A Relative Development Time Productivity Metric for HPC Systems

Andrew Funk, Jeremy Kepner
MIT Lincoln Laboratory

Victor Basili, Lorin Hochstein
University of Maryland

Ninth Annual Workshop on High Performance Embedded Computing

MIT Lincoln Laboratory
Lexington, MA

20 - 22 September 2005

This work is sponsored by Defense Advanced Research Projects Administration, under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.

Outline

- **Overview**
- Analysis
- Summary

- ***General Productivity Formula***
- ***Relative Development Time Productivity Metric***
- ***Experiments***

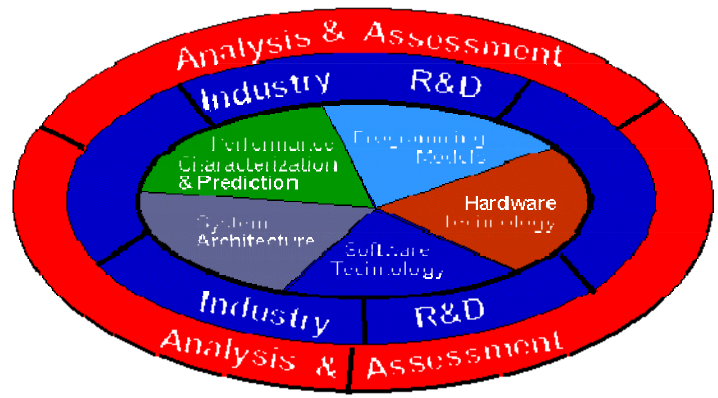
High Productivity Computing Systems

Goal:

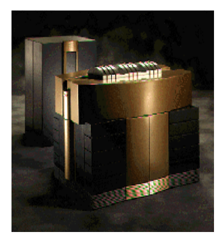
- **Provide a new generation of economically viable high productivity computing systems for the national security and industrial user community (2010)**

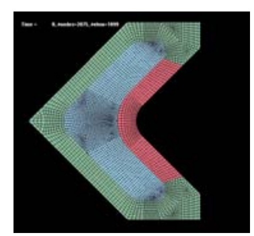
Impact:

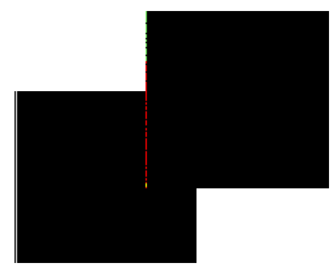
- **Performance** (time-to-solution): speedup critical national security applications by a factor of 10X to 40X
- **Programmability** (idea-to-first-solution): reduce cost and time of developing application solutions
- **Portability** (transparency): insulate research and operational application software from system
- **Robustness** (reliability): apply all known techniques to **protect against outside attacks**, hardware faults, & programming errors

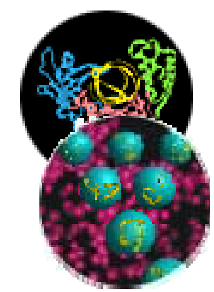


HPCS Program Focus Areas









Applications:

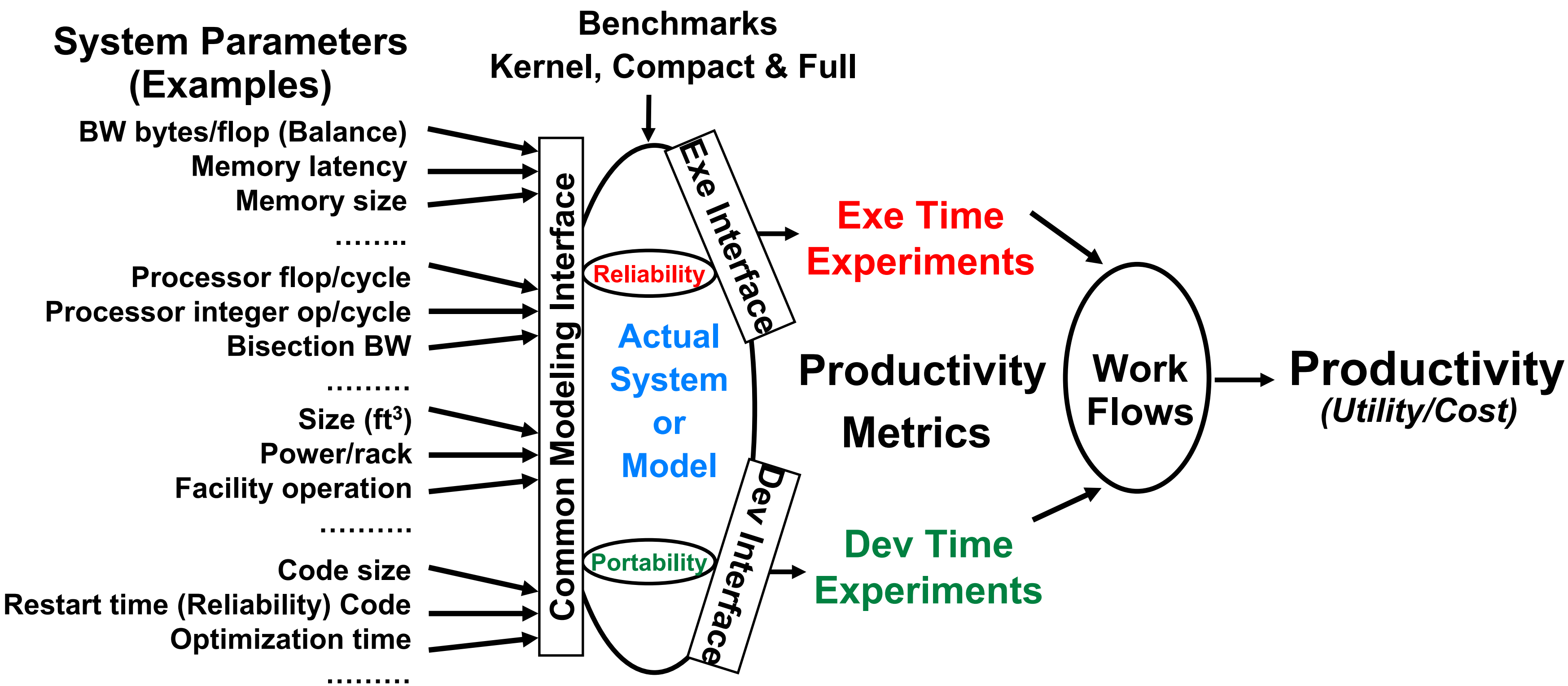
- Intelligence/surveillance, reconnaissance, cryptanalysis, weapons analysis, airborne contaminant modeling and biotechnology

Fill the Critical Technology and Capability Gap

Today (late 80's HPC technology)…..*to*…..Future (Quantum/Bio Computing)

Relative Development Time Productivity

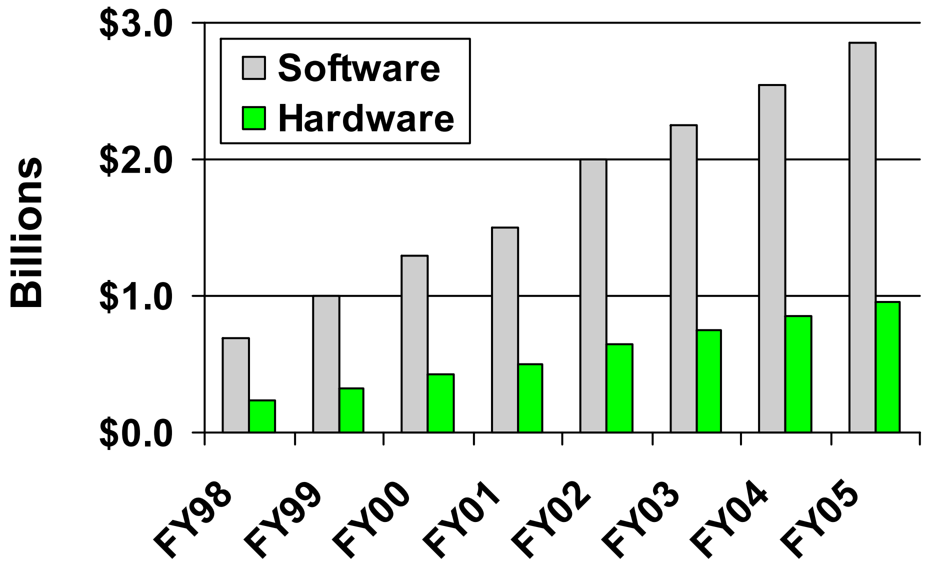
Evaluating Productivity



- **Unique combined focus from the beginning on:**
  - **Designing petascale systems for a broad range of missions**
  - **Improving the usability of such systems**
- **Developing a methodology for measuring these improvements is the focus of the Productivity Team**

Measuring Productivity

- HPC and HPEC communities have experience measuring execution performance
- Software development is often the dominant cost driver associated with developing DoD High Performance Embedded Computing (HPEC) Systems



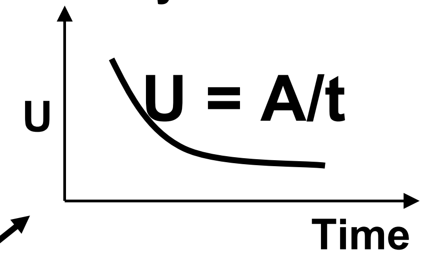
Source: HPEC market study, 2001

- Need metrics that incorporate both execution performance and software development cost for HPC and HPEC systems

General Productivity Formula

Utility is value user places on getting a result at time T

Utility Function



Overall system productivity

$$\Psi \equiv \frac{U}{C} = \frac{U(T)}{C_M + C_O + C_S}$$

Overall system cost

Machine cost

Operating costs include admin time, electric and building costs

Software costs include time spent by users developing their codes

Developing small codes (~1000 lines)

Relative Development Time Productivity Metric (Small Codes)

$$\Psi_{\text{small codes}} = \frac{\text{Application Performance}}{\text{Cost of writing code}}$$

- Speedup is major concern
- Operating and machine costs not seen

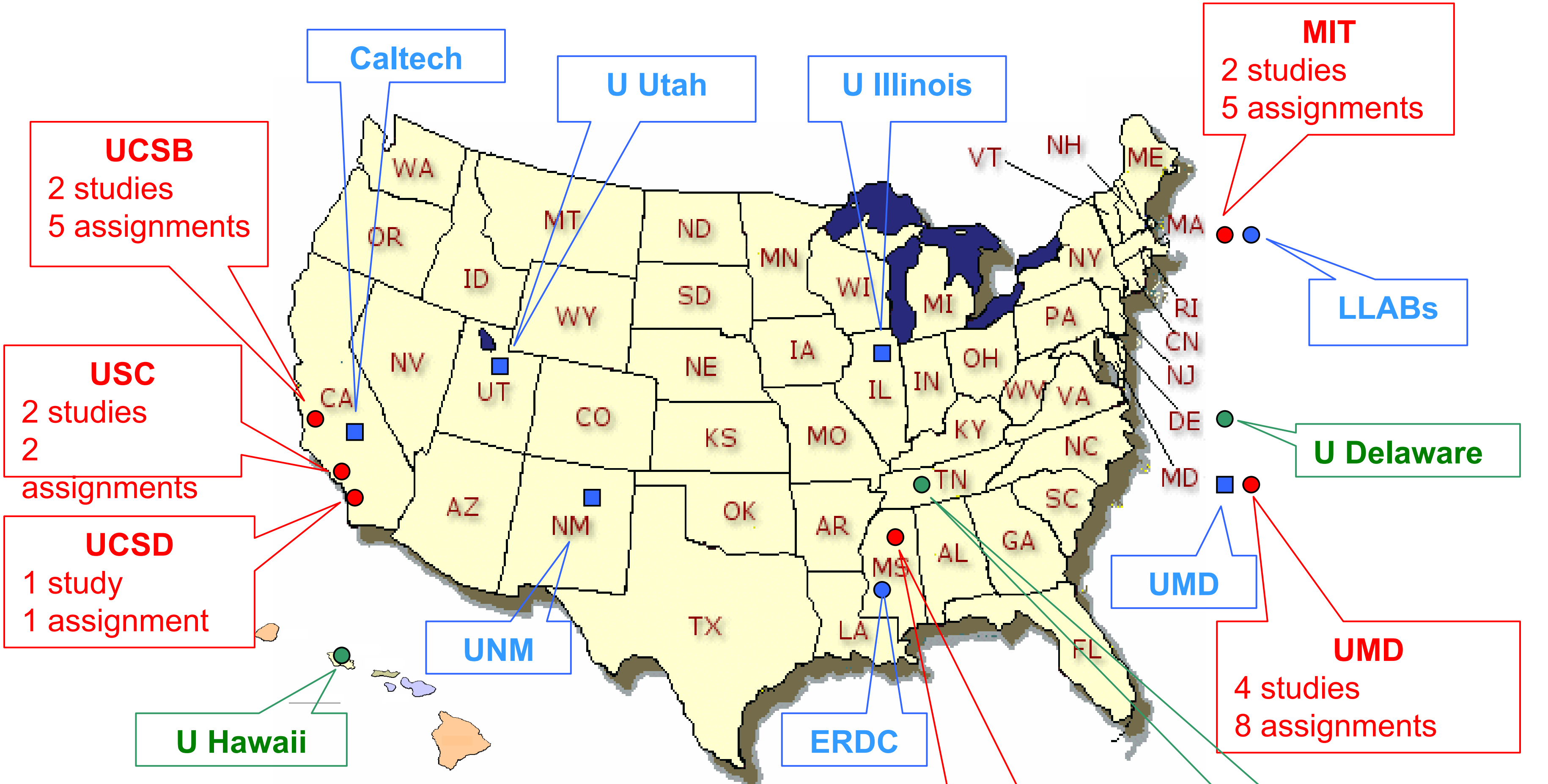
$$\Psi_{\text{relative}} = \frac{\text{Speedup}}{\text{Relative Effort}}$$

- Relative code size used for relative effort

Experiments

- **The Relative Development Time Productivity metric ($\Psi_{relative}$) was applied to:**
  - **NAS Parallel Benchmarks (NASA)**
    - 8 kernels and pseudo-apps from Computational Fluid Dynamics (CFD)
    - C/Fortran, MPI, OpenMP, Java, High Performance Fortran (HPF)
  - **HPC Challenge (University of Tennessee)**
    - High Performance Linpack (HPL, Top500), FFT, Stream, Random Access
    - Serial C and C+MPI, Serial and parallel high level language (Matlab)
  - **Classroom assignments (University of Maryland)**
    - Various textbook parallel programming exercises
    - Serial C and Matlab, MPI, OpenMP, Matlab*p

Studies are national in scope



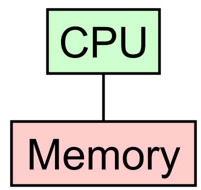
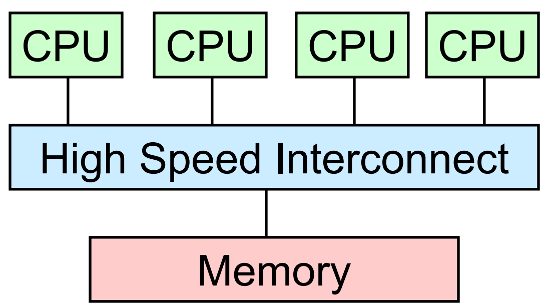
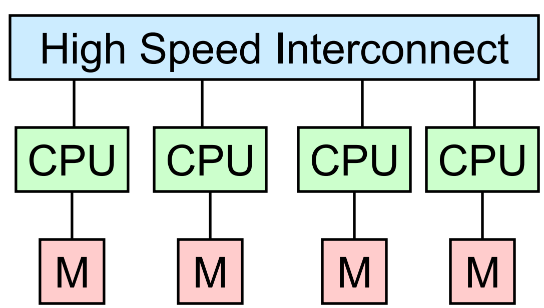
University studies **Case studies or Interviews**

Vendor interaction ○ **In progress/Completed**

□ **Planned**

MIT Lincoln Laboratory

Programming Models and Languages

Memory Model / Architecture	Programming Languages Studied
Serial 	C/C++ Fortran Java Matlab
Shared Memory 	C/Fortran + OpenMP Multithreaded Java High Performance Fortran (HPF) Co-Array Fortran (CAF) ZPL
Distributed Memory 	C/Fortran + MPI Matlab*P pMatlab

Outline

- Overview
- **Analysis**
  - ***NAS Parallel Benchmarks***
  - *HPC Challenge*
  - *Classroom Assignments*
- Summary

The NAS Parallel Benchmarks

The NAS Parallel Benchmarks (NPB) are a set of 8 programs designed to help evaluate the performance of parallel supercomputers. The benchmarks, which are derived from computational fluid dynamics (CFD) applications, consist of five kernels and three pseudo-applications

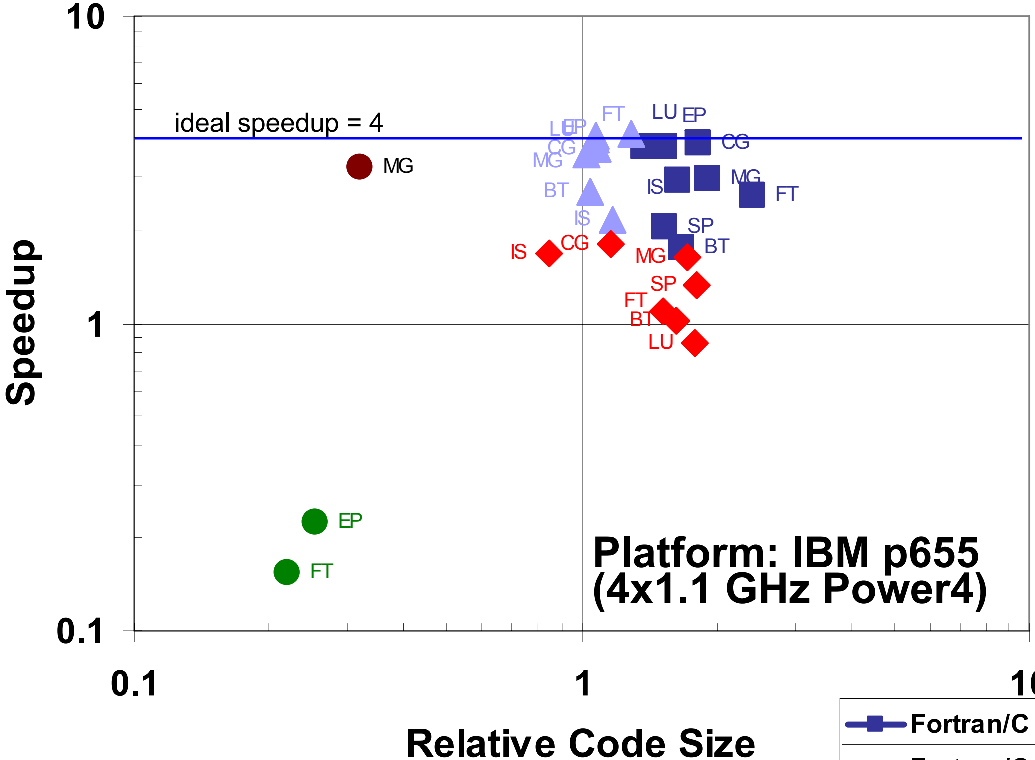
Benchmark	Code size*	Description
BT app	2,528	Block Tridiagonal solution to 3D compressible Navier-Stokes equations
CG	574	Solving unstructured sparse linear system by Conjugate Gradient method
EP	141	Embarrassingly Parallel random number generation
FT	560	A 3-D fast-Fourier Transform partial differential equation benchmark
IS	450	Parallel Sort over small Integers
LU app	2,554	Lower and Upper triangular solution to Navier-Stokes equations
MG	863	A simple 3D Multi-Grid benchmark
SP app	2,120	Scalar Pentadiagonal solution to Navier-Stokes equations

* measured in Source Lines of Code (SLOC)

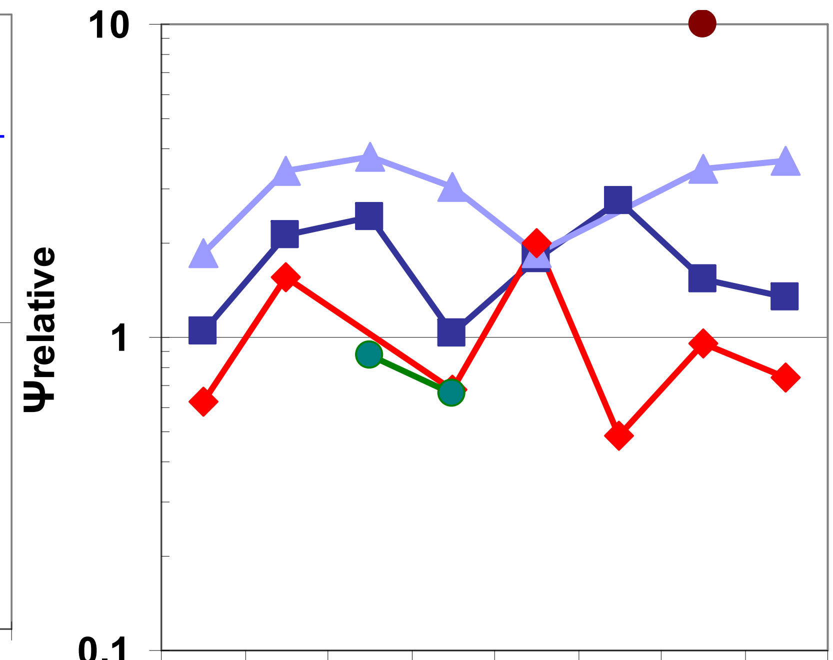
http://www.nas.nasa.gov/Software/NPB/

NAS Parallel Benchmarks

Speedup vs Relative Code Size



Ψrelative vs Benchmark

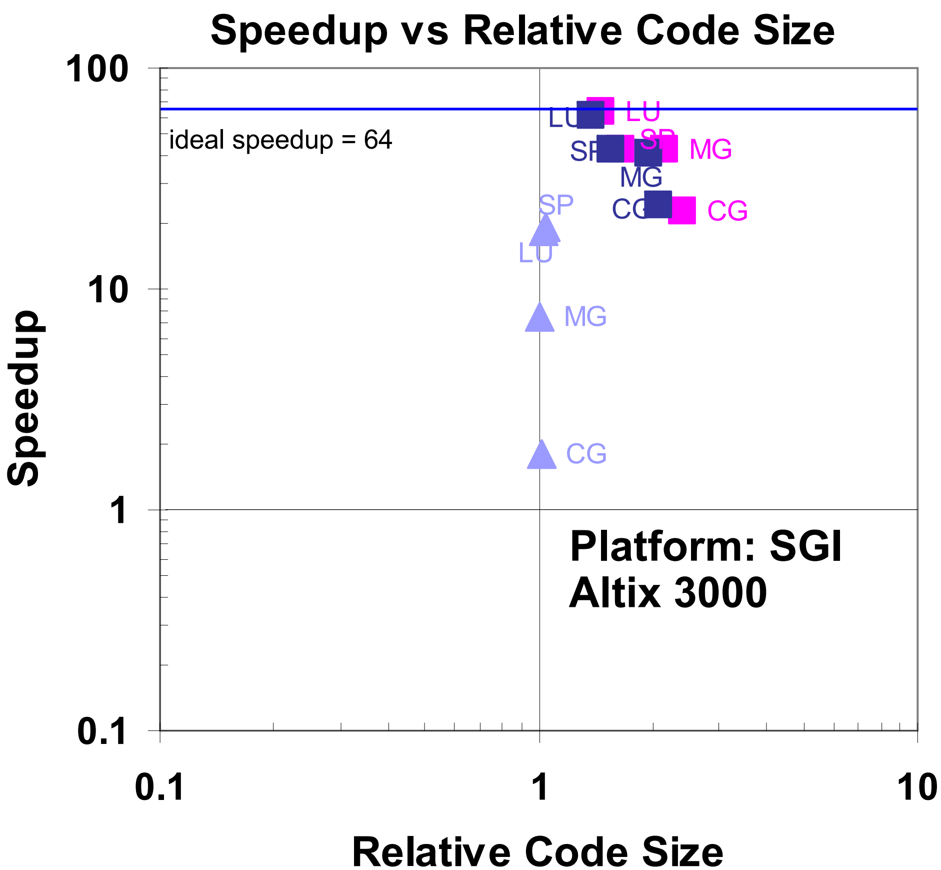


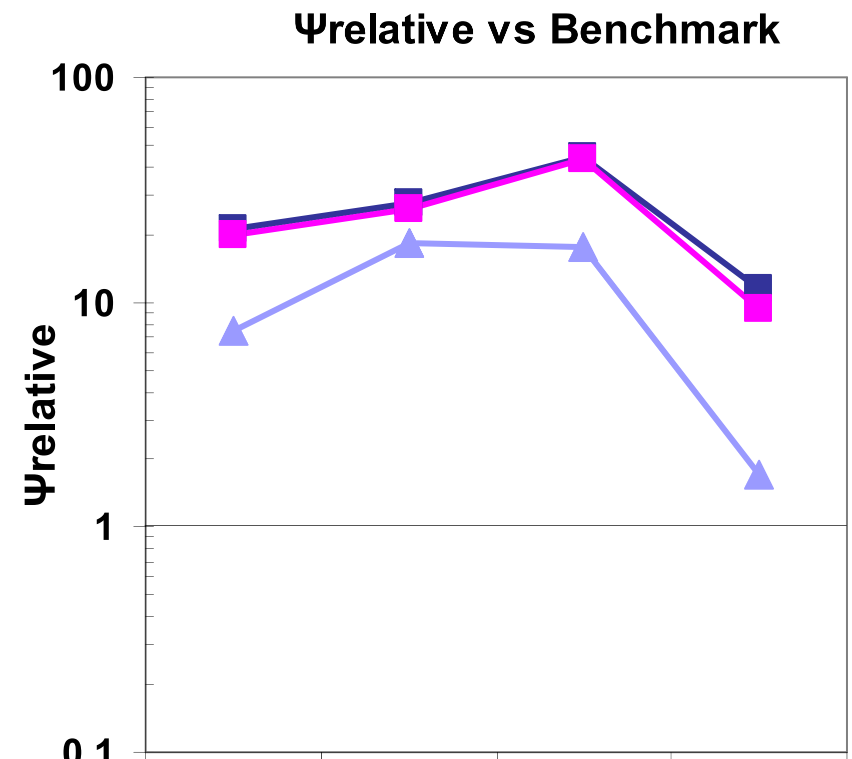
	BT	CG	EP	FT	IS	LU	MG	SP
Fortran/C + MPI	1.05	2.12	2.42	1.03	1.78	2.72	1.54	1.34
Fortran/C + OpenMP	1.87	3.42	3.76	3.02	1.87		3.47	3.68
Java	0.63	1.56		0.68	2.00	0.49	0.96	0.74
Serial Matlab			0.88	0.66				
ZPL							9.989	

These results indicate OpenMP is more productive than other approaches for small numbers of CPUs in a shared memory architecture

http://www.nas.nasa.gov/Software/NPB/

NAS Parallel Benchmarks





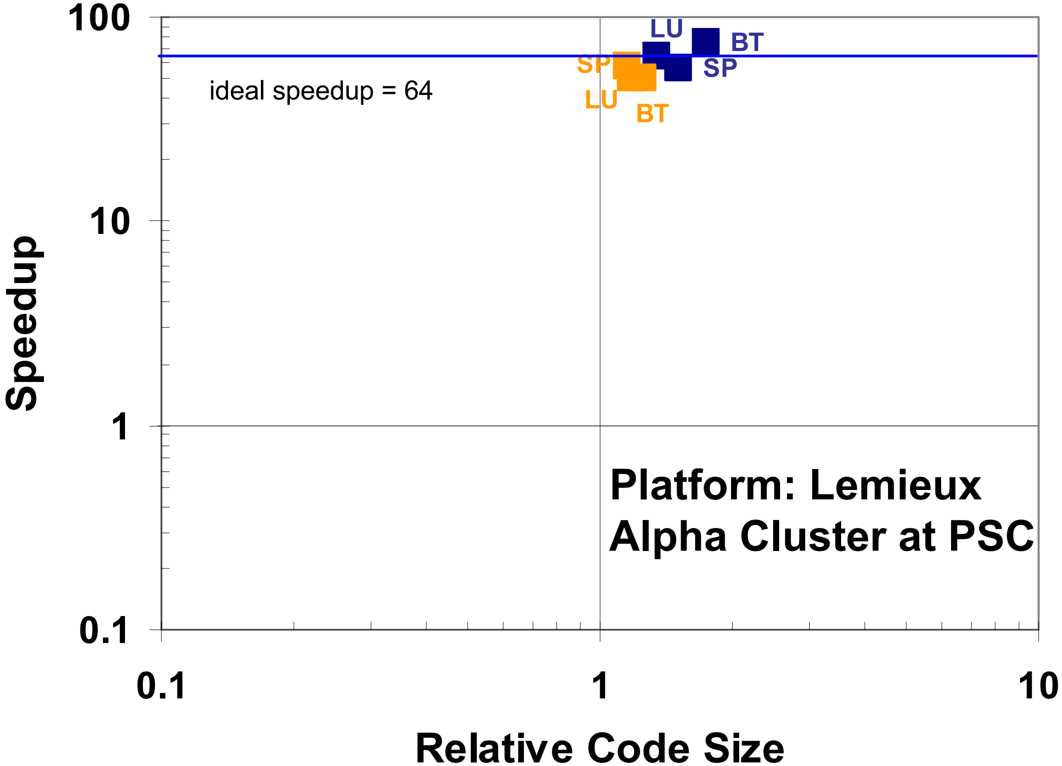
	MG	SP	LU	CG
MPI	21.06	27.69	44.55	11.41
OpenMP	7.39	18.28	17.59	1.73
CAF	19.99	25.80	42.87	9.36

These results show that for larger systems MPI and Co-Array Fortran (CAF) scale well

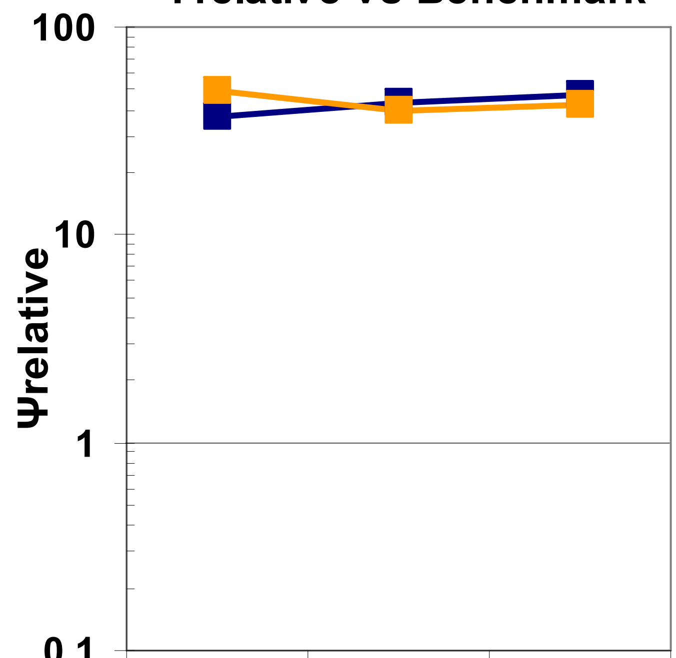
http://www.nas.nasa.gov/Software/NPB/

NAS Parallel Benchmarks

Speedup vs Relative Code Size



Ψrelative vs Benchmark



	SP	BT	LU
MPI	37.34	42.87	47.36
dHPF	49.37	39.98	42.36

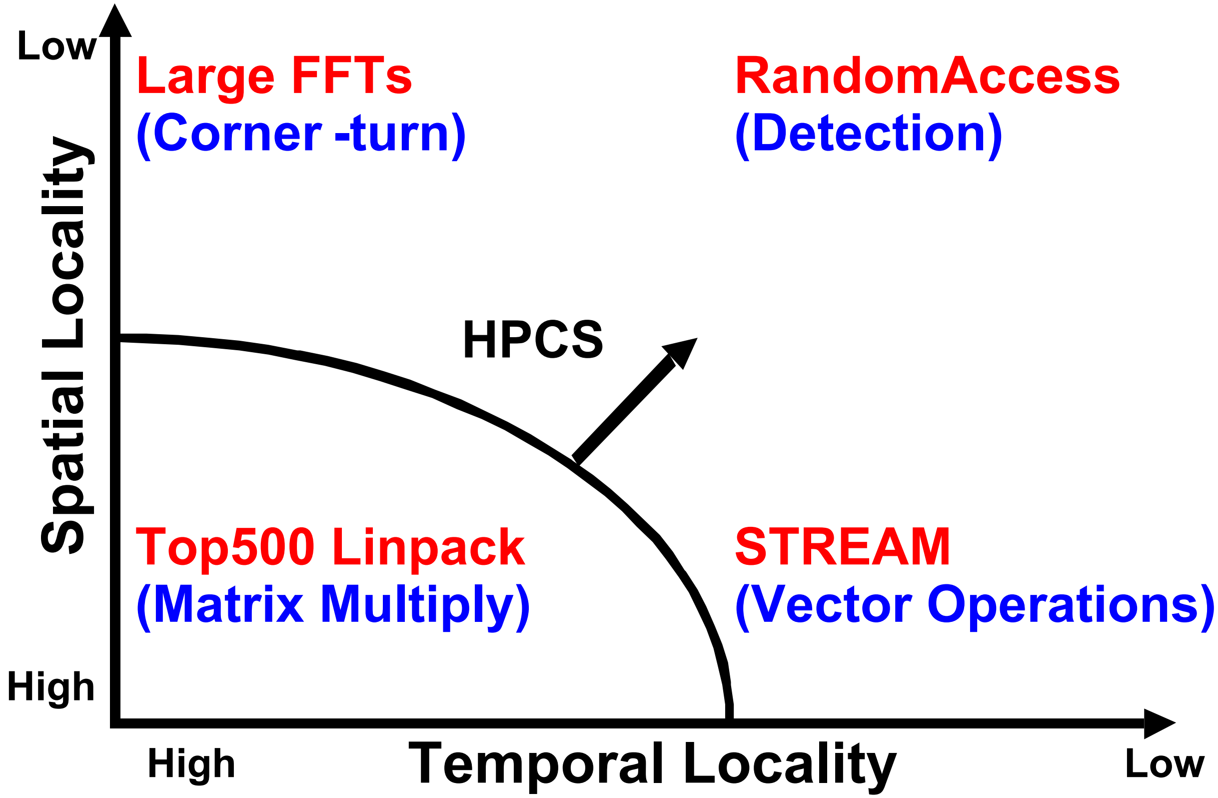
MPI and dHPF (High Performance Fortran) exhibit similar $\Psi_{relative}$, which can be achieved either by increasing performance or by reducing effort

http://www.nas.nasa.gov/Software/NPB/

Outline

- Overview
- **Analysis**
  - *NAS Parallel Benchmarks*
  - ***HPC Challenge***
  - *Classroom Assignments*
- Summary

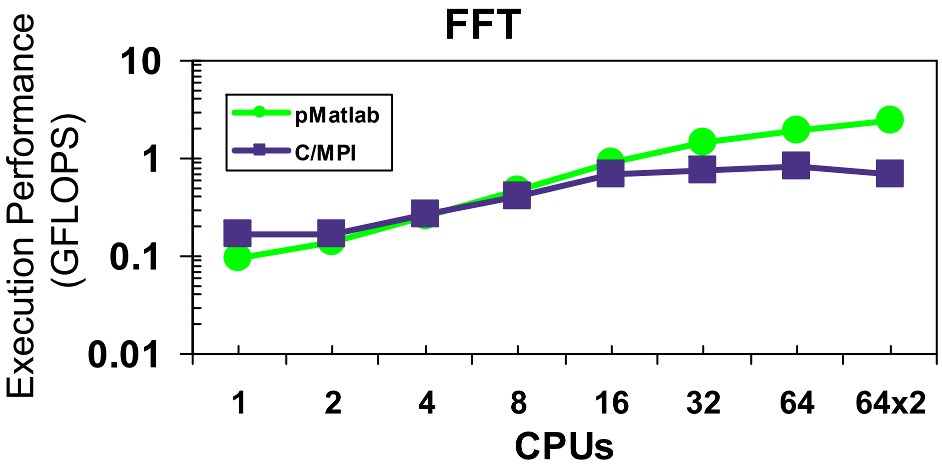
HPC Challenge Benchmark Memory Access Characteristics

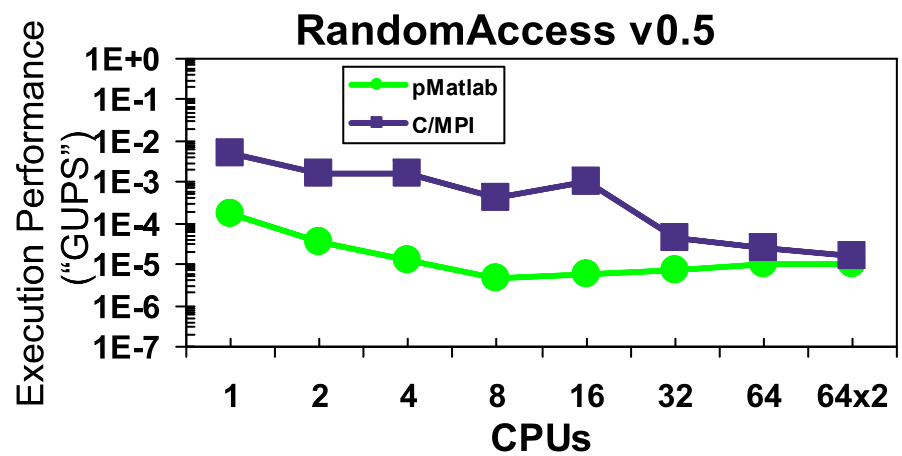


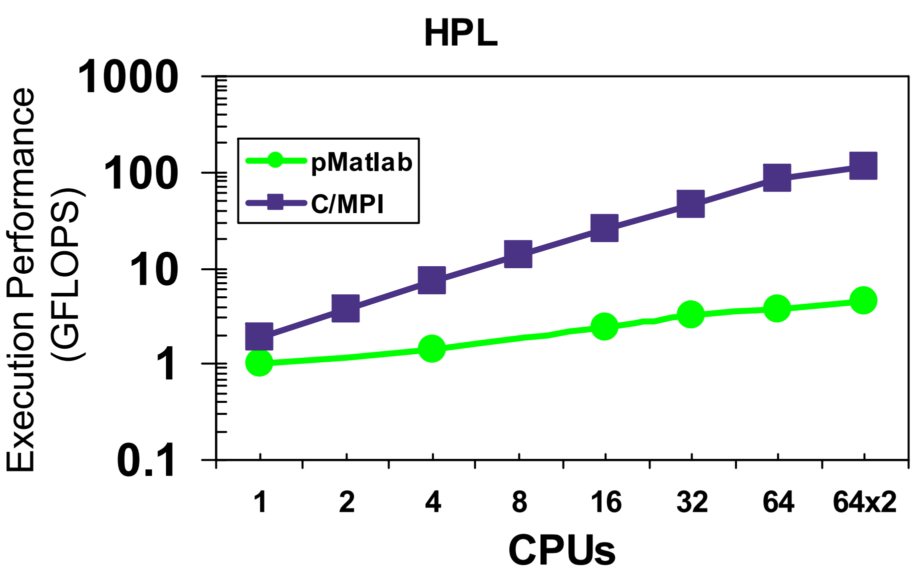
- The HPC Challenge benchmark suite bounds computations of high and low spatial and temporal locality
- Available for download at www.HighProductivity.org

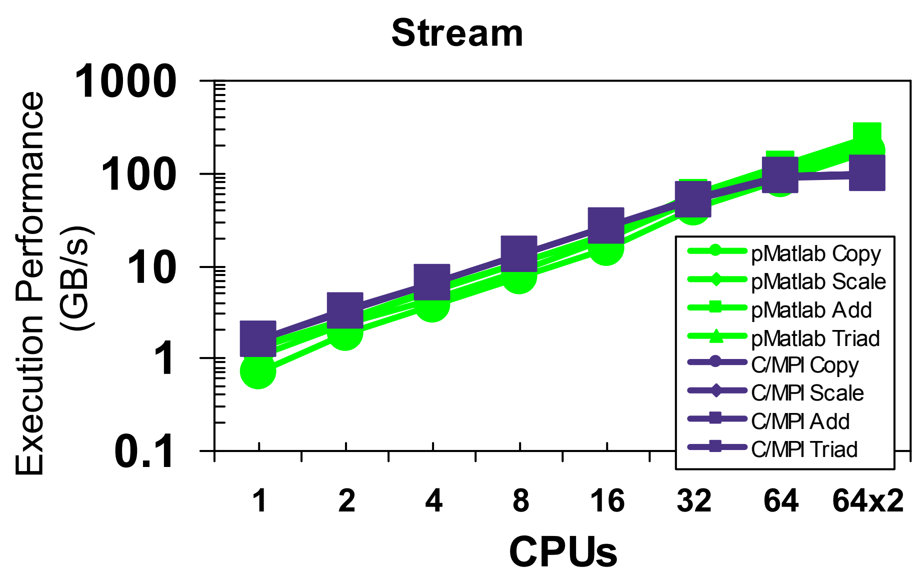
Try it on your favorite system

HPC Challenge





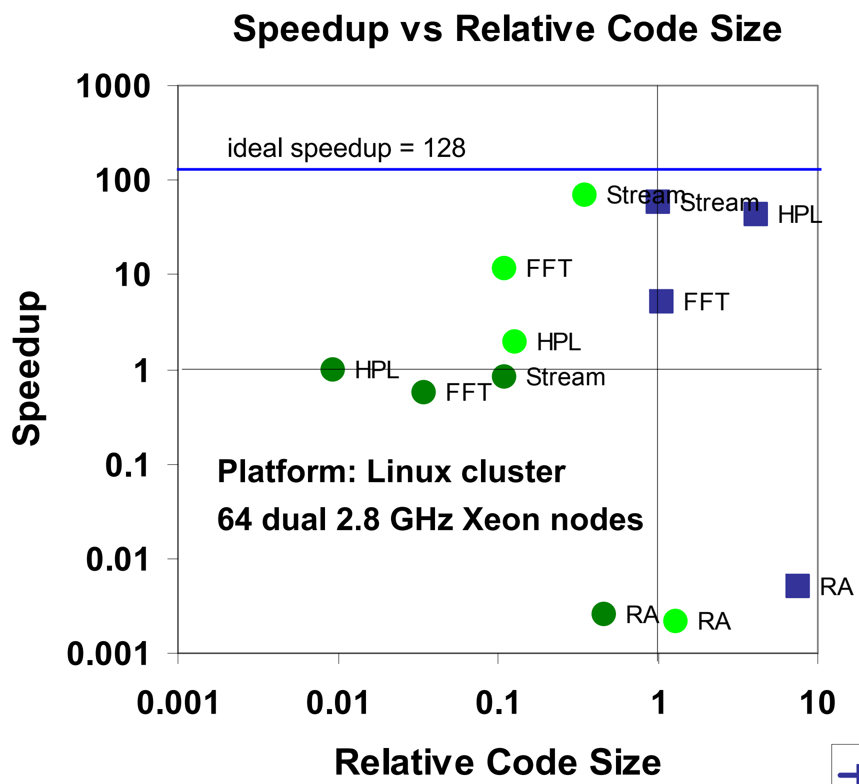


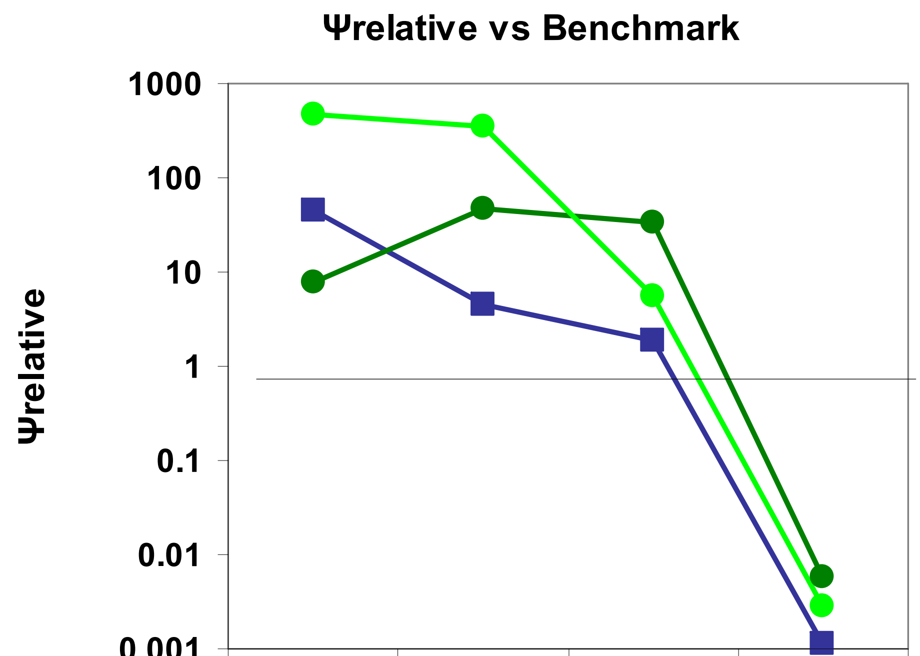


Performance of C+MPI and pMatlab is comparable

http://icl.cs.utk.edu/hpcc/

HPC Challenge





	Stream	FFT	HPL	Random Access
C + MPI	45.04	4.44	1.91	0.00
Serial Matlab	7.68	47.99	33.94	0.01
pMatlab	465.36	355.56	5.56	0.00

- For some benchmarks pMatlab has higher $\Psi_{relative}$
- Note: Scaled speedup – using largest problem size that fits in memory

http://icl.cs.utk.edu/hpcc/

Outline

- Overview
- **Analysis**
  - *NAS Parallel Benchmarks*
  - *HPC Challenge*
  - *Classroom Assignments*
- Summary

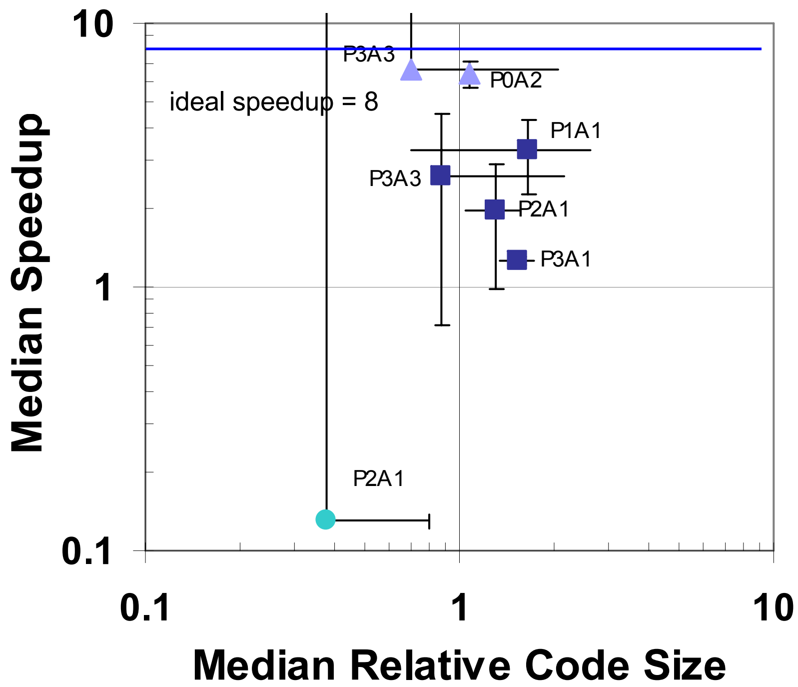
Classroom Experiments

Class	Problem	Assignment	Students reporting
P0A1	Game of Life	Create serial and parallel versions using C and MPI	16
P0A2	Weather Sim	Add OpenMP directives to existing serial Fortran code	17
P1A1	Game of Life	Create serial and parallel versions using C and MPI	11
P2A1	Buffon-Laplace Needle	Create serial versions using C and Matlab, and parallel versions using MPI, OpenMP, and StarP	11
P2A2	Grid of Resistors	Create serial versions using C and Matlab, and parallel versions using MPI, OpenMP, and StarP	11
P3A1	Buffon-Laplace Needle	Create serial versions using C and Matlab, and parallel versions using MPI, OpenMP, and StarP	17
P3A2	Parallel Sorting	Create serial and parallel versions using C and StarP	13
P3A3	Game of Life	Create serial and parallel versions using C, MPI, OpenMP	8

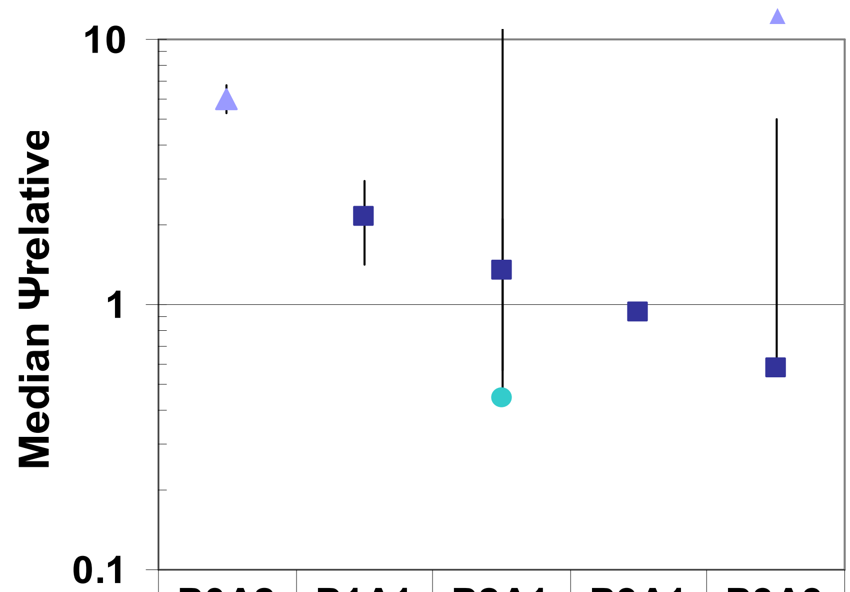
- 4 classes, 8 assignments, 104 student submissions

Classroom Data Summary

Median Speedup vs Relative Code Size



Median Ψrelative vs Benchmark

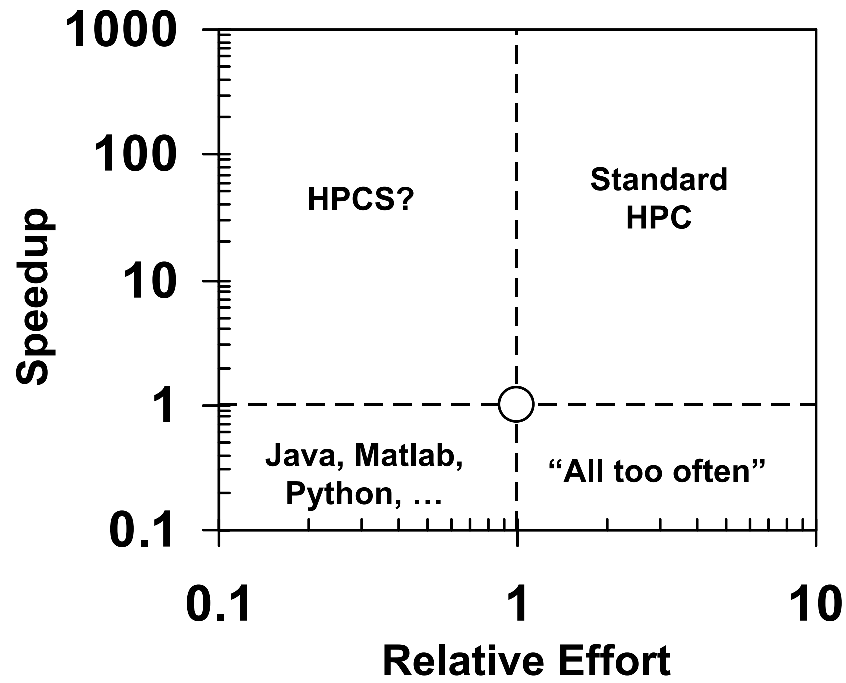


	P0A2	P1A1	P2A1	P3A1	P3A3
MPI		2.16	1.34	0.94	0.57
OpenMP	6		0.032	0.03	12.54
StarP			0.440		

As with NPB, these results indicate OpenMP is more productive than other approaches for small numbers of CPUs in a shared memory architecture

Summary

- Established a common metric, $\Psi_{relative}$, for analyzing productivity of parallel software development
- Applied metric to data, with results consistent across benchmarks and class assignments
- Technique will enable evaluating productivity of programming models for new HPC and HPEC systems
- $\Psi_{relative}$ metric, along with hardware performance and other factors, will give a more complete picture of overall system productivity



Wednesday, 21 September - Session 3: Advanced Parallel Environments

- *X10 Programming*, Vivek Sarkar, IBM
- *MathWorks Recent and Future Solutions for High Productivity,* Roy Lurie and Cleve Moler, MathWorks
- *Advanced Hardware and Software Technologies for Ultra-long FFTs,* Hahn Kim et. al., MIT Lincoln Laboratory
- *An Interactive Approach to Parallel Combinatorial Algorithms with Star-P*, John Gilbert, UCSB, et. al.

References

We wish to thank all of the professors whose students participated in this study, including Jeff Hollingsworth, Alan Sussman, and Uzi Vishkin of the University of Maryland, Alan Edelman of MIT, John Gilbert of UCSB, Mary Hall of USC, and Allan Snavely of UCSD.

- High Productivity Computer Systems http://www.HighProductivity.org
- Kepner, J. "HPC Productivity Model Synthesis." *IJHPCA Special Issue on HPC Productivity*, Vol. 18, No. 4, SAGE 2004
- Humphrey, W. S. *A Discipline for Software Engineering*. Addison-Wesley, USA, 1995
- Boehm, A. Constructive Cost Model (COCOMO). http://sunset.usc.edu/research/COCOMOII/
- NAS Parallel Benchmarks. http://www.nas.nasa.gov/Software/NPB/
- ZPL. http://www.cs.washington.edu/research/zpl/home/
- Wheeler, D. SLOCcount. http://www.dwheeler.com/sloccount/
- HPC Challenge. http://icl.cs.utk.edu/hpcc/
- Haney, R. et. al. "pMatlab Takes the HPC Challenge." Poster presented at High Performance Embedded Computing (HPEC) workshop, Lexington, MA. 28-30 Sept. 2004
- Choy, R. and Edelman, A. *MATLAB*P 2.0: A unified parallel MATLAB*. MIT DSpace, Computer Science collection, Jan. 2003. http://hdl.handle.net/1721.1/3687