“CPU Units” proposal

Andrew McNab
University of Manchester
LHCb and GridPP

Context

- Part of the HEPiX Benchmarking WG's role is to recommend a benchmark to WLCG that would be suitable for accounting (including pledges)
- Already some mentions/discussion of replacing 32-bit HS06 at the pre-GDB, GDB, GDB Steering Group and WLCG Accounting TF last week, and WLCG MB this week
  - Specifically, the Accounting TF has been asked to report on what is involved in changing benchmarks in APEL, portal etc
- Last week I mentioned an idea for how to handle accounting benchmarking changes more smoothly
  - This makes it easier to change things, when the technology requires (eg another Haswell-like scenario)
- This question is an MB/GDB problem, but relevant to the context of this WG of course

“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 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
- But to really benefit from this flexibility, we should use “at-boot” benchmarks
  - Makes it easy for sites to re-benchmark their hardware

Ideal “at boot” benchmarks

- So we can easily distribute them in RPMs (etc)
  - Should be Open Source
  - Have no dependencies beyond standard OS
  - Be small enough
- So we can run them at boot time
  - Fast enough that running at each boot is practical (minutes not hours)
- So we can collect the results automatically
  - Support some standard API like MJF
- **Turns benchmarking from a commissioning activity into an operational activity**

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

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?