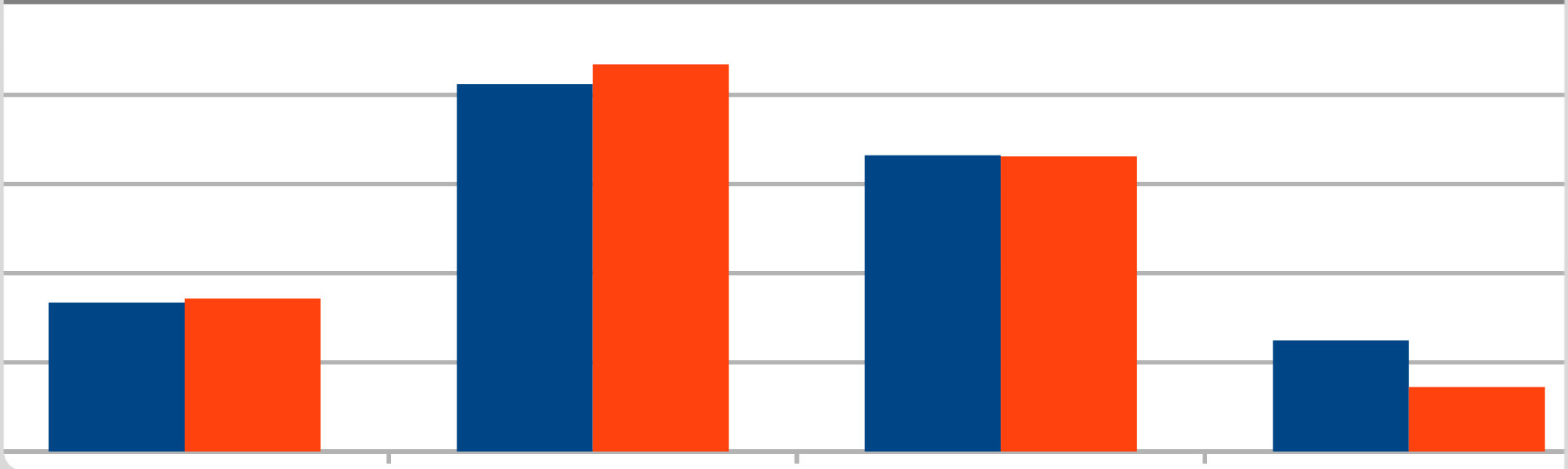
Karlsruhe Institute of Technology

Benchmarking by Monitoring
2017-05-05

Manfred Alef

STEINBUCH CENTRE FOR COMPUTING (SCC)



Still Open Question: Influence of Number of Job Slots

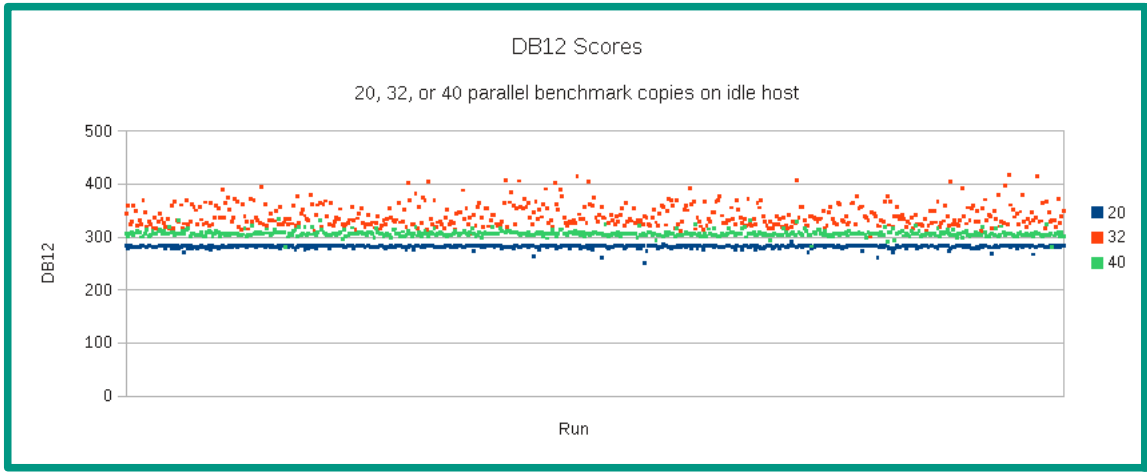
- No common strategy by sites about the number of job slots per WN
  - → #job_slots == #physical_cores (HT on or off)
  - → #physical_cores < #job_slots < #logical_processors (HT enabled)
  - → #job_slots == #logical_processors (HT enabled)
- Undisclosed overcommitment of cloud IaaS hypervisors (see the examples in previous talks by Domenico Giordano)

Still Open Question: Influence of Number of Job Slots

- Impact of the number of configured job slots per physical core on the performance?
  - ➔ HS06:
    - ≈ 17% increase in performance in case #job_slots ≈ 1.5 * #cores
    - ≈ 25% increase if #job_slots = 2 * #cores = #logical_cores
  - ➔ DB12-at-boot:
    - Obviously no increase in performance if #job_slots > #cores
  - ➔ What about the corresponding application performance (in units of events/s)?

Still Open Question: Influence of Number of Job Slots

- Impact of the number of configured job slots per physical core on the performance?
  - ➔ HS06:
    - ≈ 17% increase in performance in case #job_slots ≈ 1.5 * #cores
    - ≈ 25% increase if #job_slots = 2 * #cores = #logical_cores
  - ➔ DB12-at-boot:
    - Obviously no increase in performance if #job_slots > #cores
  - ➔



MJF-DB12 with default settings (only 1 iteration) – differences decreasing with higher number (>20) of iterations

Benchmark Collection

- Benchmarks used in this assessment:
  - ➔ HS06
  - ➔ DB12-at-boot
    - Original Python script
    - Numpy (Vincenzo Innocente)
    - CPP (Domenico Giordano)
  - ➔ **1 benchmark copy per job slot provided by the WN**

Application Performance

- Assessment by site monitoring
  - ➔ **Runtime of pilot payloads estimated by monitoring top commands (aliroot, root.exe, athena.py, cmsRun, python)**
    - Healthy nodes only
    - Quick-and-dirty assessment, doesn't take into account the job type (simulation, reconstruction, ...), or whether the jobs are belonging to the same production or not
      - 1 measurement per hour, average of 6 days
    - Ignoring pre- and postprocessing
    - Analysis per VO

Benchmarking Farm

- GridKa has configured around 1.5 job slots per physical core (default if host provides Hyperthreading)
  - ➔ Few exceptions: 1.6 slots per core for optimized multi-core support, 1 slot per core at Opteron hosts (no Hyperthreading)
- Other sites are using different policies (1 or 2 or ... slots per core)
- Undisclosed ratio in cloud IaaS environments, see D. Giordano's examples

- Temporary special configuration of GridKa compute farm some months ago for benchmarking reasons
  - ➔ WNs with either 1, 1.5 (1.6), and 2 job slots per physical core
  - ➔ Appropriate static benchmark scores (HS06, DB12-at-boot) available from MJF

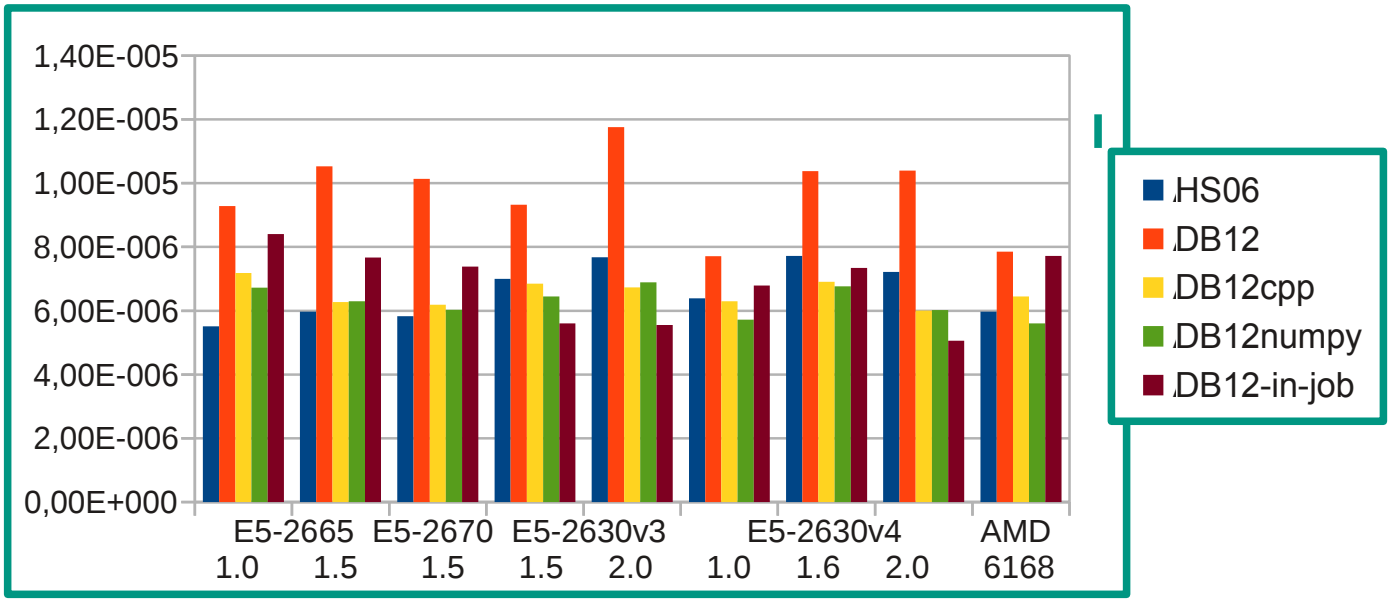
Benchmarking Farm

- Available hardware models:
  - ➔ Intel:
    - E5630: 1.5 job slots per physical core
    - E5-2665 (Sandy Bridge): 1 or 1.5 slots per core
    - E5-2670 (Sandy Bridge): 1.5 slots per core
    - E5-2630v3 (Haswell): 1.5 or 2 slots per core
    - E5-2660v3 (Haswell): 1 or 1.6 slots per core
    - E5-2630v4 (Broadwell): 1, 1.6, or 2 slots per core
  - ➔ AMD Opteron (1 job slot per core):
    - 6168 (2 sockets)
    - 6174 (4 sockets)
    - 6376 (4 sockets)

Results

Estimated Job Performance vs. Benchmarks

- Alice:
  - ➔ No significant dependency* on number of job slots per core
  - ➔ Haswell or Broadwell a bit faster than Sandy Bridge
  - ➔ No better performance on Opteron



* Comparing with HS06 and the CPP and Numpy flavors of DB12

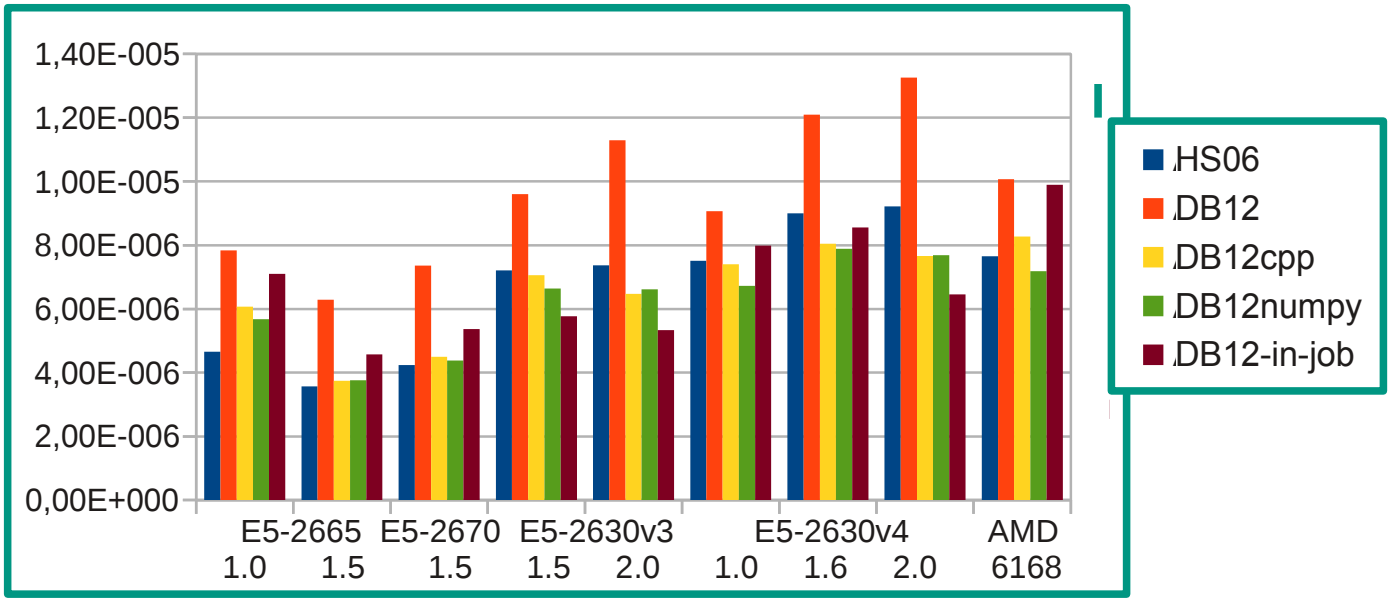
Estimated Job Performance vs. Benchmarks

- Alice:
  - ➔ No significant dependency* on number of job slots per core
    - Haswell or Broadwell a bit faster than Sandy Bridge
  - ➔ No better performance on Opteron

Remark:
The benchmarking working group has concentraded itself on Intel architecture because of the end-of-life status of AMD Opteron processors. AMD is however coming back soon with the new Zen architecture (Naples), and at least the final decision about the successor of HS06 should include benchmark results as well as job performance of Zen nodes.

Estimated Job Performance vs. Benchmarks

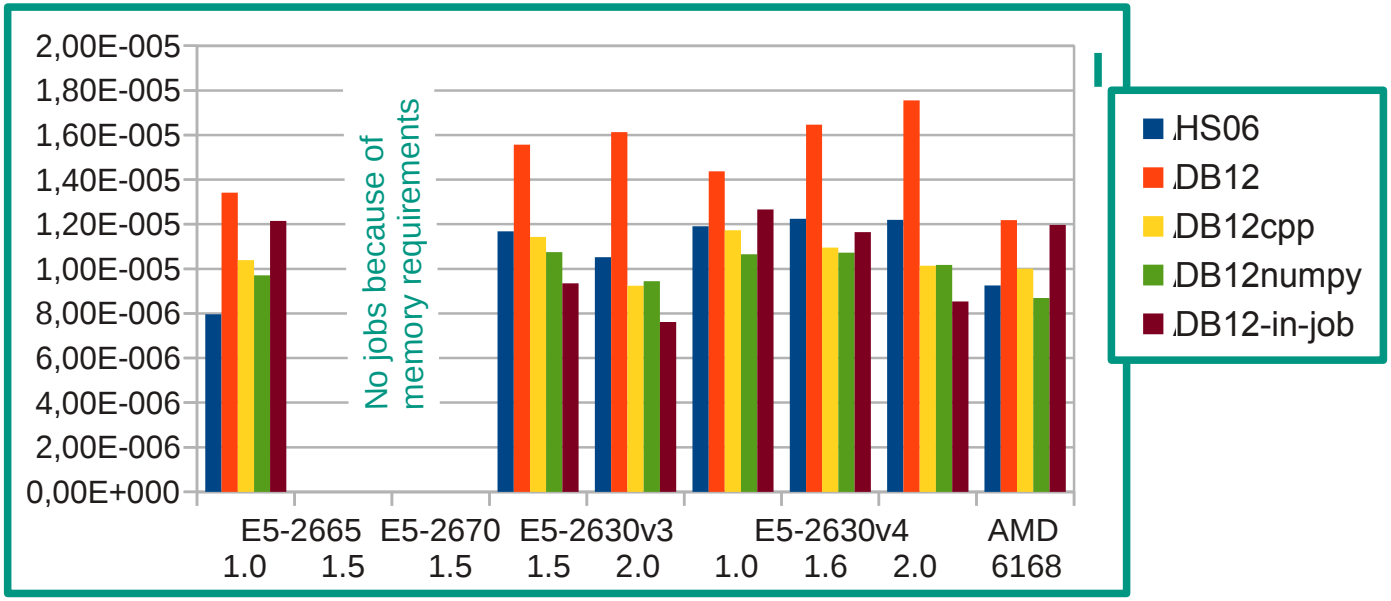
- Atlas:
  - ➔ No significant dependency* on number of job slots
  - ➔ Boost on Haswell, Broadwell, or Opteron of up to 100% and more



* Comparing with HS06 and the CPP and Numpy flavors of DB12

Estimated Job Performance vs. Benchmarks

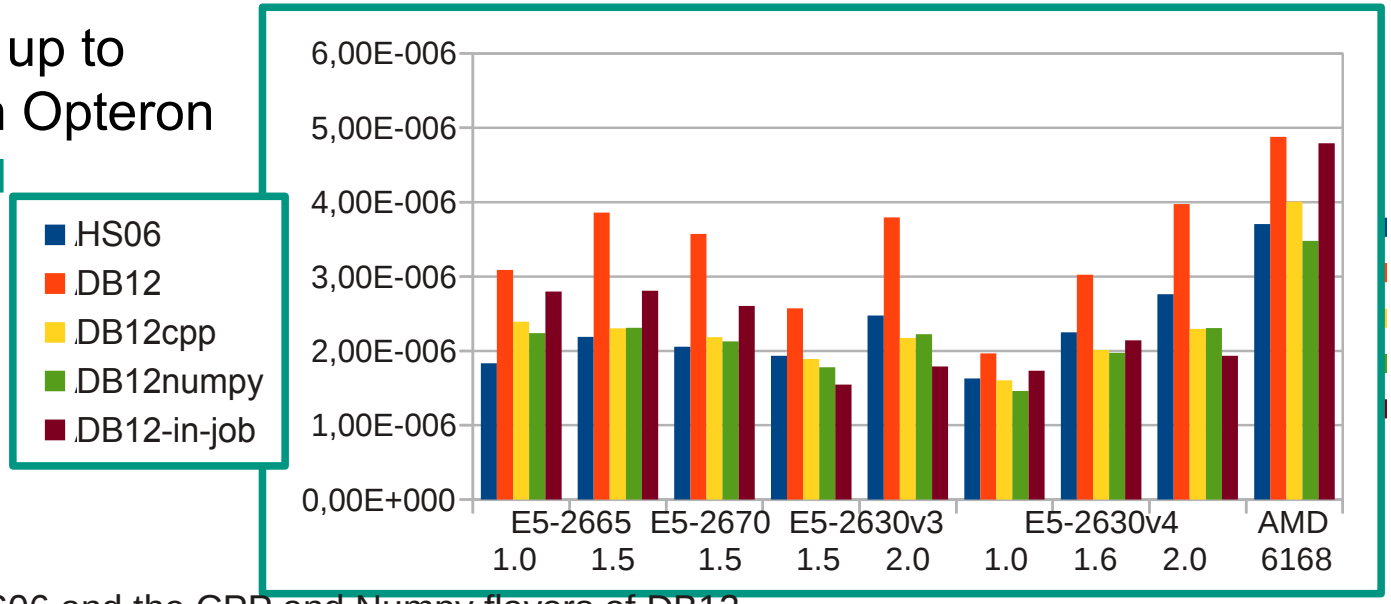
- CMS: *Low number of jobs*
  - ➔ Benefit from more job slots per core as expected*
  - ➔ Slightly better performance on Haswell or Broadwell (compared with SB)



* Comparing with HS06 and the CPP and Numpy flavors of DB12

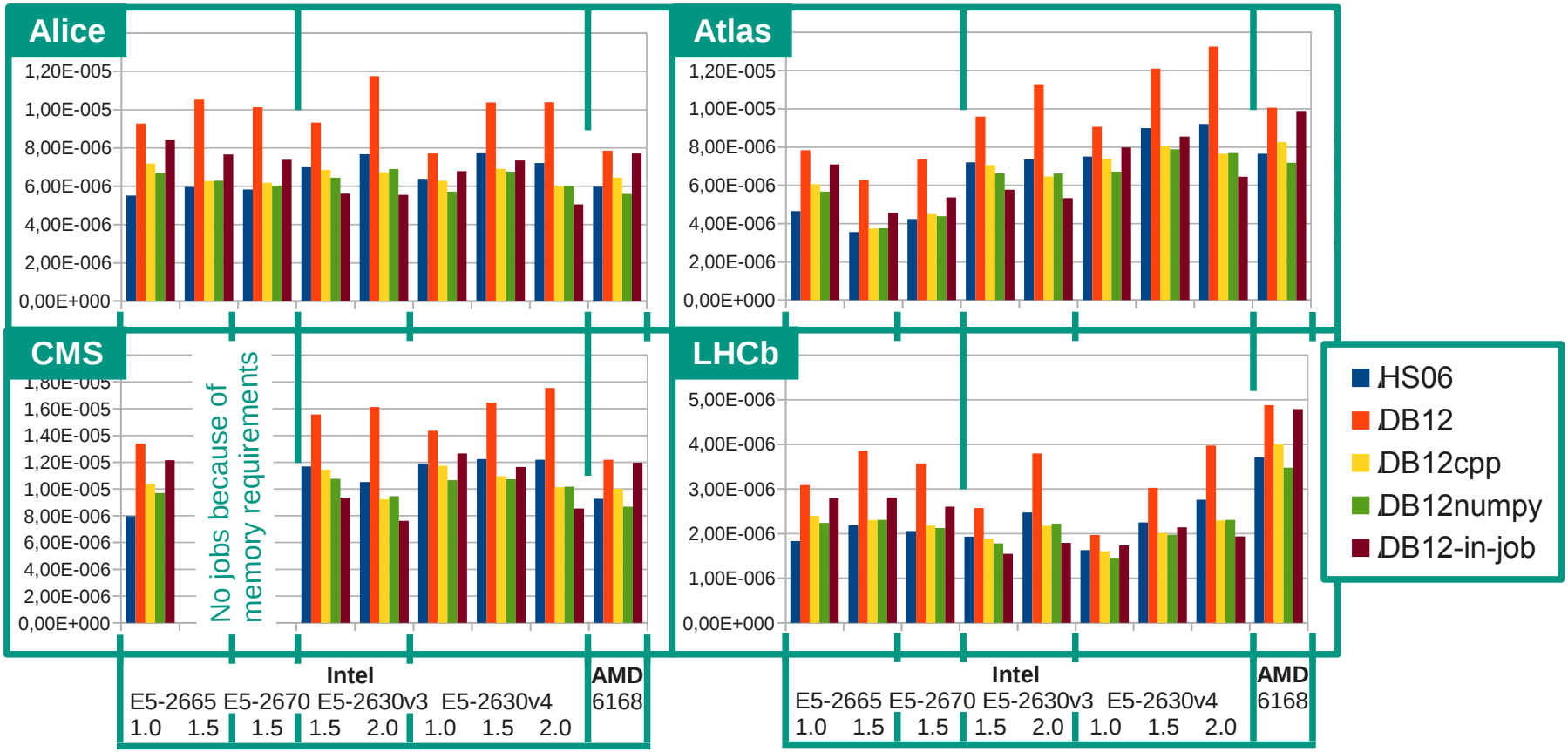
Estimated Job Performance vs. Benchmarks

- LHCb:
  - Best benefit* of more job slots per core, no significant dependency on number of job slots found (at least for Haswell and Broadwell)
    - Very bad correlation with original DB12-at-boot!
  - ➔ No boost on Haswell or Broadwell
  - ➔ Boost of up to 100% on Opteron



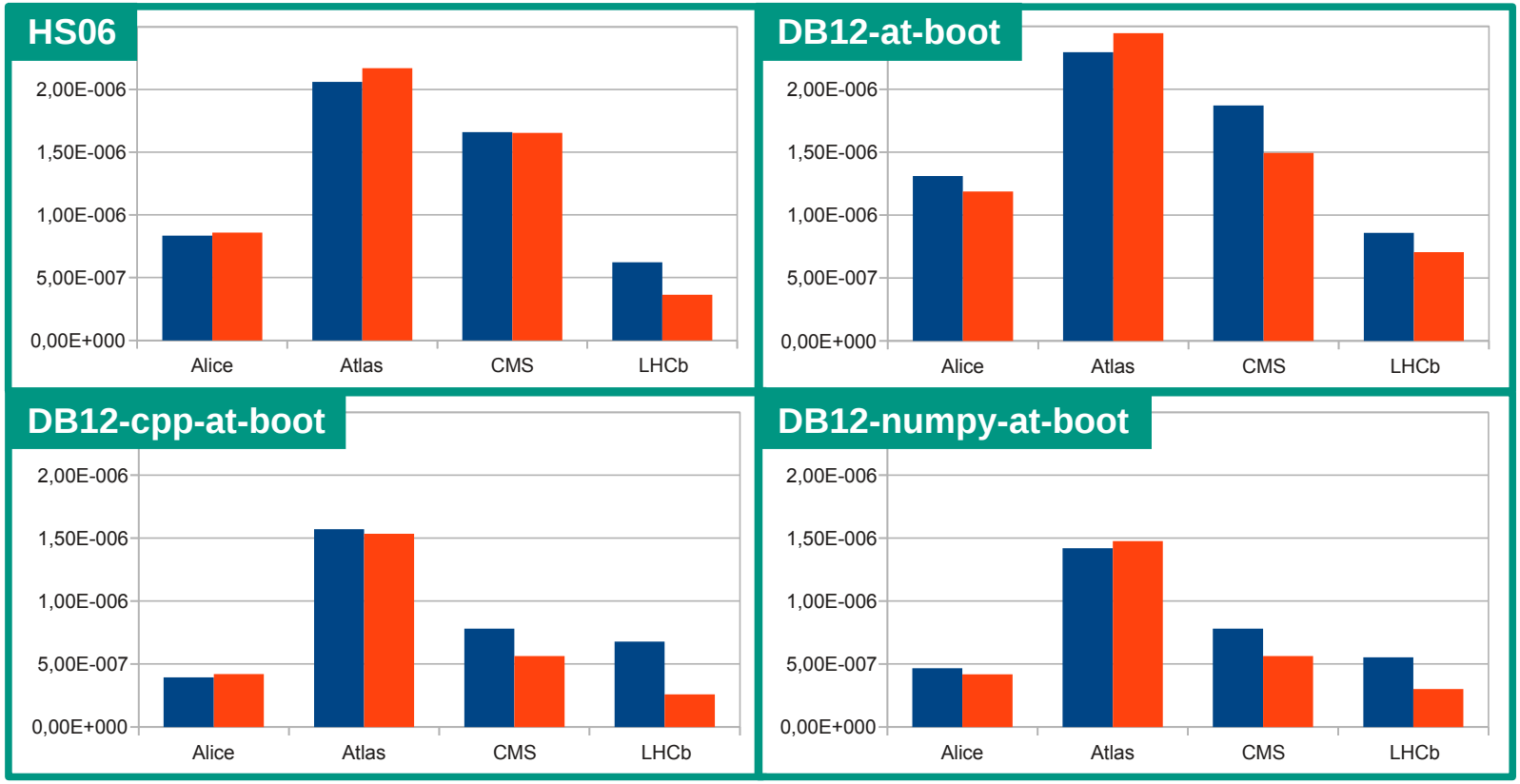
* Comparing with HS06 and the CPP and Numpy flavors of DB12

Estimated Job Performance* vs. Benchmarks



* Jobs at GridKa, average 2017-04-28 ... 2017-05-05

Benchmarks vs. Performance per Experiment



Conclusions

- Quick-and-dirty assessment of job performance by site monitoring over a few days
- Over-commitment of compute hardware (the initial question):
  - → LHCb benefits by up to 100%, no differences in job performance depending on number of slots
  - → Performance of Alice, Atlas, and CMS correlating with HS06, DB12-cpp, and DB12-numpy (not with original DB12-at-boot)
- Correlation of job performance with static benchmarks:
  - → CPP and Numpy versions of DB12 are scaling better with job performance than HS06 which is better than original DB12
  - → Best correlation of job performance with benchmarks for Alice and LHCb, and CMS at least for CPP and Numpy version of DB12

Conclusions

- Platforms:
  - ➔ Job performance per benchmark score (compared with Sandy Bridge):
    - Intel Haswell/Broadwell hosts: significantly better only for Atlas
    - AMD Opteron hosts: much better for Atlas and LHCb, but no advantages for Alice and CMS
      - No deeper analysis by the working group because of the end-of-life status of AMD Opteron processors
      - Repeat assessments when the new AMD Zen architecture (Naples) becomes available

Conclusions

- Successor of HS06:
  - ➔ The proposed, original DB12-at-boot benchmark isn't a suitable candidate to replace HS06
  - ➔ Either the CPP or the Numpy version of DB12-at-boot should be considered