The Need for Diversity in HPC

Paul Lu
Associate Professor
Dept. of Computing Science
University of Alberta
Edmonton, Alberta, Canada
paullu@cs.ualberta.ca

Original Material Copyright 2008
Presented April 14, 2008, HPCS, Ottawa, Ontario
(this version with some revisions)

Disclosure

- My research uses large-scale shared-memory machines
- SGI was my vendor partner for the winning Cluster Challenge team at Supercomputing 2007
- My team won Second Prize in VMware’s Ultimate Virtual Appliance contest in 2006
- WestGrid and Compute Canada are currently in a pre-RFP period.

Original Material Copyright 2008

Acknowledgements

- Jonathan Schaeffer, Duane Szafron, Nelson Amaral, Cam Macdonell, Maria Cutumisu, Pawel Gburzynski, Brent Gorda, Greg Wilson, my CMPUT 399 class
- Graphs from Top500.org
- Slides from David Patterson
- Information from Ken Koch
  - LANL Roadrunner Summary 3/12/2007

Original Material Copyright 2008

The Audience

- High-performance computing and simulation *users*
- Not necessarily hardware or OS researchers

Original Material Copyright 2008

Overview and Basic Argument

1. A monoculture is bad in the long term.
2. Losing expertise is the greatest risk.
3. Consequences: Hardware, Software, Research.

Original Material Copyright 2008

Not going to emphasize…

- Databases and Web servers. Focus on computational science.
- Capability vs. capacity computing
- MPI vs. shared-memory programming **models**
- Business models and viability
- The optimal amount of diversity (likely, it should 1 < d < 5)
  - Nor what the right trade-off point is between maintaining expertise vs. buying/paying for it later (assuming you can get it)

Original Material Copyright 2008

Monoculture

Really?

A bad thing?

What monoculture?

- Today, the dominant platform for high-performance computing (HPC) is:
  - x86 commodity processors
  - Linux
  - Ethernet, Infiniband
  - Fortran, C/C++

Original Material Copyright 2008

Cluster Challenge 2007 (1)

- All 6 teams used x86 CPUs. 5 of 6 teams used Intel Xeon.
- 5 of 6 teams used Linux
  - Not the same 5 teams
- 5 of 6 teams used Infiniband
  - One team used Myrinet

Original Material Copyright 2008

Cluster Challenge 2007 (2)

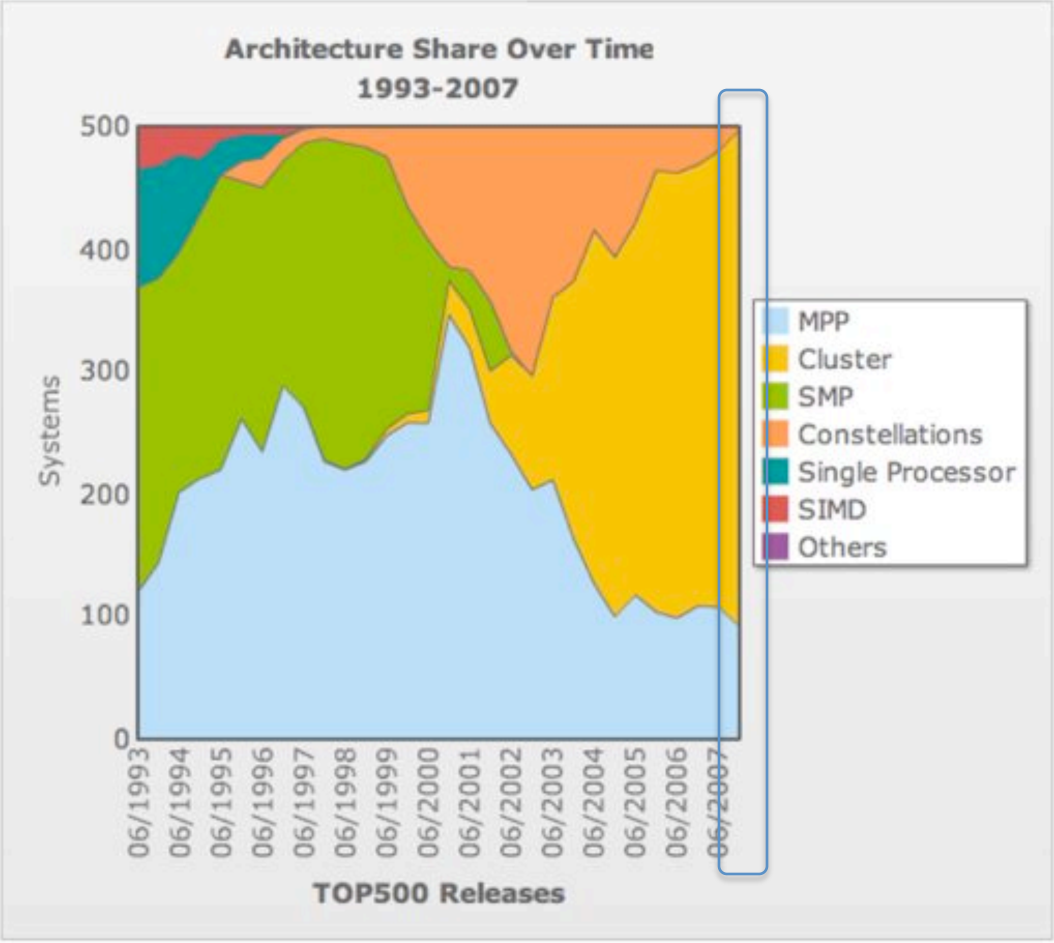
- All 6 teams used x86 CPUs. 5 of 6 teams used Intel Xeon. → Commodity processors
  - Great price-performance
- 5 of 6 teams used Linux → Strength in numbers
  - Not the same 5 teams
  - "Everything runs on Linux"
- 5 of 6 teams used Infiniband → Near-commodity interconnects
  - One team used Myrinet
  - Good price-performance
  - "Good" software support

Mainstream technologies have their advantages!

Original Material Copyright 2008

Top 500 List “Monoculture”

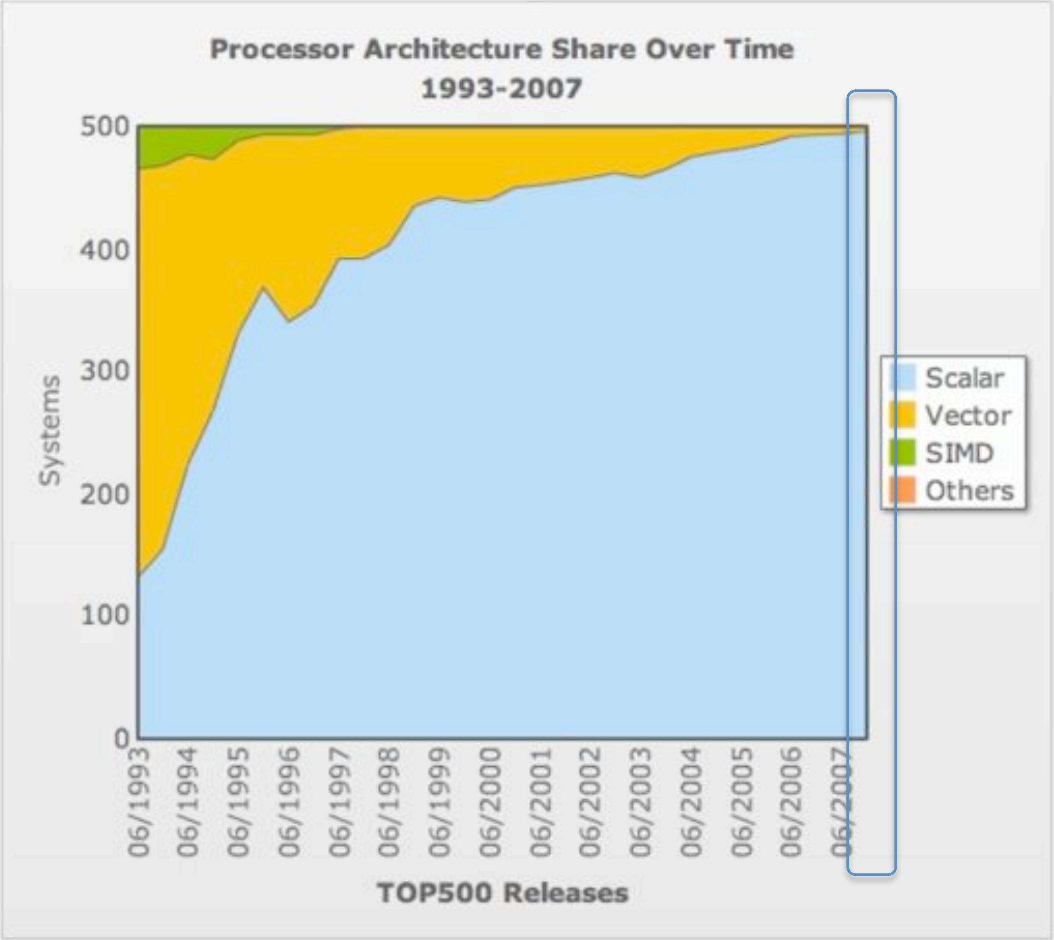
(from top500.org)



Original Material Copyright 2008

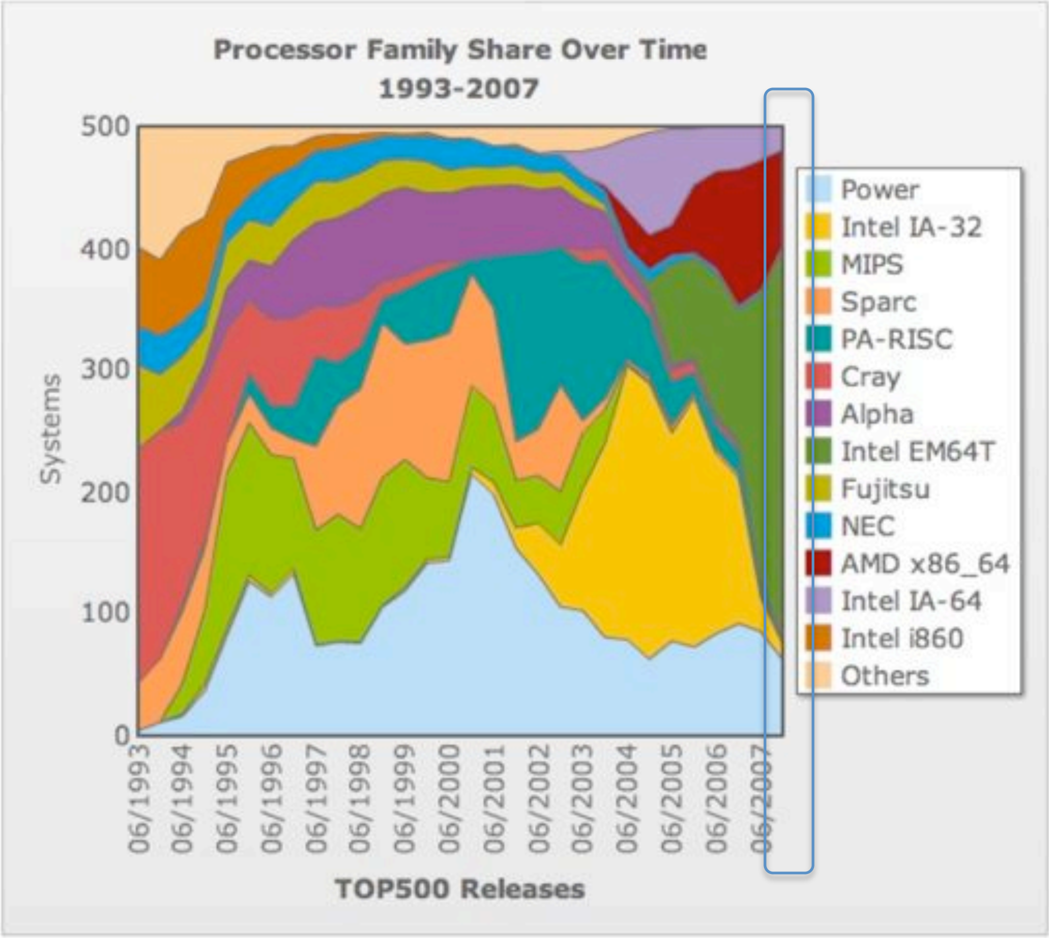
Top 500 List “Monoculture”

(from top500.org)



Top 500 List “Monoculture”

(from top500.org)



Original Material Copyright 2008

Monocultures

- Hardware
  - Commodity x86 CPUs
  - Currently, 8-way or less, quad-core processors
  - Ethernet or Infiniband
- Software
  - Linux

Original Material Copyright 2008

Monoculture, bad?

- Clusters are great, but…
- What if there was only…
  - one computer programming language?
  - one operating system?
  - one GUI?
  - one text editor?
  - one browser?

Original Material Copyright 2008

Example: Browsers

- Not good to have *only* one, viable, dominant Web browser, despite the "benefits"…
  - But, it is free (i.e., great price-performance)
  - And, everyone will be compatible with it
- Because, what if the developers turn their focus elsewhere?
  - Mobile computing, smart phones, social networks

Original Material Copyright 2008

Monoculture, good?

- Can there be too much diversity? Yes.
  - Networks: **Ethernet**, Token Ring, FDDI?
  - OS: BSD, System V, Windows, Mac OS, **Linux**?
  - Threading Libraries: Solaris, SGI, **Pthreads**?
  - Message Passing Libraries: PVM, **MPI**
- But, usually good to have a viable second choice

Original Material Copyright 2008

Good vs. Bad Monocultures

"Monocultures" can be good too

- Standardization of *specifications* is good
  - e.g., HTML
  - In HPC, Message-Passing Interface (MPI), Pthreads, and OpenMP are important standards.
- Diversity of *implementations* (**and** architectures) should be maintained
  - Open-source software helps, but is not a substitute for multiple implementations and innovation in architecture

Original Material Copyright 2008

Expertise

What if we forgot how to do something?

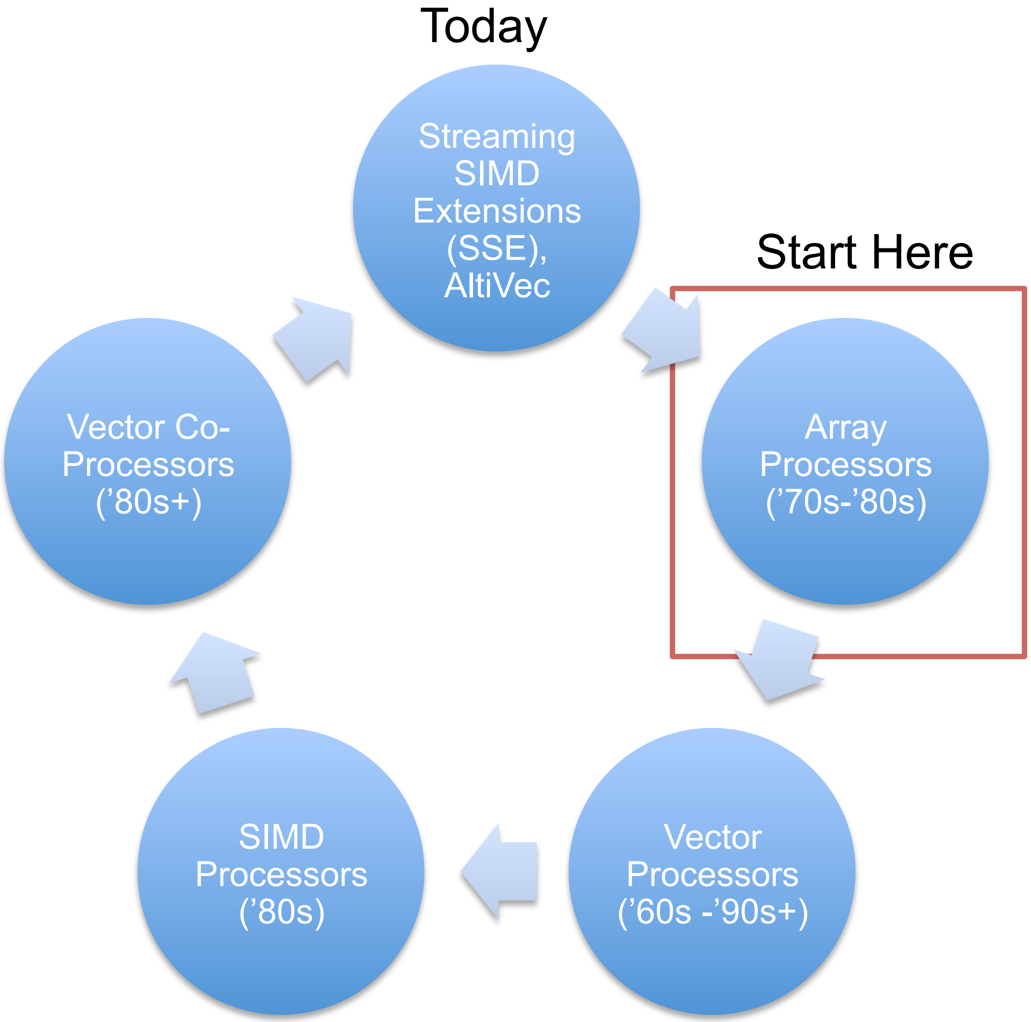
Original Material Copyright 2008

Technology Tends to Be Cyclical

- Languages
  - Fortran, C/C++, HPF, OpenMP C and Fortran, scripting languages, Fortran-something
- Processor architecture
- Co-processors and special processors

Original Material Copyright 2008

Life, Death, and Life of SIMD


Today
Streaming SIMD Extensions (SSE), AltiVec
Start Here
Array Processors ('70s-'80s)
Vector Processors ('60s -'90s+)
SIMD Processors ('80s)
Vector Co-Processors ('80s+)

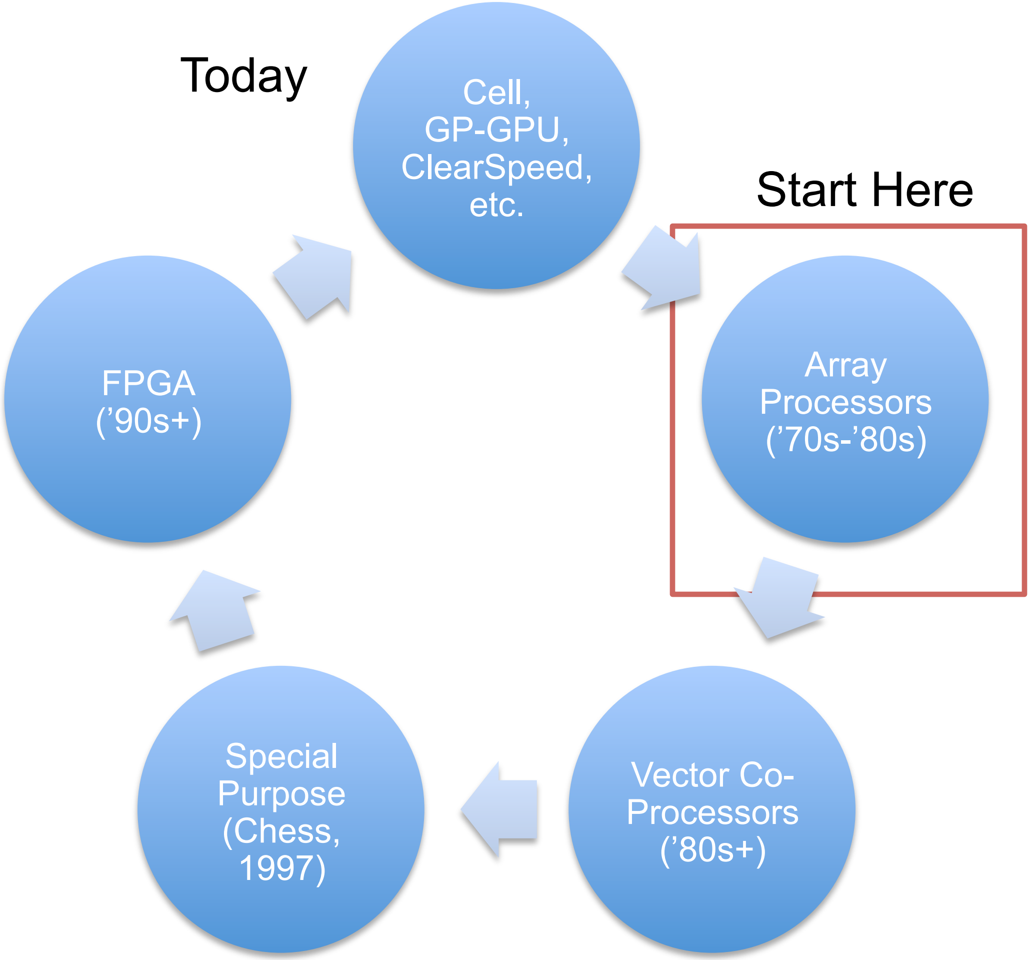
Original Material Copyright 2008

Life, Death, and Life of SIMD

- Fast, general-purpose processors usually win in the marketplace
- But, SIMD keeps returning to provide a performance edge
- Notably, with SSE, it is “bundled” with fast, general-purpose processing power
- GROMACS molecular dynamics code uses SSE/3DNow for a boost

Original Material Copyright 2008

Life, Death, and Life of Co-Processors



Today

Start Here

Cell,
GP-GPU,
ClearSpeed,
etc.

Array Processors ('70s-'80s)

Vector Co-Processors ('80s+)

Special Purpose (Chess, 1997)

FPGA ('90s+)

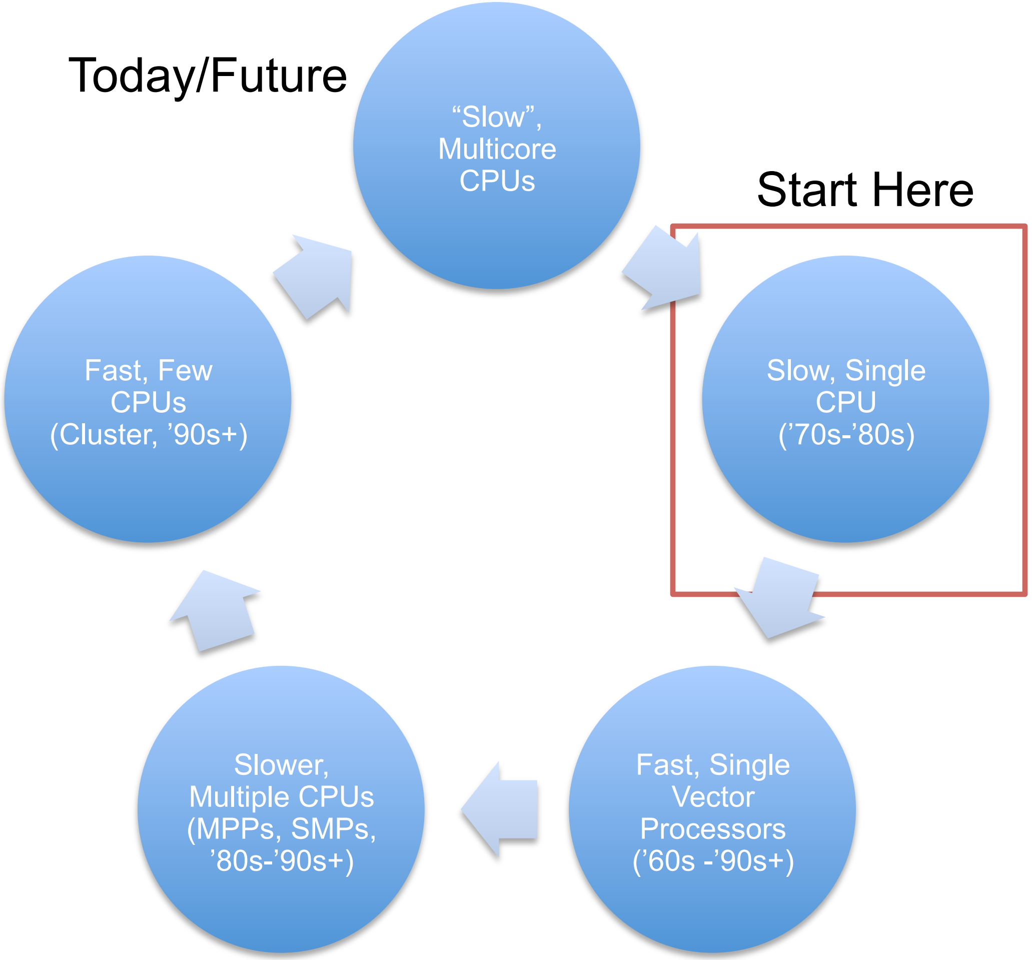
Original Material Copyright 2008

Life, Death, and Life of Co-Processors

- As with SSE, GP-GPU may become a “bundled” feature of fast processors
- Loss of low-level architectural and coding expertise is quite common
- Of course, do NOT jump on every bandwagon. But, remember how to jump.

Original Material Copyright 2008

Life, Death, and Life of "Massive Parallelism"



Today/Future
"Slow", Multicore CPUs
Start Here
Slow, Single CPU ('70s-'80s)
Fast, Single Vector Processors ('60s -'90s+)
Slower, Multiple CPUs (MPPs, SMPs, '80s-'90s+)
Fast, Few CPUs (Cluster, '90s+)

Original Material Copyright 2008

Life, Death, and Life of "Massive Parallelism"

- Next era will NOT provide "automatic" performance improvements each year via clock speed.
- Multicore may not be as well-suited for capacity computing
  - Multiple jobs sharing a processor/socket contend for (depending on architecture) memory bandwidth, I/O bandwidth, etc.
  - Performance mainly through better use of cores per-job (i.e., parallelism)
  - Requires innovation in OS, scheduling, etc.

Original Material Copyright 2008

What if we lost the skills?

- When was the last time your group wrote a *new* Fortran program?
- Is the SSE/AltiVec capability of your CPU used at all?
  - Intel is moving towards Advanced Vector Extensions, 256-bits
- Who in your group can scale your application if you had 64 cores sharing a common memory?

Original Material Copyright 2008

What if we missed the boat?

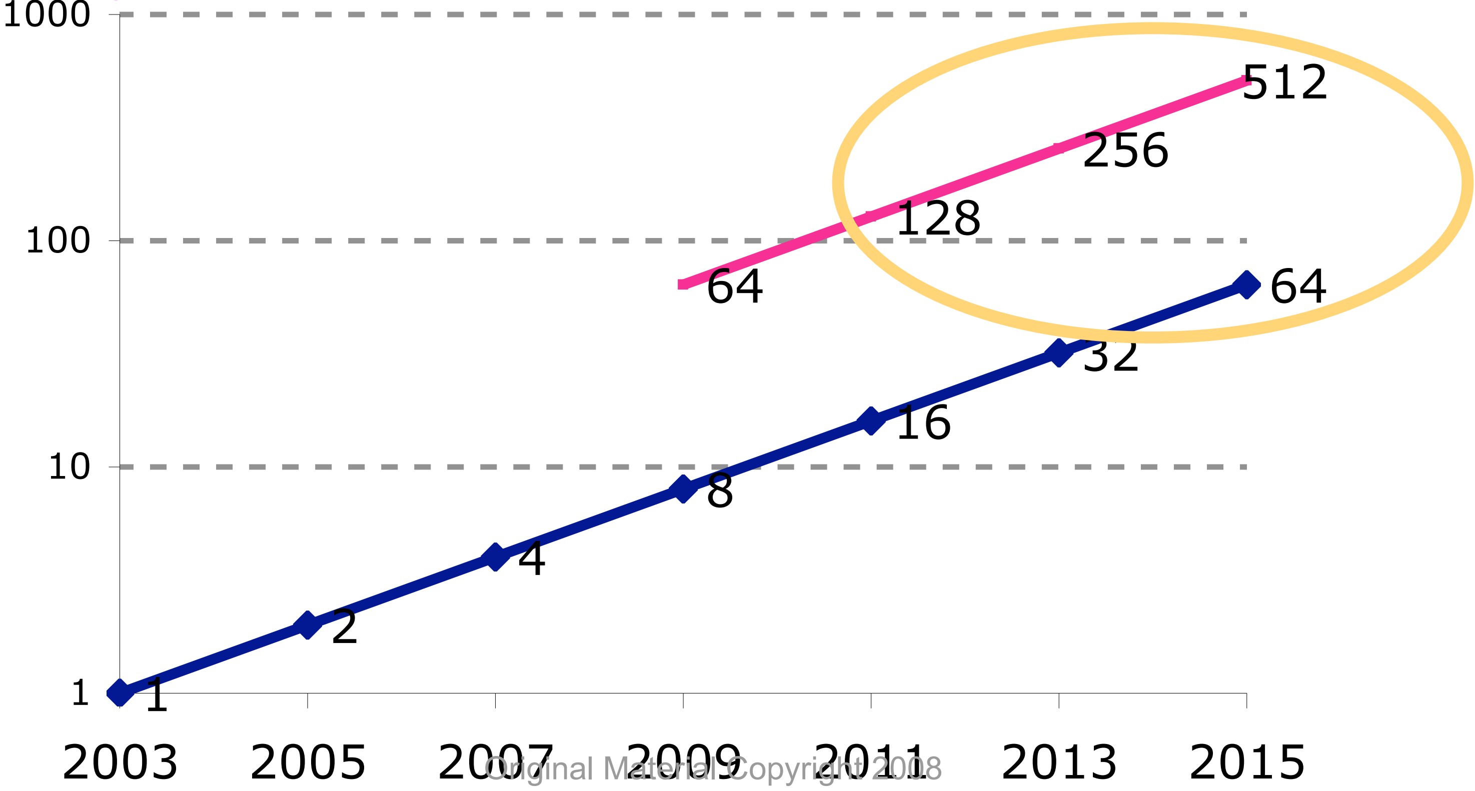
- We can foresee the days of 64 cores and more on one socket
- These look a lot like SMPs
- How will we deal with multicore and manycore systems if our platforms are currently 8- or 16-way?

Original Material Copyright 2008

Why Target 64+ Cores?

(adapted from David Patterson, 2007)

- Multicore: 2X / 2 yrs ⇒ ≈ 64 cores in 8 years
- Manycore: 8X to 16X multicore



Original Material Copyright 2008

Current Multicores

(adapted from David Patterson, 2007)

Name	Clovertwn	Opteron	Cell	Niagara 2
Chips*Cores	2*4 = 8	2*2 = 4	1*8 = 8	1*8 = 8
Clock Rate	2.3 GHz	2.2 GHz	3.2 GHz	1.4 GHz
Peak MemBW	21 GB/s	21 GB/s	26 GB/s	41 GB/s
Peak GFLOPS	74.6 GF	17.6 GF	14.6 GF	11.2 GF
Naïve SpMV (median of many matrices)	1.0 GF	0.6 GF	--	2.7 GF
Efficiency %	1%	3%	--	24%

Sparse Matrix * Vector operations

Original Material Copyright 2008

Current Multicores

(adapted from David Patterson, 2007)

Name	Clovertwn	Opteron	Cell	Niagara 2
Chips*Cores	2*4 = 8	2*2 = 4	1*8 = 8	1*8 = 8
Clock Rate	2.3 GHz	2.2 GHz	3.2 GHz	1.4 GHz
Peak MemBW	21 GB/s	21 GB/s	26 GB/s	41 GB/s
Peak GFLOPS	74.6 GF	17.6 GF	14.6 GF	11.2 GF
Naïve SpMV (median of many matrices)	1.0 GF	0.6 GF	--	2.7 GF
Efficiency %	1%	3%		24%

Expertise is required to approach peak FLOPS!

Sparse Matrix * Vector operations

Consequences

Would it really be so bad?

“Hardware” Consequences

If we only have 8- and 16-way nodes in a cluster today…

- Who will have the skills to scale up an applications for 100’s of cores?
- Who can make use of co-processors?
  - Chess processors a key part of Deep Blue’s success
  - SSE and GP-GPUs as part of CPU
- Will we waste cores and functional units?

Original Material Copyright 2008

“Software” Consequences

If Linux is the “only” OS

- Who will work on scalability and NUMA issues?
  - Linux may become focused on mobile devices, or Web servers, or the next trend
- What if that desirable feature is not available on Linux (e.g., ZFS and license conflict)?
- Will we have “golden handcuffs”?

Original Material Copyright 2008

Research Consequences

- Using co-processors, your research competitor beats you to the answer
  - Deep Blue (1997) co-processors in chess
- Stagnant innovation
  - The history of browsers, compilers, etc.
- No platforms for capability applications and algorithm development
  - Latency-sensitive, irregular applications

Current Procurements

I am troubled by the following line of reasoning in HPC procurements:

- Clusters are the best in price-performance.
- We have lots of people who need the cycles.
- Therefore, let’s spend the money on more clusters.
- Since, we don’t need any/many SMPs nor other architectures.
- Furthermore, somebody else should be paying for diversity, since I don’t even have enough for me.

I think this is very short-sighted.
To know the price of everything, but the value of nothing.

Original Material Copyright 2008

Recommendations

- Accept that clusters *are* the new backbone of HPC.
- But, if the cluster was, say, 10% smaller, will people actually notice?
- So, leave room (and budget) as investment in diversity
  - For example, SMPs
    - Great for many applications *now.* They *will* get used!
    - Develop expertise for multi/manycores in foreseeable future
  - For example, different OSes
    - Linux is great, but it is being pulled in many directions
    - Other code bases should not be abandoned
- I am NOT saying that everybody should buy one of everything. Just don't spend everything on only one thing.

Original Material Copyright 2008

Concluding Remarks

1. A monoculture is bad in the long term.
2. Losing expertise is the greatest risk.
3. Consequences: Hardware, Software, Research.

Key Idea: Balance, for the long-term.

Original Material Copyright 2008