Toward a Grand Unified Theory of High Performance Computing

John D McCalpin/Austin/IBM
mccalpin@us.ibm.com

IBM Systems Group Hardware Development
Background

- The HPC server market is getting more competitive and more price/performance sensitive

- Is it possible to understand the decisions that lead to these market changes?
Outline

• Recent Market Changes in High End Computing
• Fundamental Metrics
  – Value & Cost
• Secondary Metrics
  – Performance, Scaling, Efficiency
• Issues for future systems at extreme scale
Outline

- Recent Market Changes in High End Computing
- Fundamental Metrics
  - Value & Cost
- Secondary Metrics
  - Performance, Scaling, Efficiency
- Issues for future systems at extreme scale
Recent Market Changes in HPC

• Some long-term stability
  – >80% of HPC server revenue below $1M price point

• Some long-term trends
  – Downmarket shift in revenue distribution
  – Clusters continuing to replace Custom systems

• Some dramatic changes
  – x86 architectures growing at a phenomenal rate
Note that “RISC Cluster”, “SMP”, and “Industry Std” cut across all price bands.
Source: IBM Analysis of IDC “tracker” reports, 2000-2004
Capability Class System Types over Time

Source: IBM Analysis of IDC “tracker” reports, 2000-2004
Linux Clusters on TOP500 List

Source: IBM Analysis of TOP500 lists, 2002-2004
Outline

• Recent Market Changes in High End Computing
• Fundamental Metrics
  – Value & Cost
• Secondary Metrics
  – Performance, Scaling, Efficiency
• Issues for future systems at extreme scale
Fundamental Metrics

• HPC is a human activity, so the fundamental metrics are those that impact humans

• What is the “Value” of the set of calculations?

• What is the “Cost” of the set of calculations?

• Decision-making in HPC is all about comparing Value and Cost
What is the “Value” of the calculations?

- **Value** is a function of
  - **Time to solution**
    - Answers now are worth more than answers later
    - Many families of curves of this class
  - **Accuracy**
    - Within a dependent calculation stream (e.g., mesh refinement)
    - Across independent calculation streams (e.g., ensembles)

- Hard to quantify, but ESSENTIAL for understanding
  - Most convenient when expressible in $$$

1/25/2005
Sample “Value” Functions

- **Value vs Time to Solution**
  - “deadline”
  - “multi-tier”
  - “monotone”

- **Value vs Error**
  - “ballpark”
  - “multi-tier”
  - “monotone”

- Consider tradeoffs between these two
What is the “Cost” of the calculations?

- Cost has two major parts:
  - **SW development & maintenance**
    - \( S_{SW} = T_{develop} \times C_{develop} + T_{maintain} \times C_{maintain} \)
    - C are in units of $ / unit time
  - **HW depreciation, support & administration**
    - \( S_{HW} = T_{runtime} \times C_{HW} + T_{archive} \times Q_{archive} \times C_{archive} \)
    - \( C_{HW} \) includes HW depreciation, HW maintenance, and HW administration expenses
    - Archive cost includes quantity of data and duration of archiving (it may be cheaper to recompute than to save results)
Sample “Cost” Functions

- **“Kleenex” code**
  - Dominated by development cost

- **ISV code**
  - Balance between HW cost and SW maintenance cost

- **Streaming simulations**
  - Can be dominated by output archiving cost
Value vs Cost: a prototypical example

- Consider a typical simulation of a physical system
  - Approximate solution of time-dependent 3D partial differential equations

- The project includes:
  - Cover a modest parameter space of physical and numerical parameters
  - Runs must be relatively long to get good statistics of time-varying results
  - Code exists now, with well known performance profile
  - All runs must be finished in fixed calendar time
Example (continued)

• Lots of tradeoffs can be made here….
  – Serial vs SMP parallel vs MPI parallel runs
    • Expensive results now vs cheaper results later
    • When can I start thinking about the paper?
    • When can I feed results back into the research process?
      (e.g., avoid doing unnecessary runs)

  – More accurate discretization (finer mesh or higher-order discretization) vs longer runs (better statistics) vs more coverage of parameter space
Getting quantitative about Value

• I need to define the “Value” of the solution as a function of “Time to Solution” and “Error of Solution”

• Assume I have thought deeply about this and decided on a particular functional form, e.g.:
  – $V(T, \varepsilon) \sim \max(1-T-\varepsilon,0)$

• Now we need to relate $T$ & $\varepsilon$ to the set of user-controllable parameters
Composite Performance Figures of Merit

• Q: How should one think about composite figures of merit based on a collection of low-level measurements?

• A: Composite Figures of Merit must be based on “time” rather than “rate”
  – i.e., weighted harmonic means of rates

• Why?
  – Combining “rates” in any other way fails to have a “Law of Diminishing Returns”

• The general methodology is based on a “plus or max” operator
  – Lower bound on total time is max time of any component
  – Upper bound on total time is sum of times of all components
A Specific Example Model

- Analyze applications and pick reasonable values:

\[
\text{"Balanced GFLOPS"} \equiv \frac{1 \text{ "Effective FP op"}}{\left(\frac{1 \text{ FP op}}{\text{LINPACK GFLOPS}}\right) + \left(\frac{2 \text{ Bytes}}{\text{STREAM GB/s}}\right) + \left(\frac{0.1 \text{ Bytes}}{\text{Network GB/s}}\right)}
\]

- Two cases considered:
  - Assume long messages
  - Assume short messages

- The relative time contributions will quickly identify systems that are poorly balanced for the target workload
Comparing p655 cluster vs p690 SMP
Assumes long messages

NOTE: these are schematic representations, not actual performance measurements or projections!
Comparing p655 cluster vs p690 SMP
Assumes short messages

NOTE: these are schematic representations, not actual performance measurements or projections!
Relating Time to Solution to User Controllable Parameters

• Assume: \[ T_{total} = T_{cpu} + T_{mem} + T_{comm} \]

• Assume that the performance model is typical for low-order discretization of PDEs:
  
  – \[ W_{cpu} = a1 \times k \text{ (order)} \times N \text{ (#steps)} \times N^3 \text{ (#grid points)} \]
  – \[ W_{mem} = a2 \times N \text{ (steps)} \times N^3 \text{ (#grid points)} \]
  – \[ W_{comm} = a3 \times N \text{ (steps)} \times N^3 / P \text{ (#grid points/node)} \]

  – \[ T_{cpu} = W_{cpu} / R_{cpu} \text{ (infinite cache execution rate)} \]
  – \[ T_{mem} = W_{mem} / R_{mem} \text{ (sustained BW)} \]
  – \[ T_{comm} = W_{comm} / R_{comm} \text{ (sustained BW)} \]
A few notes on “Error”

• Sources of “Error of Solution”
  – Continuum error - solving the wrong equations
  – Statistical errors - statistical uncertainties
  – Spectral error - unresolved wavelengths
  – Truncation error - inaccurate derivative estimates
  – Roundoff errors - numerical & solver errors
Relating Error to User Controllable Parameters

• Assume Functional forms for error estimates:
  – Continuum error = 0.001
  – Statistical error ~ N^{-1/2} (1/square root of run length)
  – Spectral error ~ N^{-4} (slope of spectrum)
  – Truncation error ~ N^{-k} (discretization accuracy)
  – Roundoff error = 0.0001

• $\varepsilon = 0.0011 + e_1 * N^{-1/2} + e_2 * N^{-4} + e_3 * N^{-k}$
  – Determining coefficients is *your* problem
Relating “Cost” to User Controllable Parameters

• Many different possible cost models, depending on the type of HW resources used
  – Here assume cost ~ P * Time to Solution
  – Different kinds of nodes have different CPU vs Bandwidth tradeoffs and costs

• Now cost is implicitly tied to scaling
  – Time to Solution appears in both numerator and denominator of Value / Cost
Value / Cost

- Value = max(1-T-\(\varepsilon\),0)
- Cost = P * T

- \( \frac{V}{C} = \frac{\text{max}(1-T-\varepsilon,0)}{P*T} \)

- Remember that:
  \[ T = T_{\text{cpu}} + T_{\text{mem}} + T_{\text{comm}} \]
  \[ = \frac{W_{\text{cpu}}}{R_{\text{cpu}}} + \frac{W_{\text{mem}}}{R_{\text{mem}}} + \frac{W_{\text{comm}}}{R_{\text{comm}}} \]
  \[ = \frac{k*N^4}{(P*R_{\text{cpu}})} + \frac{N^4}{(P*R_{\text{mem}})} + \frac{(N^4/P)}{(P*R_{\text{comm}})} \]

- Etc….
Value / Cost (continued)

• Finally, $V / C$ has a quantitative expression
  – Can be optimized by varying $k$, $P$, and $N$
  – Can be optimized by choosing nodes with different costs and different ratios of $R_{cpu}$, $R_{mem}$, $R_{comm}$

• The details are left as an exercise to the reader
And the point is?

• Value and Cost are fundamental

• Quantitative expression of Value is probably intractable in general
  – But “seat of the pants” estimation is required in order to make any rational decision

• Therefore quantitative expression of Value/Cost is probably also intractable
  – And also required
Outline

• Recent Market Changes in High End Computing
• Fundamental Metrics
  – Value & Cost
• Secondary Metrics
  – Performance, Scaling, Efficiency
• Issues for future systems at extreme scale
Performance

• Can we express “performance” as a weighted combination of orthogonal basis vectors?

• Shockingly brief overview here....
What about Price/Performance

• An interesting secondary metric is Price/Performance (or Cost/Performance)
• This is not a primary metric because arbitrarily small values of Price/Performance do not bring arbitrarily large values of Value/Cost
• Even this simple metric has some interesting complexities....
Price/Performance and Machine Balance

• Everybody talks about “balanced systems”, but what does this really mean?

• Consider a simple two-component model for system performance:
  – $T_{total} = T_{cpu} + T_{mem}$
  – $= \frac{W_{cpu}}{R_{cpu}} + \frac{W_{mem}}{R_{mem}}$
  – System Balance: $\beta = \frac{R_{mem}}{R_{cpu}}$
  – Application Balance: $\gamma = \frac{W_{mem}}{W_{cpu}}$
“Balanced Systems” continued

• So if you have an application with a known Application Balance, does that tell you anything about the System Balance of desirable systems?

• Idea #1: $\beta = \gamma$
  – Result: $T_{cpu} = T_{mem}$
  – System is “balanced” in execution time
  – Was this a good idea?
    • Maybe it would have been better to make one part faster?
Extended Model for Balance

- Let’s take cost into account:
  - \($_{cpu} = \text{cost of one unit of cpu performance}\)
  - \($_{mem} = \text{cost of one unit of memory performance}\)
- Now we want to minimize cost/performance
- Cost/Performance = Cost * Time
  - \(= ($_{cpu} * R_{cpu} + $_{mem} * R_{mem}) * (W_{cpu} / R_{cpu} + W_{mem} / R_{mem})\)
- Define \(\delta = \text{$_{mem} /$_{cpu}$}\)
Another idea

• Idea #2: $\beta = 1/\delta$
  – Result: Cost of CPU = Cost of Memory
  – System is “Balanced” in cost
  – Was this a good idea?
    • Maybe it would have been better to have less of the expensive part and more of the cheap part?
Do the math….

• For an application workload with fixed $\gamma$, minimizing the price/performance says that the optimum system has a balance of

\[
\beta = (\gamma/\delta)^{1/2}
\]

• This is not obvious!
  – It says that more bandwidth is good when it is cheap and more cpu performance is good when it is cheap
And the point is?

- Simplistic “rules of thumb” for machine balance can be very wrong, even in cases that are very familiar
- More complex performance models are likely to have even more counterintuitive behavior
- The parameter space is large enough that each customer might have a unique profile
Outline

• Recent Market Changes in High End Computing
• Fundamental Metrics
  – Value & Cost
• Secondary Metrics
  – Performance, Scaling, Efficiency
• Issues for future systems at extreme scale
WARNING

• The following slides contain forward-looking material

• This represents my personal thinking about important issues in future HPC systems of very large size

• Any interpretation of these slides as indicating specific product plans on the part of IBM is a hallucination on your part, not a commitment on IBM’s part.

• Predictions are hard, especially about the future.
Technical Issues: Programming Languages

• MPI must be effectively supported
  – Currently running into scaling limits with domain decomposition (NERSC, Sandia)
  – Lack of Failure Tolerance is a serious problem
  – Collective performance is inadequate for Terascale systems – how can we go to Petascale?
  – Short-message occupancy is too high
  – Short-message latency is too high
Technical Issues: Programming Languages

• **Advanced Programming Models**
  – Must be adopted by most or all major vendors
    • Cray will “open source” CHAPEL
    • What about X10?

• **Two paths probably required:**
  – Incremental: UPC, CAF, Titanium
  – Revolutionary: CHAPEL, X10, other?
    • Functional languages
    • High-level languages
    • Domain-specific languages
Advanced Programming Models

• Most advanced programming models require high performance access to a global namespace
  – Low latency, low occupancy, high concurrency

• Hardware/software co-design is essential
  – Too expensive to do everything in HW (e.g. coherency)
  – Too slow to do everything in SW

• Features?
  – Transactional coherency?
  – Atomic operations?
  – Active Messages?
System HW Costs

• Moore’s Law is slowing down
  – From 80%/year to much less
  – Maybe 20%/year? Maybe a bit more?

• A Rational Response:
  1. Build a well-balanced system that scales with base technology, and
  2. Make a few carefully chosen higher-risk investments to try to change the rules, starting with the most expensive components
System HW Cost Ranking

- For “traditional” High End System Balances:
  - Network bisection bandwidth is expected to be the most expensive component
  - Then cost of DRAM banks for non-contiguous memory accesses
  - Then cost of CPUs
  - Then cost of DRAMs for unit-stride bandwidth
  - Then cost of DRAMs for minimum required capacity
- Where does I/O BW & capacity fit?
System Balance vs Cluster Size

- Algorithm scaling properties cause significant shifts in performance balances as systems grow to extreme scale.

- General Trends as number of nodes increases:
  - Less memory required per node
  - Less compute per step per node
  - More Communications Bandwidth per FLOP
  - Shorter Average Message Lengths
  - More Global Collective operations per FLOP

- Current extreme scale systems are increasingly performance-limited by collective operations.
Balance Shift Example

- Example: 3D Partial Differential Equation Solver
- Scale from 8 nodes to 32768 nodes (4096x)
  - Scale problem size by 8x in each dimension (x, y, z, t)
    - Memory reduced by 8x on each node
- Compute stays fixed (8x more steps on 8x fewer points)
- Bulk Communication BW increases by 2x (8x more steps on 4x less surface area)
- Global operations increase by 8x (number of steps)
  - Binary tree goes from 3 levels to 15 levels! 5x increase
  - Latency per level increases by ? Guess 2.5x?
Scale-up by 4096x (continued)

- First-order performance model
  - T_cpu (compute & local memory)
  - T_BW (bulk cluster communications)
  - T_latency (non-overlapped short messages)

- Scaled Perf Ratio:
  Time(32768 nodes) / Time(8 nodes) =
  \[
  \frac{T_{cpu} + 2 \times T_{BW} + 100 \times T_{latency}}{T_{cpu} + T_{BW} + T_{latency}}
  \]

- #1 Priority: Eliminate or Tolerate LATENCY
Processors

• Continuing divergence of HPC & Commercial design points
  – HPC: one thread controlling many functional units and generating many concurrent cache misses
  – Commercial: many threads controlling few functional units each and each generating 1-2 concurrent cache misses.

• Recent increased interest in simpler cores
  – Too much heat from big cores
  – Too much Development Cost for complex cores
Reliability

• Reliability may be a dominant factor at Petascale
• Standard “nodes” have no need to be reliable enough for long MTBF in Petascale clusters
  – Checkpoint/restart is not credible at 1 hour crash frequency
  – New programming paradigm needed?
• Undetected error rate is much lower, but perhaps more disturbing.
  – At what point do we have to start double-checking computations?
IBM PERCS Project

• Baseline: More “Type T” systems
  – SMP nodes with traditional interconnect

• DARPA HPCS funding is allowing “Type C” thinking
  – HW/SW co-design of programming languages, compiler/runtime implementations, I/O devices and libraries – all in process now
  – Target delivery in 2008-2010 time frame, with PFLOPS scale system in 2011
Summary

• Given:
  – The HPC Market continues to change
  – Value vs Cost tradeoffs are complex and resist quantification & generalization
  – Extreme Scale systems have significantly different characteristics than more modestly sized systems

• Modularity of both cost and performance are key to spanning a broad range of application areas and cluster sizes
  – But much of the effort must fall on the customer…. 