Computer Systems Performance Analysis and Benchmarking (37-235)

Analytic Modeling

Simulation

Measurements / Benchmarking

Lecture/Assignments/Projects:

Dr. Christian Kurmann

Textbook:

Raj Jain, “The Art of Computer Systems Performance Analysis”, 1991 Wiley & Sons, New York

Topic of Today:

- **Memory Systems Benchmarks (ECT-memperf)**
- **Modelling of an application (OPAL)**

Performance Evaluation, -Modeling and -Prediction of a Message Passing Molecular Simulation Code

M. Taufer, T. Stricker

Laboratory for Computer Systems

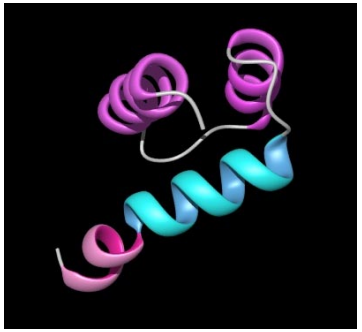
ETH - Swiss Institute of Technology

CH-8092 Zurich

Eidgenössische Technische Hochschule Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Swiss Federal Institute of Technology Zurich

Motivation



- Molecular simulation code: Opal
- Different parallel platforms to run on:

Cray J90

Cray T3E

Different Clusters of PCs

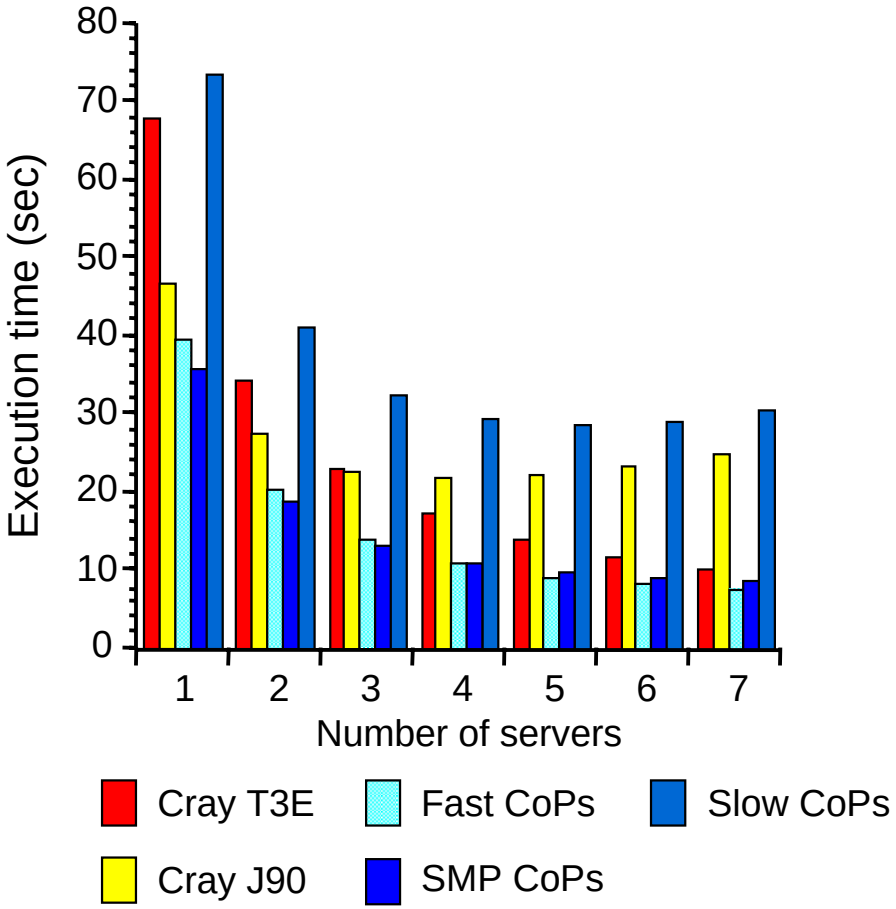
which ones will work?

which one is most cost effective?

Motivation

- Little is known about the interaction between the application and the platform
- Our solution is using:
  - analytical modeling
  - measurements

  during:
  - application design
  - parallelization
  - performance analysis



Outline

- Introduction to the code
  - application: Opal
  - middleware: Sciddle, PVM
- Performance modeling
  - parameter space
  - time complexity
  - measurements
  - calibration
- Performance prediction on different platforms
- Summary
- Conclusion

Application: Opal

- Molecular dynamic simulation of proteins and nucleic acids
  - P.Luginbühl, P.Güntert, M.Billeter, K.Wüthrich. *The new program Opal...* Journal of Biomolecular NMR, [1996]
- Parallel version:
  - replicated-data (RD) method
  - client-server paradigm
  - list of active pairs (pairs of mass centers)
    - radius as cut-off parameter
  - simulation phases: update and energy computation

Related Work

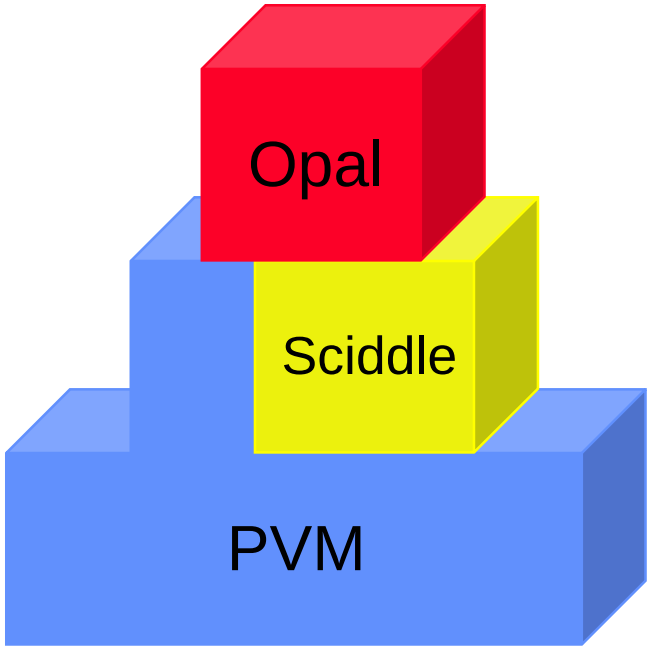
- Similar packages:
  - Amber

- Alternative parallelization:
  - replicated-data (RD) method
  - geometric- or space-decomposition (SD) method
  - force-decomposition (FD) method

Our work is only about performance modeling and not about computational chemistry or numerical algorithms.

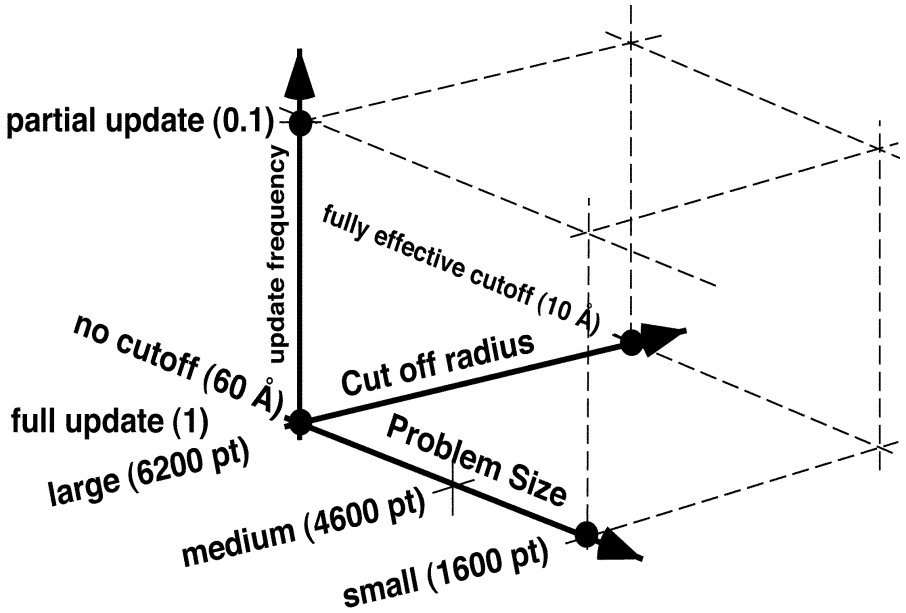
Middleware

- Sciddle for PVM:
  - remote procedure call system
  - extension to PVM
  - highly portable communication library
  - ☹ for a single J90 vector SMP
    shared memory support is better
  - ☺ for a cluster of *four* J90s:
    message passing over Hippi
  - ☺ for a cluster of PCs:
    message passing over
    - Ethernet (UDP/IP)
    - Myrinet (Fast Messages)



Parameter Space of an Application Run

- Experimental factors
  - problem size
  - number of servers
  - update parameter
  - cut-off parameter
- Response variables
  - parallel computation time
  - sequential computation time
  - communication time
  - synchronization time
  - idle time
- Full factorial design



Analytical Modeling for the Execution Time of Opal

$$t_{opal} \approx s \begin{cases} \frac{1}{2p}(a_2 u(1-2\gamma)^2 + a_3)n^2 + \left(\frac{\alpha}{a_1}p(u+2) - \frac{1}{2p}(a_2 u(1-2\gamma)+a_3) + a_4\right)n + 2(u+1)(pb_1 + b_5) \\ \frac{1}{2p}(a_2 u(1-2\gamma)^2)n^2 + \left(\frac{\alpha}{a_1}p(u+2) + \left(\frac{1}{p}a_3\bar{n} - \frac{1}{2p}a_2 u(1-2\gamma)\right) + a_4\right)n + 2(u+1)(pb_1 + b_5) \end{cases}$$

- parallel computation time:

$$t_{par\_comp} \approx \begin{cases} O(n^2) \text{ no cut-off} \approx O(p^{-1}) \\ O(n) \text{ with cut-off} \end{cases}$$

- communication time:

$$t_{comm} \approx O(n) \approx O(p)$$

Measurements on Cray J90

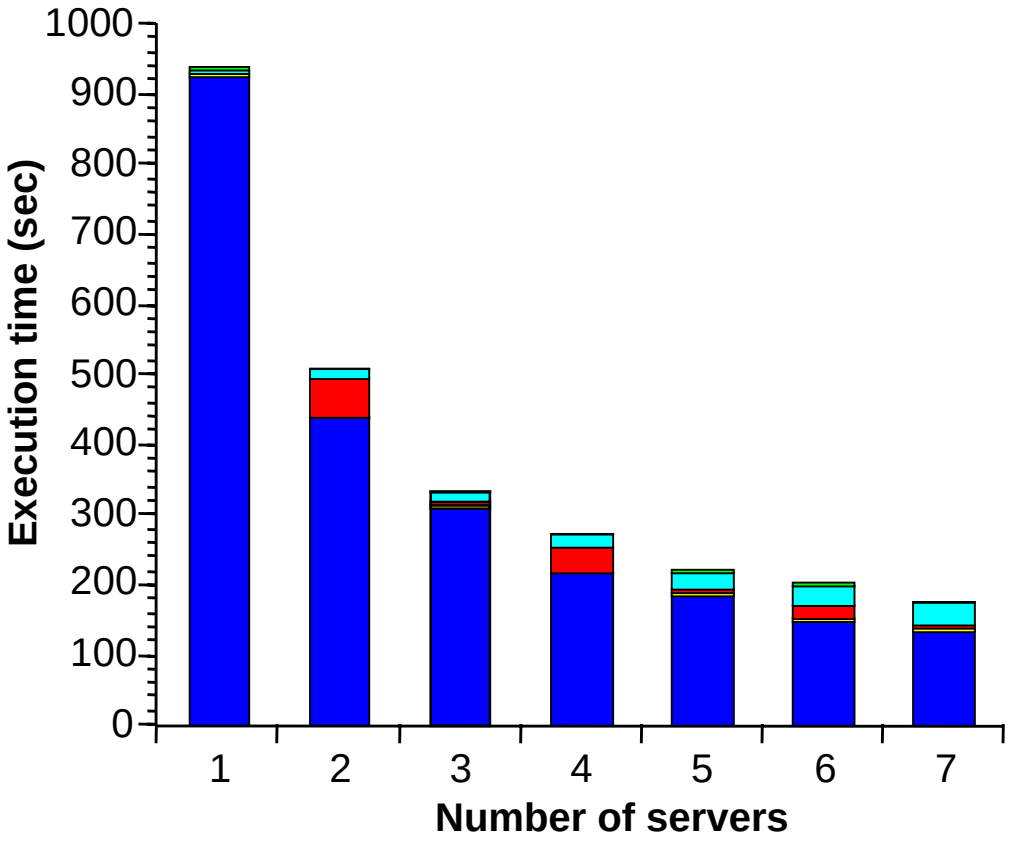
Problem with middleware

- High overlap of computation and communication
  - no support for:
    - detailed quantification of elapsed time
    - correct accounting of elapsed time
- Precise timing model
  - give up some overlap
  - introduction of additional synchronization: PVM barrier

Less than 5% slowdown over the code with overlap

Understanding the Execution Time

No cut-off

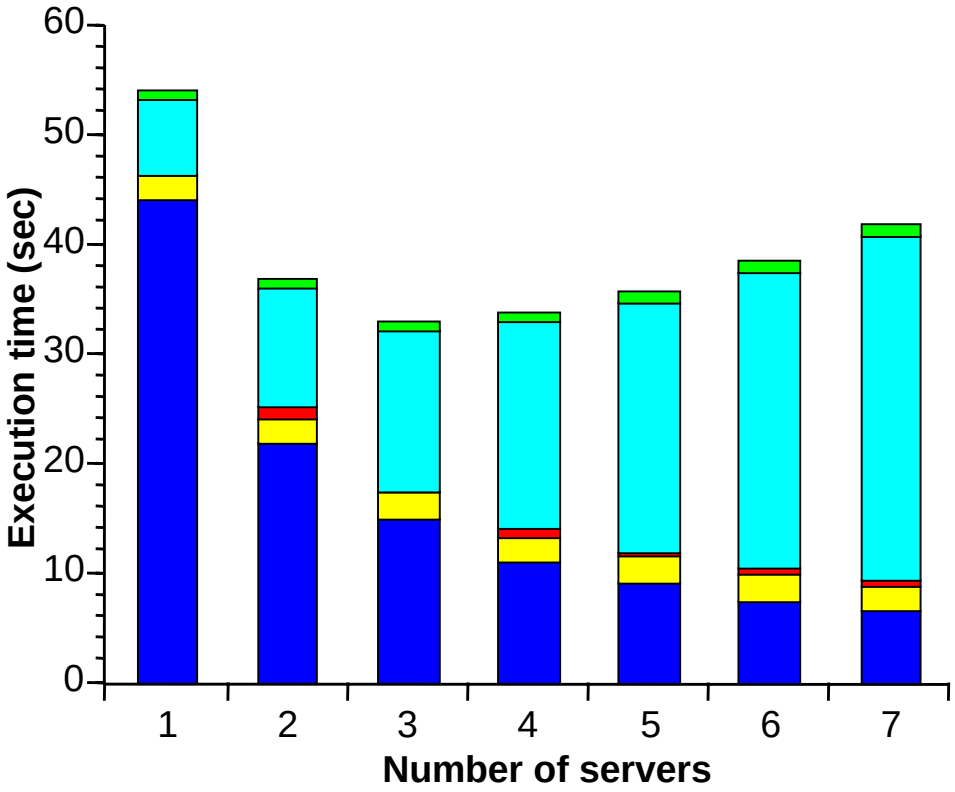


- **parallel comp. time**
- **sequential comp. time**
- **idle time**
- **communication time**
- **synchronization time**

Platform:
Cray J90 Classic

Understanding the Execution Time

With cut-off

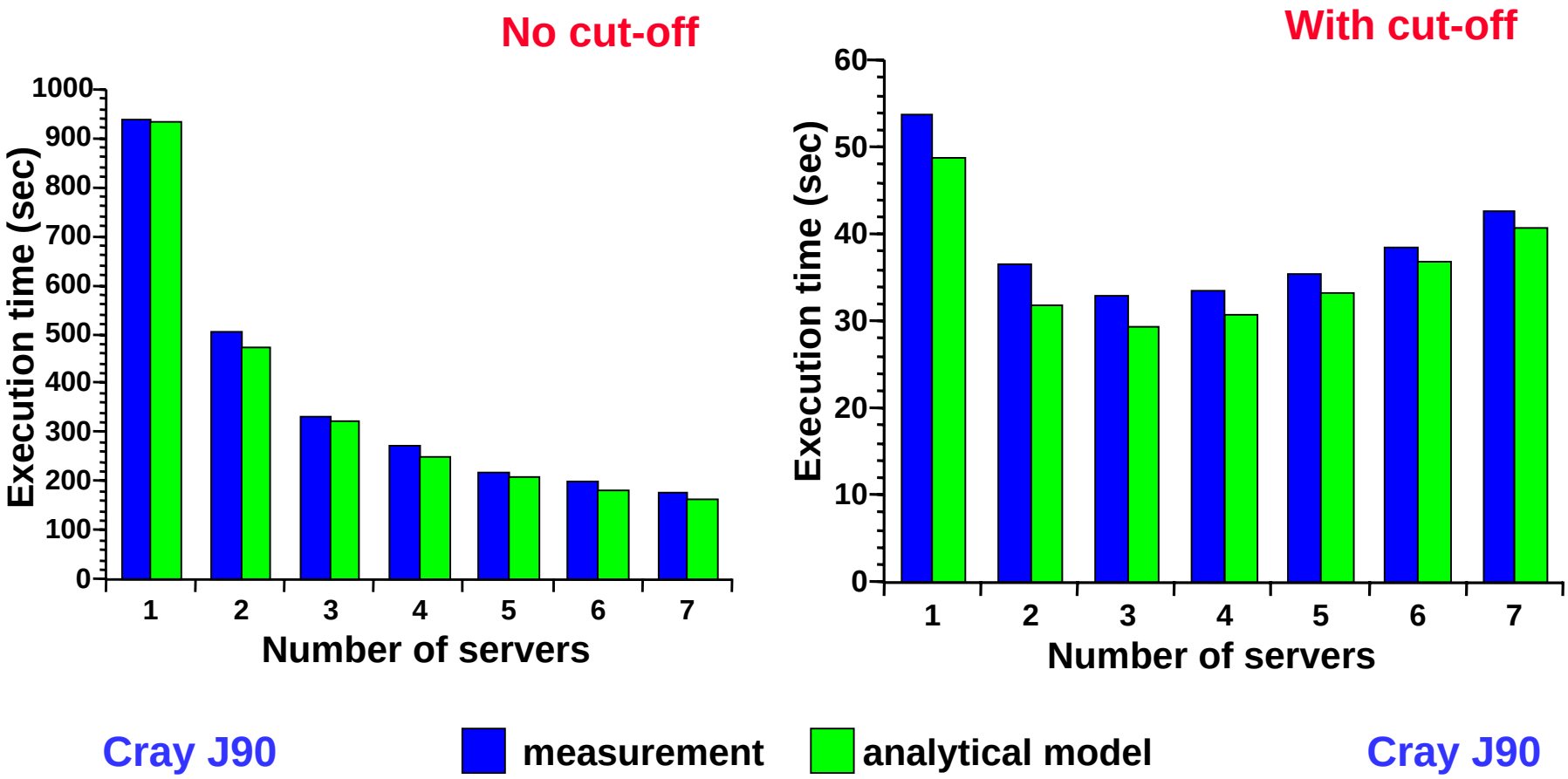


- **parallel comp. time**
- **sequential comp. time**
- **idle time**
- **communication time**
- **synchronization time**

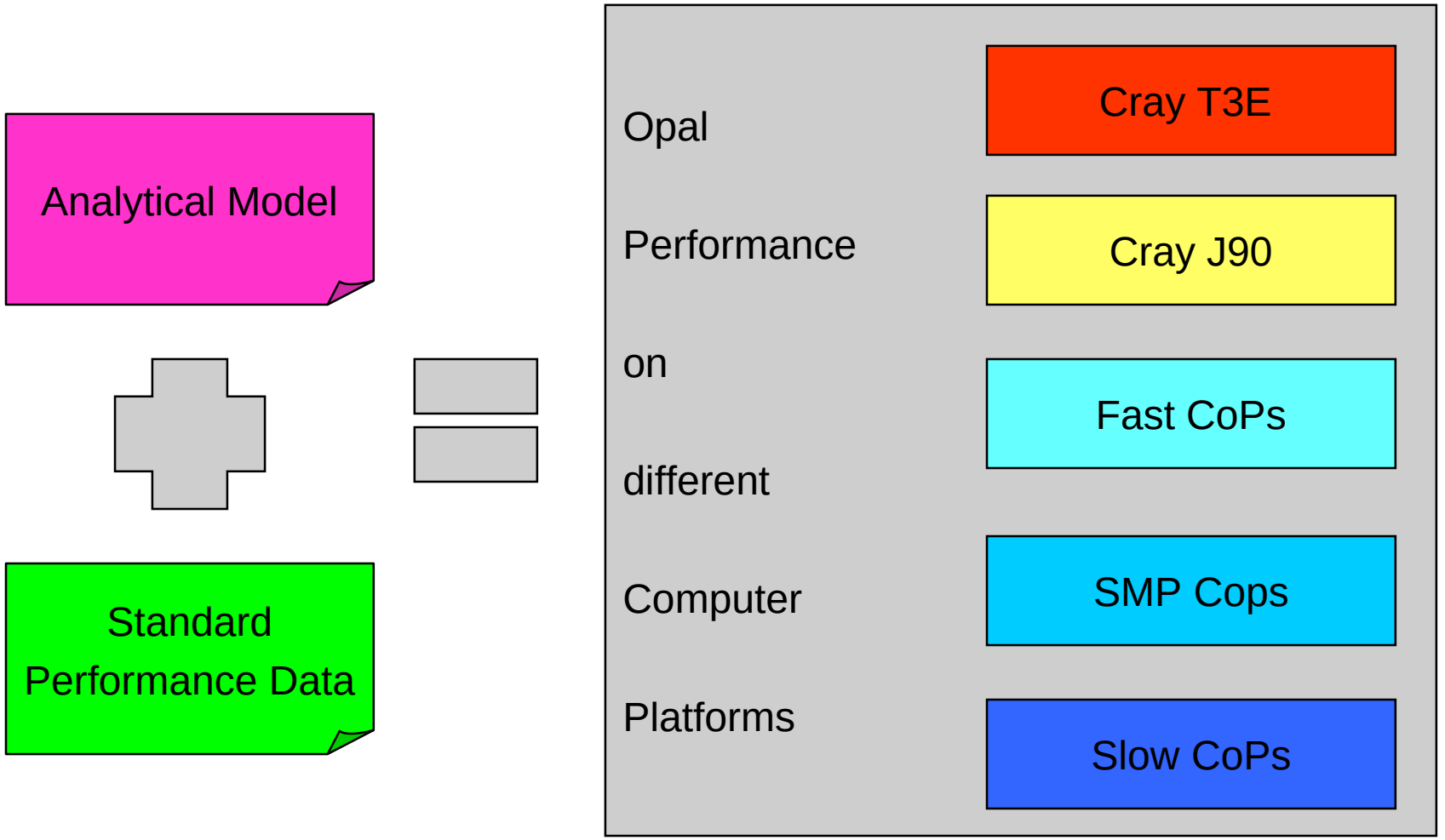
Platform
Cray J90 Classic

Calibration between Analytical Model and Measurements

Measured time vs. analytical model time



Performance Prediction on different Platforms



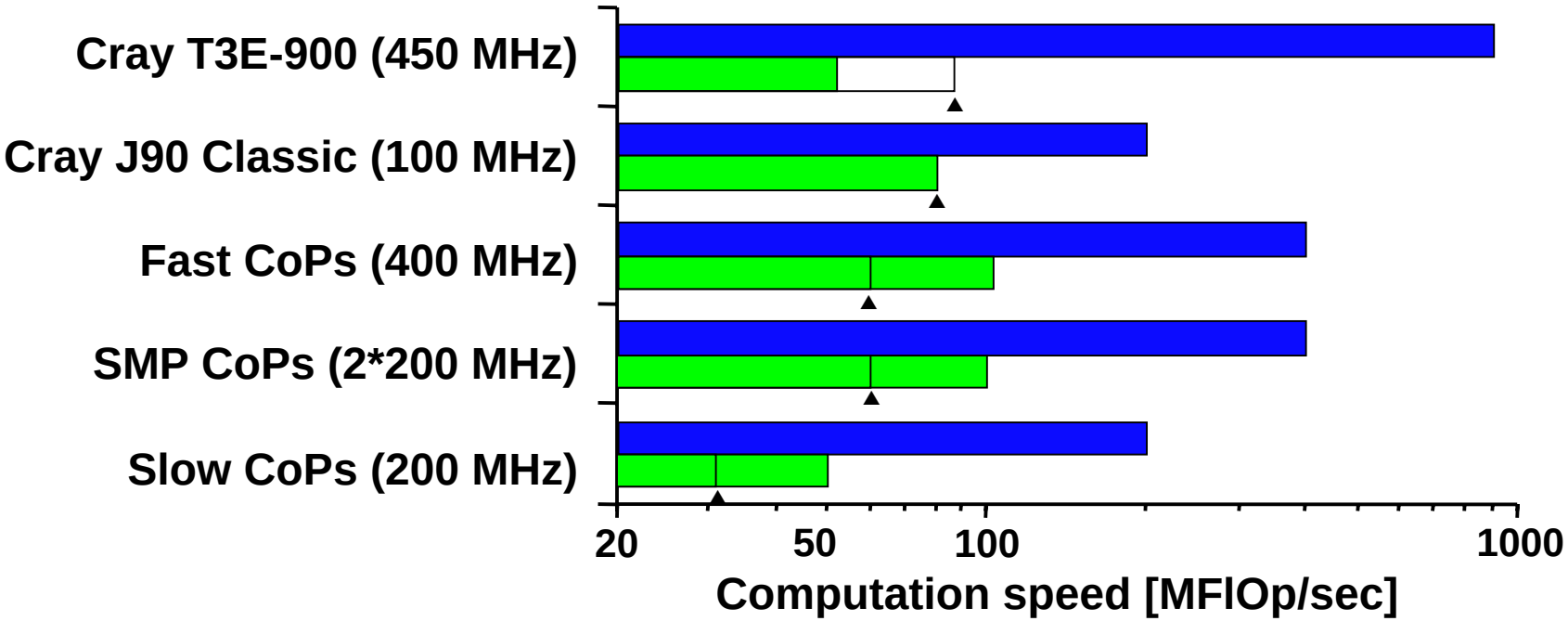
Performance Prediction on different Platforms

- Cray T3E → high-end MPP supercomputer
- Cray J90 → low cost vector supercomputer
- fast CoPs → medium cost solution
  Gigabit/s Myrinet interconnection
- SMP CoPs → built with commodity components
  SCI shared memory interconnection
- slow CoPs → low cost solution
  Ethernet 100BaseT

Opal as Micro-benchmark

- Inner loop of Opal on one processor
- 64 bit arithmetic (IEEE, Cray) - correct results
- Amount of work for 10 iterations
  - CoPs, Pentium II: **327 MFlOp**
  - Cray J90: **497 MFlOp**
  - Cray T3E, Alpha 21164 **811 MFlOp**
- Measured with performance counting hardware

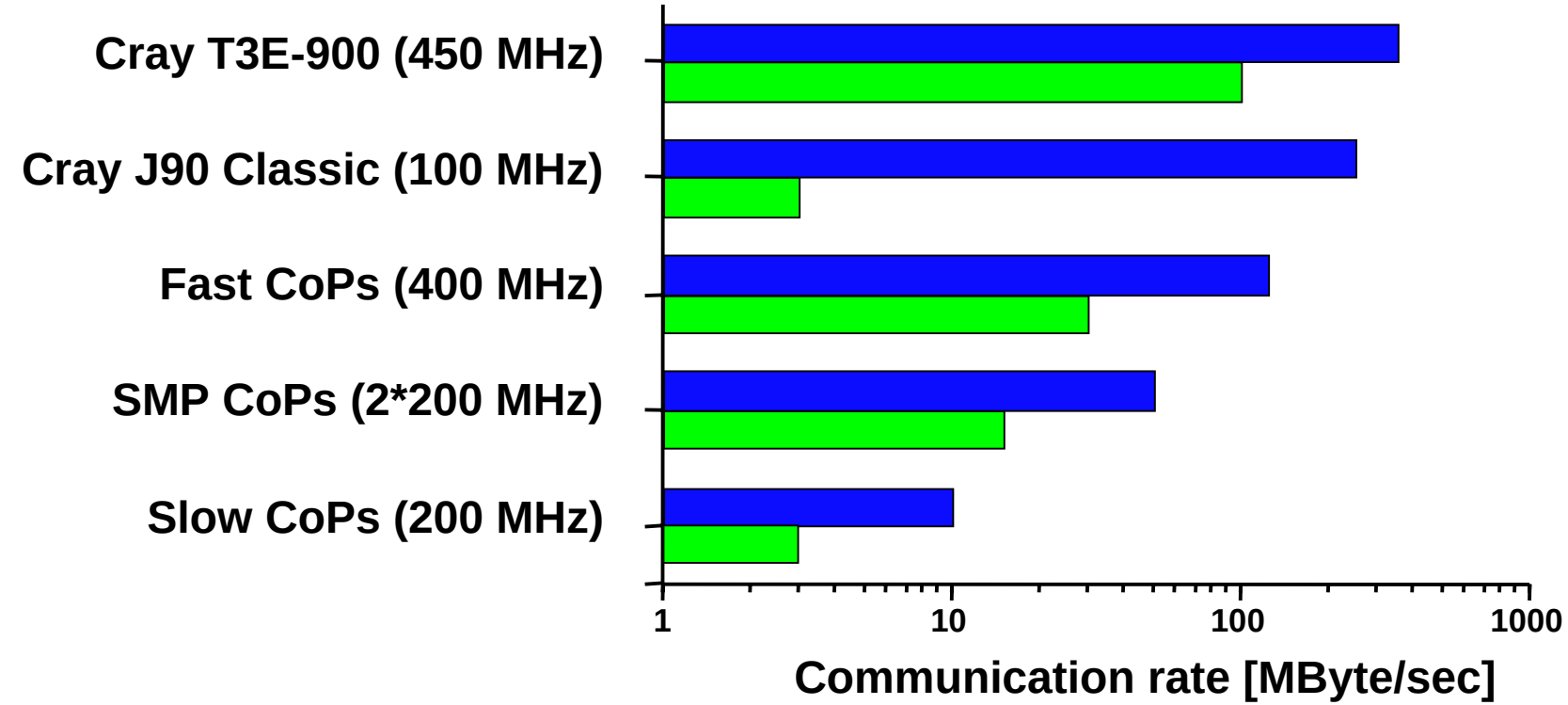
Machine Parameters: CPU Speed [MFlOp/sec]



Theoretical peak speed per processing node [MFlOp/sec]

Measured and adjusted computation speed per node [MFlOp/sec]

Machine Parameters: Communication Speed [MByte/sec]

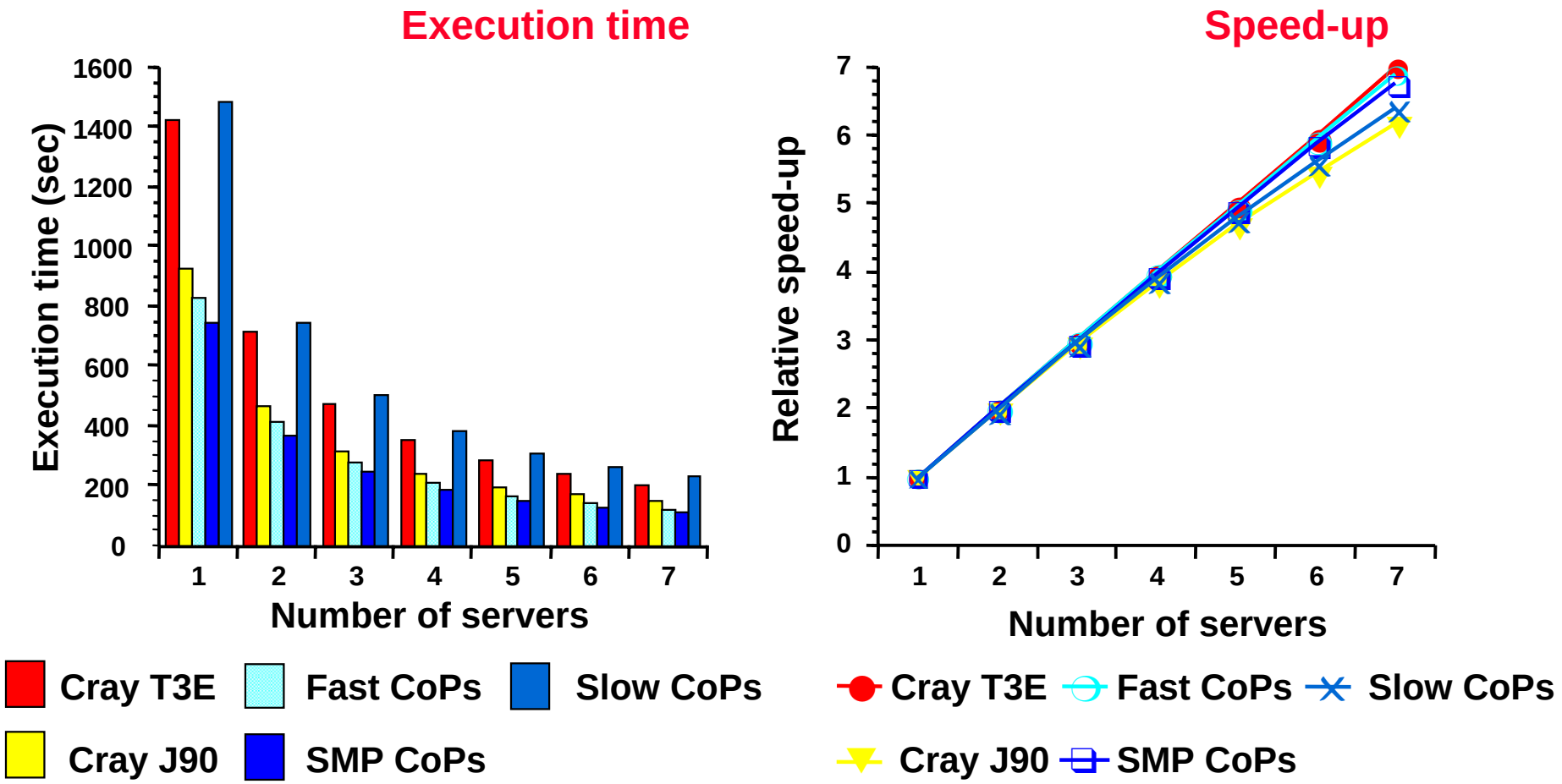


Theoretical peak communication speed on a link [Mbyte/sec]

Observed communication speed in PVM / MPI / Sciddle [Mbyte/sec]

Performance Prediction on different Platforms

No cut-off



Performance Prediction on different Platforms

With cut-off

Summary

- An analytical complexity model and a careful instrumentation for performance monitoring leads to a better understanding of the resource demands of a parallel application.
- Interesting anomalies in the implementation:
  - load imbalance for even number of servers
  - differing number of floating point operations for different processor types
- Our model allows predictions with good certainty on different Platforms:
  - how would the application run on: fast CoPs, SMP CoPs, slow CoPs.

Conclusion

- Middleware is dangerous:
  - it *interferes* with performance evaluation
    (i.e., the accounting of the execution time components).
- Looking just at speed-up is not sufficient to prove good performance:
  - keep an eye on the *total computation time*
- Prediction for execution times and speed-up figures indicate that a well designed *Cluster of PCs* can achieve an efficiency similar to the *Cray J90* and *Cray T3E* ***for the application Opal***.