Pacific Northwest
NATIONAL LABORATORY
Proudly Operated by Battelle Since 1965

A Critique of Performance Metrics

Adolfy Hoisie

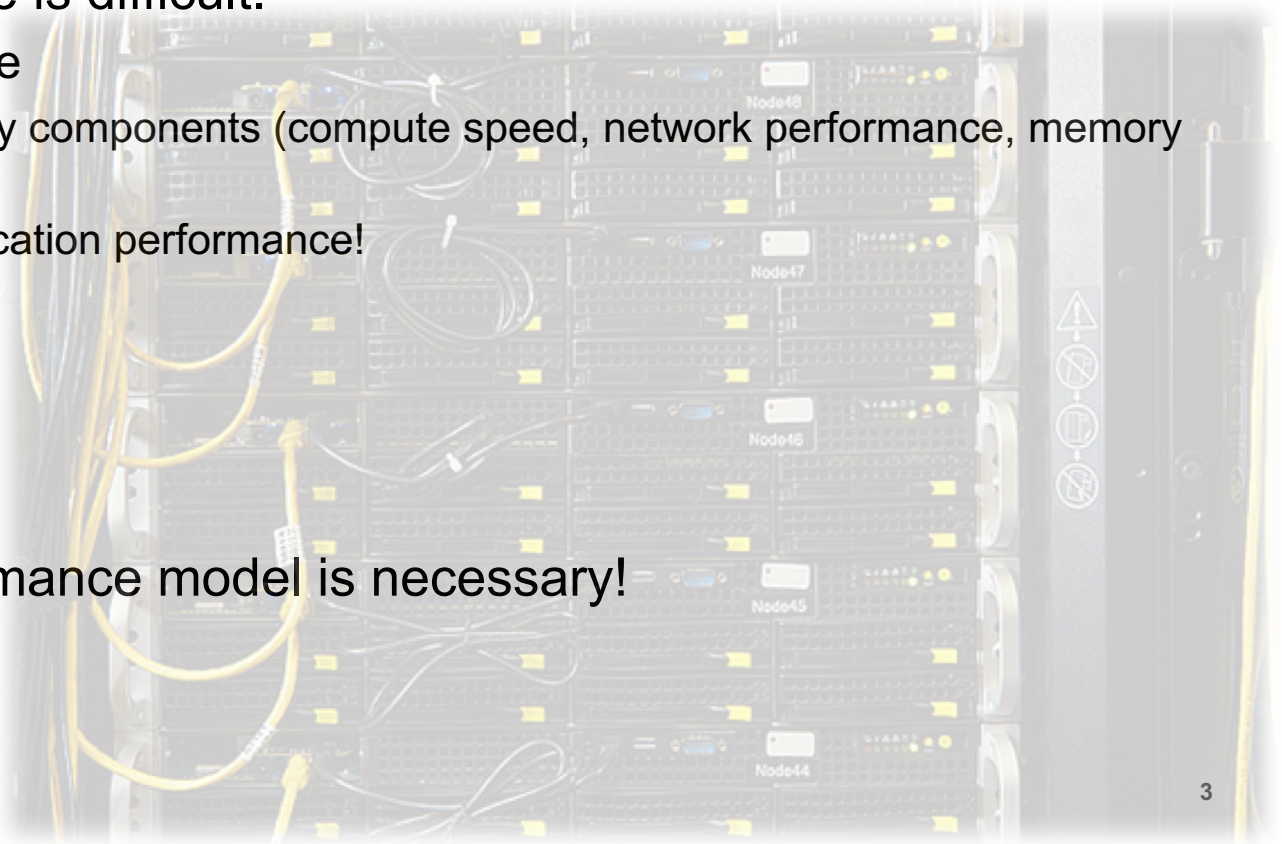
U.S. DEPARTMENT OF
ENERGY

Why the Great Interest in Performance Metrics?

- Reliance on performance metrics is tempting because:
  - Metrics appear to allow performance to be distilled into a single number
    - System *X* capable of peak performance of *N* Pflop/s
  - Metrics appear to allow rapid comparisons between systems
    - System *X* achieves 30% higher performance on LINPACK than System *Y*
  - Metrics appear to yield intuitive insight into system performance.
- However…

The Performance Metrics Road is Fraught With Peril!

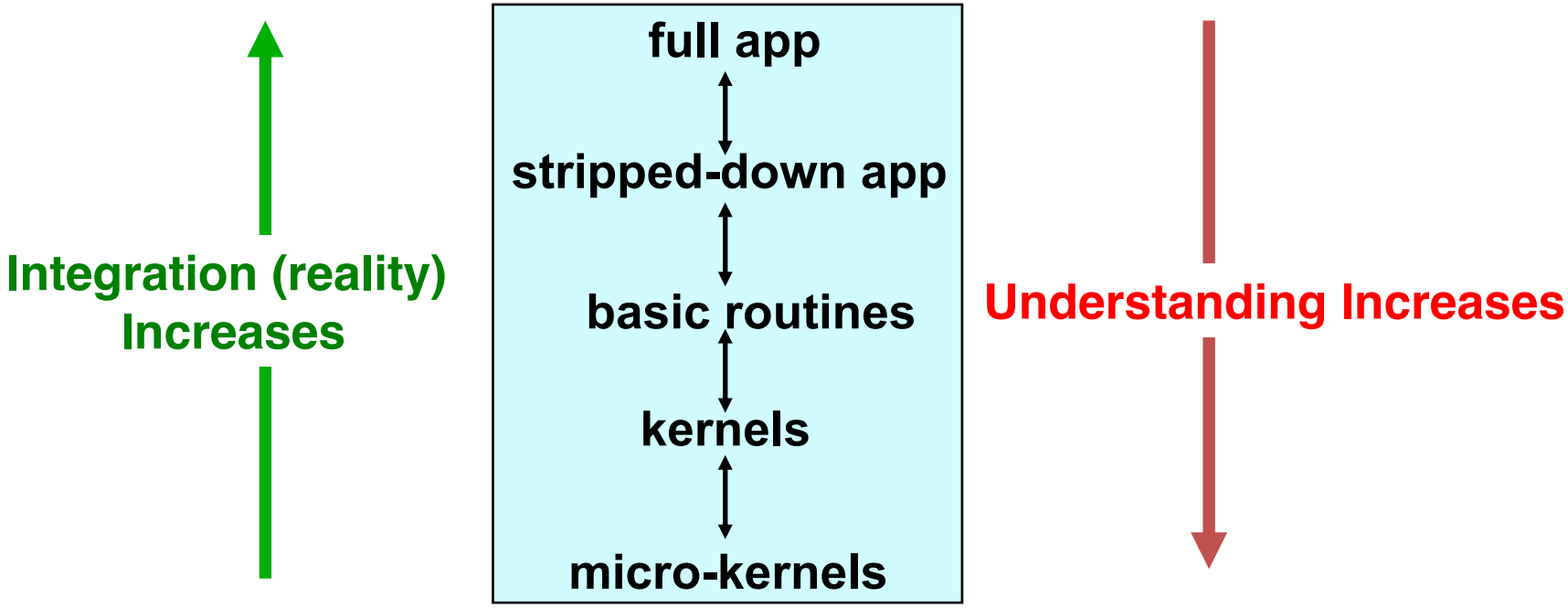
- There are so many metrics out there.
  - Some indication of the complexity of parallel application performance
- Creating metrics to describe parallel performance is difficult.
  - Metrics describe only ***aspects*** of total performance
    - Total system peak performance is impacted by many components (compute speed, network performance, memory performance, etc.)
    - Yet, we ultimately are interested in achievable application performance!
- Performance metrics are easily abused.
  - E.g., Flop/s easily manipulated with problem size
- To get the full picture, a workload-specific performance model is necessary!



Metrics Trade Realism for Understanding

Micro-kernels:

- Attempt to generalize performance
  - May represent characteristics of a large number of applications
- Are the easiest to understand and discuss
  - However, this is a poor representation of reality!



Direct Measures/Metrics

- Absolute time
  - Difference between start and finish
    - Measured as maximum dedicated wall-clock time over all processors
      - However, what constitutes “dedicated?”
      - Easiest metric to measure
  - Best performance measure
    - Used frequently by developers to track performance improvements
    - For comparisons between systems
    - For historical comparison
  - Yet, it tells us little about how well the resources are being used
    - Cannot be used to predict performance
      - Due to architectural changes
      - Due to software changes
    - Does not give any performance insight!

Efficiency as a Metric

- Measure of how well resources are being used
- Of limited validity by itself
  - Can be artificially inflated
  - Biased toward slower systems and unoptimized algorithms
- Example 1: Efficiency of applications

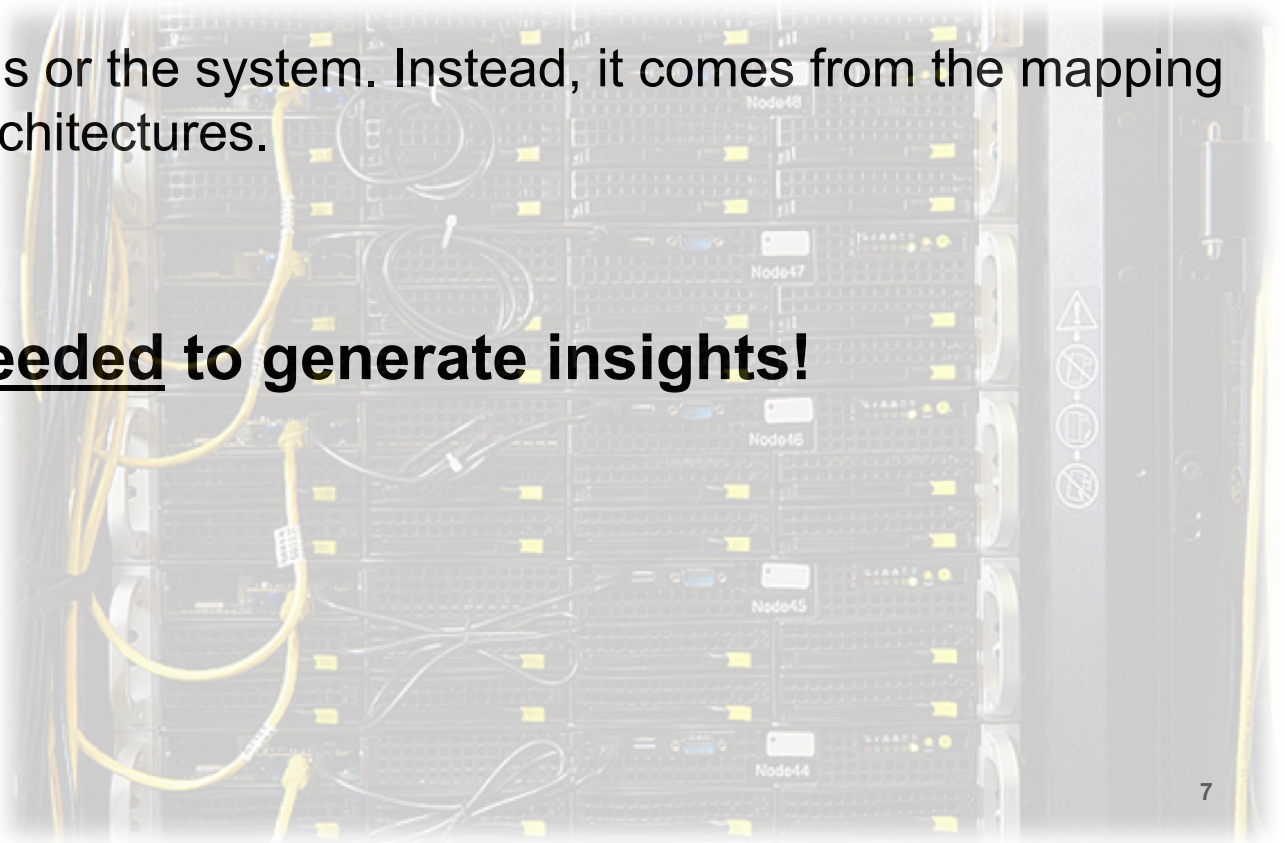
	Solver Flops	Flops	Mflop/s	% Peak	Time (s)
Original	64 %	29.8 x 10^9	448.8	5.6 %	66.351
Optimized	25 %	8.2 x 10^9	257.7	3.2 %	31.905

- Example 2: Efficiency of systems
  - Code A on System X
    - (250 MHz, 500 MFLOPS Peak per CPU, 2 FLOPS per CP):
    - Time = 522 sec; MFLOPS = 26.1 (5.2% of peak)
  - Code A on System Y
    - (900 MHz, 3600 MFLOPS Peak per CPU, 4 FLOPS per CP):
    - Time = 91.1 sec; MFLOPS = 113.0 (3.1% of peak)

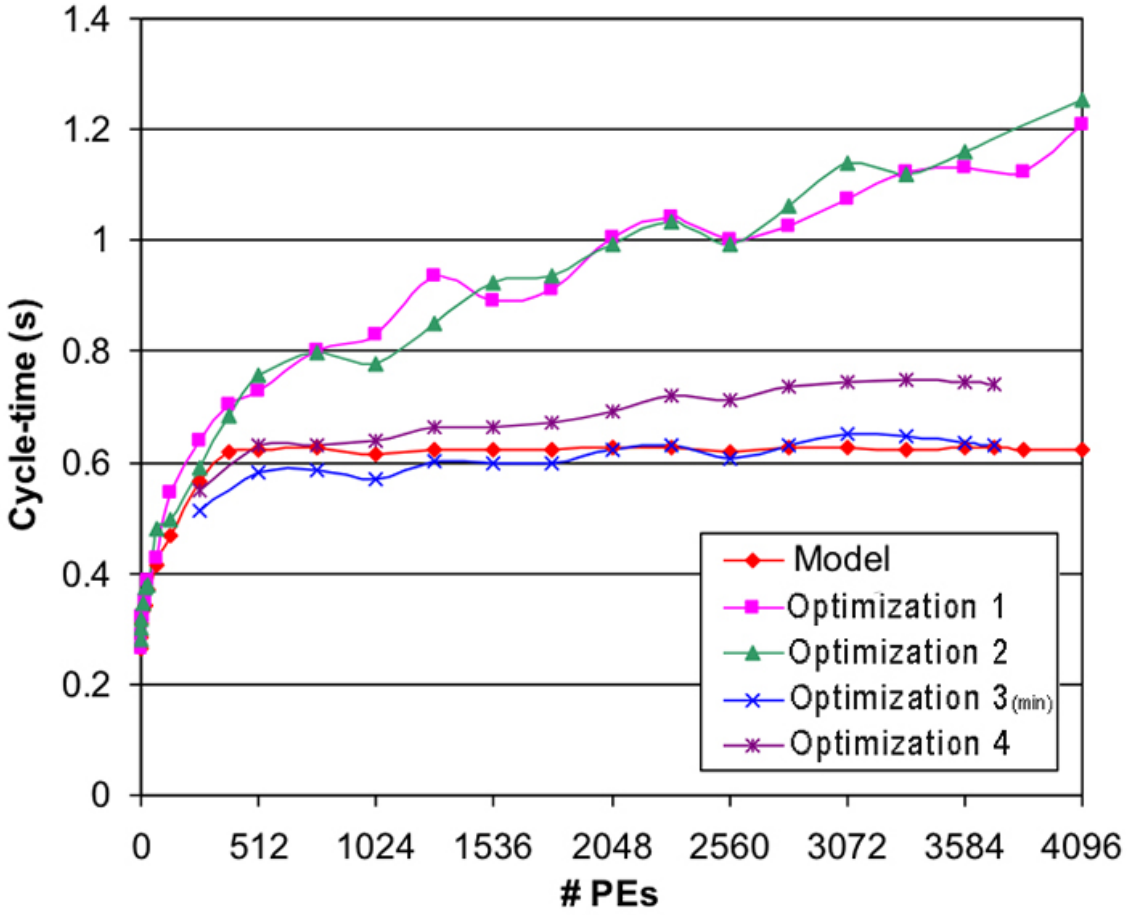
From Metrics to Models

- Application-oriented metrics are affected by algorithmic changes, input deck, software engineering.
- System-oriented metrics are affected by various system knobs, optimizations, and transient effects.
- Performance does not come from the applications or the system. Instead, it comes from the mapping of the algorithms/applications onto the system architectures.

A performance model is <u>needed</u> to generate insights!

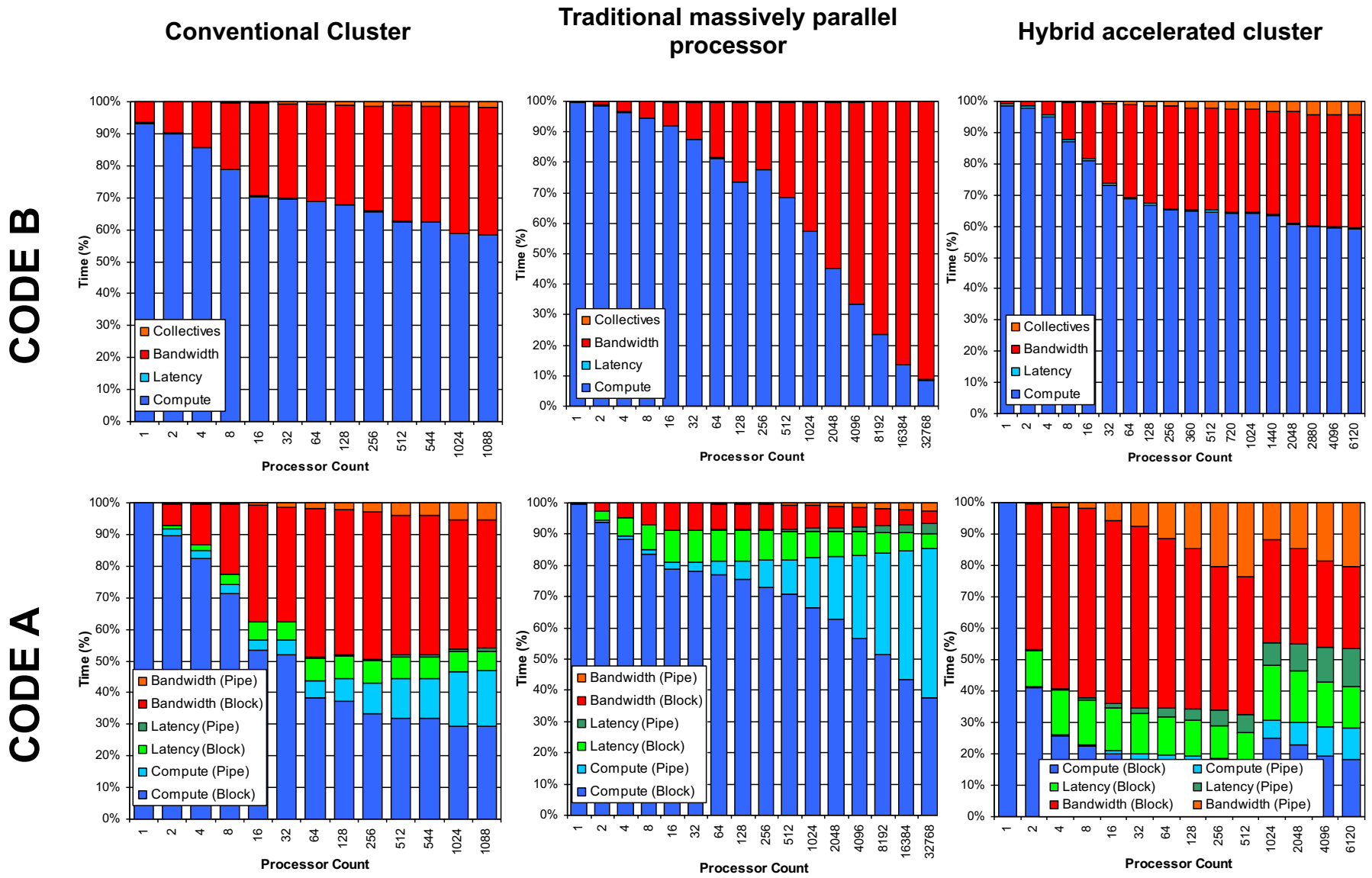


Achieved vs. Achievable Performance



- **Performance within ~10% of the expectation**
- **Without a model, we would not have identified—and solved—the performance issues!**

Insights



Simple Metrics Do Not Provide the Whole Story

- The problem is not the metrics themselves but how they are used.
- It is always dangerous to use a single metric by itself.
  - This is especially true when examining relative performance
    - How does System *A* compare with System *B*?
  - Keep in mind that micro-kernels and benchmarks only approximate reality
    - Application performance may be markedly different

To gain true insight into application performance, a performance model is necessary.