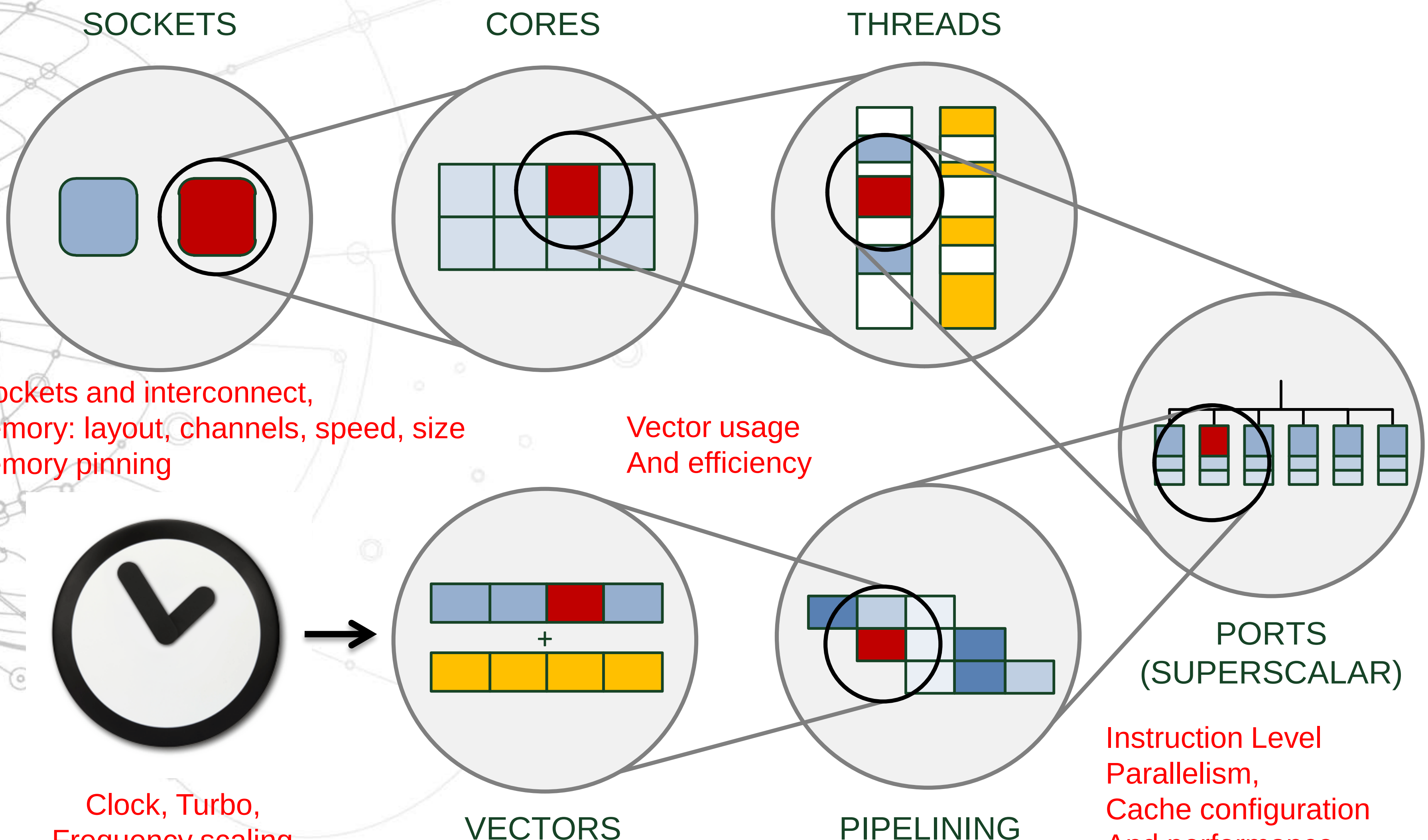
How much can a modern computer do?

Solution

Look inside and think about it

Number of cores
Package topology
Virtualization
C-States

#threads
Topology, OS numbering
On or off
Shared or partitioned resources

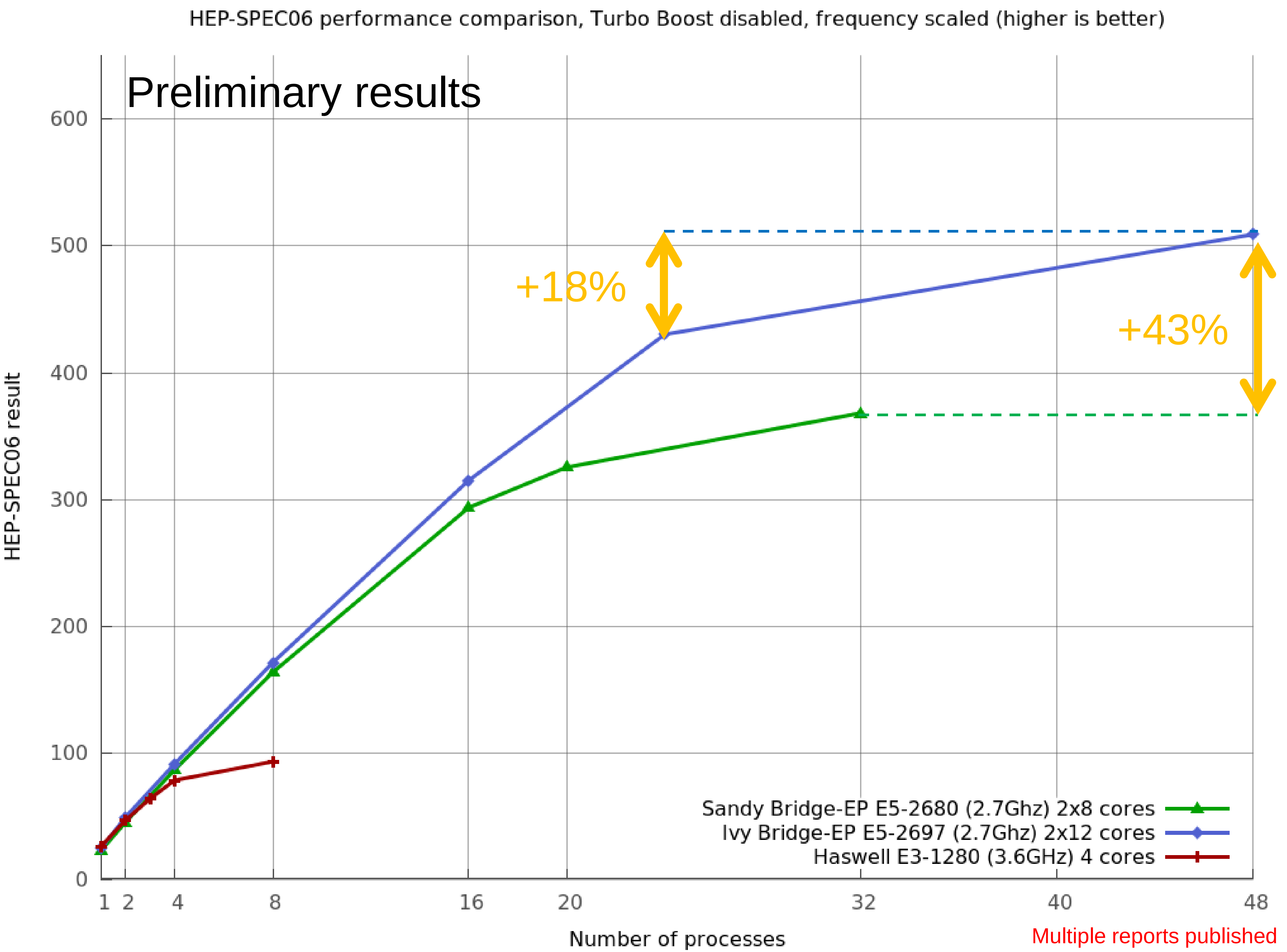


#sockets and interconnect,
Memory: layout, channels, speed, size
Memory pinning

Vector usage
And efficiency

Clock, Turbo,
Frequency scaling

Instruction Level
Parallelism,
Cache configuration
And performance


HEP-SPEC06 performance comparison, Turbo Boost disabled, frequency scaled (higher is better)
Preliminary results
+18%
+43%
HEP-SPEC06 result
0
100
200
300
400
500
600
1
2
4
8
16
20
32
40
48
Number of processes
Sandy Bridge-EP E5-2680 (2.7Ghz) 2x8 cores
Ivy Bridge-EP E5-2697 (2.7Ghz) 2x12 cores
Haswell E3-1280 (3.6GHz) 4 cores
Multiple reports published

Problem

We don't understand why the curve bends

Solution

Measure more detail

Performance measurements

CPI	0.5332
load instructions %	27.78%
store instructions %	11.71%
load and store instructions %	39.49%
resource stalls % (of cycles)	24.77%
branch instructions % (approx)	8.32%
% of branch instr. mispredicted	0.56%
all computational uops	35.50%
% of L3 loads missed	7.14%
computational x87 instr. %	0.04%

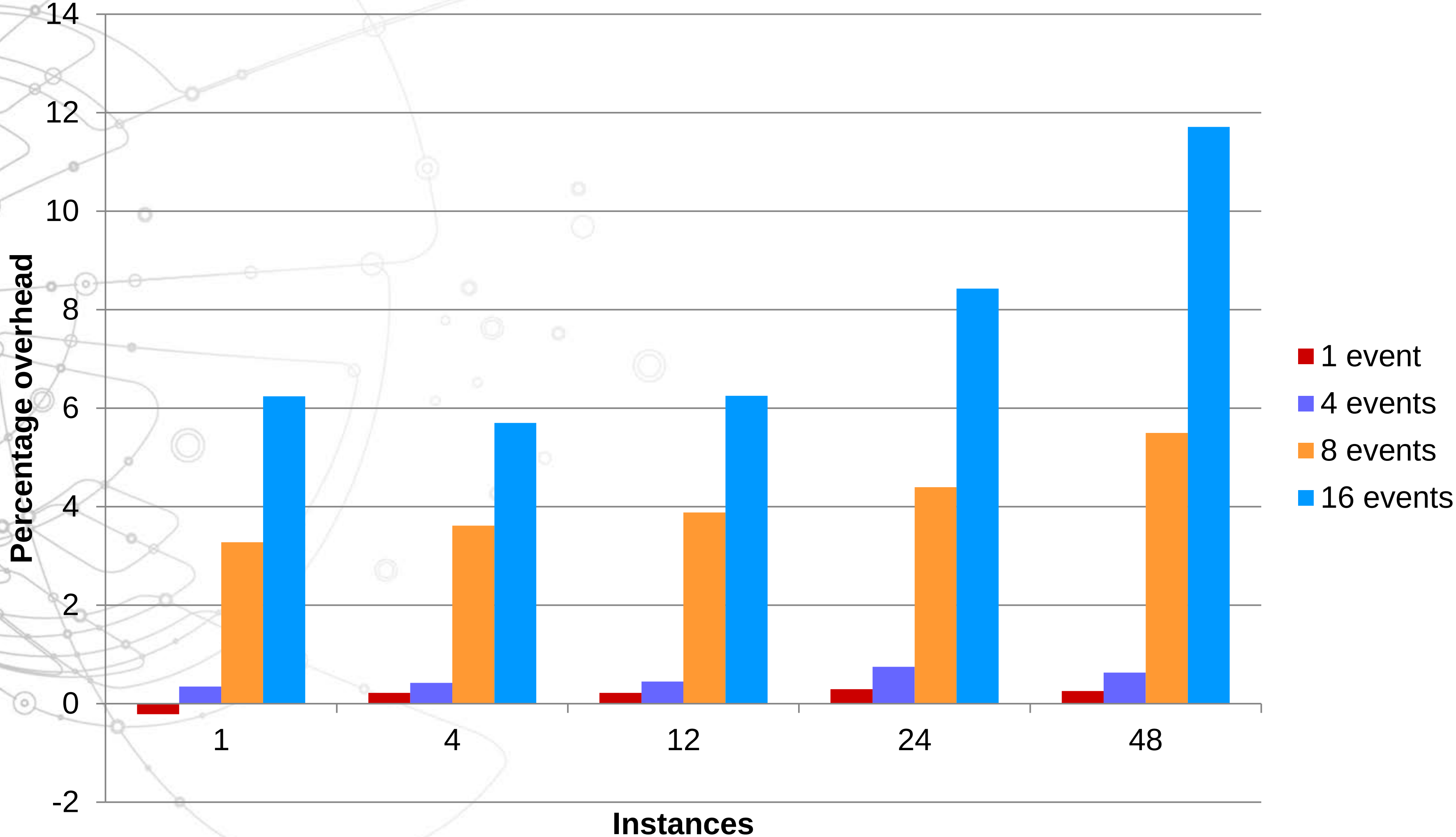
Problem

We don’t know if this is slowing down our program

Solution

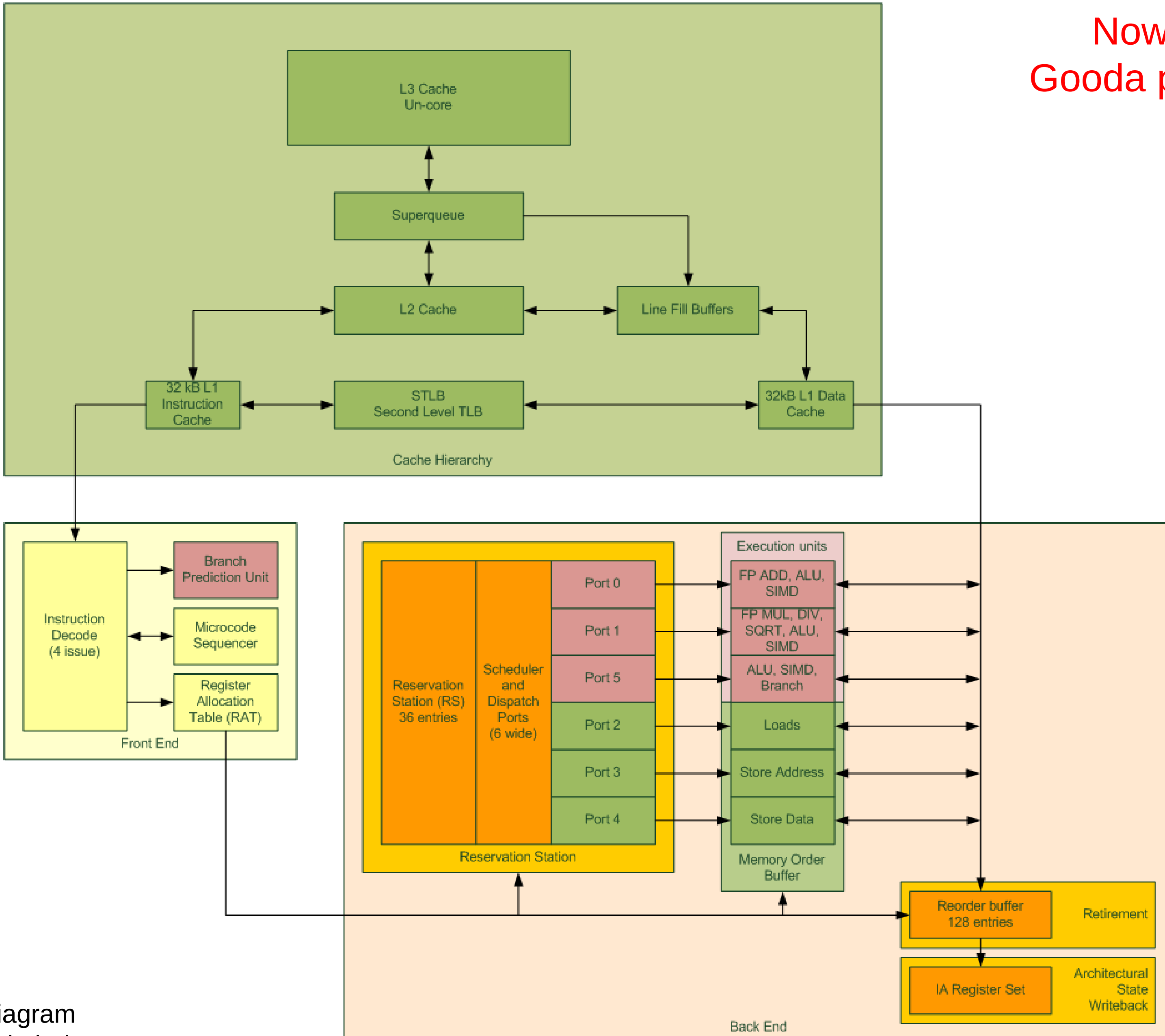
Measure the measurement

Measuring the measurement



Report ready

Now in the Gooda profiler



Westmere core diagram
A. Nowak / D. Levinthal

Problem

There are all these cache misses. SO WHAT?

Solution

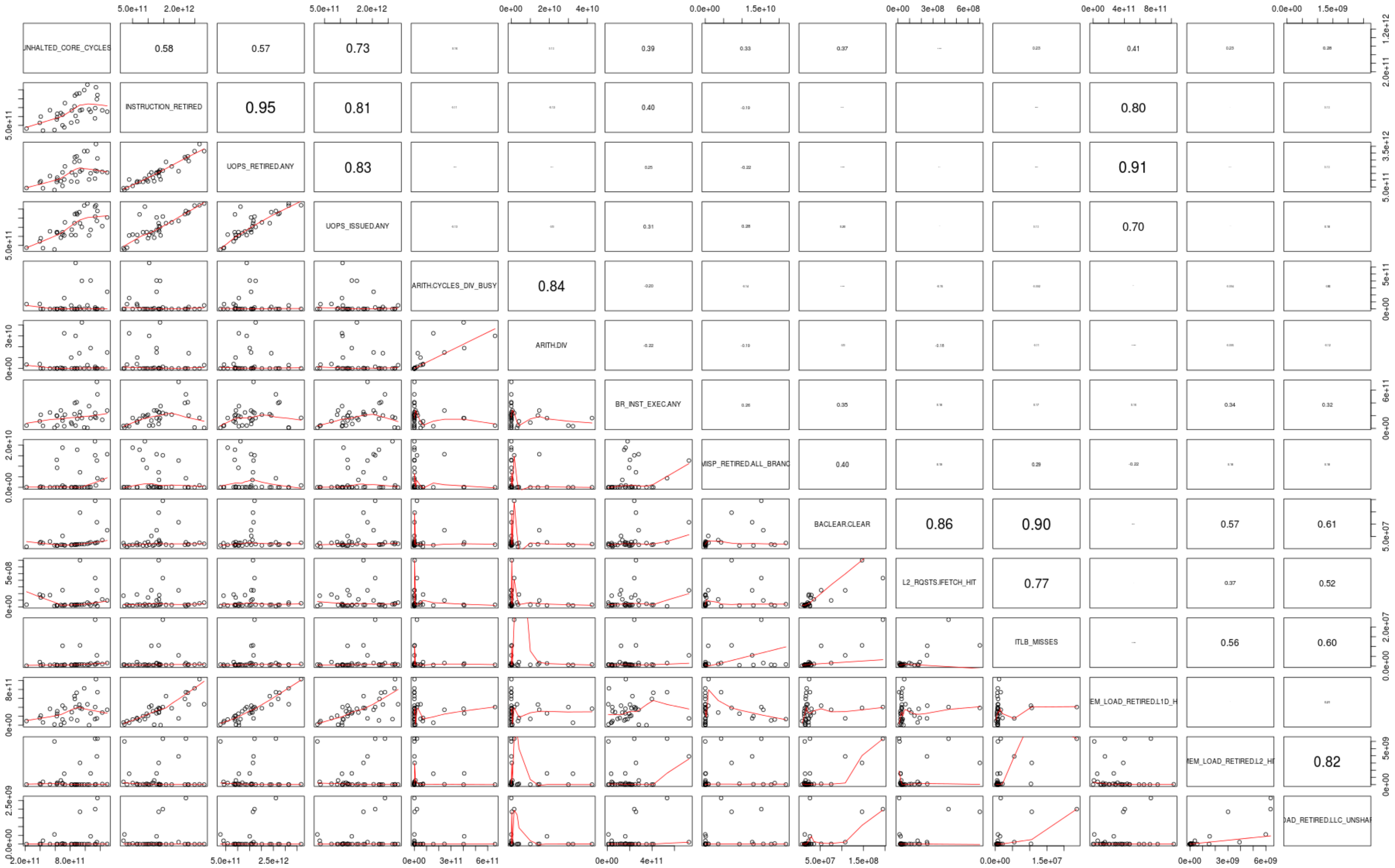
Develop expertise to interpret the result

Interpreting measurements

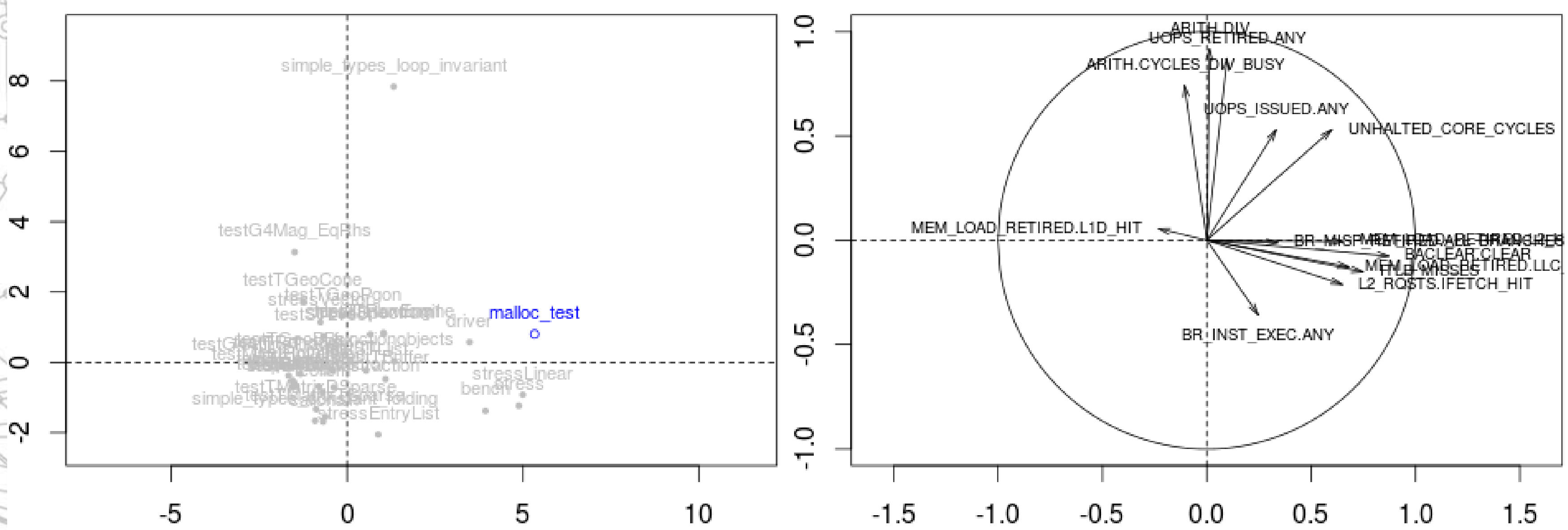
Pattern	Load, load, do something, multiply, add, store
FP	Scalar double, 10-15%
CPI	>1.0
Load/store	60% of instructions
Inst/jump	<10
Inst/call	<30-60
Memory	Largely read-only

- Conclusions:
  - Unfavorable for the x86 microarchitecture (even worse for others)
  - For the most part, code not fit for accelerators at all in its current shape

Interpreting measurements

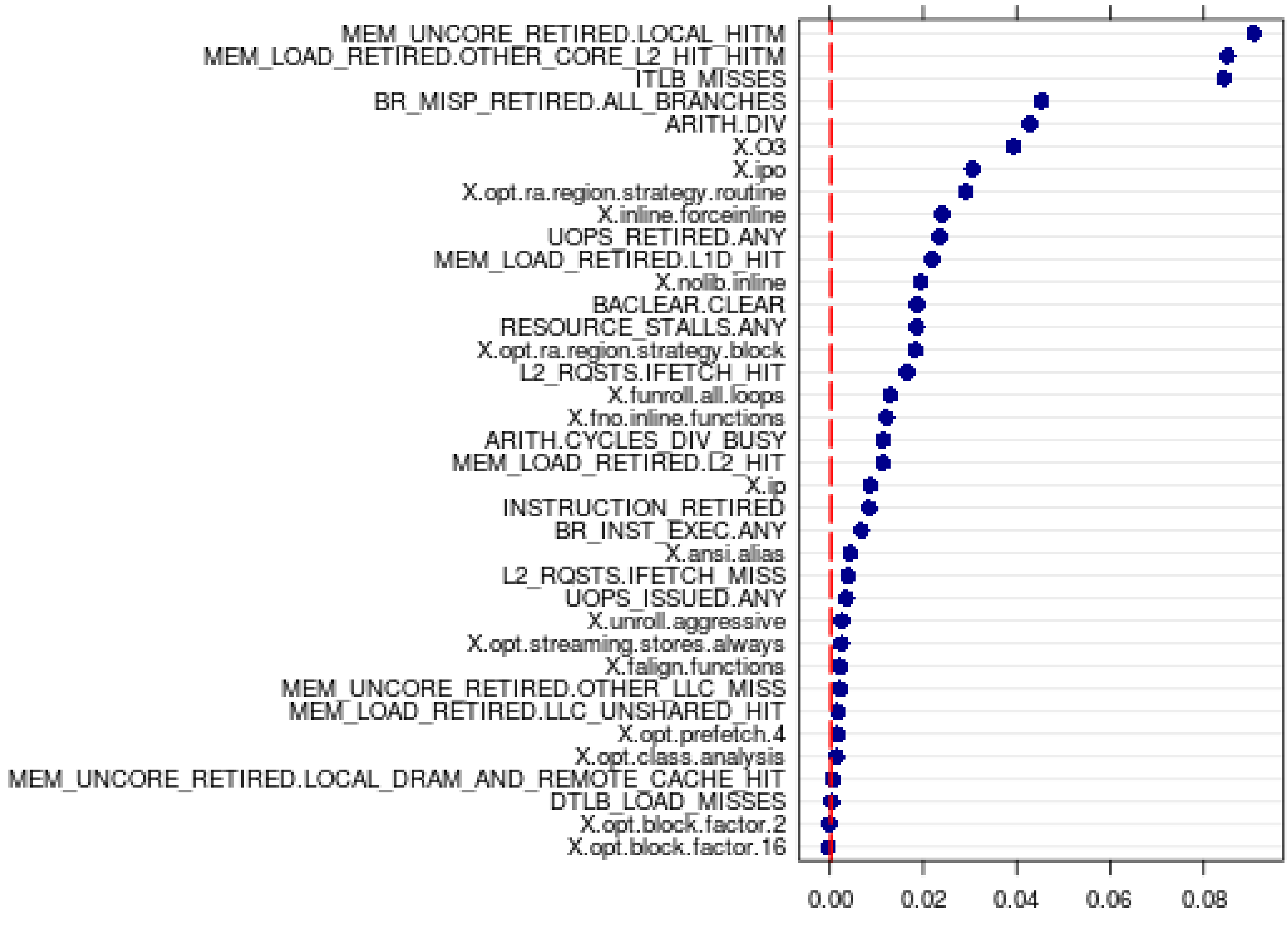


Interpreting measurements



Report published

Interpreting measurements



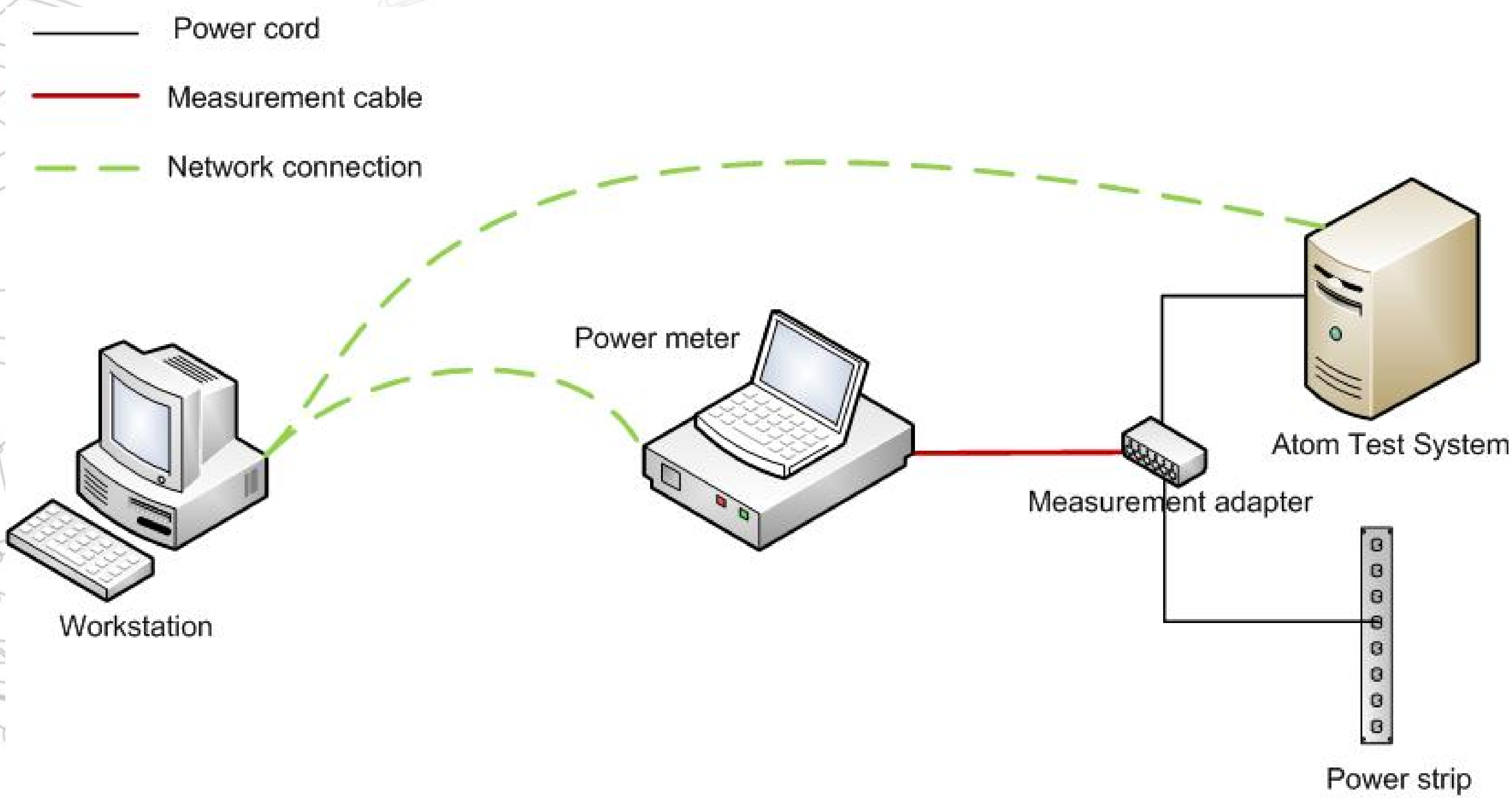
Problem

Power is a challenge in constrained environments

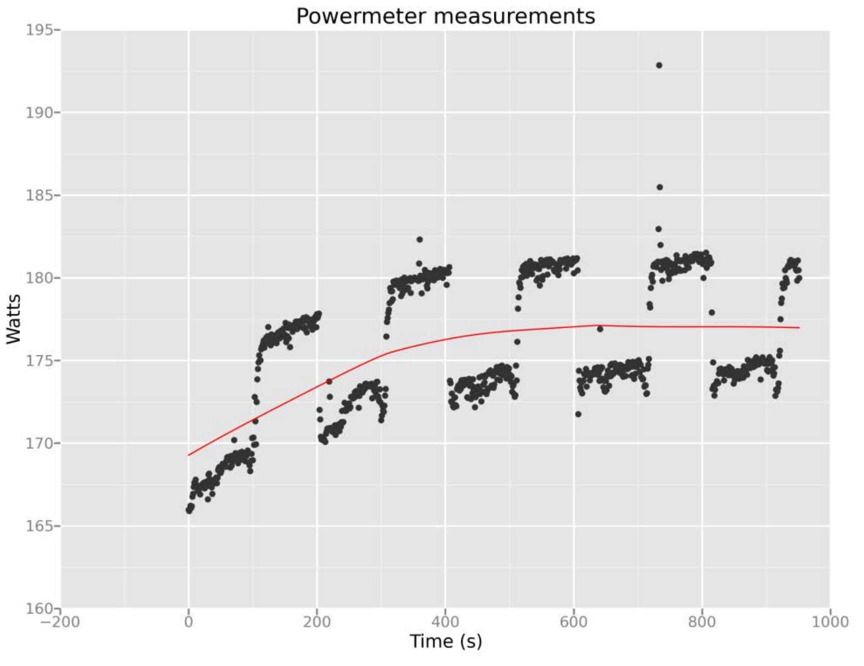
Solution

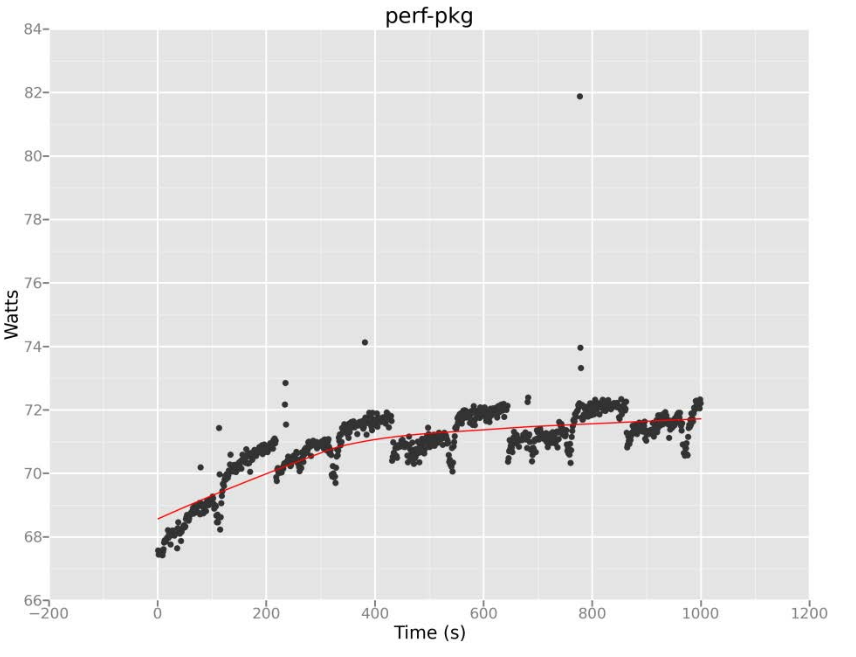
Understand power efficiency and consumption

Throwing power into the mix

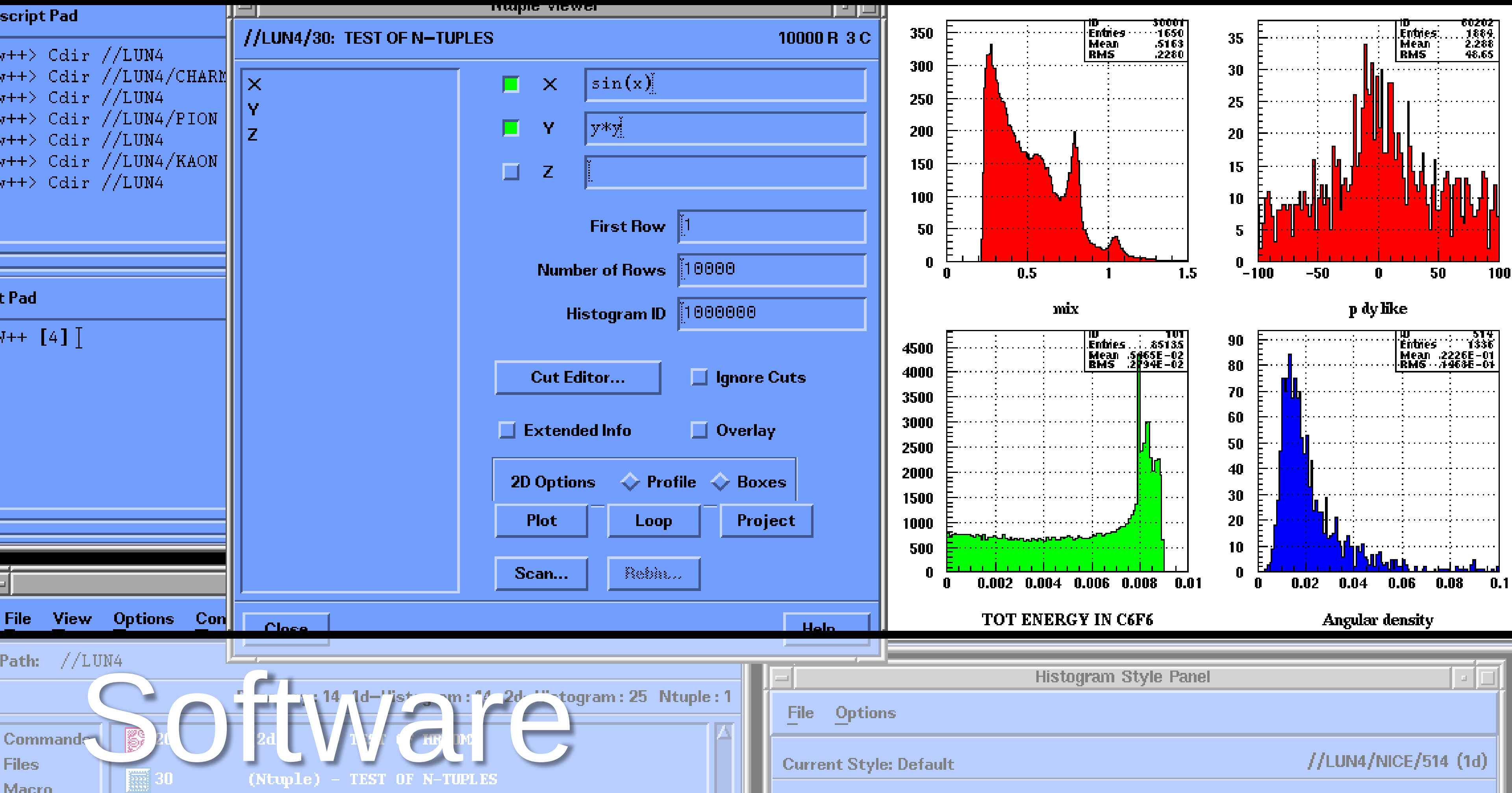


Energy counters





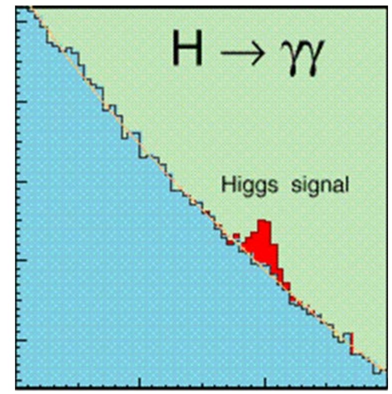
Software



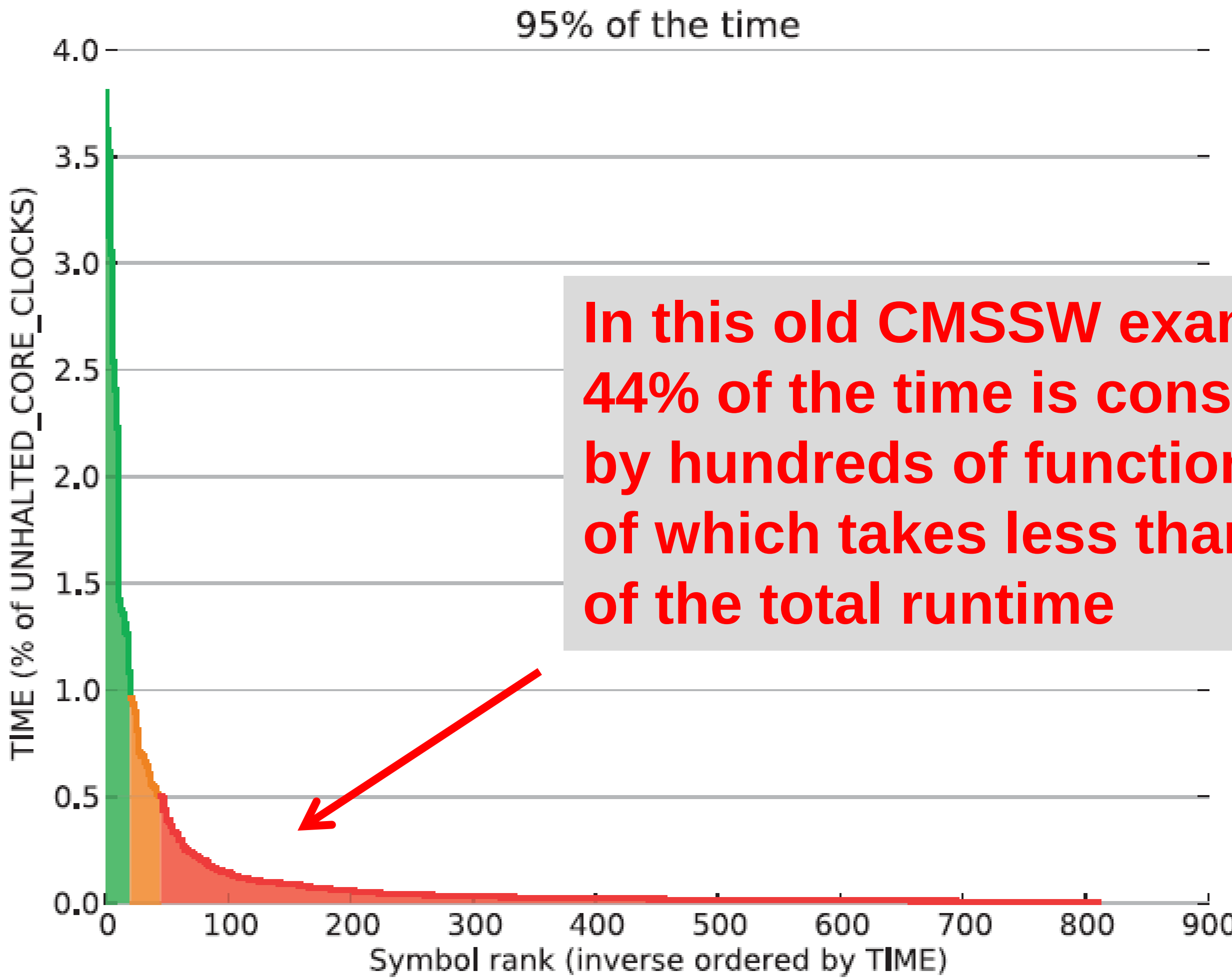
script Pad
Cdir //LUN4
Cdir //LUN4/CHARM
Cdir //LUN4
Cdir //LUN4/PION
Cdir //LUN4
Cdir //LUN4/KAON
Cdir //LUN4
Ntuple Viewer
//LUN4/30: TEST OF N-TUPLES
10000 R 3 C
X
Y
Z
sin(x)
y*y
First Row 1
Number of Rows 10000
Histogram ID 1000000
Cut Editor...
Ignore Cuts
Extended Info
Overlay
2D Options
Profile
Boxes
Plot
Loop
Project
Scan...
Rebin...
Close
Help
File
View
Options
Path: //LUN4
Commands
Files
Macro
(Ntuple) - TEST OF N-TUPLES
mix
p dy like
TOT ENERGY IN C6F6
Angular density
Histogram Style Panel
File
Options
Current Style: Default
//LUN4/NICE/514 (1d)

Software today (1)

- Independent events (collisions of particles)
  - trivial (read: pleasant) parallel processing
  - but single process model
- Bulk of the data is read-only
  - but is not shared
- Very large aggregate requirements:
  - computation, data, input/output
- Chaotic workload
  - research environment - physics extracted by iterative analysis: Unpredictable, Unlimited demand
- Compute power scales with combination of SPECint and SPECfp
  - Good double-precision floating-point (10%-20% of total) is important!
  - Good transcendental math libraries needed
- Key foundation: Linux together with GNU C++ compiler



Software today (2)



In this old CMSSW example, 44% of the time is consumed by hundreds of functions, each of which takes less than 0.5% of the total runtime

From G. Eulisse

Omnipresent parallelism - where were we just recently?

	SIMD	IPC	HW THREADS	CORES	SOCKETS
THEORY	4	4	1.35	8	4
REF	2.5	1.43	1.25	8	2
HEP	1	0.80	1.25	6	2

	SIMD	IPC	HW THREADS	CORES	SOCKETS
THEORY	4	16	21.6	172.8	691.2
REF	2.5	3.57	4.46	35.71	71.43
HEP	1	0.80	1	6	12

HEP = High Energy Physics (in this context: large HEP code)

_%

Write your percentage here

"We're ahead of the game. We've lost nothing since we never had anything."

Compiler project

- Objective: gain deep understanding of modern compiler features and efficiency
- Intel compilers – made available to CERN collaborators in 2008 and informally maintained since (with assistance from IT-PES)
- Close collaboration with the concurrency forum
  - Dozens of bugs reported against ICC
  - Active support of C++11 features

Tools

- Collaborated with Intel on VTune and Inspector tools in the alpha stage
  - “That’s the first external tool that actually works with our code”
- Work on a custom profiler
- Collaboration with HP on perfmon2 (ended)
- Collaboration with Google on the perf tool
  - Contributions to the tool, reports published
  - Analysis of efficiency
- Collaboration with PH on tuning, parallelization, tools
  - Now within the Concurrency Forum
  - Threading Building Blocks a reasonable candidate for parallelization and concurrency

Tools – custom performance analysis with pandas/numpy

Paper ready
(covers only a bit)

Problem

Compilers are not enough. Data analysis in ROOT is single threaded

Solution

Parallelize an example

Case study: parallel data analysis

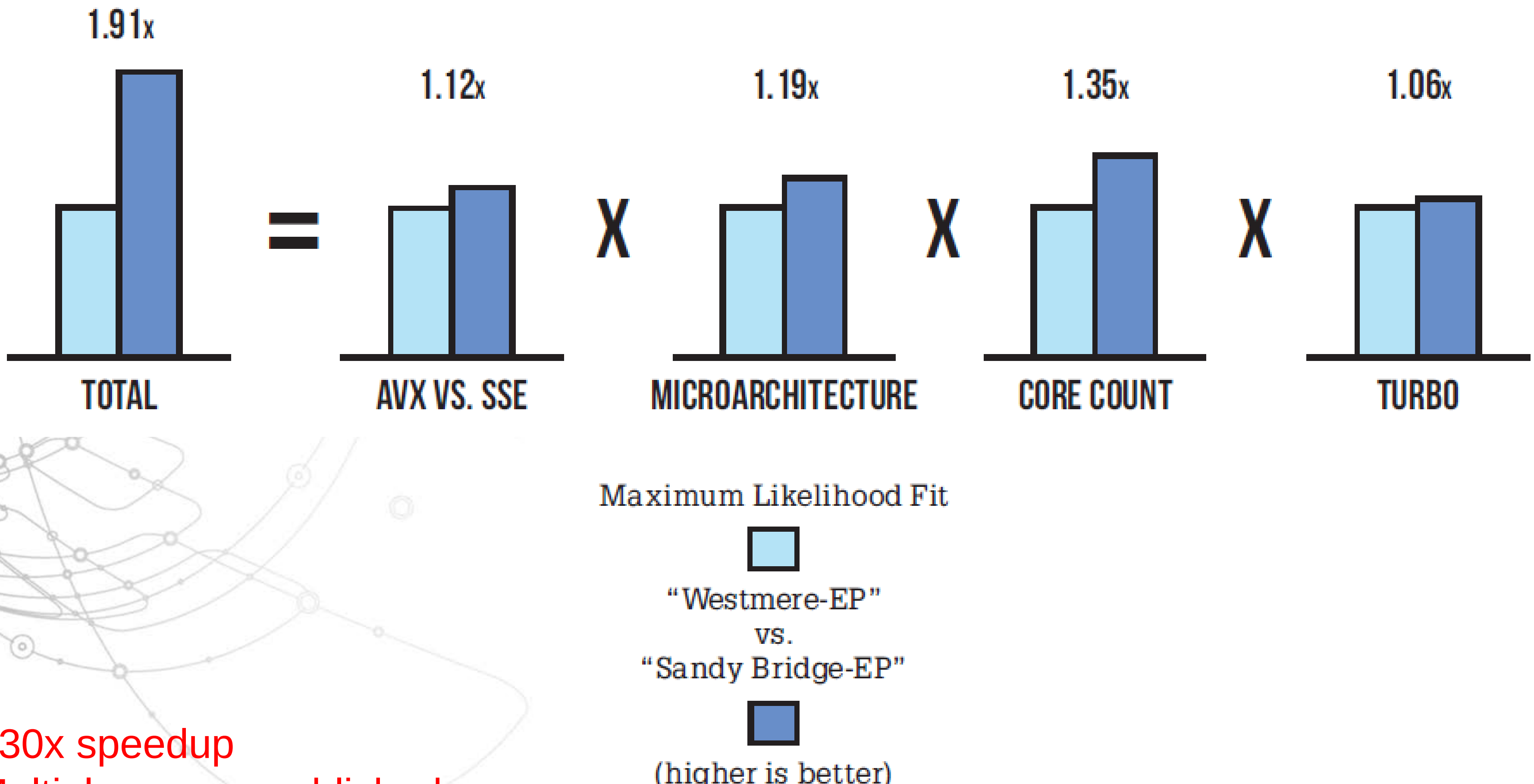
$$NLL = \sum_{j=1}^{s} n_j - \sum_{i=1}^{N} \left[\ln \sum_{j=1}^{s} \left(n_j \prod_{v=1}^{n} \mathcal{P}_j^v (x_i^v | \hat{\theta}_j) \right) \right]$$

N number of events
$\hat{x}_i$ set of observables for the event i
$\hat{\theta}$ set of parameters

n observables
s species
n_j number of events belonging to the species j

From A. Lazzaro

Case study: parallel data analysis



>30x speedup
Multiple papers published

Problem

Geant4 is single-threaded and won't fit in small memories

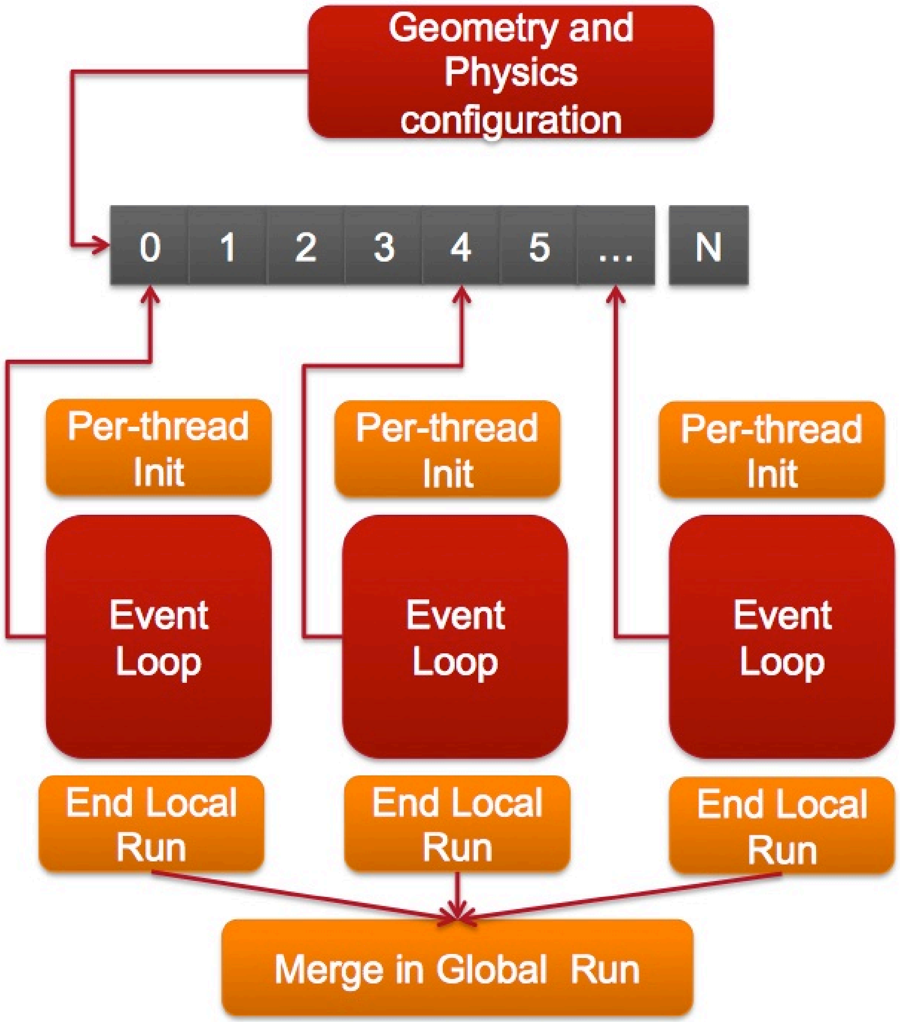
Solution

Parallelize Geant4

(IS THAT EVEN POSSIBLE?)

Case study: multi-threaded simulation

Up to 20x memory savings
Paper co-authored

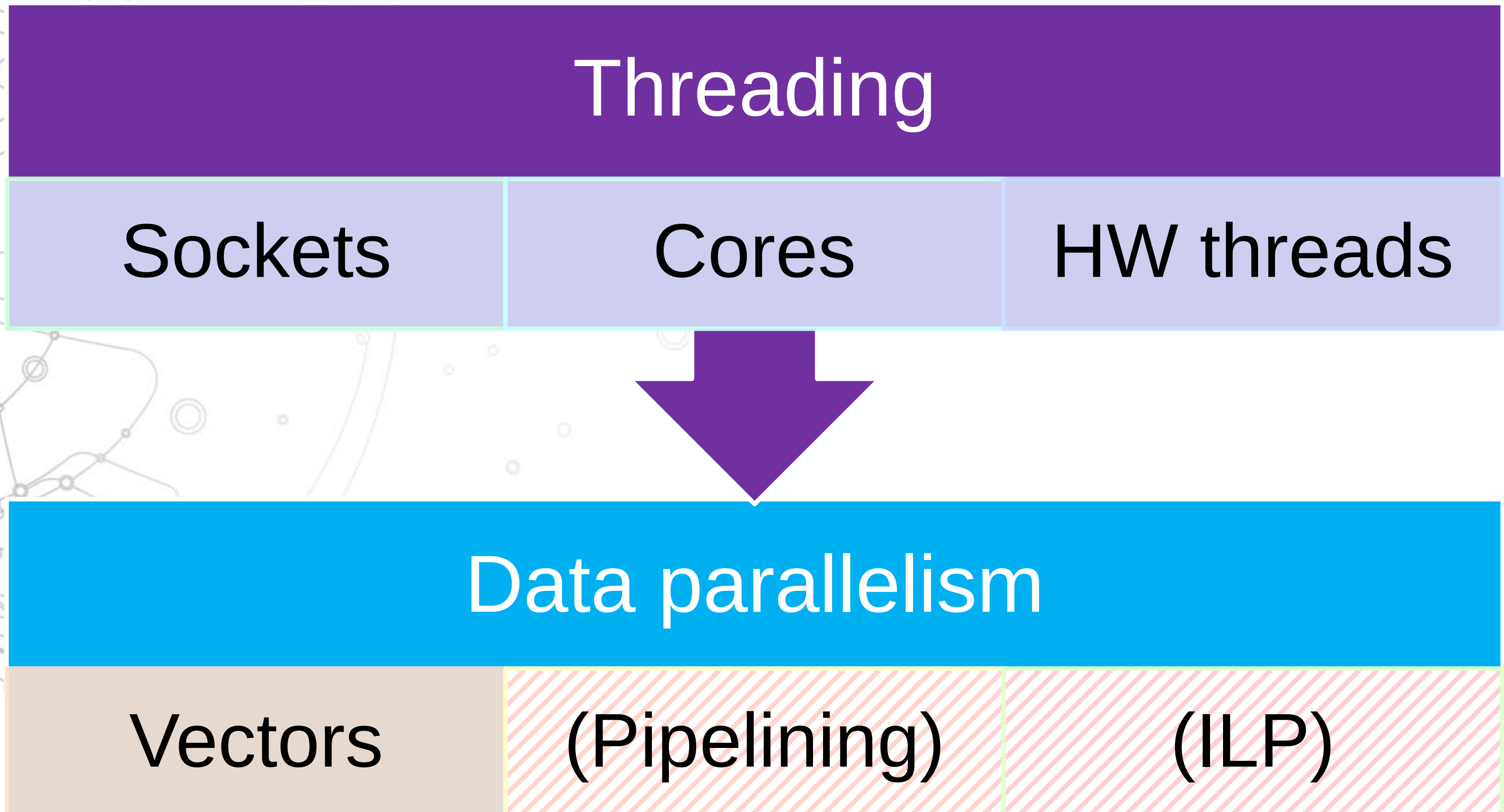


Per-event seeds pre-prepared in a "queue"

Threads compete for next event to be processes (new in ref-08)

Command line scoring and G4tools automatically merge results from threads

From Geant4


Threading
Sockets
Cores
HW threads
Data parallelism
Vectors
(Pipelining)
(ILP)

Problem

Our simulation does not take advantage of vectors

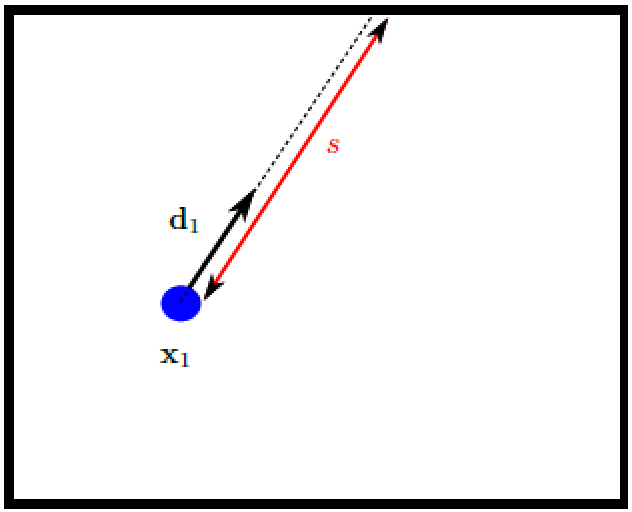
Solution

Look for a way to add data parallelism

Case study: vectorized simulation

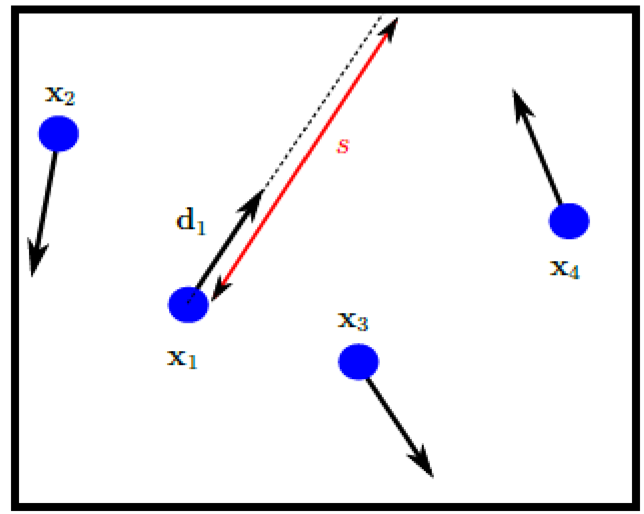
- typical geometry task in particle tracking: **get distance to boundary**

```
double
Box::DistFromInside(double *x, double *d);
```



1 particle

```
void
Box::DistFromInside_v(double *x,
double *d, double *dist, int np);
```



vector of particles

Good speedups achieved

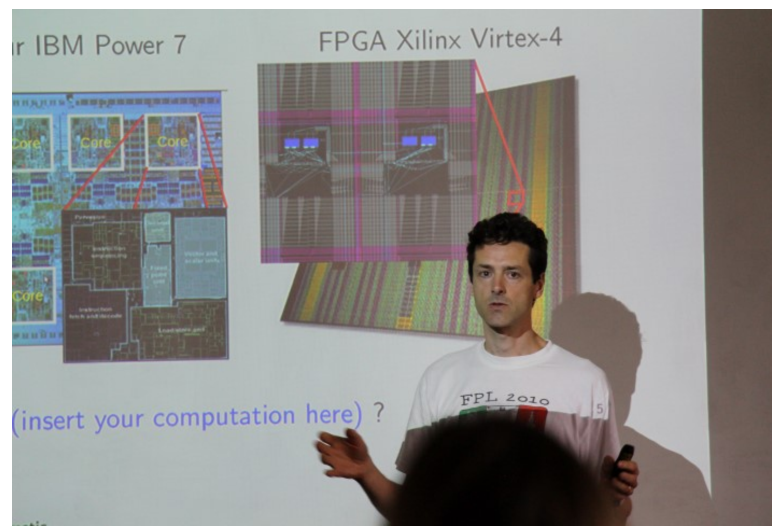
From S. Wenzel

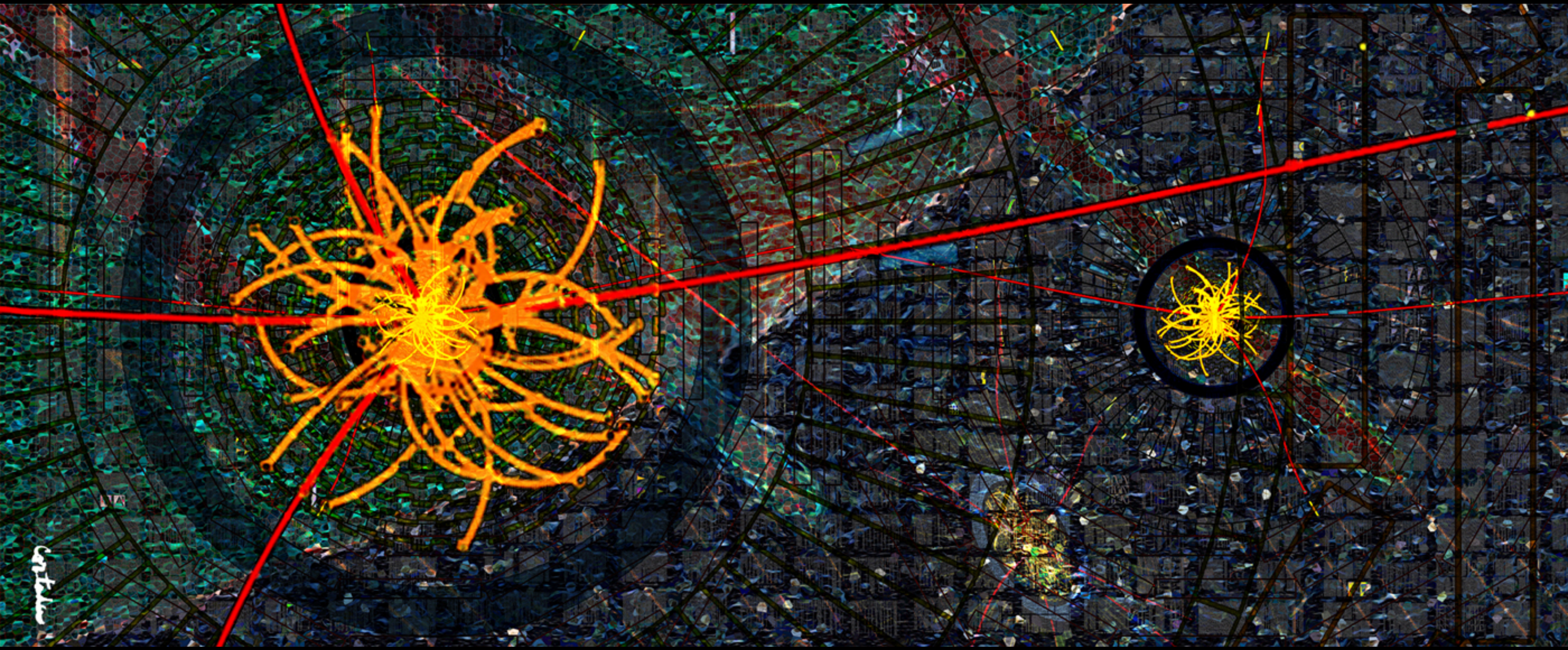
Conclusion – reality check

Level	Potential gains	Estimate
Algorithm	Major	~10x-1000x
Source code	Medium	~1x-10x
Compiler level	Medium-Low	~10%-20% (more possible with autovec or parallelization)
Operating system	Low	~5-20%
Hardware	Medium	~10%-30%

Education

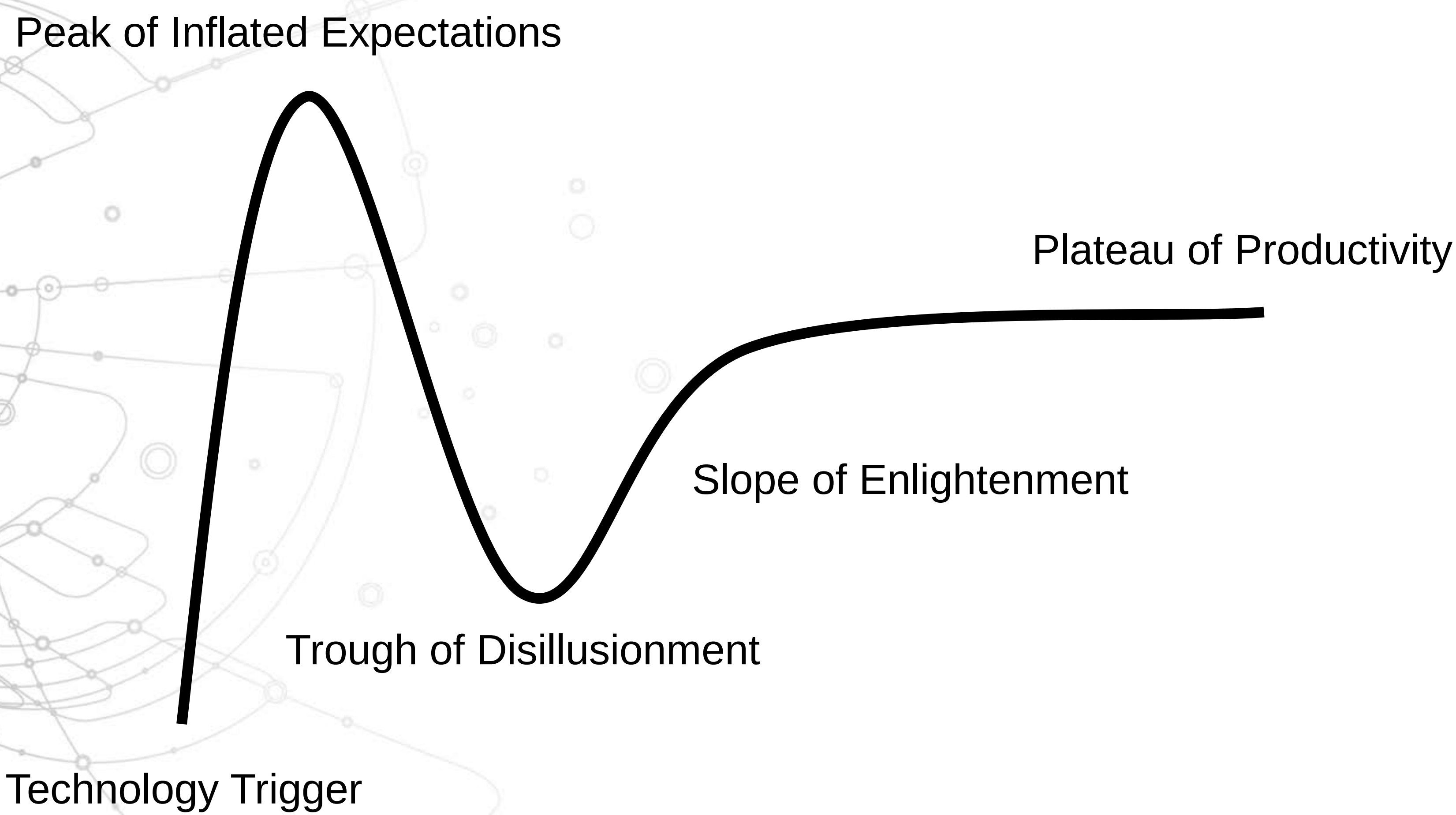
- Our workshops trained over 1’000 people since their inception 7 years ago – also at conferences, universities, schools, private companies
- Special workshops, visits, talks (>10 / year)





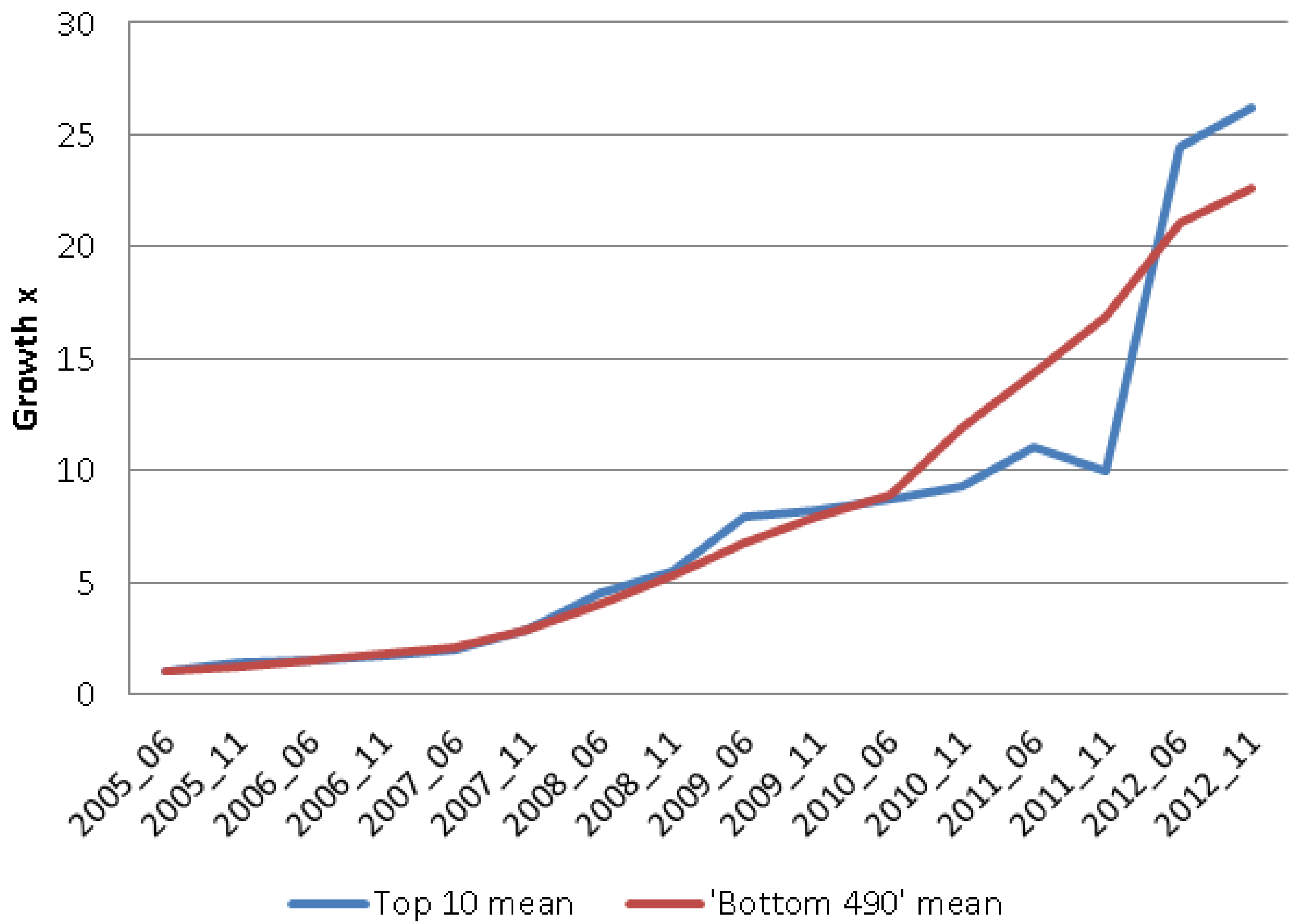
The Future

The Hype Cycle



Modeled after Gartner Inc.

Top500 CPU core count growth



Modeled after Lehto, Manninen, von Alfthan

Heterogeneity

- A whole new set of problems
- How is heterogeneity expressed in hardware?
- How far from one node to another?
- Will the floating point results match?
- How to express heterogeneity in code?
- What coding standards to use? Will code compile anywhere? Will it perform well?
- How to split up the workload?

Paper published

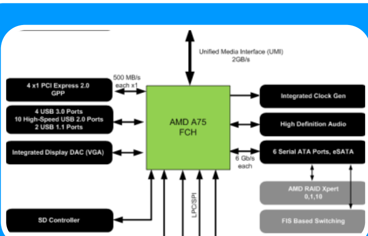
Heterogeneity

Cluster level

- Non-homogeneous nodes
- Large scale, expensive interconnect

Node level

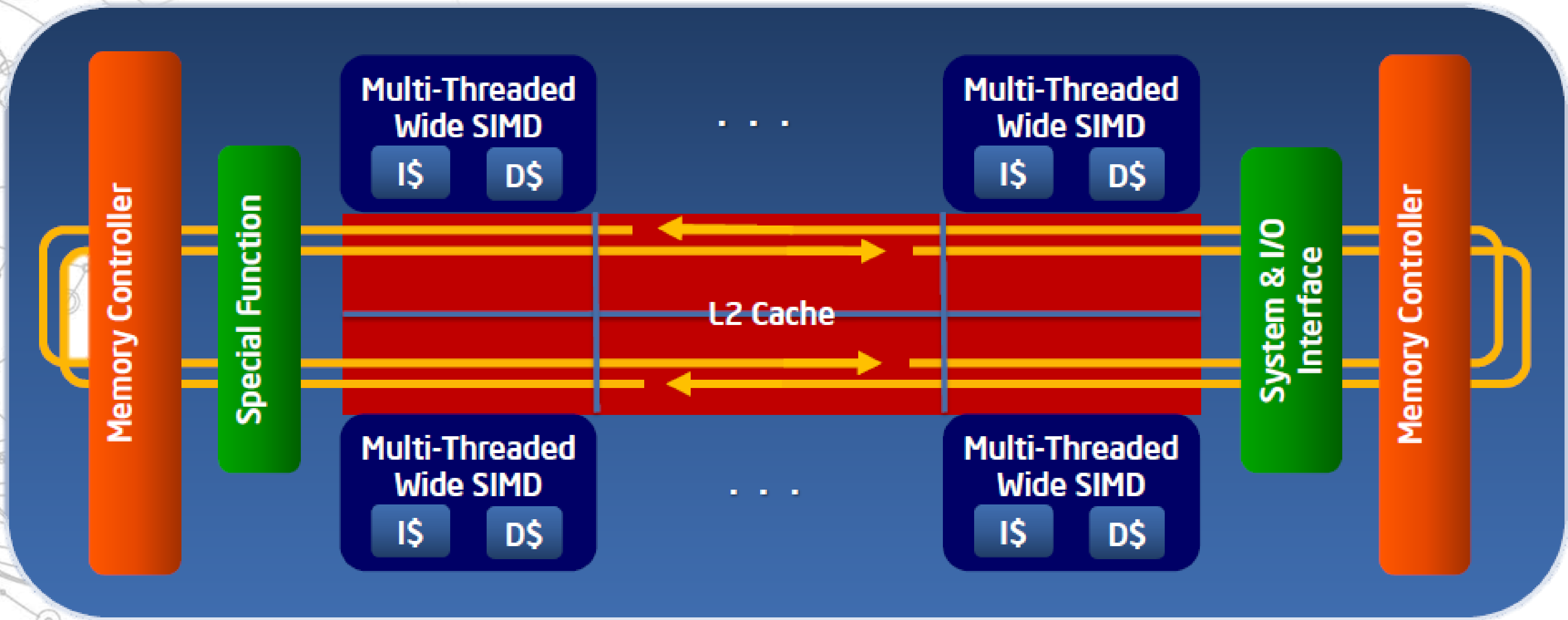
- Non-homogeneous components of a node
- Standard platform interconnect



Chip level

- Non-homogeneous components in a package/chip
- On-chip interconnect or standard bus

Xeon Phi



Source: Intel

Xeon Phi evolution at openlab

Early access
- Work since MIC alpha (under RS-NDA)
- ISA reviews in 2008

Results
- 3 benchmarks ported from Xeon and delivering results: ROOT, Geant4, ALICE HLT trackfitter

Expertise
- Understood and compared with Xeon
- Post-launch dissemination

CERN openlab

Intel MIC programming models

Native mode

workload runs entirely on a co-processor system (networked via PCIe)

Offload

Co-processor as an accelerator where host gets weak

Balanced

Co-processor and host work together

Cluster

application distributed across multiple cards (possibly including host)

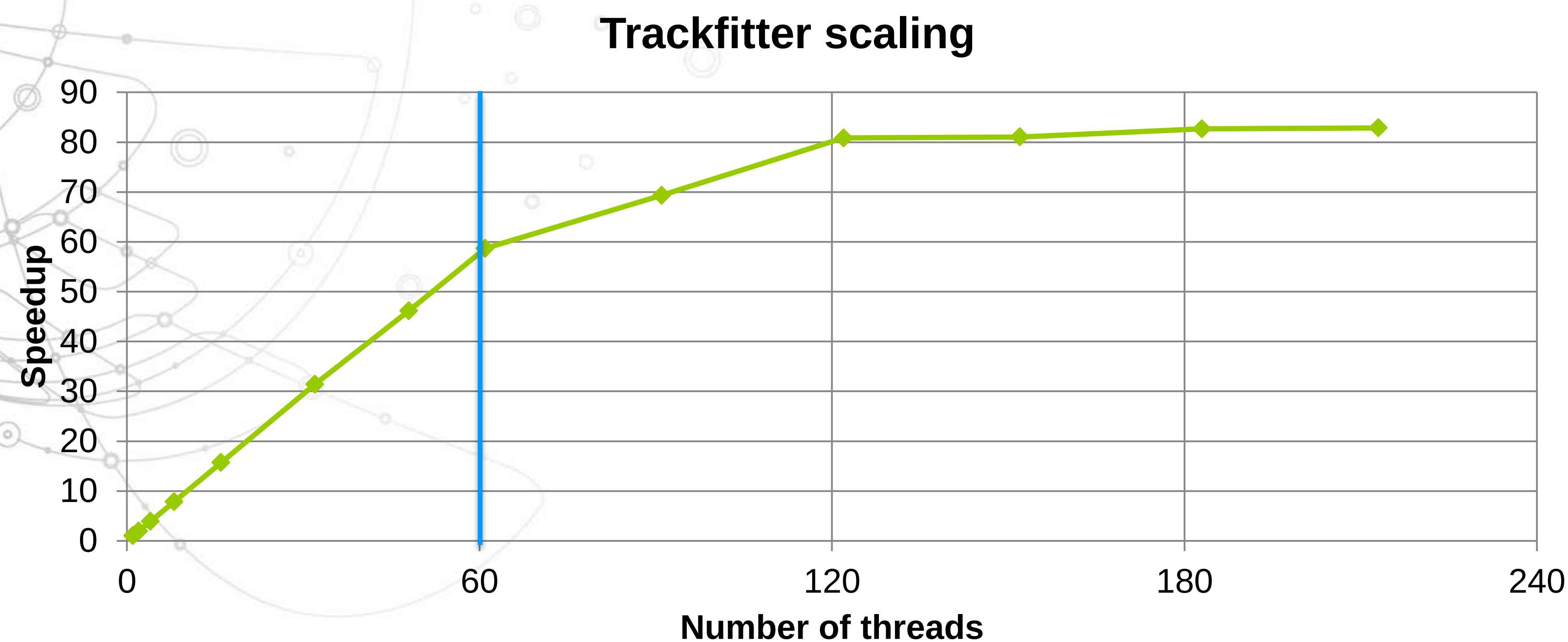
	LOC	1st port time	New ports	Tuning
TF	< 1’000	days	N/A	2 weeks
MLFit	3’000	< 1 day	< 1 day	weeks
MTG	2’000’000	1 month	< 1 day	< 1 week

Paper published

The new ALICE/CBM Trackfitter

preliminary results

x87	singleVc (vec)	OpenMP 213 threads (max perf)	DP scalar to SP vector speedup	OpenMP to SP vector speedup	OpenMP vec to x87 speedup
11.587	0.811	0.0098	14.2x	82.7x	1182x



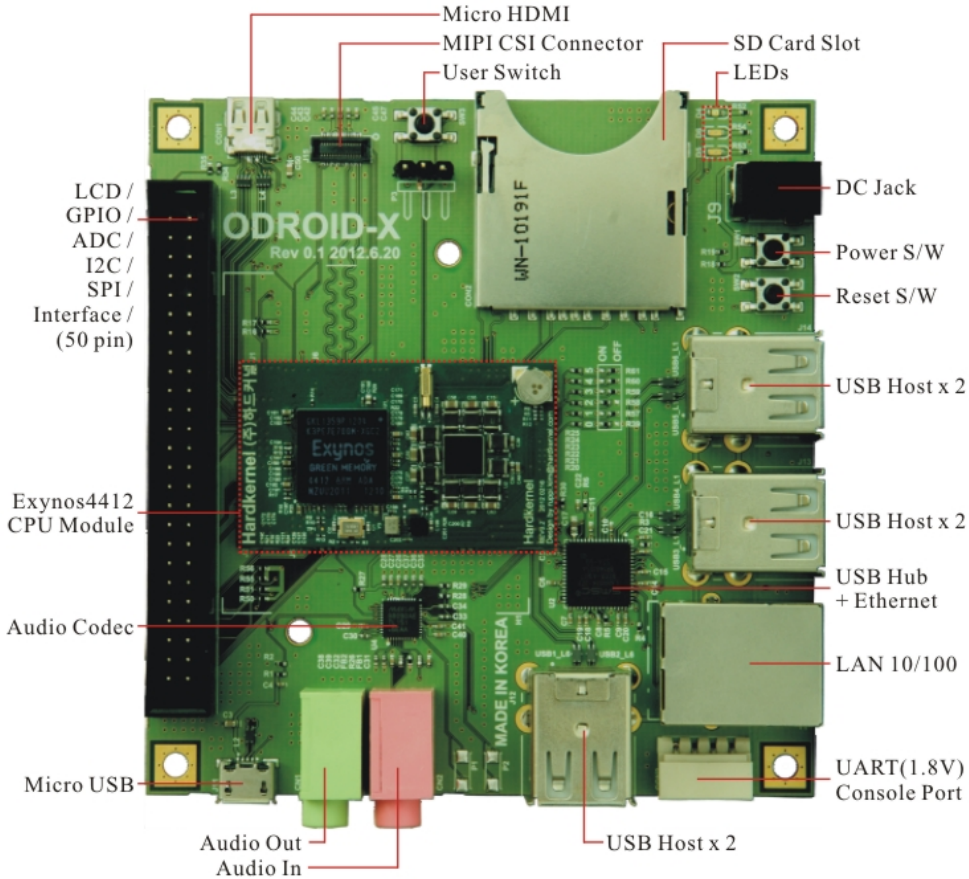
Paper published

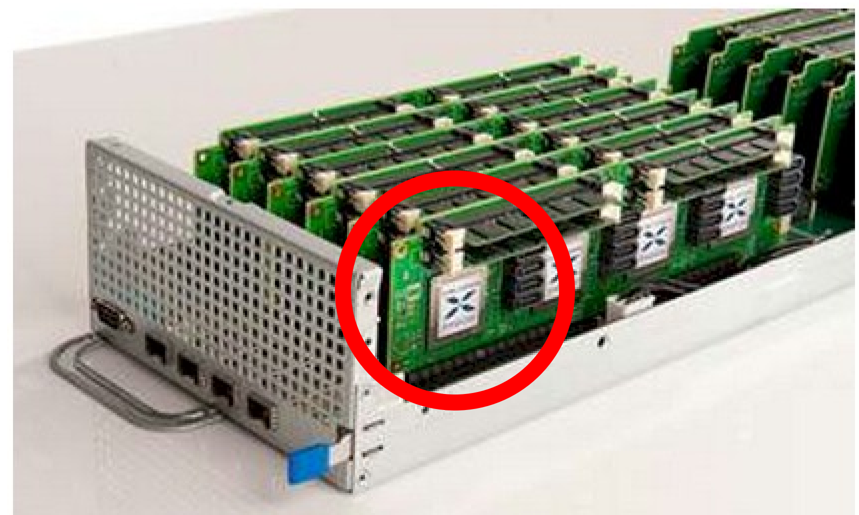
Were we of any help?

- Pre-silicon feedback (Geant4) -> arch. behavior
- System connectivity -> full system
- System integration -> ongoing (KNL)
- Comments on general OS -> Linux
- Math function usage -> better compilers and guidelines
- Documentation -> improved
- Benchmarks -> delivered
- Testimonials -> delivered
- Comments on stack -> ongoing (OSS)
- Many more…

Paper published
Feedback now in thousands of devices

ARM64


Micro HDMI
MIPI CSI Connector
User Switch
SD Card Slot
LEDs
DC Jack
Power S/W
Reset S/W
USB Host x 2
USB Host x 2
USB Hub
+ Ethernet
LAN 10/100
UART(1.8V)
Console Port
USB Host x 2
Audio Out
Audio In
Micro USB
Audio Codec
Exynos4412
CPU Module
LCD /
GPIO /
ADC /
I2C /
SPI /
Interface /
(50 pin)
ODROID-X
Rev 0.1 2012.6.20
MADE IN KOREA



Cool languages and runtimes

`Simple assignments`	`A[:] = 5;`
`Range assignment`	`A[0:7] = 5;`
`Assignment w/ stride`	`A[0:5:2] = 5;`
`Increments`	`A[:] = B[:] + 5;`
`2D arrays`	`C[:][:] = 12;` `C[0:5:2][:] = 12;`

`Function calls`	`func (A[:]);`
	`A[:] = pow(c, B[:])`
	`operators`

`Conditions`	`if (5 == a[:])`
	`    results[:] = „y”`
	`else`
	`    results[:] = „n”`

`Reductions`	`__sec_reduce_mul (A[:])`
`Gather`	`C[:] = A[B[:]]`
`Scatter`	`A[B[:]] = C[:]`

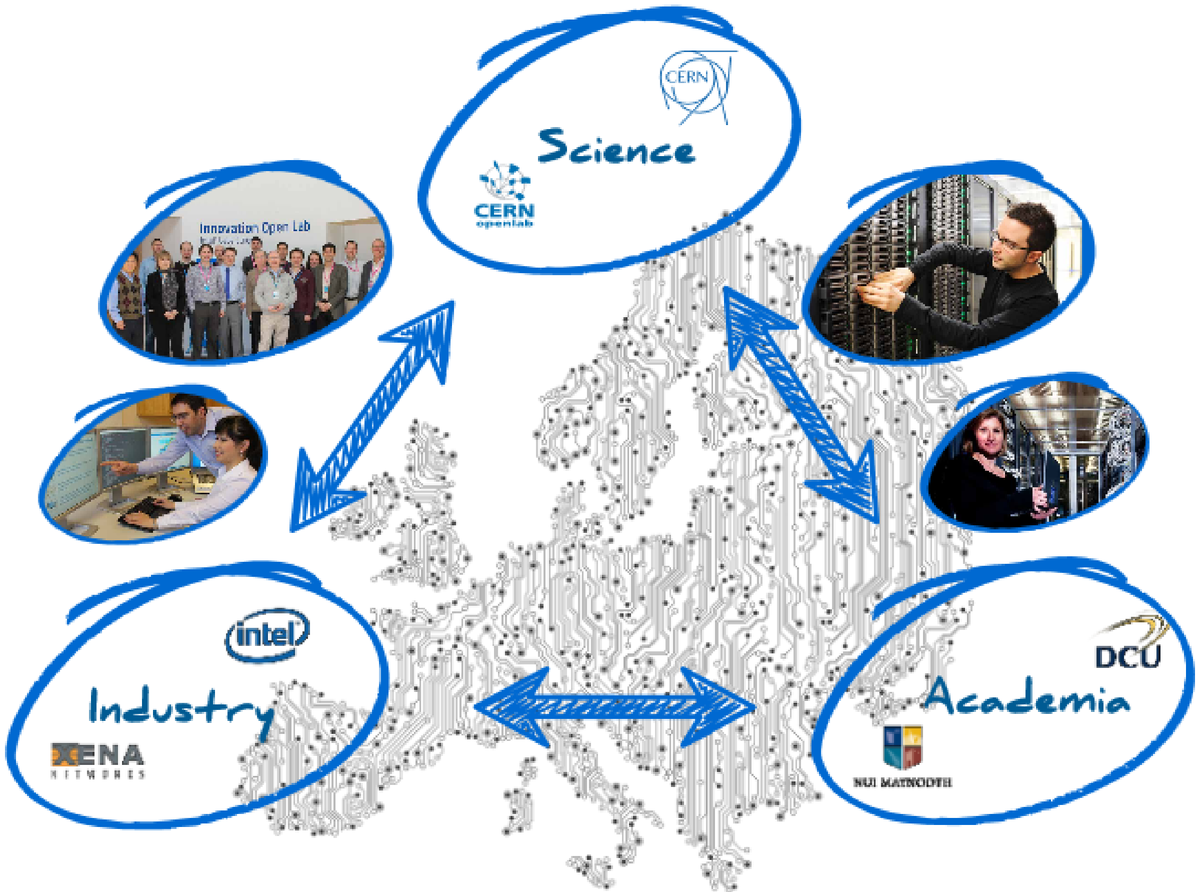
http://cilkplus.org

Report published

ICE-DIP 2013-2017:
The Intel-CERN European Doctorate Industrial Program

A public-private partnership to research solutions for next generation data acquisition networks, offering research training to five Early Stage Researchers in ICT

5 Fellows hired



Research topics:

- Silicon photonics systems
- Next generation data acquisition networks
- High speed configurable logic
- Computing solutions for high performance data filtering

CERN openlab

Q & A

Andrzej.Nowak@cern.ch

Based on the work of G. Balazs, G. Bitzes, M-M. Botezatu, J-J Fumero, S. Jarp, P. Karpinski, A. Lazzaro, A. Nowak, S. Olgunsoylu, A. Santogidis, P. Szostek, L. Valsan

and others