Systems and Technology Group

IPDPS 2007 Panel Position:

Is the Multi-Core Roadmap going to Live Up to its Promises?

Greg Pfister
Distinguished Engineer
IBM Systems and Technology Group
Austin, TX

The Legal Page

Any relationship between the opinions expressed here and any official position of the IBM Corporation…

or of anyone else in the IBM Corporation

or, for that matter, anyone else anywhere

...is purely coincidental.

And all copyrighted words or works are the property of their respective owners.

Random gratuitous clipart

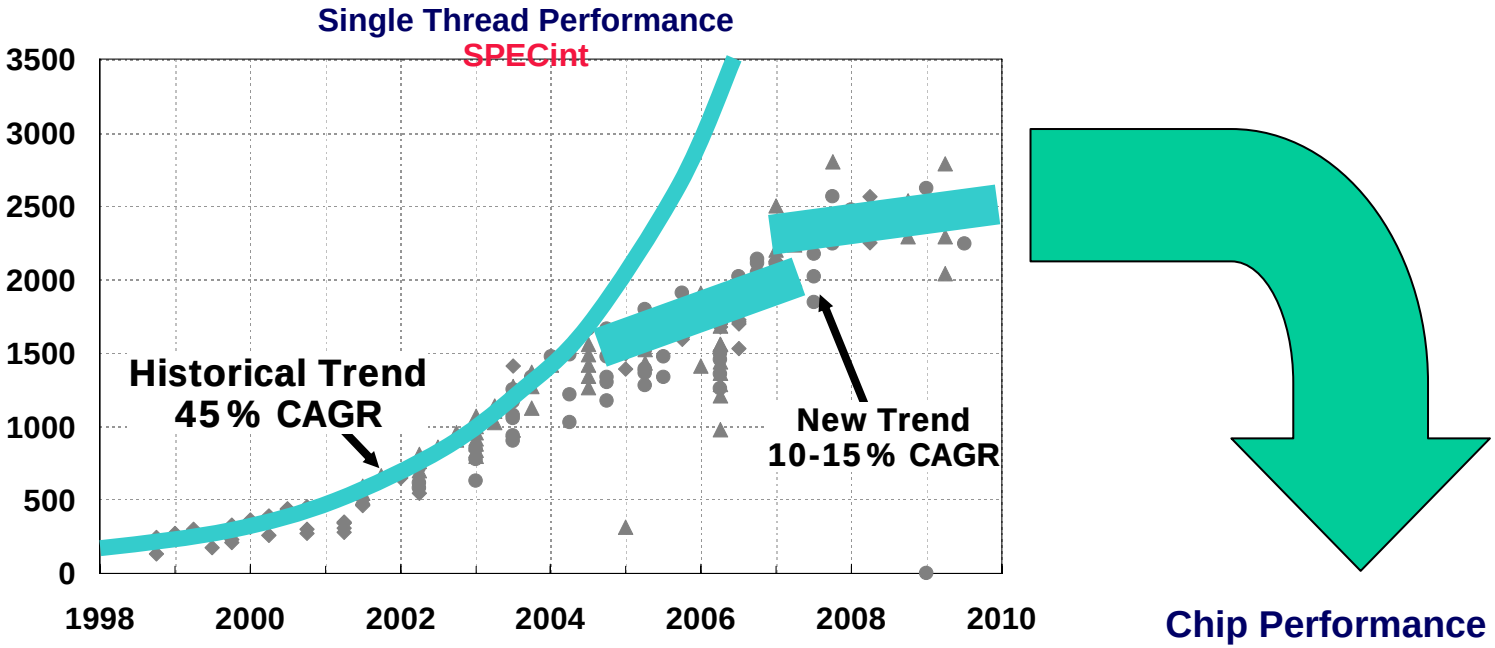
The Questions, and a Summary of My Answers

Is the computing **landscape** (technology, applications, and market) today sufficiently different to exploit multiprocessors from what it was in the past? If yes, in what sense? If not, why?	It depends.
Do we need more **research in multiprocessing** given past work? If yes, what are the biggest challenges? If not, state the reasons.	Sure.
Will progress in software/architecture make it possible to make sequential **languages** prevail? If yes, what are the top priorities in research to make that happen? If not, what are the visions for a parallel-language paradigm shift and what are the major challenges in software/architecture research to accelerate uptake in the programming community?	Yes; see below.
Would **multi-disciplinary** research (across the applications, algorithms, software, and architecture areas) be a good way to accelerate developments? Then, what areas should interact more closely and with what goals in mind?	It depends, but Yes.

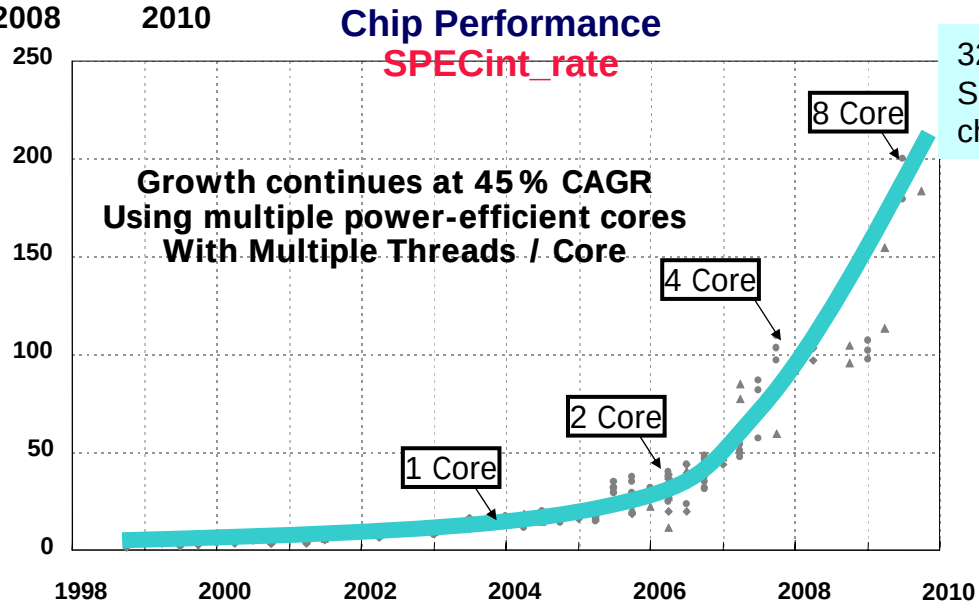
Dependencies

- What, exactly, is this “promise”?
- What market(s) are we talking about?
- Which programmers are we talking about?
- What do you mean by “discipline”?

The Promise (My Interpretation)



- Just change the metric.
- Promise, or threat?
- Or just reality.
- We – IPDPS folks – have been working this for 30+ years.
- SMP: The most conservative form of parallelism.
- The role of demos.



32-way SMP / chip?

Landscape

- Servers for commercial processing – OK.
  - Change:
    - All processing used to be on a monolithic mainframe.
    - Now uses multiple tiers, scales out well; back ends scale up.
  - Virtualization (partitioning) coming on strong.
    - Also helps power, space, system management.
- Clients – not OK.
  - My mind boggles at parallel Powerpoint.
  - Parallel Photoshop boggles memory.
  - Virtual worlds? Virtual World Web? vBusiness?
    - Implementations embarrassingly serial, but parallelism is there.

  Client-based volume has driven the industry. Be concerned.
- HPC – Not one area. Mixed bag. **VAST** simplification:
  - Much now COTS or "COTS-like" clusters. OK, like Commercial.
  - Some: stream processing (will be mainstream). OK.
  - Rest? Either will change to one of those, or run on systems purchased like military aircraft.

Research in Multiprocessing (Hardware)

- Power, power, power. And power.
- More efficient virtualization, especially I/O.
- Synchronization that simplifies programming.
  - *Possibly* transactional memory?
    - No, it doesn't help wizard programmers.
      - The wizards aren't the ones I worry about.
- Coherent memory access: bandwidth and efficiency.
- Efficient and usable accelerator attachment.
  - now usable, or efficient, not both. (Not physics.)
  - and virtualization of accelerators.
  - and virtualization of their attachment.
- Integration of explicit data streaming with and into conventional processing units.
- Did I mention power?

Languages & Multidisciplinary (1)

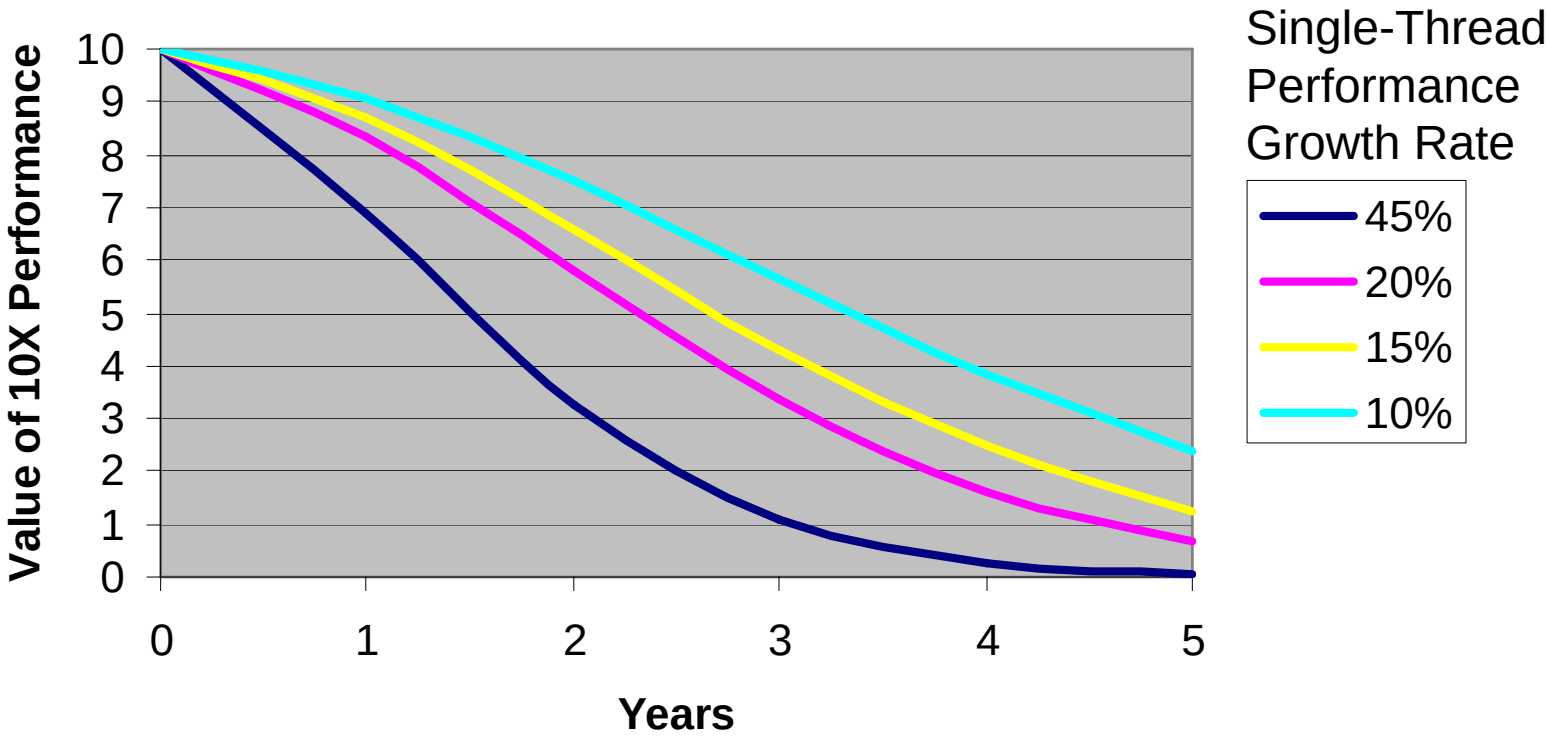
- An IBM Senior Vice President said:
  - "I am worried."
- And Lo, there came to be a Task Force.
- I arrived bearing a list:
  - 101 active and/or well-known parallel languages.
  - **None** with any traction in the market.
  - We have Fortran, C, C++, Java. And MPI, and OpenMP.
- A necessary question: What's Wrong?

Languages & Multidisciplinary (2)

- Contrast: Commercial middleware (J2EE, SOA, DBS, etc.)
  - Key: Eliminates a huge amount of the boilerplate gruntwork.
    - **Makes the programmer's job easier.**
  - And just by the way, provides *completely transparent* parallelism.
  - One major reason commercial servers are OK.
- Confusion of the "answer" with the requirements
  - Nobody *wants* to program in parallel for the sake of parallelism.
  - Parallelism in and of itself does not simplify their jobs.
  - Want: express applications more simply, clearly, easily.
- Best: Tools which
  - 1. **Help an application area. A lot.**
    - Must do or you are toast; won't be used.
  - 2. *As a side effect* provide parallelism.

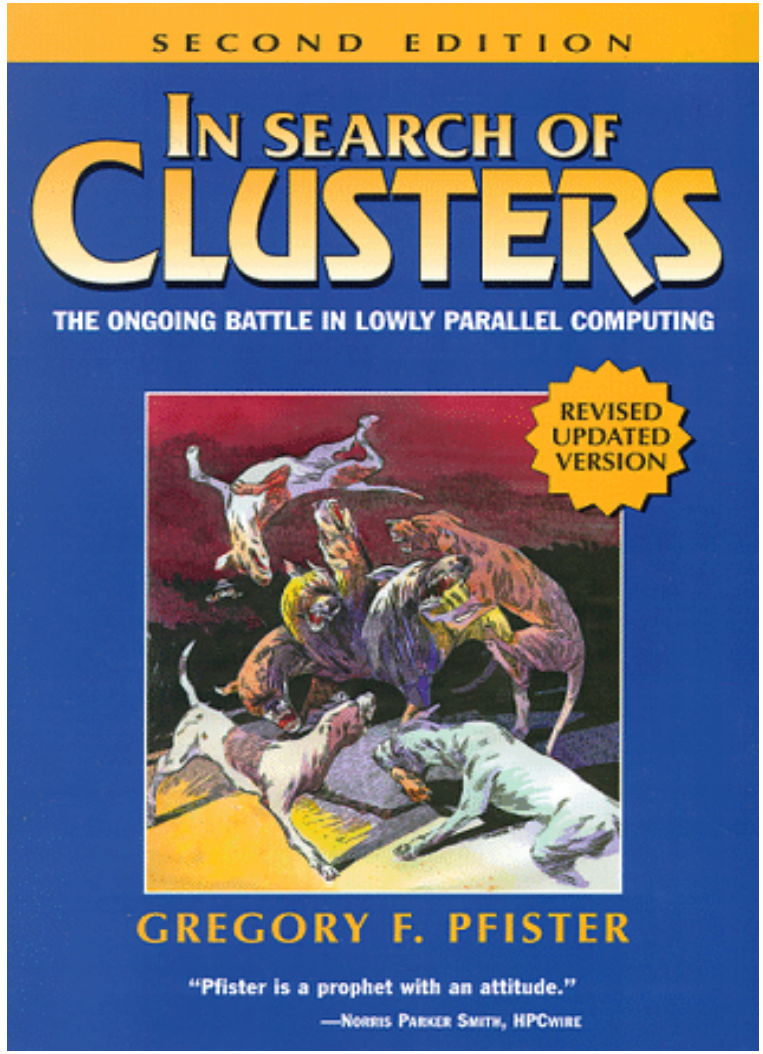
  E.g., not a stream programming tool/language, a *CFD* tool.
  Must get inside the heads of the disciplinarians.
- Implicit: Need tools appropriate to different programmers' roles:
  - OS, system libraries, middleware, application libraries, applications
  - Maybe these are best considered their own disciplines.

Accelerator / Architecture Longevity (Controversial!)



- Frequency growth slowdown ⇨ enhanced accelerator business case:
  - Past (45%): After 3 years – useless.
  - Now (<20%): as much 5 years useful lifetime
- Controversy? *Aggregate* chip performance still at 45%.
- **Has architecture ever really mattered in the long run?**
  - RISC, CISC, vector, SIMD, MIMD, whatever; who cares in 3 years?
  - **Has that changed?**

- Thank you for listening.



Just in case any of you were wondering...

(No, I can't give a presentation without plugging my book.)

Extremely nonrandom clipart