Comparison between MJF and event rates

Ph. Charpentier
CERN-LHCb

Why benchmarking from VO perspective?

MJF AND BENCHMARKING

- Two main usages:
  - Job matching / job masonry
  - Accounting
- Job Matching / masonry
  - We know the CPU-work required by a job (HS06.s)
    - Can my pilot match this job or not within the available resources?
  - We want to adjust the job duration to the CPU-work left
    - LHCb elastic MC jobs
    - Define at last moment the number of events
      - NbEvts = CPUWorkLeft / CPUWorkPerEvt (from a test sample)
      - With some safety margin of course
  - Precision
    - For matching CPUWorkLeft should not be overestimated
    - For masonry it should be as precise as possible (+0 -20%)
- Accounting
  - Compare with estimates and WLCG/EGI accounting
    - If possible more precise than WLCG/EGI

What are the ingredients?

MJF AND BENCHMARKING

- Time left
  - In queue or lifetime of a VM
    - From batch system or MJF
    - Sometimes time left, sometimes CPU work left…
- CPU power available to the job (CPUPower)
- CPUWorkLeft = CPUTimeLeft * CPUPower

- What is available?
  - From batch systems: CPUTimeLeft
    - not always real seconds (could be work for a certain power, e.g. 1 SI2k at CERN)
    - Sometimes it provides also the assumed CPU power (but not always)
  - CPU Power
    - Fast benchmark (LHCb)
    - WN type / power table (but almost impossible, see Ulrich's presentations)
    - MJF but…

LHCb

What is the CPU power available to a job?

- Notation:
  - HS06: total benchmark result (e.g. from MJF)
  - NbLogCores / NbPhysCores: logical / physical cores
  - NbSlots: number of single core slots allowed
  - CPUPower: available CPU power per "allocated core"
- Assume:
  - HS06 measured running NbSlots instances of benchmark
  - We are running 'single core' applications (i.e. mostly using a single core as there are always several threads)
- Simple case: no hyperthreading
  - CPUPower = HS06 / NbPhysCores
    - How do we know? OK if NbPhysCores == NbLogCores == NbSlots
- Hyperthreading enabled
  - Assume NbPhysCores < NbSlots ≤ NbLogCores
    - No overcommitment
    - More slots than physical cores
  - CPUPower = HS06 / min(NbSlots, NbLogCores)

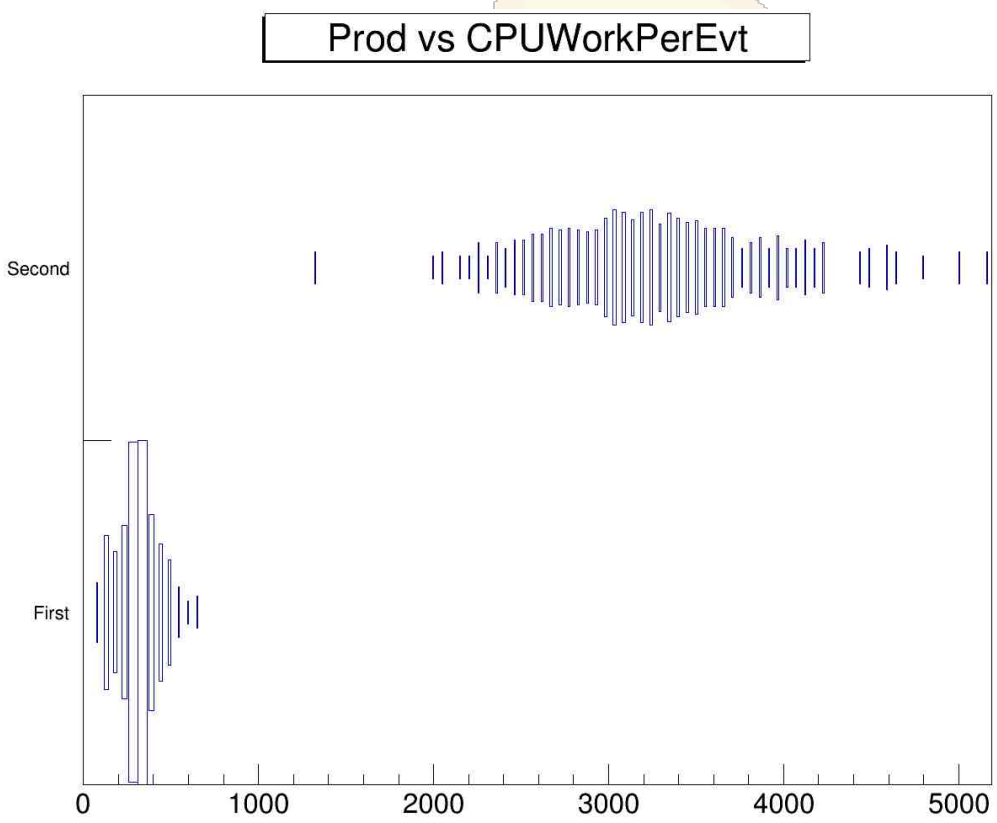
Benchmarking with jobs

MJF AND BENCHMARKING

- Chose MC production: event generation (low multiplicity) including fixed B decay mode + detector simulation (Geant4)
  - Event only differ by their random seeds
  - Select "large" jobs i.e. more than 100 events
  - Initialisation / finalisation negligible (typically equivalent to one event), very low IO
  - CPUwork may vary from 300 to 5000 HS06.s
- For each job we collect:
  - Total CPU time, number of events
  - LHCbDirac CPUPower evaluation
  - MJF CPUPower when available (CERN, GRIDKA)
    - Caveat: wrong formula applied currently:
      - CPUPower = HS06 / NbLogCores
    - Correction to apply:
      - GRIDKA: factor 2/1.5 on Intel (as 1.5 slots per physical core, not 2)
      - CERN: WN dependant NbLogCore / min(NbLogCore,NbSlots)
        - Information gathered from LSF

Results from Dirac CPUPower

- Short (2mn) script run by each job, mostly computational
  - Absolute calibration old (several years ago with old CPUs)
- Selected two recent productions with quite different CPU requirements

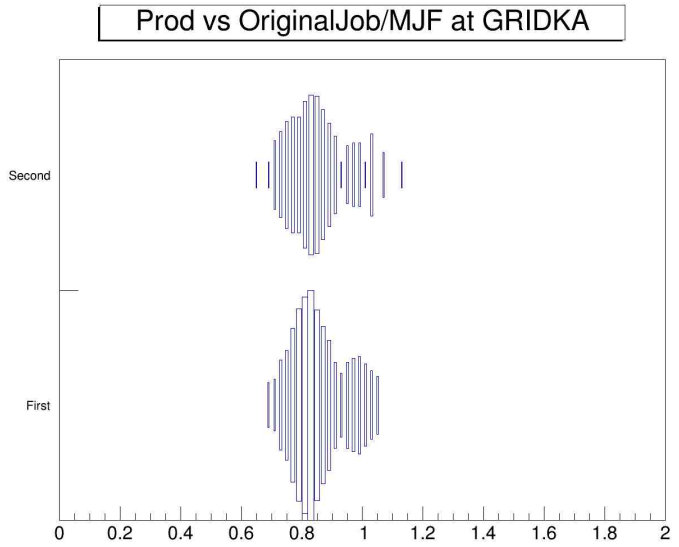


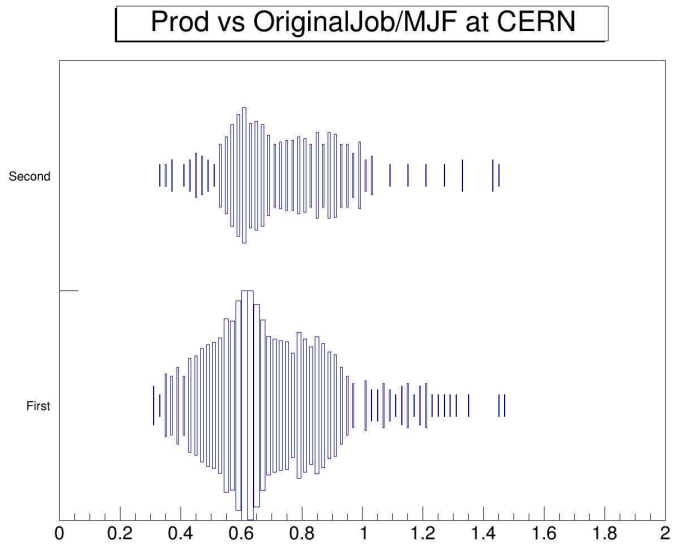
CPUPower extraction from jobs

- For each production, the event rate is proportional to CPUPower
  - CPUTime * CPUPower / NbEvents = CPUWorkPerEvt
  - CPUPower = k_{prod} * NbEvents /CPUTime
  - k_{prod} arbitrarily taken from the CPUWorkPerEvt using Dirac's CPUPower (only relative calibration of the two productions)
- Definitions for plots:
  - MJFPower : CPUPower from MJF (corrected with NbSlots)
    - OriginalMJFPower : obtained from jobs with wrong formula
  - DiracPower : CPUPower from Dirac
  - JobPower : CPUPower from jobs
  - Job/MJF : ratio of MJF to Job-Power
  - WNModel : CPU model as returned by the OS
    - E.g. Intel(R)Xeon(R)CPUL5520@2.27GHz
  - WNType (at CERN) : using CERN convention for WNs
    - E.g. i6_24_62c2h (but not complete type yet used)
    - Can be extended with NbSlots, NbLogCores
      - E.g. ('i6_24_62c2h', '12', '24')

Original values

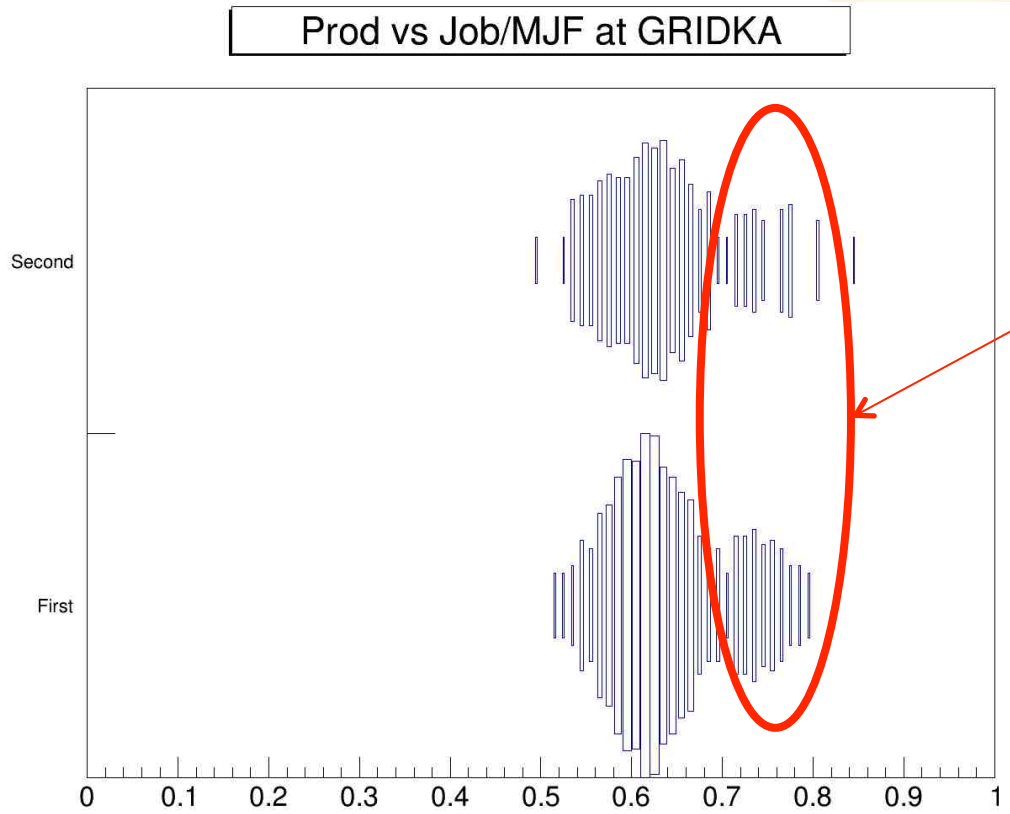
- Different absolute normalization at CERN and GRIDKA…
- Large tails at CERN
- … formula used was wrong:
  - HS06/NbLogCores
- At GRIDKA, 1.5 slots per physical core, i.e. ratio 2/1.5
  - For Intel-based WNs (selected)
- Normalisation: 0.84 / 0.63
  - Which fits exactly
- Correct MJFPower by the number of slots
  - Not always easy to find at CERN, but…



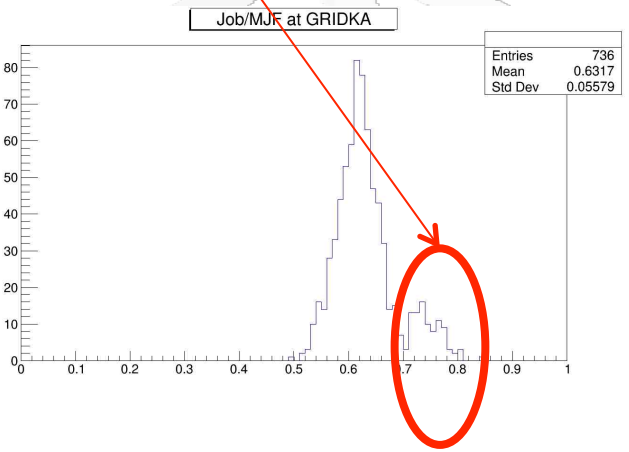


Results about GRIDKA

- Simplest site: low number of WNModels, fixed ratio NbSlots/NbLogCores
  - Exclude one WNType found to have bad settings (fixed by the site), only used Intel-based WNs

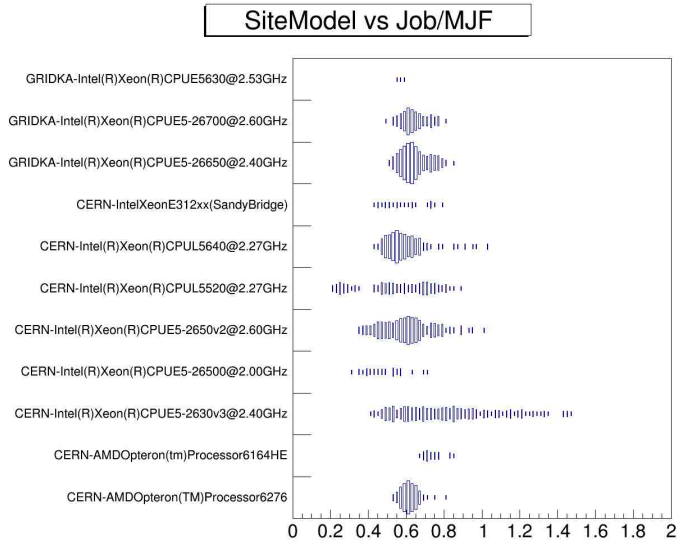


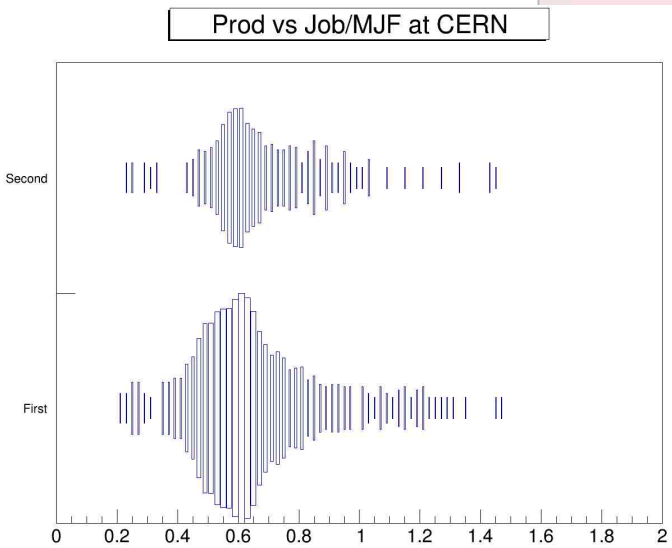
Not understood yet…

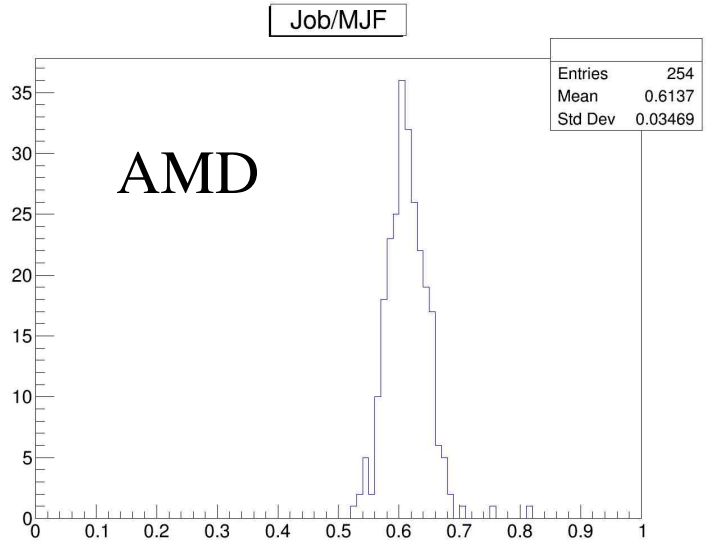


Results at CERN

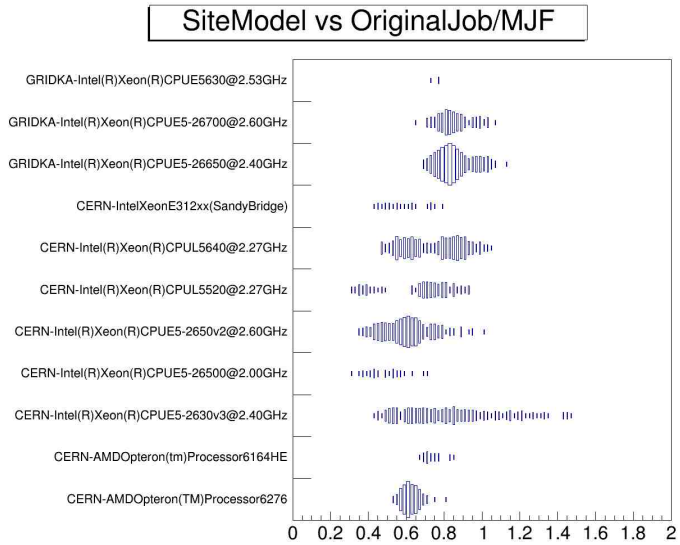
- Looks better for some models than for others (right)
- The two productions agree well
- Extremely sharp peak when selecting AMD nodes at CERN
  - Shows it is a good probe (5.5%)

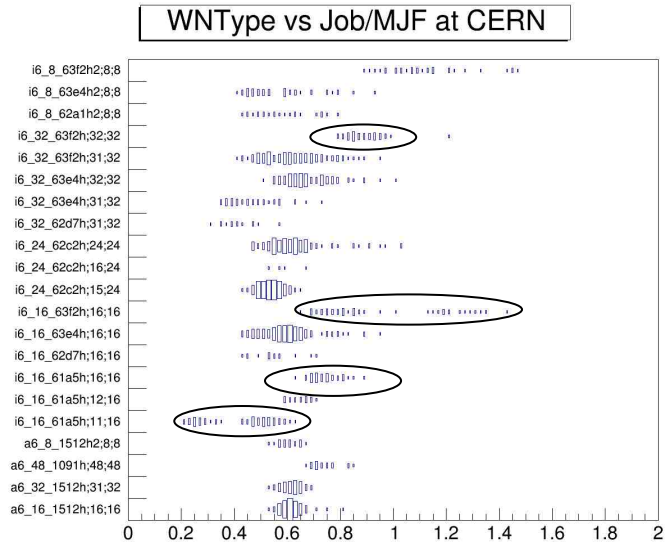






Subtle effects

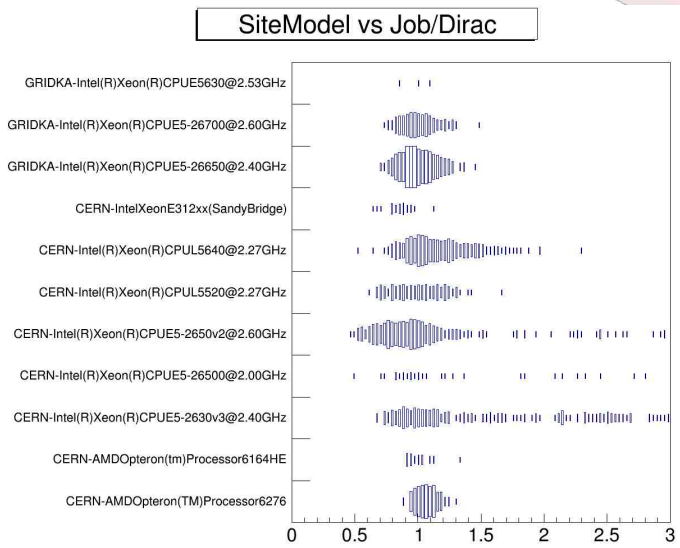


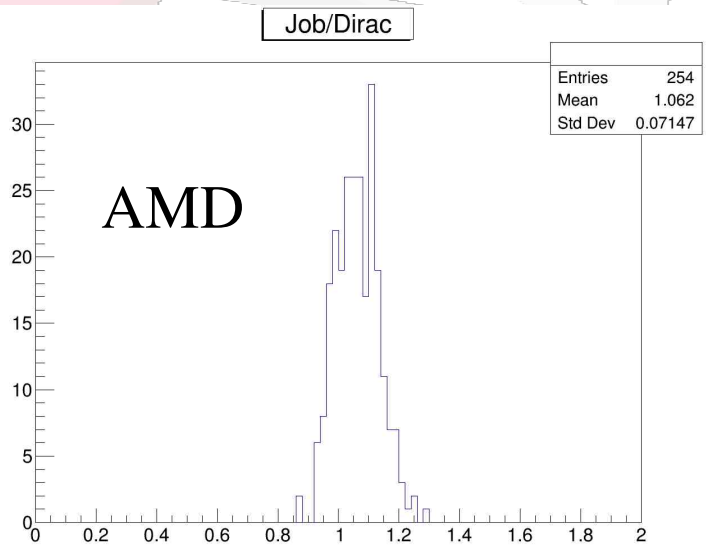


- HW model is not a good metrics (left)
- At CERN, nodes have a type, but also NbSlots and NbLogCores are important
  - Better (right) for most types, but still some strange effects (surrounded)
  - Should check carefully if the information is correct
  - The LSF cpuf parameter uses NbLogCores and not min(NbSlots, NbLogCores)

Using fast benchmarking

- Dirac fast benchmarking currently used
- Comparison between job and Dirac benchmarking is not so bad (left)
  - Same problematic WNModels…
  - Very good also for AMD (6.7%) and at GRIDKA
    - Effects like load / variable clock frequency for energy saving ?
    - Reminder: absolute normalisation is not important (as used for defining the Job absolute scale!)





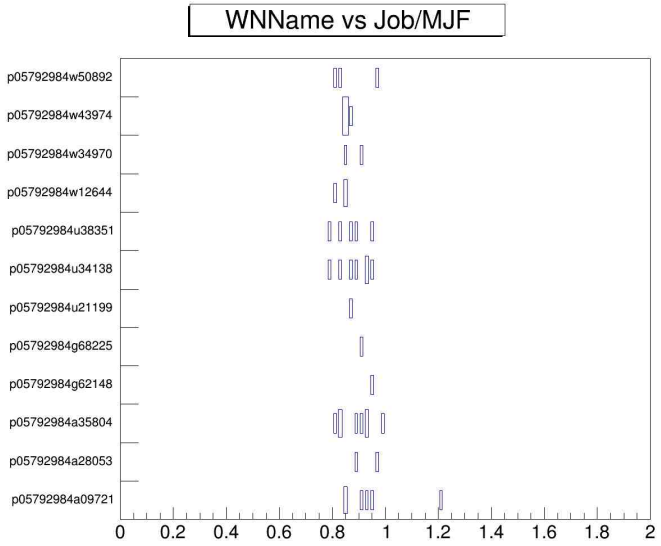
Conclusions

MJF AND BENCHMARKING

- Information from MJF is important for LHCb
  - Matching, masonry and accounting
  - More important even for multicore job slots (super-masonry)
- Many parameters enter into a correct determination of CPUPower
  - Do we agree on the formula?
  - Do all sites agree to run HS06 in the same conditions as jobs run (i.e. as many instances as job slots)?
    - Beware: WN classification may be overcomplicated (see Ulrich's talk @ preGDB earlier on)
    - Should each box be benchmarked?
- Fast benchmarking is a compromise, but less precise
  - We should probably recalibrate our benchmark point!
    - Anyway for matching it is irrelevant as we use the same ruler for getting the CPU requirements!

Detailed study

- Below: CPU type i6_32_63f2h_32_32
  - Too high value
  - Means too low MJF estimate
    - Not a problem for matching
  - Special case of WNs with name starting with 'p' ?



- Below: CPU type i6_16_61a5h_11_16
  - Clearly two sets of WNs
  - I can provide lists
  - Found a difference:
  - RESOURCES: (intel share aishare cvmfs wan exe lcg cpuperf slot11)