How to get CERN into the TOP500 list

CERN openlab II monthly review
27 February 2007

Andreas Hirstius

Background

- CERN recently purchased a large number of 3GHz Woodcrests
- Each box has a theoretical max. performance of 48 GFlops
  - 4 cores á 12 GFlops (3GHz * 4FP ops per cycle)
  - ~30000 GFlops theor. max with all delivered machines
- To enter the next TOP500 list we would need ~3500-4000 GFlops
  - a relatively small efficiency should be sufficient to enter the list
- So our motivation was: We could, so why not try it 
- BUT:
  - Parallel applications are not very common at CERN
  - ... pretty much no experience with MPI (software used for parallelization)

The benchmark

- The standard benchmark used is HPL – High Performance Linpack
  - the software solves a linear system of order n: $\mathcal{A}x = b$
  - a matrix size N is chosen according to the available memory
  - the available cores are arranged into a P-by-Q grid of processes
    - P and Q largely control the load balance ≙ performance
  - the actual work is distributed in NB-by-NB sized blocks
    - the choice of NB has also significant influence on performance
  - 14 more parameters that can be used for fine tuning
    - those parameters are far less important
  - Values / Examples come later…

What does a cluster usually looks like

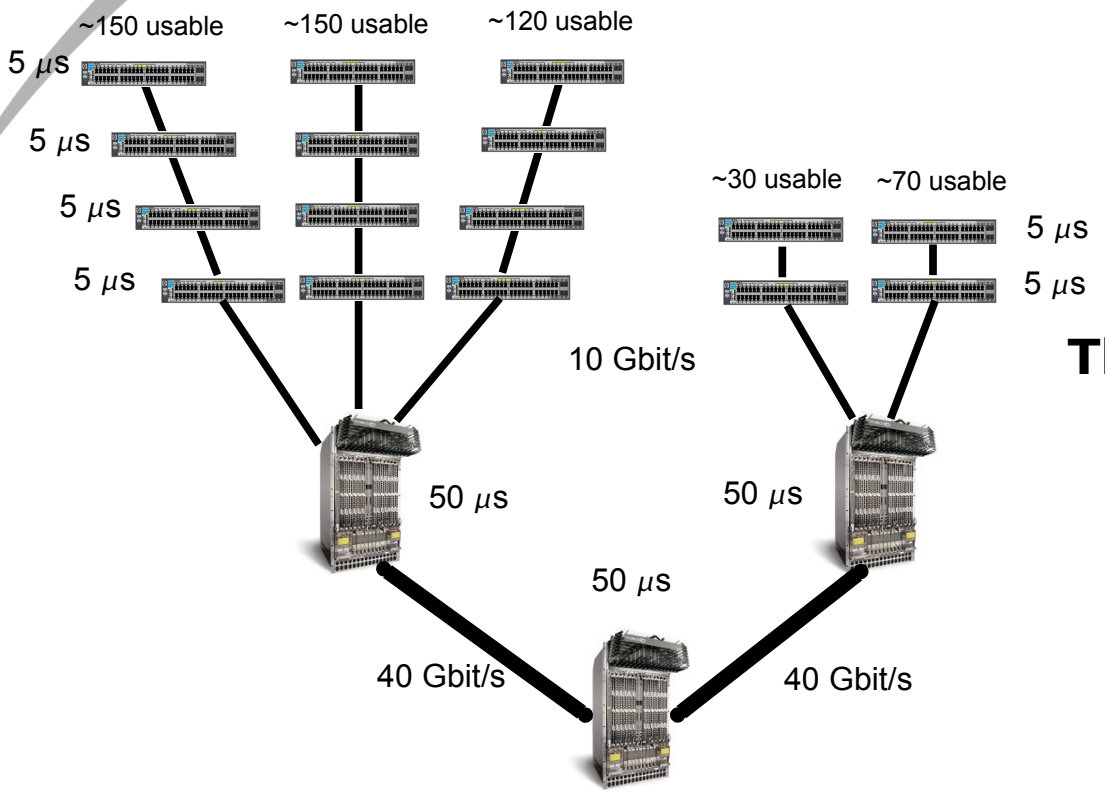
- Large/Huge multiprocessor machines with proprietary interconnect
  - Blue Gene (#1 in the list has 131072 processors!!!)
  - Cray
  - Altix
- Large number of small multiprocessor machines with fast interconnect
  - InfiniBand, Quadrics, Myrinet (latency $\mathcal{O}(\mu s)$)
- Ethernet based clusters have usually a specialised network setups
  - using switches with very low latency ... overall $\mathcal{O}(10\ \mu s)$
  - 43% of the systems, but only 22% of total performance

The most important thing for a cluster:

The interconnect has a very low latency in the order of a few μs
(Ethernet based cluster have larger latency ... and lower efficiency)

Our setup has latencies up to 600 μs !!
That's an eternity for a parallel job...

The setup - hardware



The Machines:

- about 530 machines available
  - three different vendors
  - 3GHz Woodcrest
  - 8GB RAM
  - 1Gb NICs

The Network setup:

- Edge: HP ProCurve 3500yl
  (3400cl for machines from one vendor)
  - delay per switch: 5 µs
- Core: Force10 E1200
  - delay per router: 50 µs !!

The setup – software

- The machines were installed with the std. CERN setup
  - SLC4 for x86_64
  - all daemons running, incl. monitoring
    - considered “very bad” for HPL performance
  - Job submission was using LSF
    - special queue was installed
    - usually a single user can not submit so many jobs
  - NO special tuning at all !
- Intel MPI
- Intel MKL (Math Kernel Library)
- High Performance Linpack (HPL)

How it started...

- Initial tests were started with ~260 machines
  - get the setup up and running
    - MPI and HPL
    - setup LSF
  - get familiar with the software and the parameters
  - test scalability up to 256 machines (1024 cores)
    - we were unsure about the scalability in our environment
      ... remember our latencies are about a factor 100 larger than in a “normal” cluster
- The results were very promising, so more machines were made available

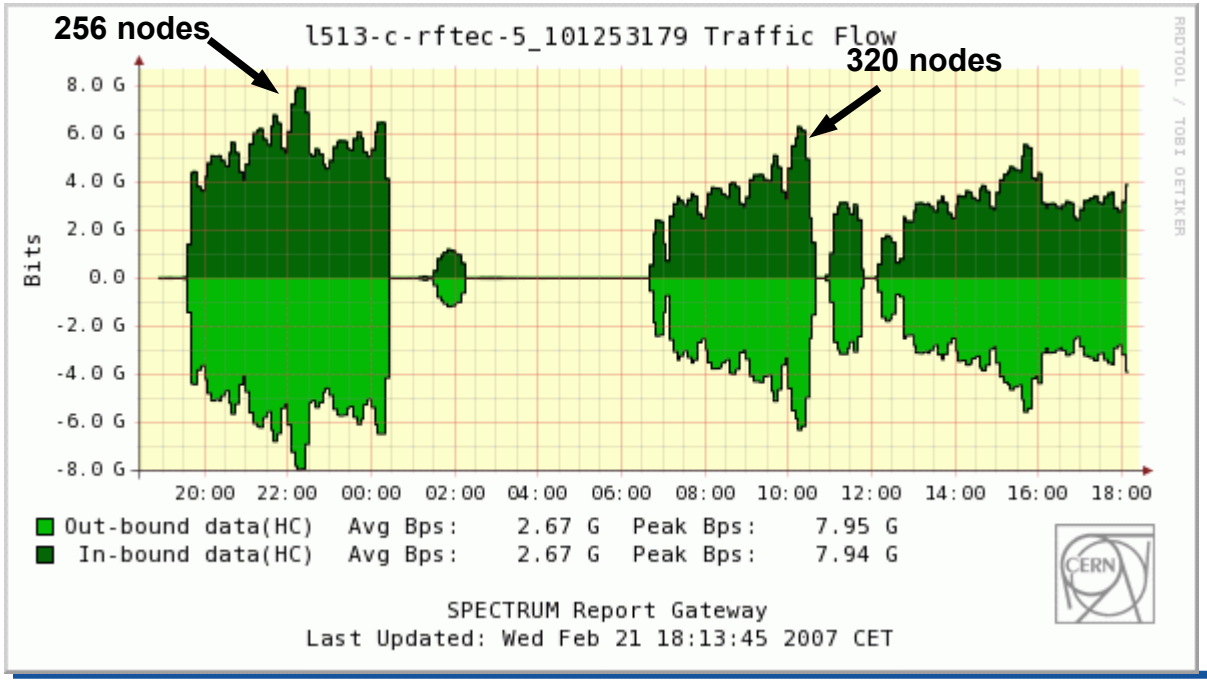
number of cores	4	8	16	64	256	1024
GFlops	35.9	67.2	119	435.2	1735	6227
rel. increase in #cores	1	2	2	4	4	4
rel. increase in Gflops	1	1.87	1.77	3.66	3.99	3.59
scaling factor	1	0.94	0.89	0.91	1.00	0.90
efficiency (theor/meas.)	0.75	0.7	0.62	0.57	0.56	0.51

... and continued ...

- ~530 machines (2120 cores) were available
- for about four days no successful run with more than 256 machines (1024 cores) 
- MPI crashed when it tried to establish all necessary communication channels
- Intensive debugging at CERN and by Intel
- The problem could be traced to the batch of machines delivered by one of the vendors
  - the machines which are connected to the HP ProCurve 3400cl switches
    - we think that the driver for the NIC could be the problem
    - ... or the switches ...
    - ... or both.

... and went on.

- only about 340 machines remained available
- but even now the runs were very unstable and slow
- it turned out that communication was again the problem
  - LSF assignes the machines “randomly”
  - since we are limited by our connectivity this is dangerous
  - machines had to be carefully ordered ...



- The traffic via the 10Gbit uplink to the router is the limiting factor
- “unordered” nodes for a run with 1024 cores run at ~8Gbit/s
- An “ordered” list of 320 nodes (1280 cores) run at only ~6.1Gbit/s

The impact of communication

The inital computations do not require so much communication...

```
Column=002704 Fraction=0.005 Mflops=9448558.39
Column=005304 Fraction=0.010 Mflops=9860783.61
Column=008008 Fraction=0.015 Mflops=10003344.26
Column=010608 Fraction=0.020 Mflops=9985213.33
Column=013312 Fraction=0.025 Mflops=10056021.72
```

... but the communications at the end of the run have a significant impact on overall performance

```
Column=315432 Fraction=0.595 Mflops=8677419.04
Column=368368 Fraction=0.695 Mflops=8582679.81
Column=421408 Fraction=0.795 Mflops=8486964.16
Column=474448 Fraction=0.895 Mflops=8399663.57
Column=527384 Fraction=0.995 Mflops=8335859.30
```

Tuning the parameters – the Voodoo

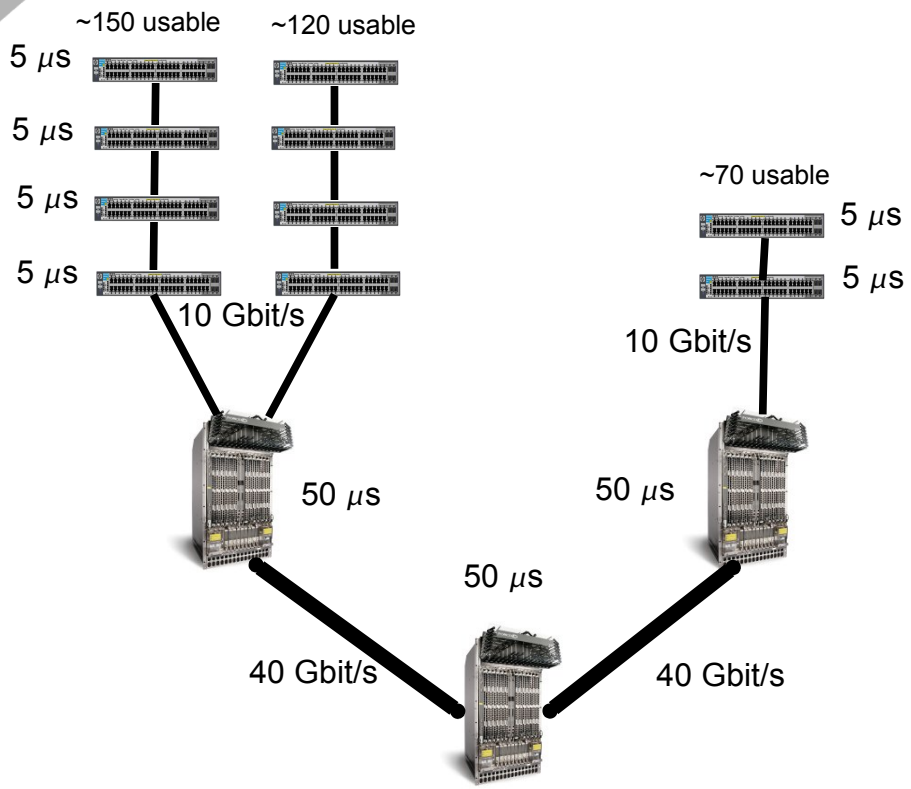
Initial tuning was done with 1024 cores ...

The parameters	GFlops
N: 445000; NB: 104; P: 32; Q: 32	4471
N: 445000; NB: 104; P: 16; Q: 64	5934
N: 445000; NB: 104; P: 8; Q: 128	5142
N: 445000; NB: 96; P: 16; Q: 64	4840
N: 455000; NB: 128; P: 16; Q: 64	6164
N: *460000*; NB: *128*; P: 16; Q: 64	**6227**

... but with 1360 cores everything was different

The parameters	GFlops
N: 530000; NB: 128; P: 16; Q: 85	8209
N: 530000; NB: 104; P: 20; Q: 68	8198
N: *530000*; NB: *104*; P: 16; Q: 85	**8329**
N: 540000; NB: 104; P: 16; Q: 85	7940
N: 530000; NB: 104; P: 20; Q: 68	8042

The final setup and result



- 340 machines

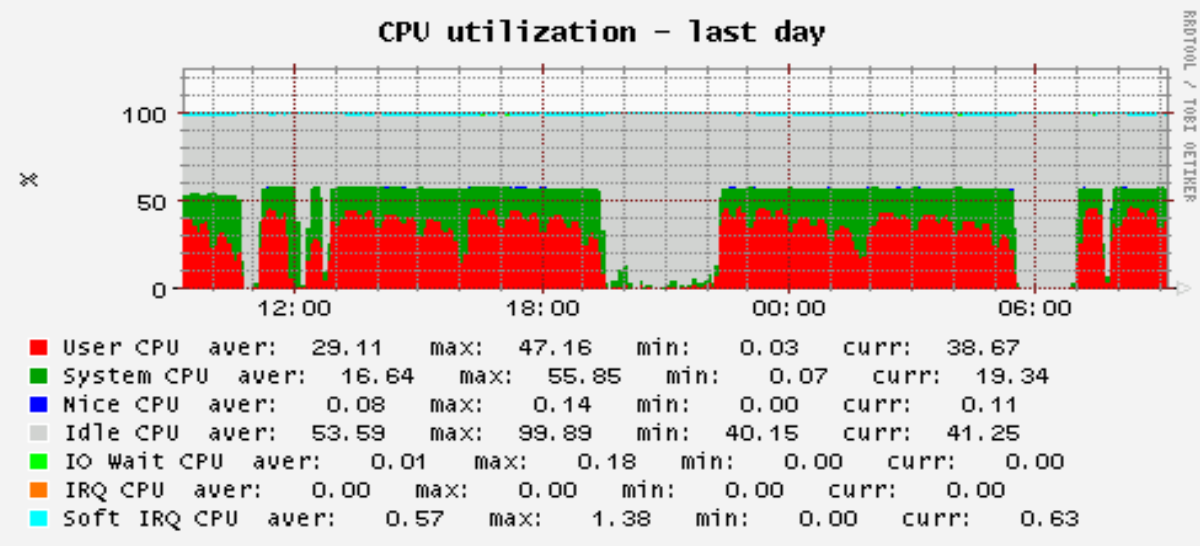
  ➔ 1360 cores

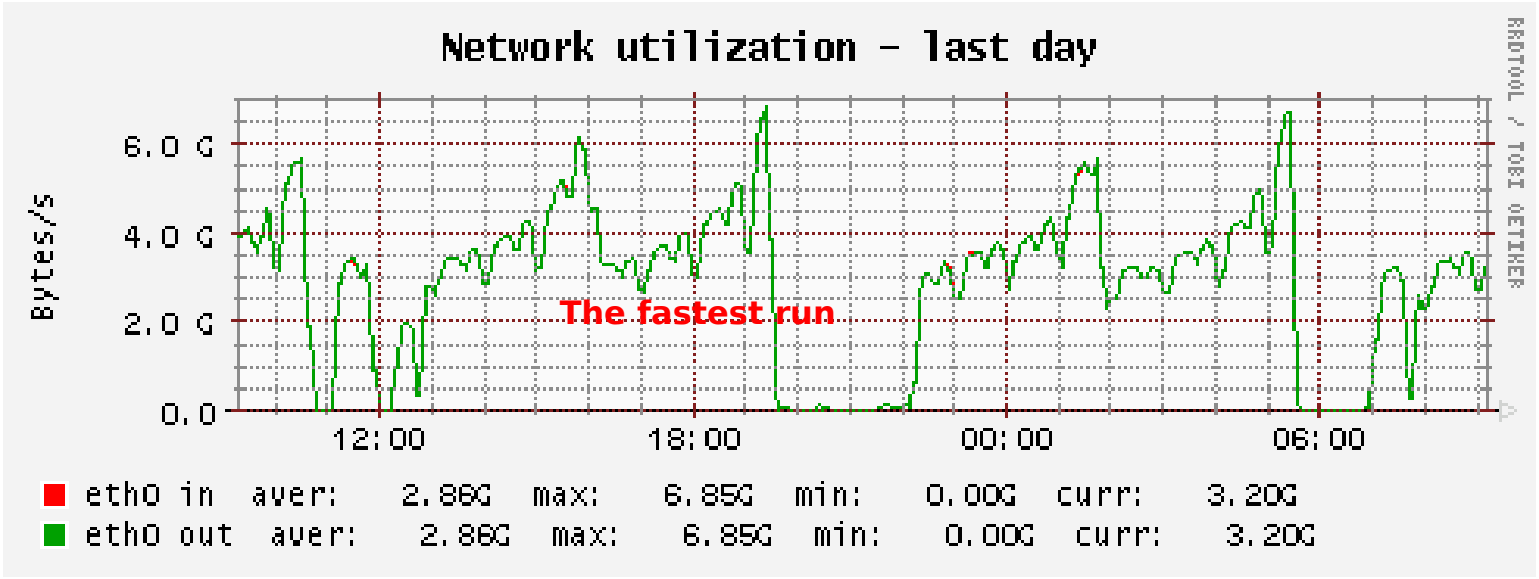
... delivering:

8329 GFlops

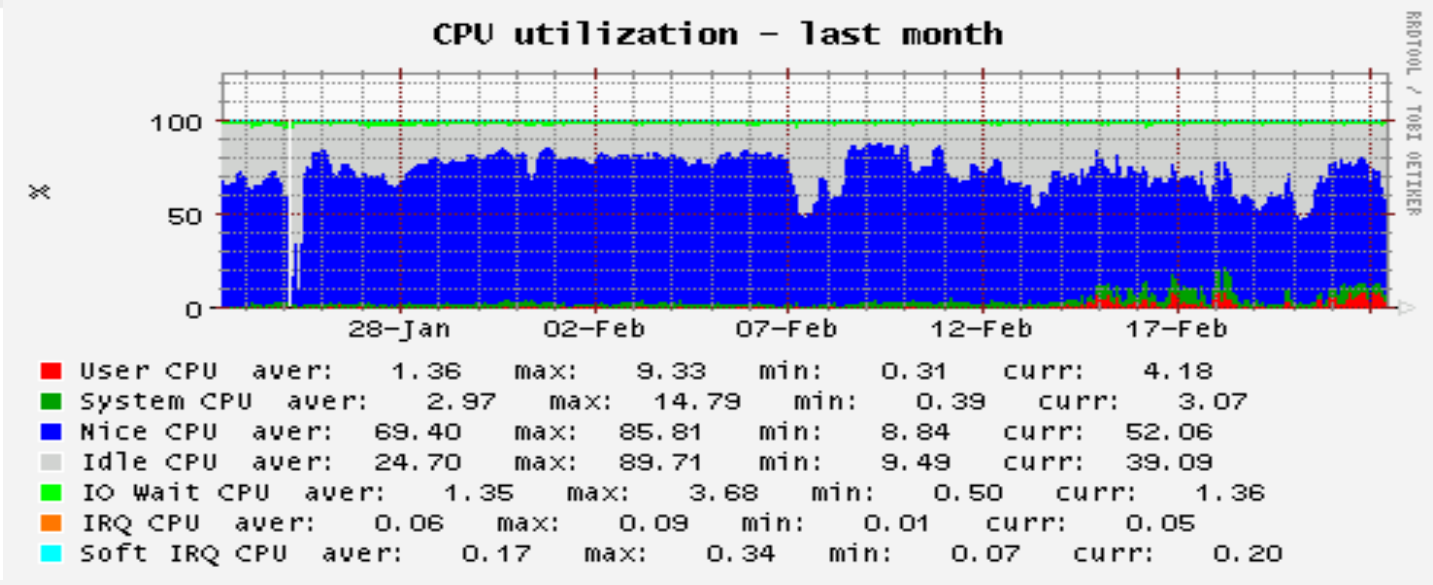
- 6.12 GFlops per core
- 51% efficiency !

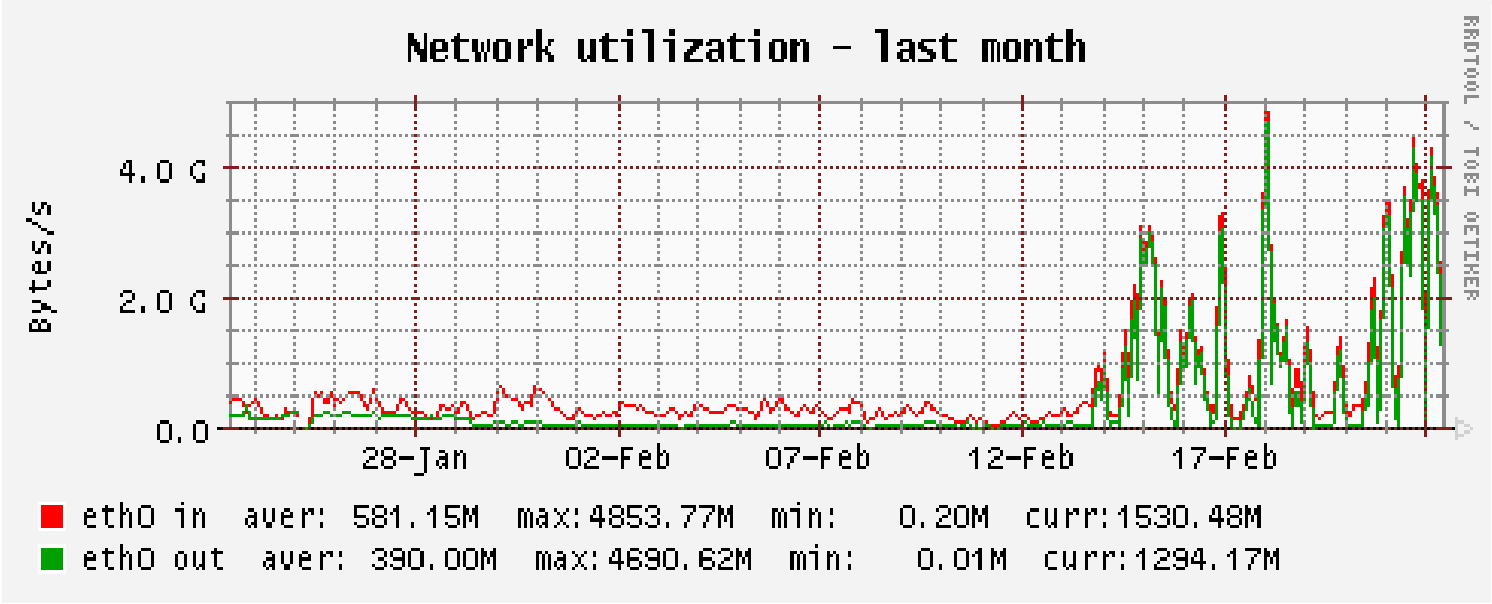
… how it looks in the monitoring





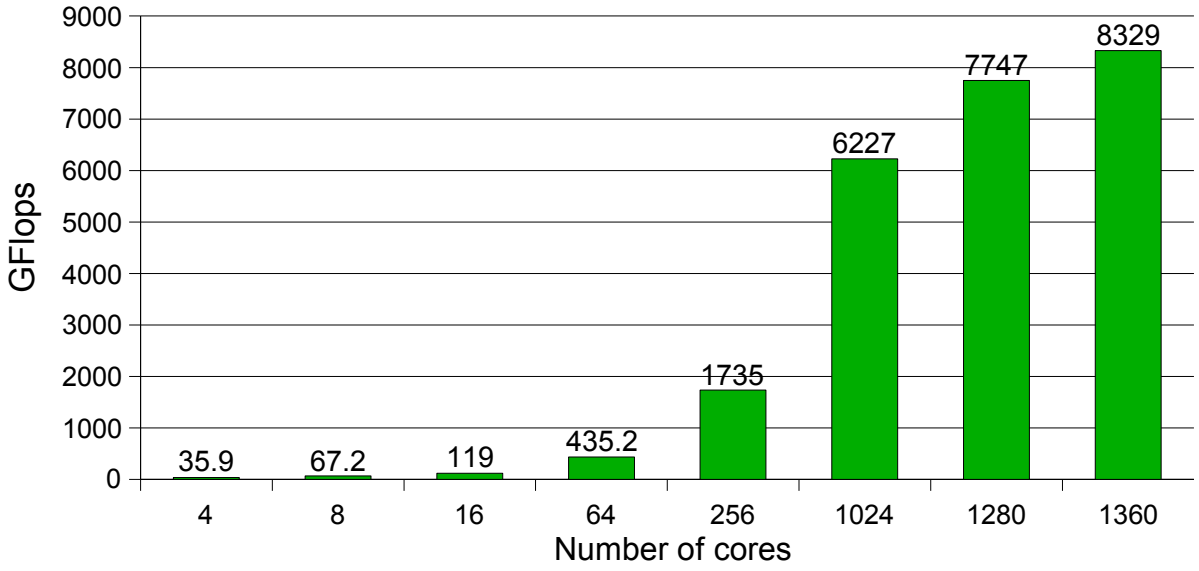
… compared to the entire lxbatch farm





Summary of results

number of cores	4	8	16	64	256	1024	1280	1360
GFlops	35.9	67.2	119	435.2	1735	6227	7747	8329
rel. increase in #cores	1	2	2	4	4	4	1.25	1.06
rel. increase in GF	1	1.87	1.77	3.66	3.99	3.59	1.24	1.08
scaling	1	0.94	0.89	0.91	1.00	0.90	1.00	1.01
theoretical max.	48	96	192	768	3072	12288	15360	16320
efficiency (theor./real)	0.75	0.70	0.62	0.57	0.56	0.51	0.50	0.51



Only the run with 1360 cores is optimised!
(at least as much as possible in the available timeframe)

Conclusions

CERN IT achieved a remarkable performance with High Performance Linpack and Intel MPI

8329 GFlops with 1360 cores

(6.12 GFlops per core ≙ 51% efficiency)

- setup not optimised (h/w or s/w wise)
- for our type of (network) setup extremely good result
  - other GigE based clusters: 19 - 67 % efficiency
- would be rank #79 in current list
- being submitted to the TOP500 committee
  (as soon as the submission webpage is online again)
- HPL is extremely sensitive to it's parameters ...

Acknowledgements

- Intel:
  - Sergey Shalnov
  - Intel MPI people
- CERN
  - Ulrich Schwickerath
    - LSF and general help (debugging, etc.)
  - Veronique Lefebure
    - installation of machines
  - Nick Garfield (and others in CS group)
    - looking into the network setup