The Landscape of Parallel Computing Research: A View from Berkeley

Ying Yu

Dept of Electrical and Computer Engineering

University of Delaware

CISC 879 : Software Support for Multicore Architectures

Outline

- Motivation
- Old & New Conventional Wisdom
- 7 Questions to Frame Parallel Research
- Applications & Dwarfs
- Hardware
- Programming Model
- Systems Software
- Metrics for Success
- Conclusions

Motivation

- The comparison of the outdated conventional wisdoms and their new replacements.
- Moore’s Law continues, thousands of processors can be put on a single, economical chip.
- Communication between these processors within a chip can have very low latency and high bandwidth.
- The open source software movement means that the software stack can evolve much more quickly than in the past.

Outline

- Motivation
- Old & New Conventional Wisdom
- 7 Questions to Frame Parallel Research
- Applications & Dwarfs
- Hardware
- Programming Model
- Systems Software
- Metrics for Success
- Conclusions

Old & New Conventional Wisdom

- Old CW: Power is free, but the transistors are expensive.

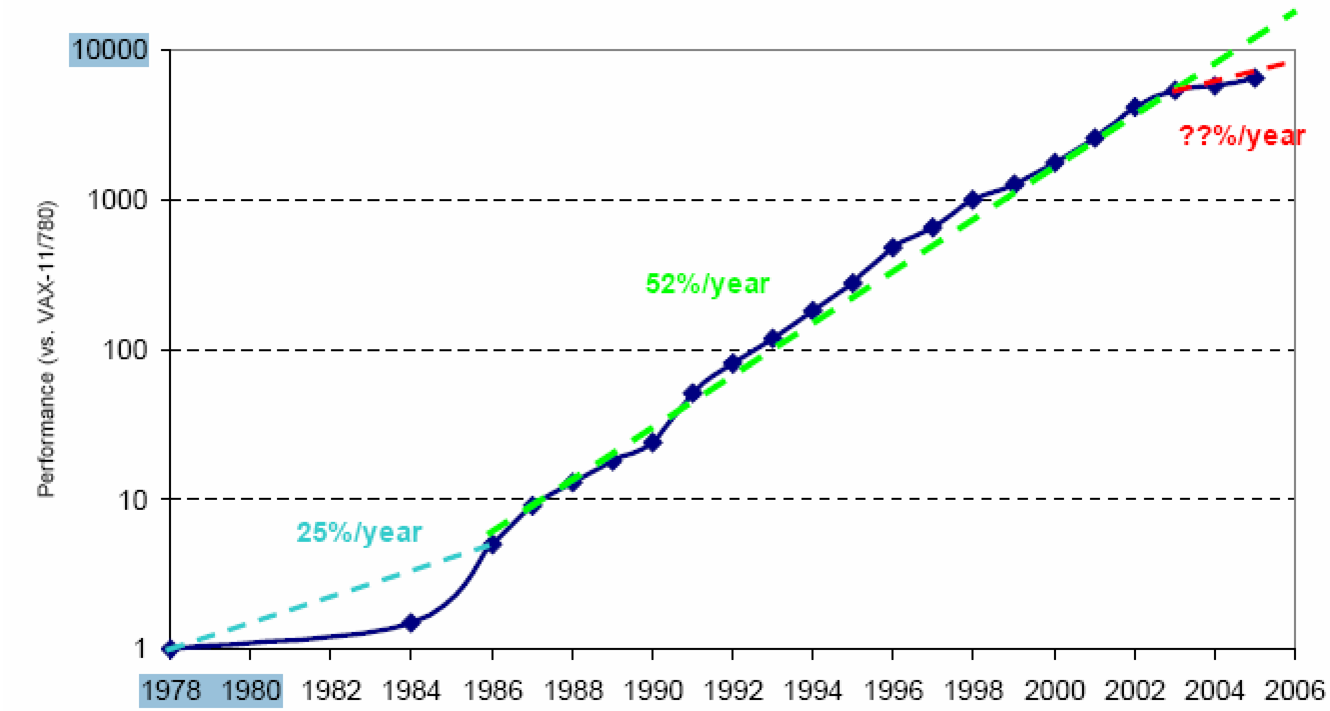
  New CW: "Power Wall", Power is expensive, but transistors are " free"
- Old CW: Multiplies slow, but loads and stores fast.

  New CW: "Memory wall", Loads and stores are slow, but multiplies fast.
- Old CW: We can reveal more ILP via compilers and architecture innovation.

  New CW: "ILP wall", Diminishing returns on finding more ILP.
- Old CW: 2X CPU Performance every 18 months.

  New CW: Power Wall + Memory Wall + ILP Wall= Brick Wall

Uniprocessor Performance (SPECint)

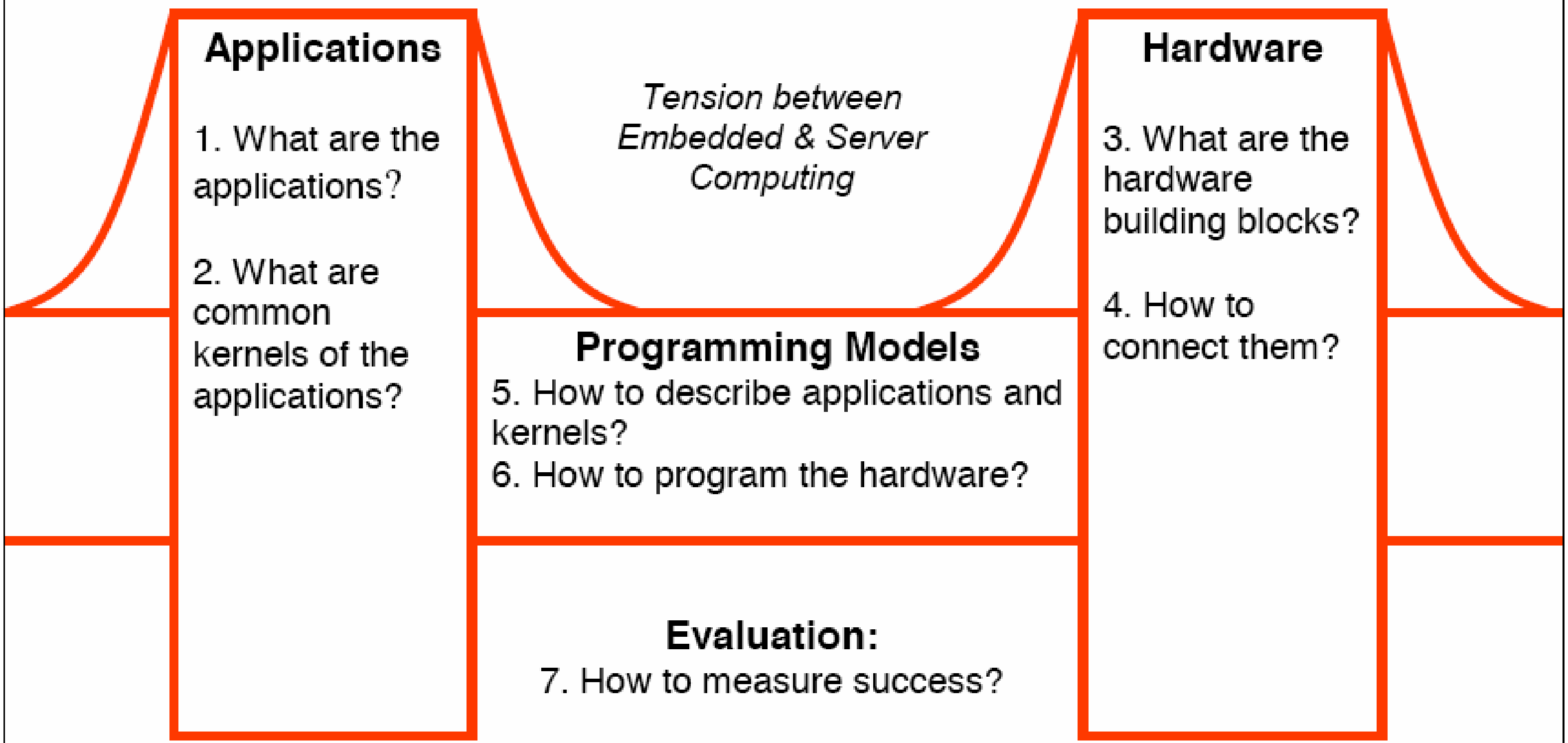


- VAX : 25%/year 1978 to 1986
- RISC + x86: 52%/year 1986 to 2002
- RISC + x86: ??%/year 2002 to present

Outline

- Motivation
- Old & New Conventional Wisdom
- 7 Questions to Frame Parallel Research
- Applications & Dwarfs
- Hardware
- Programming Model
- Systems Software
- Metrics for Success
- Conclusions

7 Questions to Frame Parallel Research



Applications

1. What are the applications?

2. What are common kernels of the applications?

Tension between Embedded & Server Computing

Hardware

3. What are the hardware building blocks?

4. How to connect them?

Programming Models

5. How to describe applications and kernels?

6. How to program the hardware?

Evaluation:

7. How to measure success?

Figure 1. A view from Berkeley: seven critical questions for 21st Century parallel computing. (This figure is inspired by a view of the Golden Gate Bridge from Berkeley.)

Outline

- Motivation
- Old & New Conventional Wisdom
- 7 Questions to Frame Parallel Research
- Applications & Dwarfs
- Hardware
- Programming Model
- Systems Software
- Metrics for Success
- Conclusions

Phillip Colella's "Seven dwarfs"

- Dense Linear Algebra
- Sparse Linear Algebra
- Spectral Methods
- N-Body Methods
- Structured Grids
- Unstructured Grids
- Monte Carlo

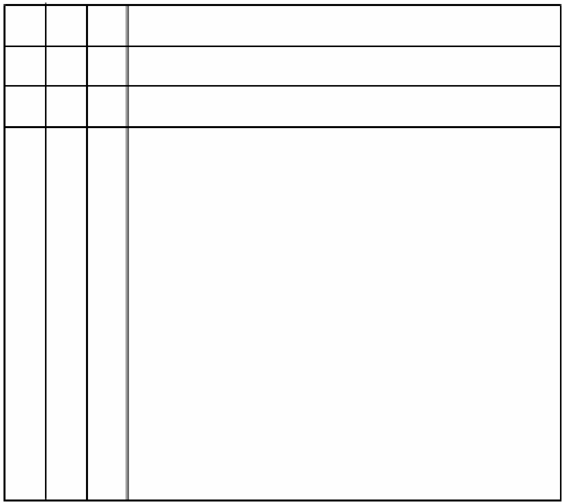
Dense Linear Algebra

Description:

These are the classic vector and matrix operations, traditionally divided into Level 1 (vector/vector), Level 2 (matrix/vector), and Level 3 (matrix/matrix) operations. Data is typically laid out as a contiguous array and computations on elements, rows, columns, or matrix blocks are the norm.

For Example:

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 \\ 0 & 1 & 2 & 3 \end{bmatrix} x \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 30 \\ 40 \\ 80 \\ 20 \end{bmatrix}$$

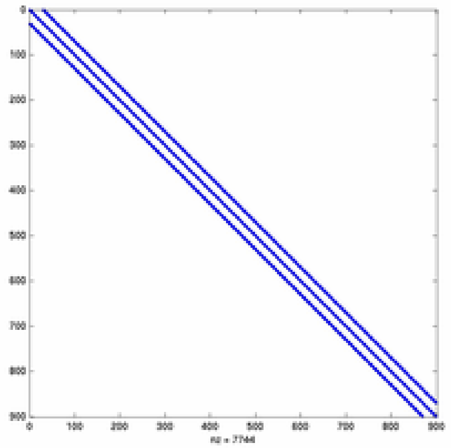


Dense linear algebra matrix diagram.

Sparse Linear Algebra

Description:

Sparse matrix algorithms are used when input matrices have such a large number of zero entries that it becomes advantageous, for storage or efficiency reasons, to “squeeze” them out of the matrix representation. Compressed data structures, keeping only the non-zero entries and their indices, are the norm here.



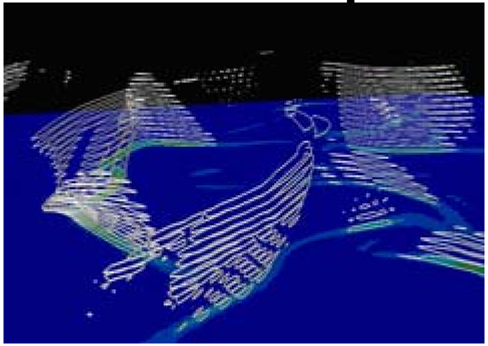
Sparse linear algebra matrix diagram.

Spectral Methods

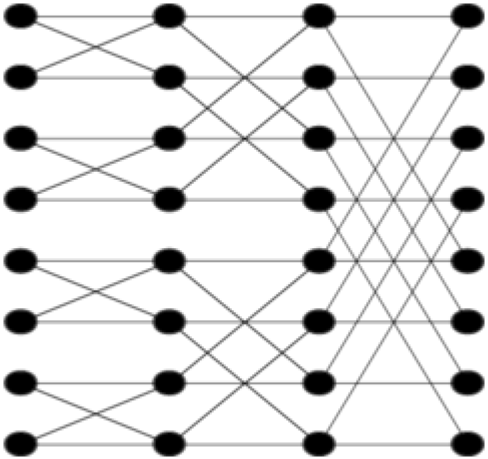
Description:

Data are in the frequency domain, as opposed to time or spatial domains, typically, spectral methods use multiple butterfly stages, which combine multiply-add operations and a specific pattern of data permutation, with all-to-all communication for some stages and strictly local for others.

For Example:



Sheet-like vortical structures that form in high-Reynolds Turbulence problem in 3D computed using a Pseudo-Spectral method.



Spectral Methods computational organization

N-Body Methods

Description:

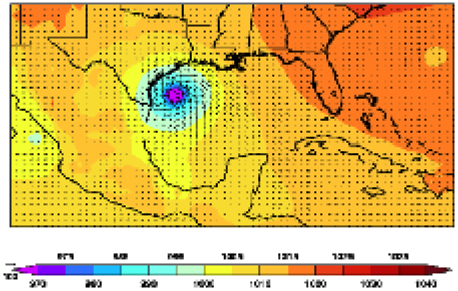
Depends on interactions between many discrete points. Variations include particle-particle methods, where every point depends on all others, leading to an O(N2) calculation, and hierarchical particle methods, which combine forces or potentials from multiple points to reduce the computational complexity to O(N log N) or O(N).

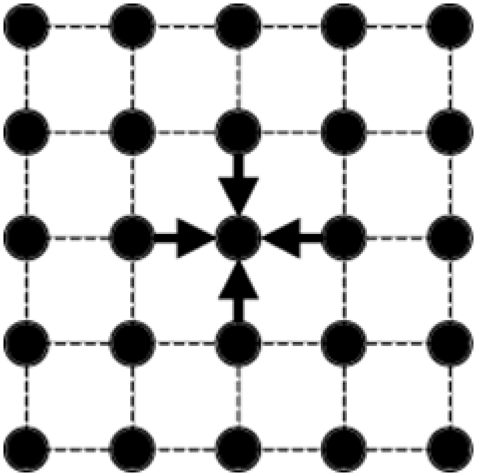
Structured Grids

Description:

Represented by a regular grid; points on grid are conceptually updated together. It has high spatial locality. Updates may be in lace or between 2 versions of the grid.

For Example:





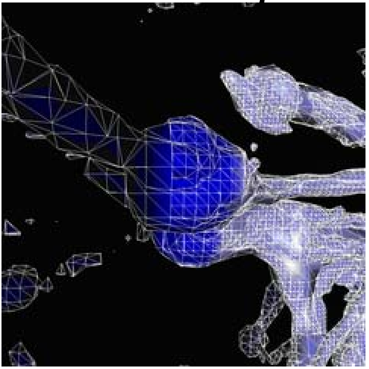
Structured Grid computational organization

Unstructured Grids

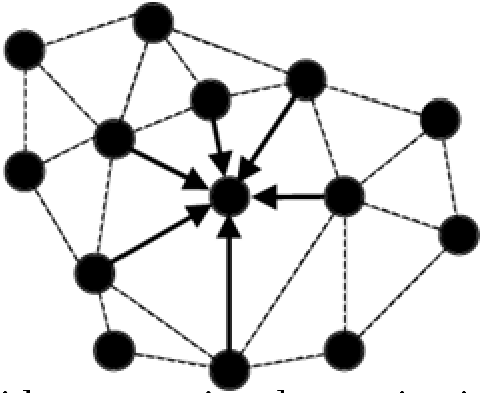
Description:

An irregular grid where data locations are selected, usually by underlying characteristics of the application. Data point location and connectivity of neighboring points must be explicit. The points on the grid are conceptually updated together. Updates typically involve multiple levels of memory reference indirection, as an update to any point requires first determining a list of neighboring points, and then loading values from those neighboring points.

For Example:



This structure arises in a cosmology calculation.



Unstructured Grid computational organization

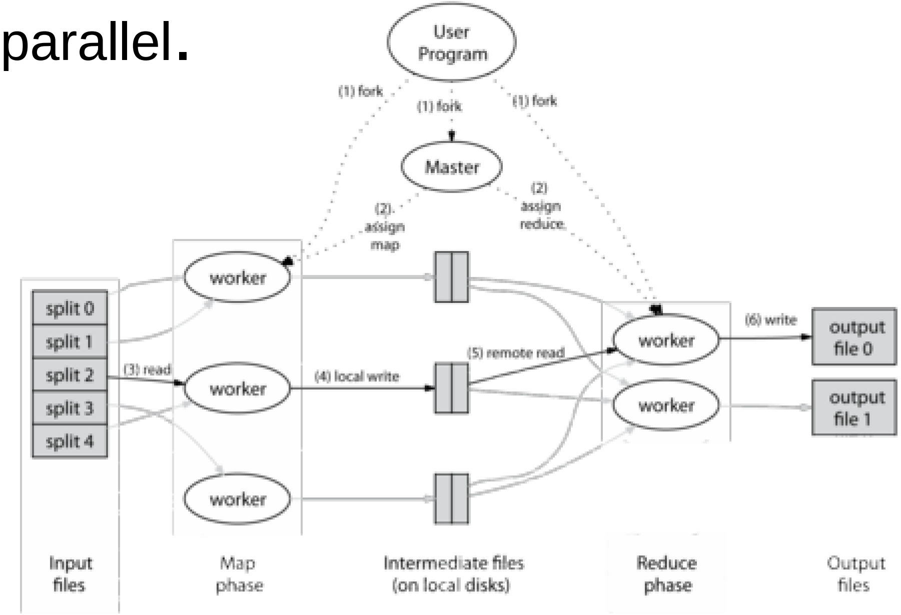
Monte Carlo

Description:

Calculations depend on statistical results of repeated random trials.

Considered embarrassingly parallel.

For Example:



Simplified Data Processing on Large Clusters"

QUESTIONS!

- How well do the Seven Dwarfs of high performance computing capture computation and communication patterns for a broader range of application?
- What dwarfs need to be added to cover the missing important areas beyond high performance computing?

Examine Effectiveness

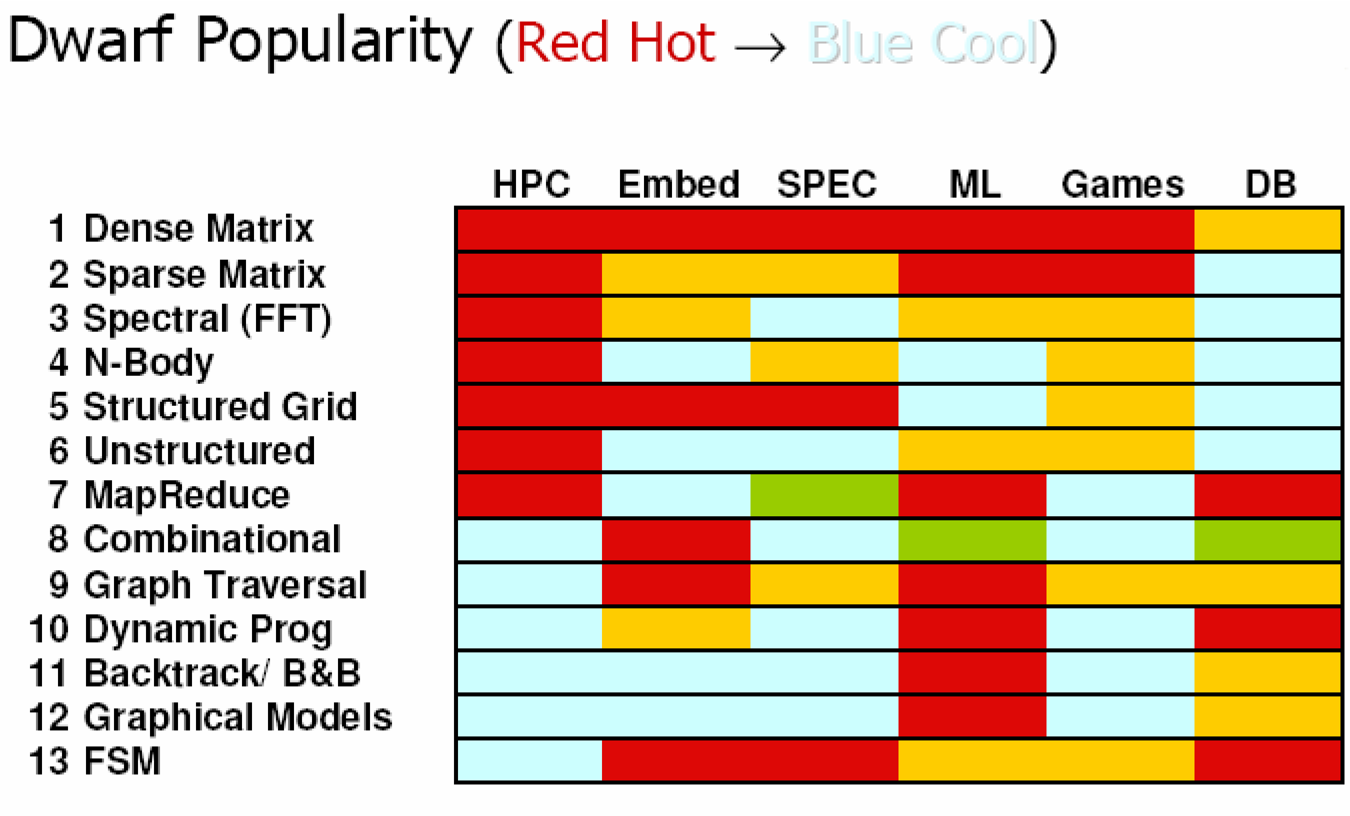
- 1. Embedded Computing (EEMBC benchmark)
- 2. Desktop/Server Computing (SPEC2006)
- 3. Machine Learning
- 4. Games/Graphics/Vision
- 5. Data Base Software

- Result: Added 7 more dwarfs, revised 2 original dwarfs, renumbered list

Next 6 Dwarfs

- Combinational Logic
- Graph Traversal
- Dynamic Programming
- Backtrack and Branch-and-Bound
- Construct Graphical Models
- Finite State Machine

Dwarf Popularity



Dwarf Popularity (Red Hot → Blue Cool)

	HPC	Embed	SPEC	ML	Games	DB
1 Dense Matrix	🟥	🟥	🟥	🟥	🟥	🟧
2 Sparse Matrix	🟥	🟧	🟧	🟥	🟥	🟦
3 Spectral (FFT)	🟥	🟧	🟦	🟧	🟧	🟦
4 N-Body	🟥	🟦	🟧	🟦	🟧	🟦
5 Structured Grid	🟥	🟥	🟥	🟦	🟧	🟦
6 Unstructured	🟥	🟦	🟦	🟧	🟧	🟦
7 MapReduce	🟥	🟦	🟩	🟥	🟦	🟥
8 Combinational	🟦	🟥	🟦	🟩	🟦	🟩
9 Graph Traversal	🟦	🟥	🟧	🟥	🟧	🟧
10 Dynamic Prog	🟦	🟧	🟦	🟥	🟦	🟥
11 Backtrack/ B&B	🟦	🟦	🟦	🟥	🟦	🟧
12 Graphical Models	🟦	🟦	🟦	🟥	🟦	🟧
13 FSM	🟦	🟥	🟥	🟧	🟧	🟥

Outline

- Motivation
- Old & New Conventional Wisdom
- 7 Questions to Frame Parallel Research
- Applications & Dwarfs
- Hardware
- Programming Model
- Systems Software
- Metrics for Success
- Conclusions

HW: What are the problems?

- Power limits leading edge chip designs

  Intel Tejas Pentium 4 cancelled due to power issues

- Yield on leading edge processes dropping dramatically

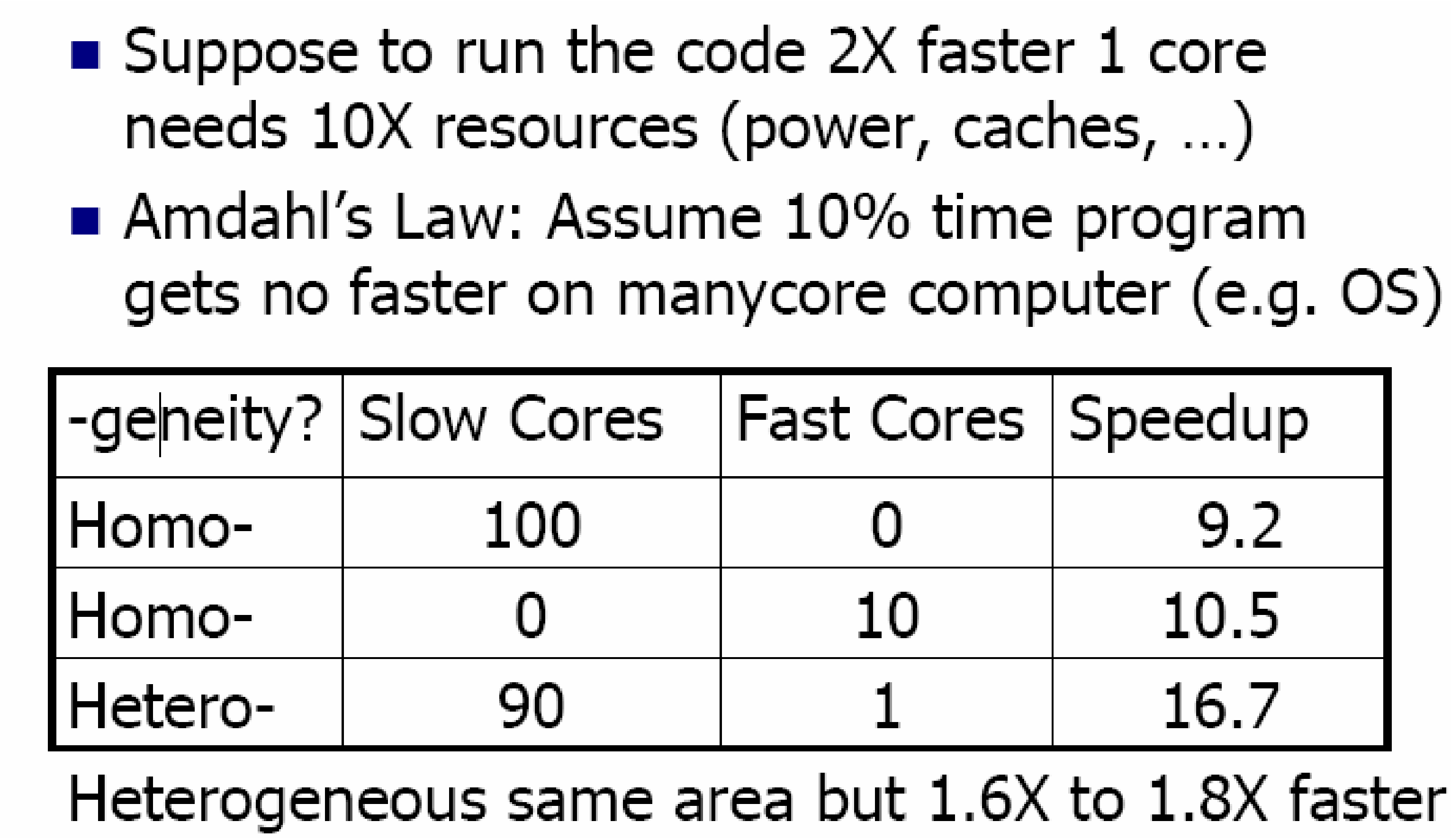
  IBM quotes yields of 10 – 20% on 8-processor Cell

- Design/validation leading edge chip is becoming unmanageable

HW Solution 1: Small is Beautiful

- An energy-efficient way to achieve performance
- An economical element that is easy to shut down in the face of catastrophic defects and easier to reconfigure in the face of large parametric variation
- A simple architecture is easier to design and functionally verity more power efficient and easier to predict
- One size fits all?
- Amdahl’s Law to Heterogeneous processors?

Heterogeneous Processors?



- Suppose to run the code 2X faster 1 core needs 10X resources (power, caches, …)
- Amdahl’s Law: Assume 10% time program gets no faster on manycore computer (e.g. OS)

-geneity?	Slow Cores	Fast Cores	Speedup
Homo-	100	0	9.2
Homo-	0	10	10.5
Hetero-	90	1	16.7

Heterogeneous same area but 1.6X to 1.8X faster

HW Solution 2: Memory

- For creating a new hardware foundation for parallel computing hardware
- For the reason to innovate in memory is that increasingly, the cost of hardware is shifting from processing to memory.
- For the DRAM capacity increasing speed slows down these decades.

→

Given the current slow increase in memory capacity, the MIPS explosion suggests a much larger fraction of total system silicon in the future should be dedicated to memory

HW Solution 3: SWITCH

- For Interconnection networks:
  - On-chip topologies to prevent the complexity of the interconnect from dominating cost of manycore systems
  - To augment the packet switches using simple circuit switches to reconfigure the wiring topology between the switches to meet the application communication requirements
  - Using less complex circuit switches
- For communication primitives:
  - Synchronization using transactional memory
  - Synchronization using Full-Empty Bits in Memory
  - Synchronization using Message Passing

Some obvious recommendations

- Counters and other instrumentation more important than in the past

- Don’t include features that significantly affect performance or energy if programmers cannot accurately measure their impact

Outline

- Motivation
- Old & New Conventional Wisdom
- 7 Questions to Frame Parallel Research
- Applications & Dwarfs
- Hardware
- Programming Model
- Systems Software
- Metrics for Success
- Conclusions

How to evaluate Programming Model

- Programming model must allow programmer to balance competing goals of productivity and implementation efficiency.
- Programming model efforts inspired by psychological research
- Models must be independent of the number of processors
- Models must support a rich set of data sizes and types
- Models must support proven styles of parallelism

Human-centric Programming Model

- Obviously, success of models strongly affected by human beings who use them
- Efficiency, productivity, correctness important, but no impact of research from psychology???
    - Used in HCI, but not in language design or SW engineering
    - E.g., a rich theory of human errors [Kantowitz and Sorkin 1983] [Reason 1990]
- E.g., Transactional Memory helps with human errors
    - System ensures correctness even when programmers make incorrect assumptions about safety of parallelizing code
- Integrating research on human psychology and problem solving critical to developing successful parallel Programming Model and environments

Outline

- Motivation
- Old & New Conventional Wisdom
- 7 Questions to Frame Parallel Research
- Applications & Dwarfs
- Hardware
- Programming Model
- Systems Software
- Metrics for Success
- Conclusions

Systems Software

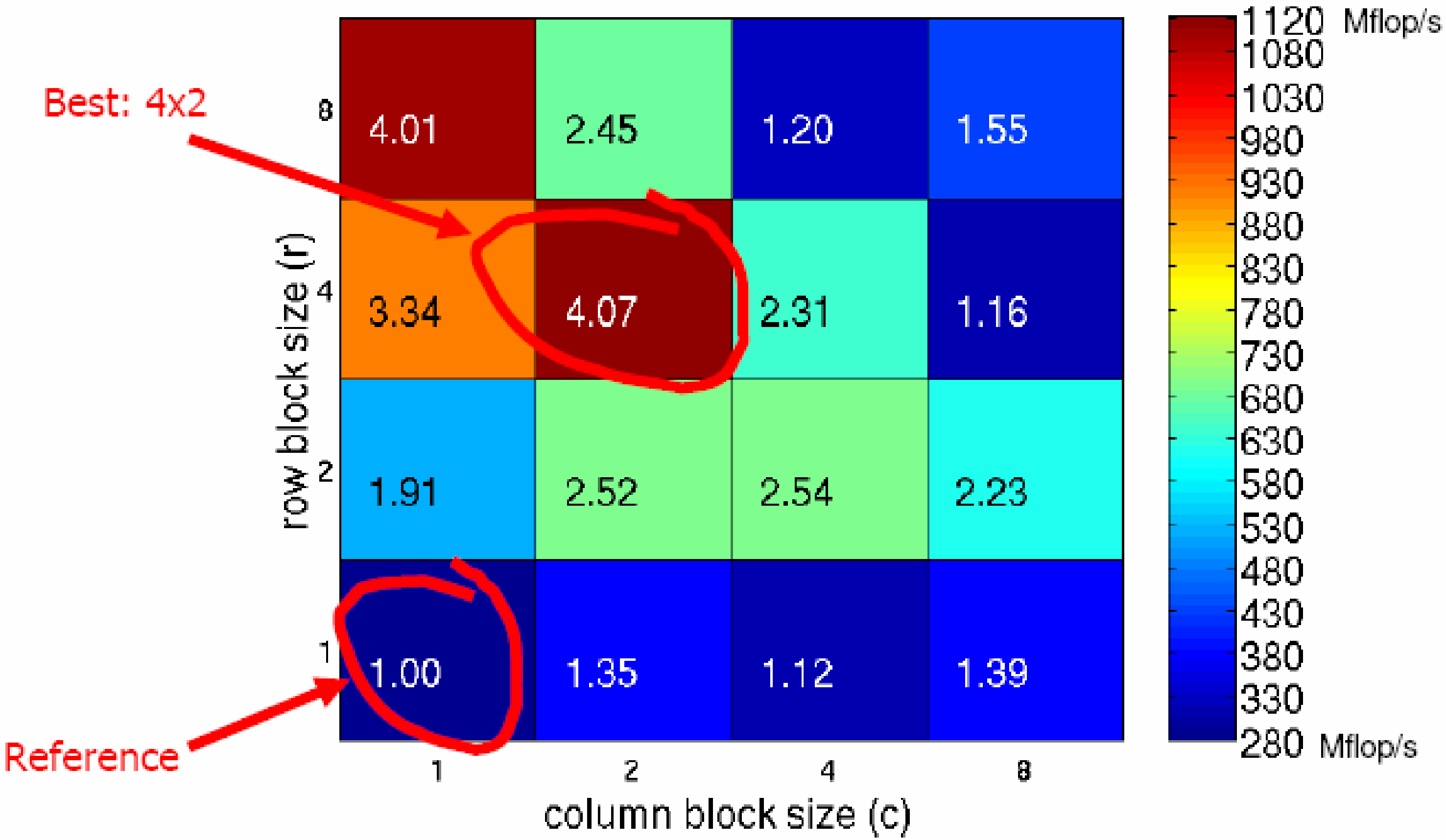
- Instead of completely re-engineering compilers for parallelism, rely more on “Auto-tuners” that search to yield efficient parallel code.

- Instead of relying on the conventional large, monolithic operating systems, rely more on virtual machines and system libraries to include only those functions needed by the applications.

Sparse Matrix – Search for Blocking

For finite element problem (BCSR) [Im, Yelick, Vuduc, 2005]

900 MHz Itanium 2, Intel C v8: ref=275 Mflop/s

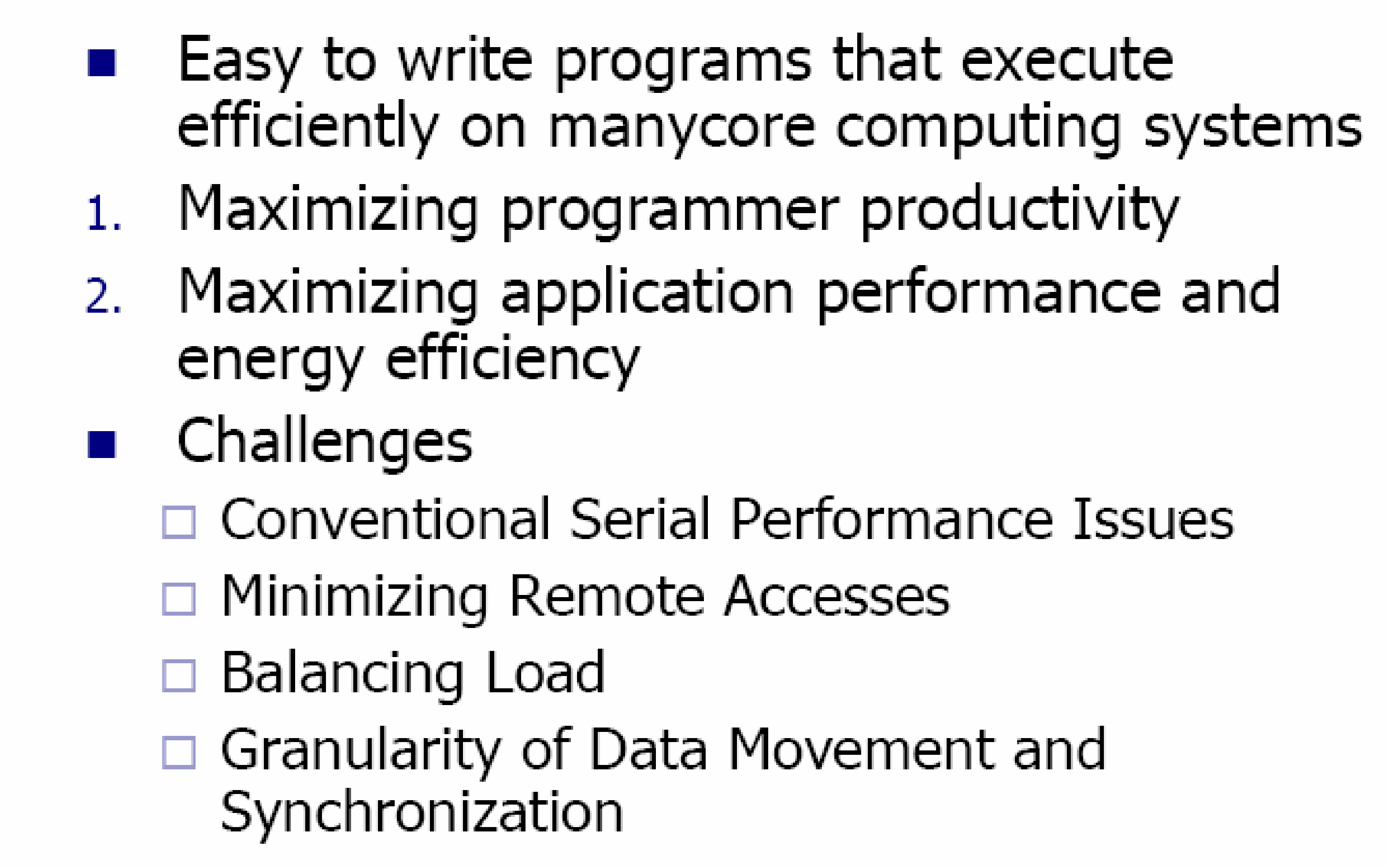


	1	2	4	8
8	4.01	2.45	1.20	1.55
4	3.34	4.07	2.31	1.16
2	1.91	2.52	2.54	2.23
1	1.00	1.35	1.12	1.39

Outline

- Motivation
- Old & New Conventional Wisdom
- 7 Questions to Frame Parallel Research
- Applications & Dwarfs
- Hardware
- Programming Model
- Systems Software
- Metrics for Success
- Conclusions

How to measure success?



- Easy to write programs that execute efficiently on manycore computing systems

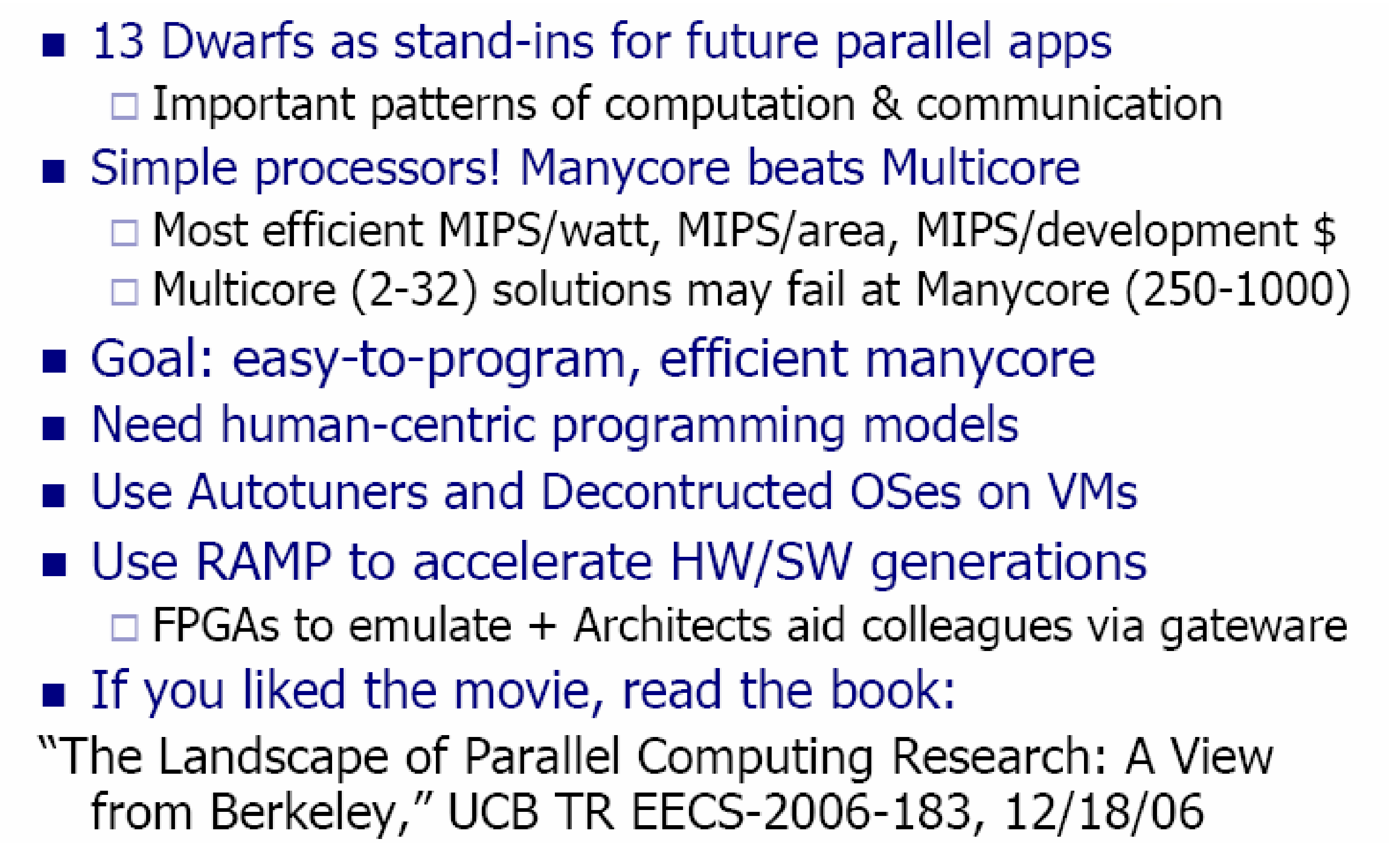
1. Maximizing programmer productivity
2. Maximizing application performance and energy efficiency

- Challenges
  - □ Conventional Serial Performance Issues
  - □ Minimizing Remote Accesses
  - □ Balancing Load
  - □ Granularity of Data Movement and Synchronization

Outline

- Motivation
- Old & New Conventional Wisdom
- 7 Questions to Frame Parallel Research
- Applications & Dwarfs
- Hardware
- Programming Model
- Systems Software
- Metrics for Success
- Conclusions

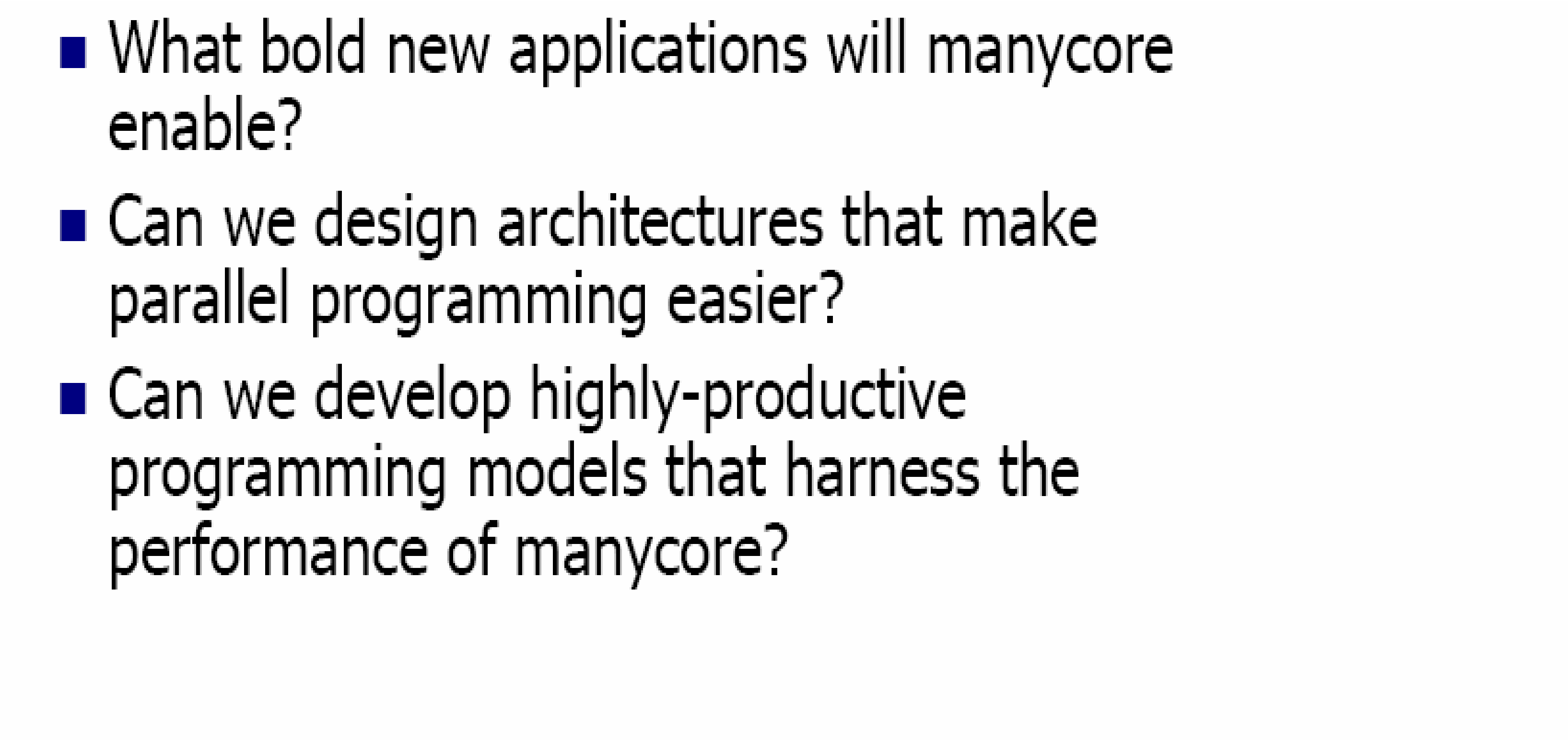
Conclusions



- 13 Dwarfs as stand-ins for future parallel apps
  - □ Important patterns of computation & communication
- Simple processors! Manycore beats Multicore
  - □ Most efficient MIPS/watt, MIPS/area, MIPS/development $
  - □ Multicore (2-32) solutions may fail at Manycore (250-1000)
- Goal: easy-to-program, efficient manycore
- Need human-centric programming models
- Use Autotuners and Decontructed OSes on VMs
- Use RAMP to accelerate HW/SW generations
  - □ FPGAs to emulate + Architects aid colleagues via gateware
- If you liked the movie, read the book:

"The Landscape of Parallel Computing Research: A View from Berkeley," UCB TR EECS-2006-183, 12/18/06

Where we go from here?



- What bold new applications will manycore enable?
- Can we design architectures that make parallel programming easier?
- Can we develop highly-productive programming models that harness the performance of manycore?

Questions?