“CPU Units” proposal

Andrew McNab
University of Manchester
LHCb and GridPP

Context

- The idea of changing the benchmarks we use for accounting and pledges has been raised
- The HEPiX Benchmarking WG has been asked to start looking for candidates
- The WLCG Accounting TF was asked to report on how hard it would be to change APEL, Accounting Portal etc
- This proposal is one way of managing that kind of change, in a way that makes it easier to change in the future
- It's important to stress that we may want to change again if there is a repeat of the Haswell step-change in performance
  - +40% for HEP applications and fast benchmarks; but not for HS06. So improved delivery to experiments is not recognised

“CPU Units” idea

- WLCG adds a “CPU Unit” (CU) in parallel with HS06 in the accounting system (APEL, accounting portal etc.)
- To start with, 1.0 CU = 1.0 HS06
- WLCG can update the definition of CU to reflect changes in the technology (eg the Haswell scenario)
  - It can be a combination of one or more benchmarks
  - New benchmarks can be included; old ones dropped
- Since CU is designed to be updated, we don't have to change the accounting system, pledges etc each time
- But this puts constraints on what revisions can be made to the CU definition

“CPU Units” revision constraints

- CU definition should be based on empirical evidence about experiment software performance across relevant hardware
- Avoid penalising sites for good faith decisions in the past
  - So sites may choose to continue to publish previously published CU values after a revision
  - Guarantees that their ability to pledge won’t go down
  - But prevents them using an old definition of CU on new hardware
- Weights/scale used within CU should be chosen to ensure that on older (oldest?) hardware:

  previous CU value = new CU value

“CPU Units” revision consequences

- On newer hardware if the new definition is sensitive to improvements in technology, then new CU value may go up
  - This is a Good Thing: it gives credit for hardware which is doing more work for experiments than we thought
  - Motivates sites to buy hardware which is better for the experiments
- WLCG has the choice about whether to stay with the same CU definition for a decade or change next year
  - Don’t have to worry about cost of changing APEL etc
  - Don’t get paralysed by the thought that we might make the wrong decision and be stuck with it for ten years
  - Can respond to evolving experiment code

Ongoing benchmarking / performance

- To make sure WLCG community are making the best purchasing decisions
  - We should monitor the performance of new architectures with “CPU Units” revisions in mind
    - That’s architectures not just particular vendors’ models
  - This already happens at some level, but CU provides a mechanism to keep it all joined up:
    - From experiment software measurements
    - To the pledges
- Ideally, a way of easily running (duplicating?) some production work on very new and unusual hardware to have real comparisons
  - eg Atom processors
  - Even where not credible to buy, they give a broader range of data points to calibrate, say, cache dependency of performance
  - Makes behaviour visible which may be masked on balanced machines

Making it easier to rerun benchmarks

- So we can easily distribute them in RPMs (etc)
  - Should be Open Source
  - Have no dependencies beyond standard OS
  - Be small enough to be distributed
- Fast enough that we can run them at boot time on each worker node?
  - Or at OS install time?
  - Or just at commissioning time - as now.
- **Ideally we should try to turn benchmarking from a commissioning activity into an operational activity**
  - Then CU revisions won't be painful to roll out
  - Sites won't be tempted to "estimate" benchmarks

Summary

- Haswell step-change in performance has highlighted some things
  - We're reluctant to update benchmark choice because of the cost of rollout
  - We're reluctant to update benchmark choice because of the effort of changing central services, pledges etc
- CPU Units proposal provides a mechanism for making updates easier
  - … partly by making promises to sites and experiments that revisions will be fair
- Can we convert benchmarking from a commissioning activity into an operational activity? Like security patches.

Backup slides

CPU Performance Benchmarking

- Fundamental aim of benchmarking is to attempt **to predict the rate at which a given computer can run applications of interest**
  - Prediction either relative ("it will be twice as fast on this CPU as that one") or absolute ("these events will take 43.5 hours to simulate")
  - So benchmarking is about constructing theories of CPU performance
  - Usual requirements apply: **theories should be as simple as possible, and make accurate, consistent, reproducible predictions**
- CPU performance depends on multiple fundamental metrics
  - Clock speed, instructions per clock cycle, complexity of instructions, branch prediction, cache sizes, cache speed, memory speed, …
- Simple model is that speed in executing a given task is a linear combination of the fundamental metrics for that CPU
  - In general, weights will be different for different applications
  - **A good benchmark for a given application has the same set of weights for the metrics as the application itself**

CPU Performance Benchmarking (2)

- However, the individual metrics' weights are not usually observable
- What we see is the overall benchmark speed and the overall application speed, and we compare those
- Benchmark suites (like SPEC06) attempt to provide multiple benchmarks with varying dependencies (weights) on the fundamental metrics of CPUs
  - Hope that benchmarks form a basis (in linear algebra terms)
  - The weights appropriate to any application can then be achieved by forming a linear combination of the basis set of benchmarks
  - eg appSpeed = 1.0 x busSpeed + 0.4 x cpuSpeed (fundamental metrics)

    = 0.4 x BM1 + 1.0 x BM2 (suite benchmarks)

    where BM1=(0.5 x bS + 0.5 x cS) and BM2=(0.8 x bS + 0.2 x cS)
- So, what benchmarks are appropriate for our application domains?
- And what is convenient? What provides a basis? Can represent any app?