CS 498
Hot Topics in High Performance Computing

Networks and Fault Tolerance

1. Introduction to Parallel Computer Architecture (I)

Our teaching style

- As interactive as possible
  - Ask many questions, immediately
  - We want you to understand all presented topics
  - Additional references and (maybe) homework for deeper understanding
- We will ask questions
  - To check if you understand
  - Please don’t let us down ☺

Structure of the Networking Part

- Section I: Introduction to HPC
- Section II: Parallel Architectures
- Section III: HPC Networking Basics
- Section IV: Topology
- Section V: Routing
- Section VI: Flow Control
- Section VII: Host Interface
- Section VIII: HPC Network Examples

Sources

- Books
  - Leighton: “Arrays, Trees, and Hypercubes”
  - Culler, Singh, Gupta: “Parallel Computer Architecture”
  - Xu: “Topological Structure and Analysis of Interconnection Networks”
  - Dally, Towles: “Interconnection Networks”
- Slide sets
  - Culler’s CS 258 (S99)
  - Rober Fiedler’s Blue Waters presentations
  - Various others, credited on slides

Structure of the FT Part

- Section I: Why Fault tolerance in HPC, What are faults in HPC systems?
- Section II: What fault tolerance techniques?
- Section III: What is checkpointing and what to checkpoint?
- Section IV When and where to checkpoint?
- Section V: How to make sure that a checkpointed parallel execution will lead to correct results after restart?
- Section VI: How to Coordinate checkpointing?
- Section VII: What about Uncoordinated checkpointing and message logging?
- Section VIII: Can we improve message logging?
- Section IX: How to Hybrid fault tolerance protocols?
- Section X: Can we predict faults, errors and failures?

Documents and tools

- Fault tolerance for Distributed system is a not a new but it is a young domain for Parallel Commuting
- Books/articles related to Faults, Errors, Failures and distributed computing
  - A. Avizienis et al. "Basic Concepts and Taxonomy of Dependable and Secure Computing", IEEE Transactions on dependable and secure computing, Vol.1, No 1 January-March 2004
  - N. Lynch "Distributed Algorithms", Morgan Kaufmann Publishers Inc. 1996 ISBN:1558603484
  - E. Elnozahi "A Survey of Rollback-Recovery Protocols in Message Passing Systems", ACM Computing Survey, Vol. 34, No. 3, pp. 375-408, September 2002."
- Article related to FT/Resilience in Parallel Computing
  - F. Cappello et al. "Toward Exascale Resilience". IJHPCA 23(4): 374-388 (2009)
  - F. Cappello "Fault Tolerance in Petascale/ Exascale Systems: Current Knowledge, Challenges and Research Opportunities". IJHPCA 23(3): 212-226 (2009)
- Tools:
  - MPICH-V (INRIA), Open MPI, MVAPICH
  - BLCR (LBNL)
  - SCR (LLNL)

What is High Performance Computing

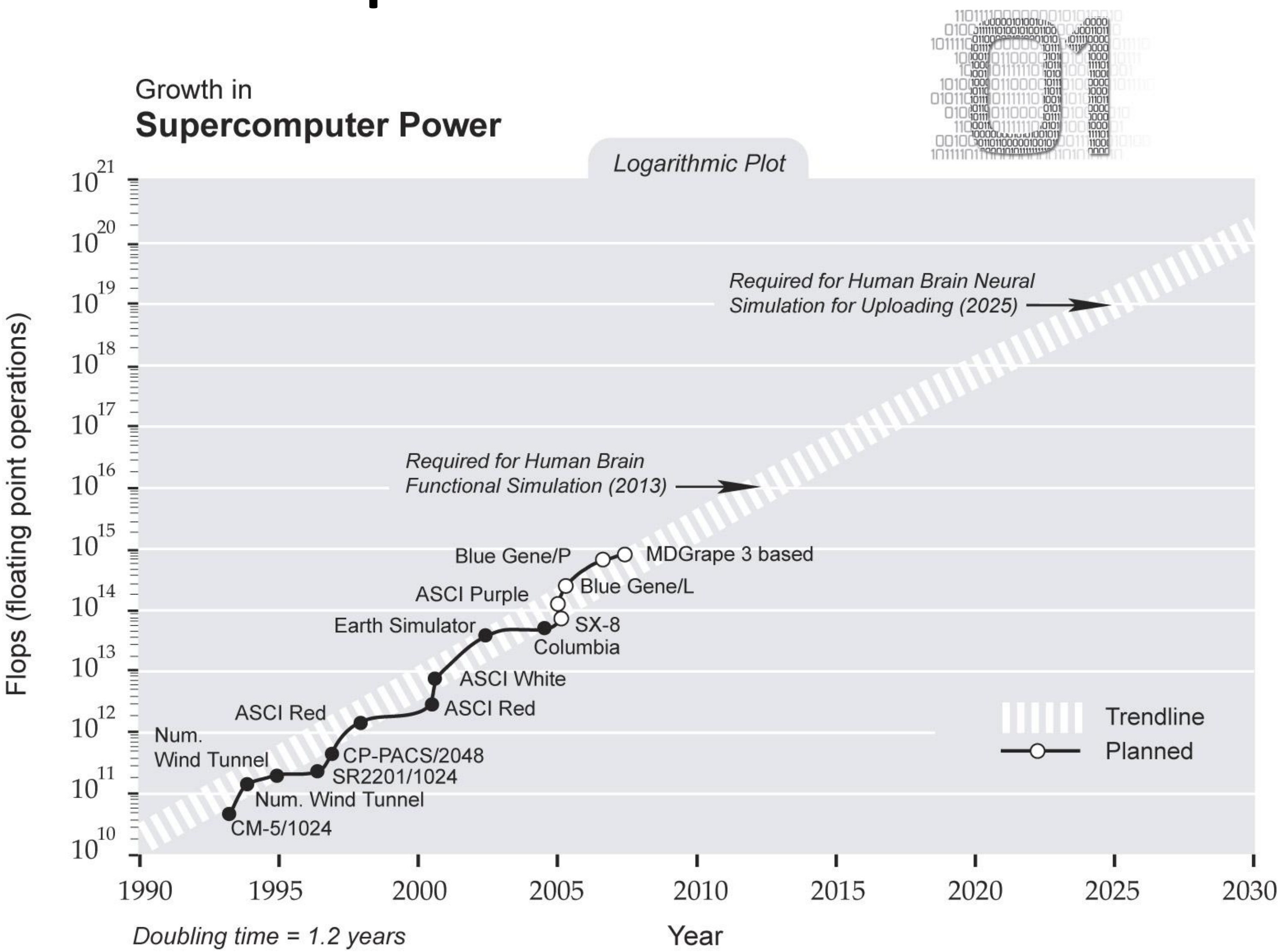
- Synonyms, aka.:
  - HPC, Supercomputing, High End Computing (HEC)

- "The largest and most powerful machines of mankind"
  - Typically huge investments (billions of $$)
  - But don't worry, history shows that you'll have the same performance on your lap 10-15 years later

Define “Powerful”

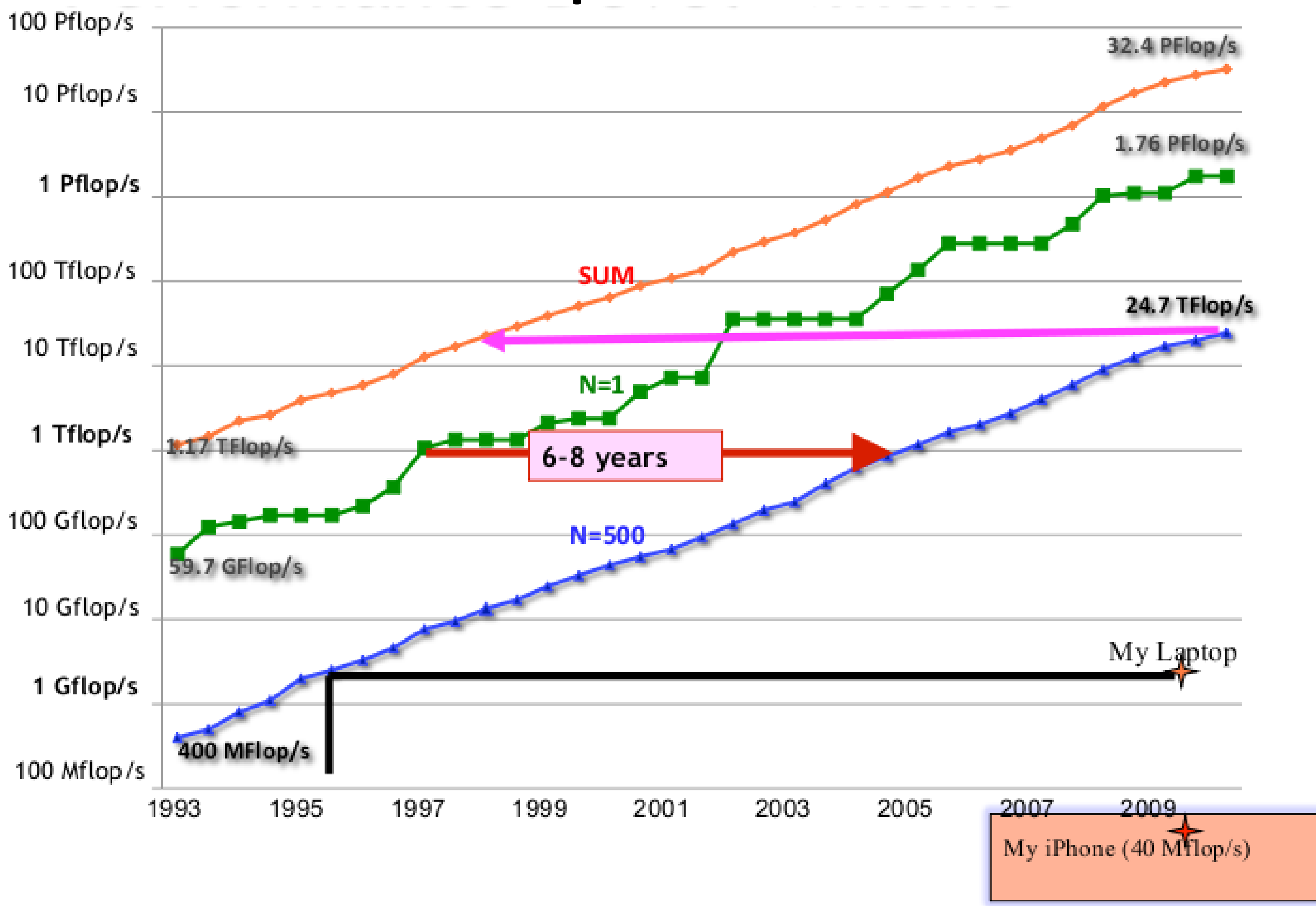
- Means high speed, high parallelism, high memory capacity, high …
- Traditional measure in scientific computing is FLOP/s (Floating Point Operations per second)
  - Disadvantage: only reflects one dimension of the optimization space
  - Advantage: historical data back to the 80’s, extrapolations were pretty exact
  - We’re on an exponential trajectory!

Exponential Growth



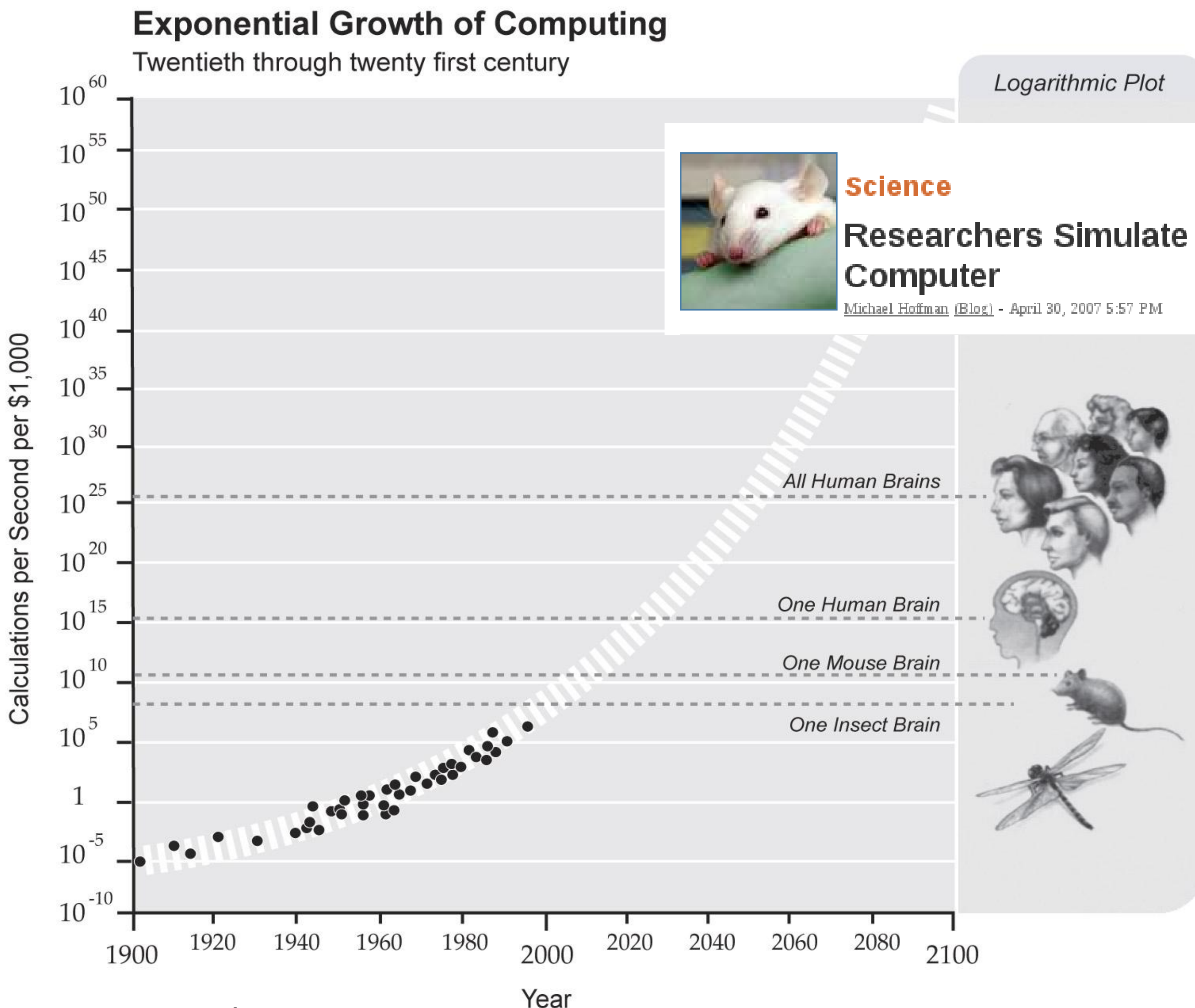
Source: www.singularity.com

Top 500



Source: Jack Dongarra, EuroMPI 2010 keynote

Singularity is close



Science

Researchers Simulate Mouse Brain on Computer

Michael Hoffman (Blog) - April 30, 2007 5:57 PM

Source: www.singularity.com

What do we need that much power for?

- Solving bigger problems than we could solve before!
  - E.g., Gene sequencing and search, simulation of whole cells, mathematics of the brain, …

- Solve small/existing problems faster!
  - E.g., large (combinatorial) searches, mechanical simulations (aircrafts, cars, weapons, …)

Scientific Computing - Problem Areas

- Most natural sciences are simulation driven are moving towards simulation
  – Theoretical physics (solving the Schrödinger equation, QCD)
  – Biology (Gene sequencing)
  – Chemistry (Material science)
  – Astronomy (Colliding black holes)
  – Medicine (Protein folding for drug discovery)
  – Meteorology (Storm/Tornado prediction)
  – Geology (Oil reservoir management, oil exploration)
  – and many more … (even Pringles uses HPC)

Commercial Computing – Problem Areas

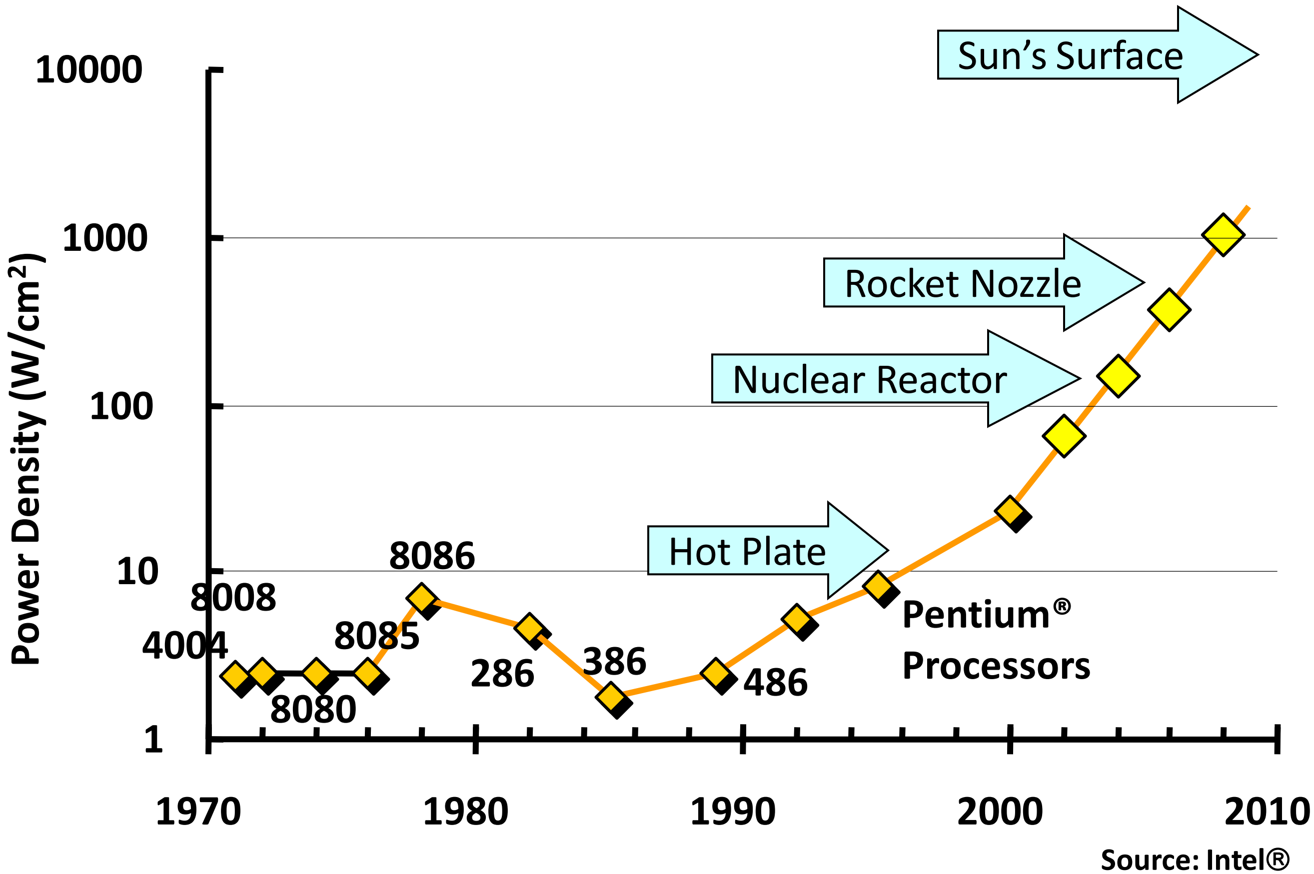
- Databases, data mining, search
  - Amazon, Facebook, Google
- Transaction processing
  - Visa, Mastercard
- Decision support
  - Stock markets, Wall Street, Military applications
- Parallelism in high-end systems and back-ends
  - Often throughput-oriented
  - Used equipment varies from COTS (Google) to high-end redundant mainframes (banks)

Commercial Computing – Industries

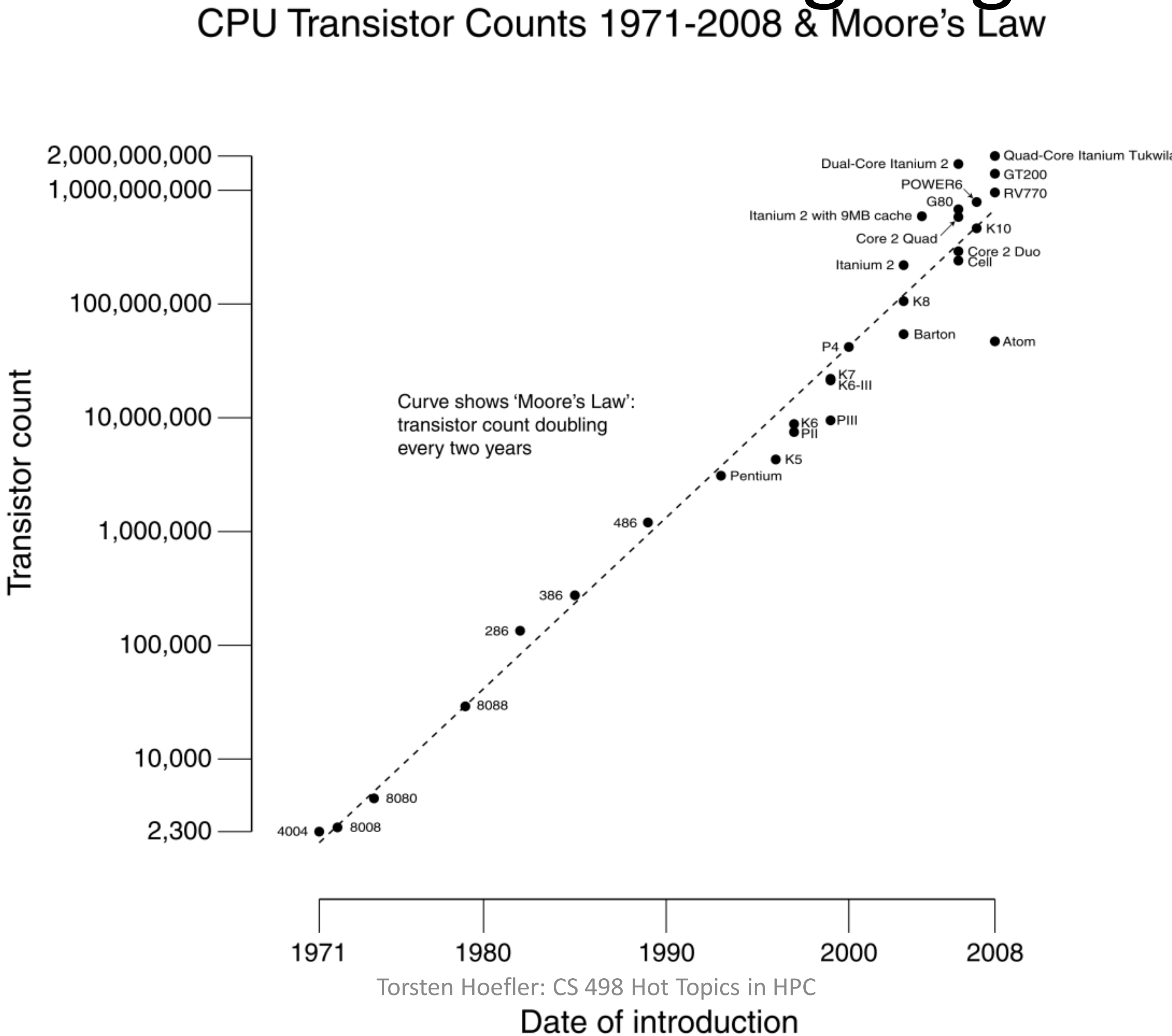
- Aeronautics (airflow, engine, structural mechanics, electromagnetism)
- Automotive (crash simulation, combustion, airflow)
- Computer-aided design (CAD)
- Pharmaceuticals (molecular modeling, protein folding, drug design)
- Petroleum (Reservoir analysis)
- Visualization (all of the above, movies, 3d)
- Financial modeling

How to increase compute power?

Clock Speed:



But Moore’s law is still going strong



What to do with more transistors?

- Architectural innovations
  - Branch prediction, Tomasulo logic/rename register, speculative execution, …
  - Help only so much ☹
- What else?
  - Simplification is beneficial, less transistors per CPU, more CPUs, e.g., Cell B.E., GPUs
  - We call this “cores” these days
  - Also, more intelligent devices or higher bandwidths (e.g., DMA controller, intelligent NICs)

Moving Toward Parallelism

- Everything goes parallel
  - Desktop computers get more cores
    - 2,4,8, soon dozens, hundreds?
  - Supercomputers get more PEs (cores, nodes)
    - > 256.000 today
    - > 1.6 million tomorrow
    - > 1 billion in a couple of years (after 2020)
  - Supercomputers get more nodes
    - ~20.000 today
    - >100.000 to 1 million in some years

What will you get out of this class?

- Understand engineering and design of modern network architectures and fault tolerance for HPC
  - Technology limits
  - Fundamental issues
    - Addressing, replication, communication, synchronization
    - Application properties, Execution/results correctness
  - Design techniques, tradeoffs and limits
    - Cache coherence, protocols, pipelining, etc.
    - FT protocols, checkpointing, log analysis, error detection/correction
  - At all scales (small to large)
  - Interactions between hardware/software
    - Moving/fuzzy boundary

Why does it benefit you?

- You'll most likely not become a computer or network architect
  - Not many needed but even less students ☺
- You may become a computer science researcher or need to develop/use large HPC apps.
  - Understanding FT is important to get high efficiency
- Fundamental issues translate across wide variety of systems
  - Parallelism an FT are inevitable
- Supercomputing is "pioneering" at the top of the pyramid
  - technology migrates downward over time
- Software/hardware interactions are important at all levels

Why study Networks?

- Role of a network architect:
  - *"Design and architect an interconnection network to maximize performance and programmability within limits of current technology and cost"*
  - The network is central to parallel computer architecture and its importance grows
- *"The networks of today's HPC systems easily cost more than half of the system and for Exascale, the network might be by far the dominating cost."*

Why study FT for HPC?

- FT need for HPC was marginal because HPC system MTBF were high enough (1 week, 1 month). This is not true anymore for large systems today (MTBF of 1day and less are seen)
- It can only get worse with the increase of the number of components and component complexity
- There is no compromise:
  - *Fault tolerance is not like other problems of HPC (performance, efficiency, power consumption, etc.) → there is no half success:*
  - *Application execution succeeds with correct results or fails!*
- Clouds are starting considering HPC applications and Cloud nodes have typically a much lower MTBF than HPC nodes.

Why study it Now?

- Everything “parallel” is **hot** and importance increases!
- Current technology trends make parallel computing inevitable! Architectures and networks are still under active development.
- Since Moore’s law just transitioned from higher frequency to higher number of components, faults, errors and failures rate will inevitably increase
- Everyone needs to understand fundamental principles and tradeoffs, not just taxonomies
  – There is still a lot to be done!

Blue Waters (quick motivation)

System Attribute	Track 2 (kraken)	DOE (jaguar)	Track 1 (Blue Waters*)
Vendor	Cray	Cray	IBM
Processor	AMD Istanbul	AMD Istanbul	Power 7
Peak Perf. (PF)	1.03	2.33	~10
Sustained Perf. (PF)	~0.1	~0.233	~1.0
Number of cores	99,072	224,256	300,000+
Amount of Memory (PB)	0.129	0.3	1.2+
Amount of Disk Storage (PB)	2.4	5	18+
Amount of Archival Storage (PB)	??	??	Grow to 500
External Bandwidth (Gb/s)	??	??	100-400

* *Reference petascale computing system (no accelerators).*