Ter@tec

Pôle européen de compétence
en simulation numérique haute performance

Solver software infrastructure for exascale applications

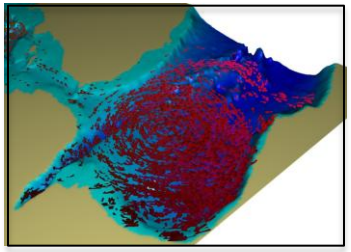
David Keyes, Applied Mathematics & Computational Science

Director, Extreme Computing Research Center (ECRC)

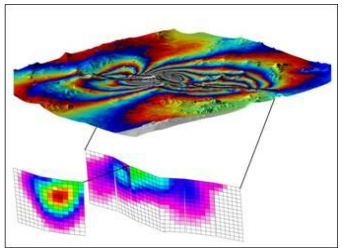
King Abdullah University of Science and Technology

Philosophy of software investment

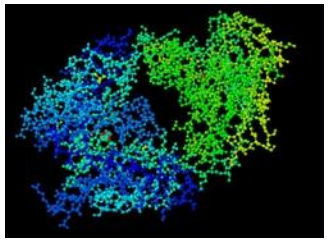
I. Hoteit



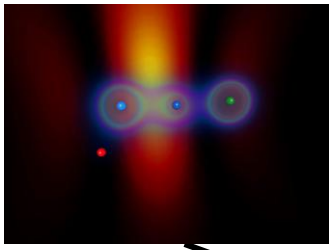
M. Mai



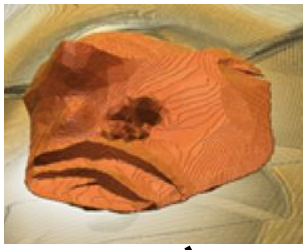
V. Bajic



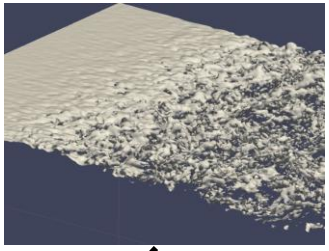
A. Fratalocchi



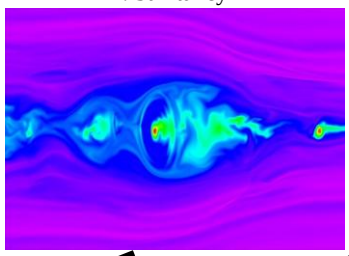
G. Schuster



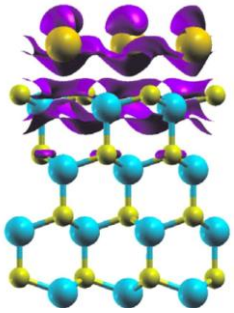
F. Bisetti



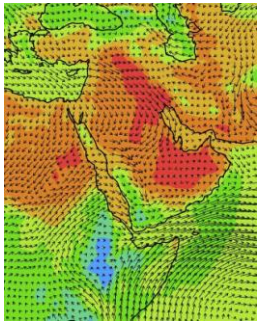
R. Samtaney

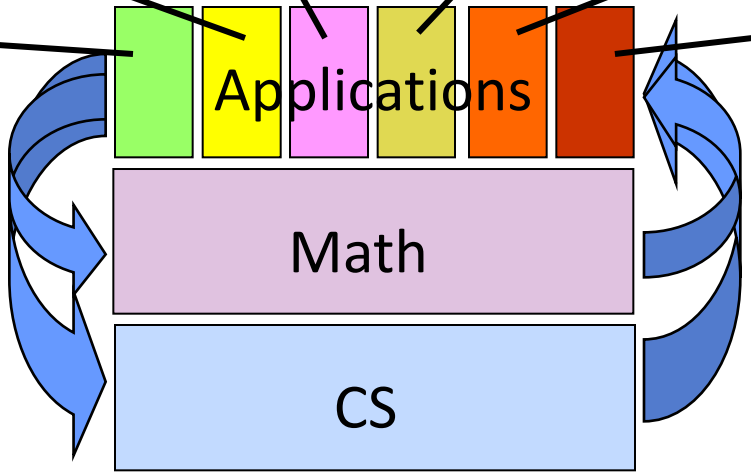


U. Schwingenschloegl



G. Stenchikov





Applications drive

Math & CS enable

France and KAUST

(top five academics are French or Francophone)

Jean-Lou Chameau

President
PhD, Stanford, 1973
came from Caltech
Légion d'honneur

Jean Fréchet

VP for Research
PhD Syracuse, 1971
came from Berkeley
NAS, NAE, Japan Prize

Yves Gnanou

Dean, PSE
PhD Strasbourg, 1985
came from Ecole Polytechnique
Légion d'honneur

Pierre Magistretti

Dean, BESE
PhD UCSD, 1982
came from EPFL

Mootaz Elnozahy

Dean, CEMSE
PhD Rice, 1993
came from IBM

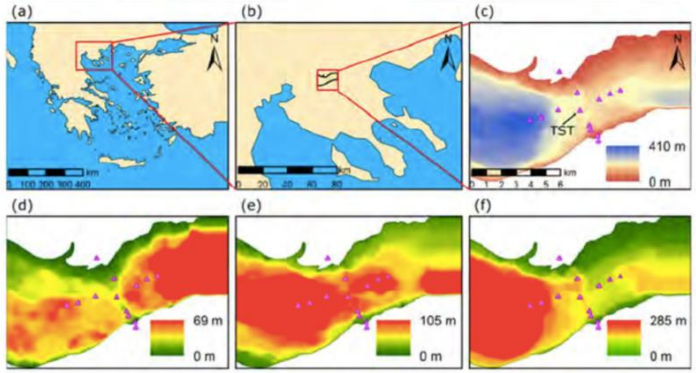
France and KAUST in HPC

l'Observatoire de Paris

Multi-objective adaptive optics project

Géosciences pour une Terre durable
brgm

Euro "seistest" project: Mygdonian basin



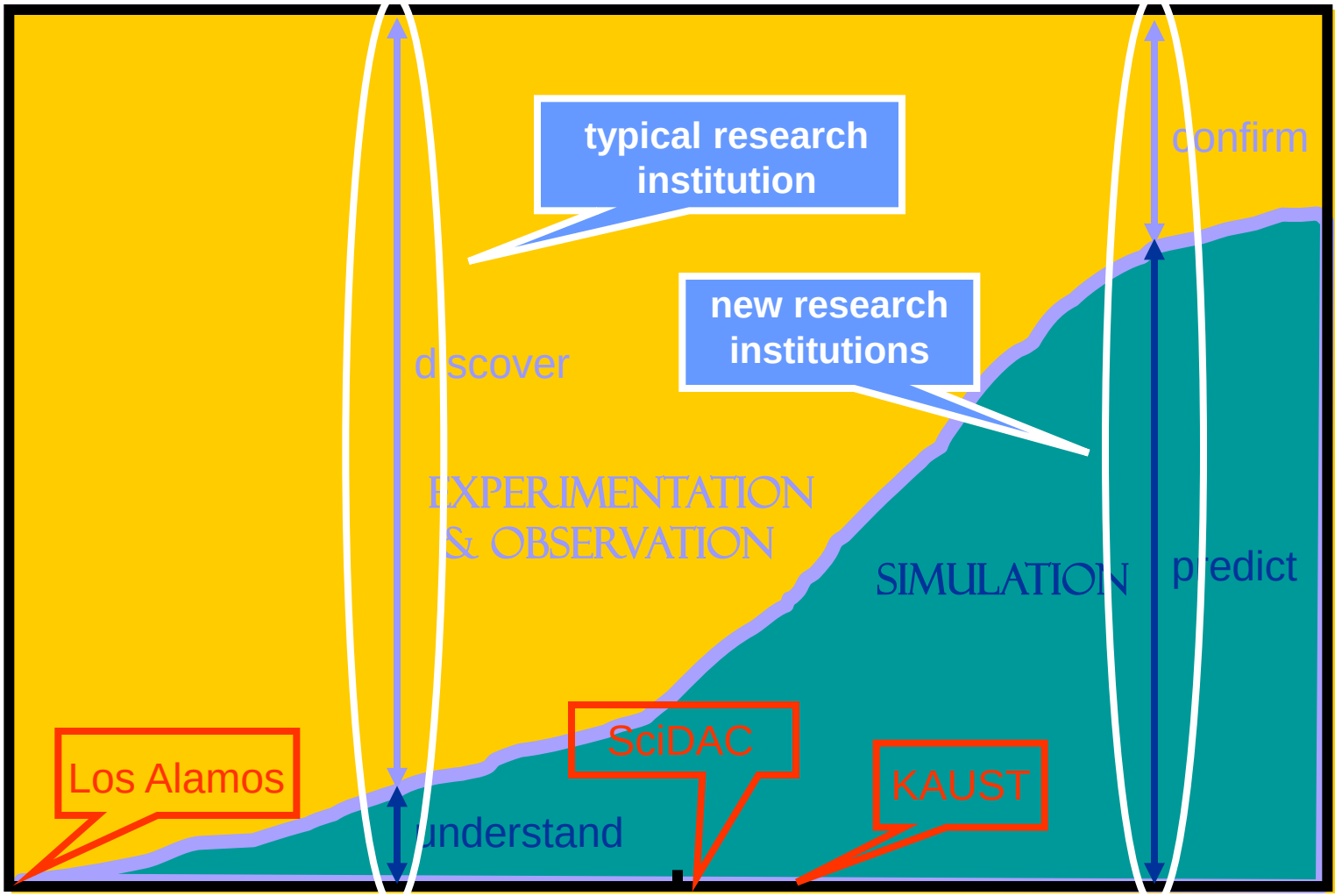
TOTAL
COMMITTED TO BETTER ENERGY

IFP Energies nouvelles — Innover les énergies

KAUST and the four scientific paradigms

(many institutions are still firing on just two of the four cylinders)

Advance of the third paradigm



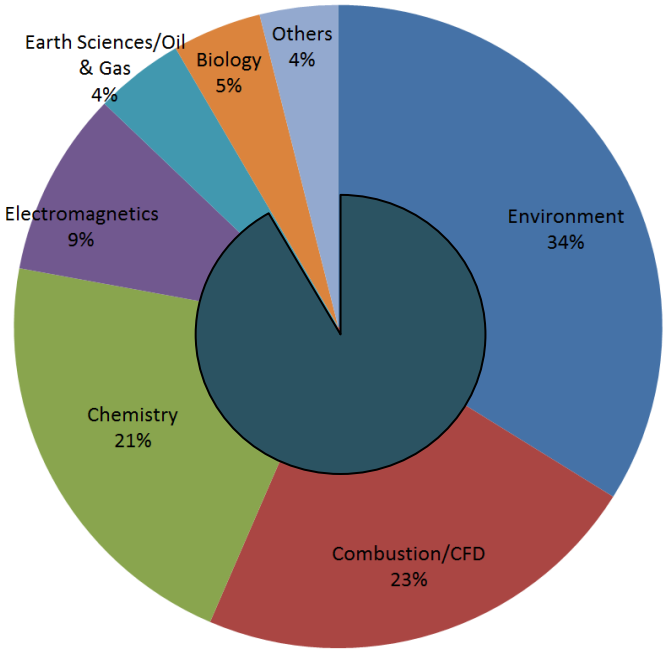
SHAHEEN

CORNEA

Shaheen has been a scientific instrument for environment and energy simulations

Science Area	Codes
Atmospheric Modeling	WRF, WRF-Chem, HIRAM
Ocean Modeling	WRF, MITgcm
CFD/Plasma	Plasmoid – in house code
CFD/Combustion	NGA, S3D
Computational Biology	In-house genomic motif identification code
Computational Earthquake Seismology	SeisSol, SPECFEM_3D_GLOBE
Computational Electromagnetism	In house explicit code
Big Data/ Analysis of Large Graphs	Mizan - in house code
Computational Chemistry	VASP, LAMMPS, Gaussian, WEIN2k, Quantum ESPRESSO
Seismic imaging/Oil & gas	In house 3D reverse time migration code

Major Shaheen application codes



Shaheen-1 utilization by area, 2009-15

Biggest application has been the Coordinated Regional Climate Downscaling Experiment (CORDEX) project to advance regional climate modeling

Presented at the COP21 global climate negotiations in Paris in December 2016

WCRP World Climate Research Programme

Site	Vendor	Computer	Country	Cores	Rmax [Pflops]	Power [MW]
National Supercomputing Center in Wuxi	NRCPC	Sunway TaihuLight NRCPC Sunway SW26010, 260C 1.45GHz	China	10,649,600	93.0	15.4
National University of Defense Technology	NUDT	Tianhe-2 NUDT TH-IVB-FEP, Xeon 12C 2.2GHz, IntelXeon Phi	China	3,120,000	33.9	17.8
Oak Ridge National Laboratory	Cray	Titan Cray XK7, Opteron 16C 2.2GHz, Gemini, NVIDIA K20x	USA	560,640	17.6	8.21
Lawrence Livermore National Laboratory	IBM	Sequoia BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	1,572,864	17.2	7.89
RIKEN Advanced Institute for Computational Science	Fujitsu	K Computer SPARC64 VIIIfx 2.0GHz, Tofu Interconnect	Japan	795,024	10.5	12.7
Argonne National Laboratory	IBM	Mira BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	786,432	8.59	3.95
Los Alamos NL / Sandia NL	Cray	Trinity Cray XC40, Xeon E5 16C 2.3GHz, Aries	USA	301,0564	8.10	4.23
Swiss National Supercomputing Centre (CSCS)	Cray	Piz Daint Cray XC30, Xeon E5 8C 2.6GHz, Aries, NVIDIA K20x	Switzer-land	115,984	6.27	2.33
HLRS – Stuttgart	Cray	Hazel Hen Cray XC40, Xeon E5 12C 2.5GHz, Aries	Germany	185,088	5.64	3.62
King Abdullah University of Science and Technology	Cray	Shaheen II Cray XC40, Xeon E5 16C 2.3GHz, Aries	Saudi Arabia	196,608	5.54	2.83

Shaheen II

KAUST Supercomputing Lab

جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology

Shaheen II Specs

- 36 cabinets of Cray XC40 with Intel Haswell 2.3 Ghz with 16 cores
- 128 GB of RAM per node
- Number of nodes: 6192
- Number of cores: 198144
- Peak Performance: 7.3 PFlops/s
- LINPACK : 5.6 PFlops/s
- 2.8 MW at peak
- 17.4 PB of Parallel File System
- I/O throughput: over 500 GB/s
- Burst Buffer capacity: 1.5 TB
- Burst Buffer throughput: over 1.2 TB/s

Ranked in Jun'16 lists:
#10 on HPL
#4 on HPGMG

Shaheen I ➔ Shaheen II

IBM Blue Gene/P	Cray XC40
June 2009	**May 2015**
Speed: **.222 Petaflop/s (peak)** Entry ranking: **#14 HPL R_{max} (2009)**	Speed: **7.3 Petaflop/s (peak, ↑ ~33X)** Ranking**: #7 HPL R_{max} (2015)**
Power: 0**.5 MW (0.44 GF/s/W)** Cooling: **air**	Power: **2.8 MW (~2 GF/s/Watt, ↑ ~5X)** Cooling: **water**
Memory: **65 TeraBytes** Amdahl-Case Ratio: **0.29 B/F/s**	Memory: **793 TeraBytes (↑ ~12X)** Amdahl-Case Ratio: **0.11 B/F/s (↓ ~3X)**
I/O bandwidth: **25 GB/s** Storage: **2.7 PetaBytes**	I/O bandwidth: **500 GB/s (↑ ~20X)** Storage: **17.6 PetaBytes (↑ ~6.5X)**
Nodes: **16,384** Cores: **65,536 at 0.85 Ghz**	Nodes: **6,192** Cores: **198,144 at 2.3 Ghz**
Burst buffer: **none**	Burst buffer: **1.5 Petabytes, 1.2 TB/s bandwidth**

A growing imbalance!

Sunway TaihuLight (#1 on Top500 HPL)

Further imbalance!

TaihuLight compared with Shaheen-2

	TaihuLight	Shaheen-2	Ratio
Cores	**10,649,600**	**196,608**	**54.1**
Peak	**125.4 PF/s**	**7.235 PF/s**	**17.3**
Primary Memory	**1.31 PB**	**0.793 PB**	**1.65**
Amdahl-Case Ratio	**0.0104 B/(F/s)**	**0.110 B/(F/s)**	**0.0945**
HPL	**93.01 PF/s**	**5.536 PF/s**	**16.8**
HPL Rank	***#1 (74.2%)***	***#10 (76.5%)***	
HPCG	**0.371 PF/s**	**0.114 PF/s**	**3.25**
HPCG Rank	***#3 (0.297%)***	***#12 (1.57%)***	
Power	**15.37 MW**	**2.834 MW**	**5.42**
Power Eff.	**8.16 GF/s/W**	**2.55 GF/s/W**	**3.20**

“A good player plays to where the ball is; a great player plays to where the ball is going to be.”

(paraphrase of Wayne Gretzsky, with “ball” for “puck”)

Aspiration for this talk

To paraphrase Wayne Gretzsky:

“Algorithms for where architectures are going to be”

Such algorithms may *or may not* be the best today; however, hardware trends can be extrapolated to their sweet spots.

Examples being developed at the Extreme Computing Research Center (ECRC)

- **ACR, a new spin on 45-year-old cyclic reduction that recursively uses H matrices on Schur complements to reduce O(N^2) complexity to O($N \log^2 N$)**
- **FMM-Pre, a 30-year-old O(N) solver for potential problems with good asymptotic complexity but a bad constant when used at high accuracy, used in low accuracy as a FEM preconditioner**
- **QDWH-SVD, a 2-year-old SVD algorithm that performs more flops but generates essentially arbitrary amounts of dynamically schedulable concurrency, and beats state-of-the-art on GPUs**
- **MWD, a multicore wavefront diamond-tiling stencil evaluation library that reduces memory bandwidth pressure on multicore processors**
- **BDDC, a preconditioner well suited for high-contrast elliptic problems that trades lots of local flops for low iteration count**
- **MSPIN, a new nonlinear preconditioner that replaces most of the global synchronizations of Newton iteration with local problems**

Background of this talk:
www.exascale.org/iesp

INTERNATIONAL EXASCALE SOFTWARE PROJECT ROADMAP 1.0

Jack Dongarra, Pete Beckman, Terry Moore, Patrick Aerts, Giovanni Aloisio, Jean-Claude Andre, David Barkai, Jean-Yves Berthou, Taisuke Boku, Bertrand Braunschweig, Franck Cappello, Barbara Chapman, Xuebin Chi, Alok Choudhary, Sudip Dosanjh, Thom Dunning, Sandro Fiore, Al Geist, Bill Gropp, Robert Harrison, Mark Hereld, Michael Heroux, Adolfy Hoisie, Koh Hotta, Yutaka Ishikawa, Fred Johnson, Sanjay Kale, Richard Kenway, David Keyes, Bill Kramer, Jesus Labarta, Alain Lichnewsky, Thomas Lippert, Bob Lucas, Barney Maccabe, Satoshi Matsuoka, Paul Messina, Peter Michielse, Bernd Mohr, Matthias Mueller, Wolfgang Nagel, Hiroshi Nakashima, Michael E. Papka, Dan Reed, Mitsuhisa Sato, Ed Seidel, John Shalf, David Skinner, Marc Snir, Thomas Sterling, Rick Stevens, Fred Streitz, Bob Sugar, Shinji Sumimoto, William Tang, John Taylor, Rajeev Thakur, Anne Trefethen, Mateo Valero, Aad van der Steen, Jeffrey Vetter, Peg Williams, Robert Wisniewski, Kathy Yelick

SPONSORS

Office of Science U.S. Department of Energy, NSF, ANR, CEA, CEPTLO, CRAY, EDF, EPSRC Engineering and Physical Sciences Research Council, FUJITSU, INRIA, GENCI, NVIDIA, RIKEN, 東京大学 THE UNIVERSITY OF TOKYO

The International Exascale Software Roadmap,

J. Dongarra, P. Beckman, et al., *International Journal of High Performance Computer Applications* **25**(1), 2011, ISSN 1094-3420.

Uptake from IESP meetings

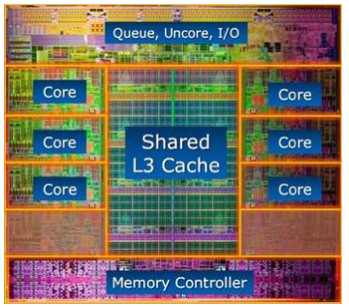
- While obtaining the next 2 orders of performance, we need an order of magnitude more Flop/s per Watt
  - target: 50 Gigaflop/s/W, today about 6.6 Gigaflop/s/W
- Draconian reduction required in power per flop and per byte will make computing and moving data less reliable
  - circuit elements will be smaller and subject to greater physical noise per signal, with less space and time redundancy for resilience in the hardware
  - more errors must be caught and corrected in software
- Power may be cycled off and on or clocks slowed and speeded
  - based on compute schedules (user-specified or software adaptive) and dynamic thermal monitoring
  - makes per-node performance rate unreliable

Why exa- is different

Which steps of FMADD take more energy?

64-bit floating-point fused multiply add **or** moving four 64-bit operands 20 mm across the die

```
      934,569.299814557                    input
x          52.827419489135904              input
-----------------------------
= 49,370,884.442971624253823
+           4.20349729193958               input
-----------------------------
= 49,370,888.64646892                      output
```



(Intel Sandy Bridge, 2.27B transistors)

Going across the die will require an order of magnitude more!

DARPA study predicts that by 2019:

- **Double precision FMADD flop: 11pJ**
- **cross-die per word access (1.2pJ/mm): 24pJ (= 96pJ overall)**

Today's power costs per operation

Operation	approximate energy cost
DP FMADD flop	100 pJ
DP DRAM read-to-register	4800 pJ
DP word transmit-to-neighbor	7500 pJ
DP word transmit-across-system	9000 pJ

Remember that a *pico* (10^{-12}) of something done *exa* (10^{18}) times per second is a *mega* (10^{6})-somethings per second

- **100 pJ at 1 Eflop/s is 100 MW (for the flop/s only!)**
- **1 MW-year costs about $1M ($0.12/KW-hr × 8760 hr/yr)**
  - **We "use" 1.4 KW continuously, so 100MW is 71,000 people**

c/o J. Shalf (LBNL)

Why exa- is different

Moore's Law (1965) does not end but Dennard's MOSFET scaling (1972) does

Table 1
Scaling Results for Circuit Performance

Device or Circuit Parameter	Scaling Factor
Device dimension t_{ox}, L, W	$1/\kappa$
Doping concentration N_a	κ
Voltage V	$1/\kappa$
Current I	$1/\kappa$
Capacitance $\epsilon A/t$	$1/\kappa$
Delay time/circuit VC/I	$1/\kappa$
Power dissipation/circuit VI	$1/\kappa^2$
Power density VI/A	1

Table 2
Scaling Results for Interconnection Lines

Parameter	Scaling Factor
Line resistance, $R_L = \rho L/Wt$	κ
Normalized voltage drop IR_L/V	κ
Line response time $R_L C$	1
Line current density I/A	κ

Robert Dennard, IBM
(inventor of DRAM, 1966)

Eventually processing is limited by transmission, as known for > 4 decades

Some exascale architecture trends

- **Clock rates cease to increase while arithmetic capability continues to increase dramatically w/concurrency consistent with Moore's Law**
- **Memory storage capacity diverges exponentially below arithmetic capacity**
- **Transmission capability (memory BW and network BW) diverges exponentially below arithmetic capability**
- **Mean time between hardware interrupts shortens**
- **➔ Billions of $ € £ ¥ of scientific software worldwide hangs in the balance until better algorithms arrive to span the architecture-applications gap**

Node-based "weak scaling" is routine; thread-based "strong scaling" is the game

- **Expanding the number of nodes (processor-memory units) beyond 10^6 is *not* a threat to algorithms that lend themselves to well-amortized precise load balancing**
  - **provided that the nodes are performance reliable**
- **The real challenge is usefully expanding the number of cores on a node to 10^3**
  - **must be done while memory and memory bandwidth per node expand by (at best) ten-fold less (basically "strong" scaling)**
  - **don't need to wait for full exascale systems to experiment in this regime – the battle is fought on individual shared-memory nodes**


Energy-aware generation
Bulk-synchronous generation

Bulk Synchronous Parallelism

Leslie Valiant, Harvard

2010 Turing Award Winner

A Bridging Model for Parallel Computation

The success of the von Neumann model of sequential computation is attributable to the fact that it is an efficient bridge between software and hardware: high-level languages can be efficiently compiled on to this model; yet it can be efficiently implemented in hardware. The author argues that an analogous bridge between software and hardware is required for parallel computation if that is to become as widely used. This article introduces the bulk-synchronous parallel (BSP) model as a candidate for this role, and gives results quantifying its efficiency both in implementing high-level language features and algorithms, as well as in being implemented in hardware.

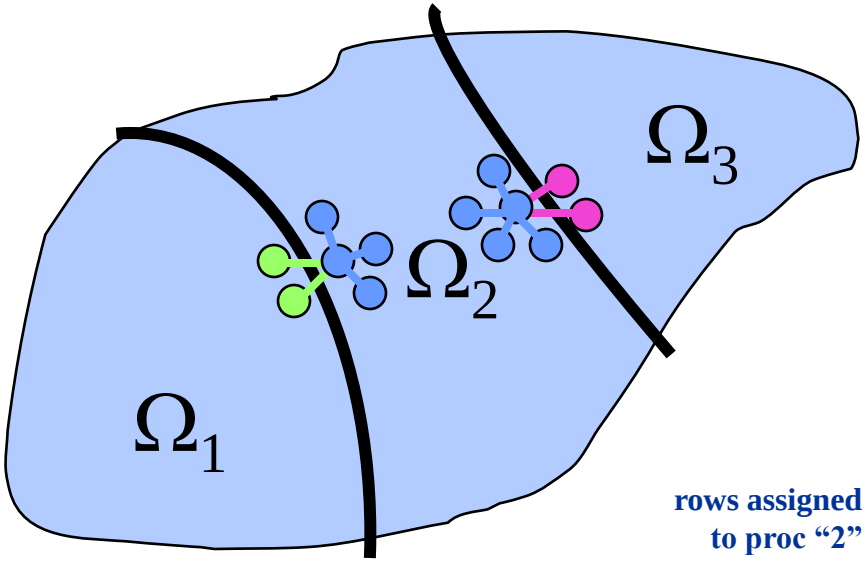
Leslie G. Valiant

Comm. of the ACM, 1990

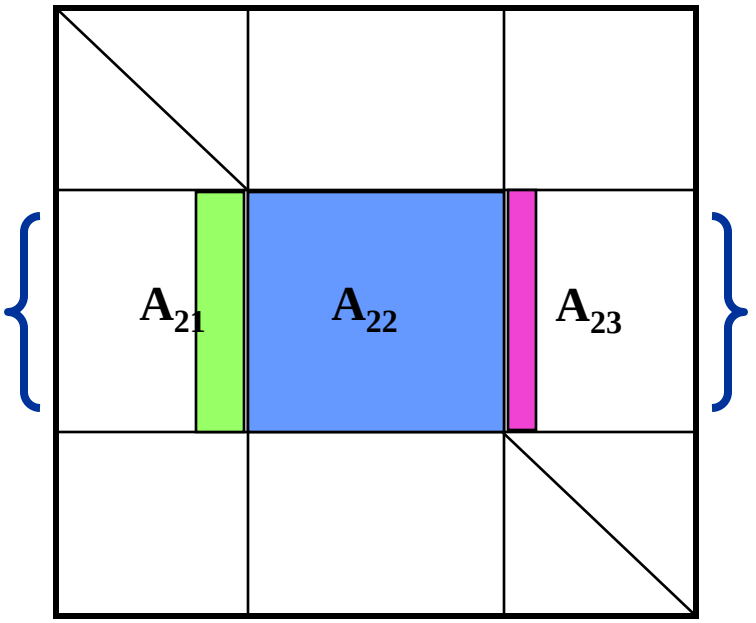
How are most simulations implemented at the petascale today?

- **Iterative methods based on data decomposition and message-passing**
  - **data structures are distributed**
  - **each individual processor works on a subdomain of the original**
  - **exchanges information at its boundaries with other processors that own portions with which it interacts causally, to evolve in time or to establish equilibrium**
  - **computation and neighbor communication are both fully parallelized and their ratio remains constant in weak scaling**
- **The programming model is BSP/SPMD/CSP**
  - **Bulk Synchronous Programming**
  - **Single Program, Multiple Data**
  - **Communicating Sequential Processes**

BSP parallelism w/ domain decomposition



rows assigned
to proc “2”



Partitioning of the grid induces block structure on the system matrix (Jacobian)

BSP has an impressive legacy

By the Gordon Bell Prize, performance on *real applications* (e.g., mechanics, materials, petroleum reservoirs, etc.) has improved *more* than a million times in two decades. Simulation *cost per performance* has improved by nearly a million times.

Gordon Bell Prize: Peak Performance Year	Gigaflop/s delivered to applications
1988	1
1998	1,020
2008	1,350,000

Gordon Bell Prize: Price Performance Year	Cost per delivered Gigaflop/s
1989	$2,500,000
1999	$6,900
2009	$8

Extrapolating exponentials eventually fails

- **Scientific computing at a crossroads w.r.t. extreme scale**
- **Proceeded steadily for decades from giga- (1988) to tera- (1998) to peta- (2008) with**
  - ***same* BSP programming model**
  - ***same* assumptions about who (hardware, systems software, applications software etc.) is responsible for what (resilience, performance, processor mapping, etc.)**
  - ***same* classes of algorithms (*cf*. 25 yrs. of Gordon Bell Prizes)**

Extrapolating exponentials eventually fails

- **Exa- is qualitatively different and looks more difficult**
  - **but we once said that about message passing**
- **Core numerical analysis and scientific computing will confront exascale to maintain relevance**
  - **not a "distraction," but an intellectual stimulus**
  - **potentially big gains in adapting to new hardware environment**
  - **the journey will be as fun as the destination**

Main challenge going forward for BSP

- **Almost all "good" algorithms in linear algebra, differential equations, integral equations, signal analysis, etc., require frequent synchronizing global communication**
  - **inner products, norms, and fresh global residuals are "addictive" idioms**
  - **tends to hurt efficiency beyond 100,000 processors**
  - **can be fragile for smaller concurrency, as well, due to algorithmic load imbalance, hardware performance variation, etc.**
- **Concurrency is heading into the billions of cores**
  - **Already 10 million on the most powerful system today**

Implications for algorithms

- **Plenty of ideas exist to adapt or substitute for favorite solvers with methods that have**
  - **reduced synchrony (in frequency and/or span)**
  - **greater arithmetic intensity**
  - **greater SIMD-style shared-memory concurrency**
  - **built-in resilience ("algorithm-based fault tolerance" or ABFT) to arithmetic/memory faults or lost/delayed messages**
- **Programming models and runtimes may have to be stretched to accommodate**
- **Everything should be on the table for trades, beyond disciplinary thresholds ➔ "co-design"**

Bad news/good news (1)

- **One will have to explicitly control more ot the data motion**
  - **carries the highest energy cost in the exascale computational environment**
- **One finally will get the privilege of controlling the *vertical* data motion**
  - ***horizontal* data motion under control of users already**
  - **but *vertical* replication into caches and registers was (until recently with GPUs) mainly scheduled and laid out by hardware and runtime systems**
  - **TaihuLight *cache-free*, with user-controlled scratchpads**

Bad news/good news (2)

- **"Optimal" formulations and algorithms may lead to poorly proportioned computations for exascale hardware resource balances**
  - **today's "optimal" methods presume flops are expensive and memory and memory bandwidth are cheap**
- **Architecture may lure scientific and engineering users into more arithmetically intensive formulations than (mainly) PDEs**
  - **tomorrow's optimal methods will (by definition) evolve to conserve whatever is expensive**

Bad news/good news (3)

- Fully hardware-reliable executions may be regarded as too costly/synchronization-vulnerable
- Algorithmic-based fault tolerance (ABFT) will be cheaper than hardware and OS-mediated reliability
  - developers will partition their data and their program units into two sets
    - a small set that must be done reliably (with today's standards for memory checking and IEEE ECC)
    - a large set that can be done fast and unreliably, knowing the errors can be either detected, or their effects rigorously bounded
- Examples already in direct and iterative linear algebra
- Anticipated by Von Neumann, 1956 ("Synthesis of reliable organisms from unreliable components")

Bad news/good news (4)

- **Default use of (uniform) high precision in nodal bases on dense grids may decrease, to save storage and bandwidth**
  - **representation of a smooth function in a hierarchical basis or on sparse grids requires fewer bits than storing its nodal values, for equivalent accuracy**
  - **we will have to compute and communicate "deltas" between states rather than the full state quantities, as when double precision was once expensive (e.g., iterative correction in linear algebra)**
  - **a generalized "combining network" node or a smart memory controller may remember the last address, but also the last values, and forward just the deltas**
- **Equidistributing errors properly to minimize resource use will lead to innovative error analyses in numerical analysis**

Bad news/good news (5)

- **Fully deterministic algorithms may be regarded as too synchronization-vulnerable**
  - **rather than wait for missing data, we may predict it using various means and continue**
  - **we do this with increasing success in problems without models ("big data")**
  - **should be fruitful in problems coming from continuous models**
  - **"apply machine learning to the simulation machine"**
- **A rich numerical analysis of algorithms that make use of statistically inferred "missing" quantities may emerge**
  - **future sensitivity to poor predictions can often be estimated**
  - **numerical analysts will use statistics, signal processing, ML, etc.**

What will first "general purpose" exaflop/s machines look like?

- ***Hardware*: many potentially exciting paths beyond today's CMOS silicon-etched logic, but not commercially at scale within the decade**
- ***Software*: many ideas for general-purpose and domain-specific programming models beyond "MPI + X", but not penetrating the mainstream CS&E workforce for the next few years**
  - **"X" is CUDA, OpenMP, OpenACC, OpenCL, etc., or MPI, *itself***

Philosophy

- **Algorithms must adapt to span the gulf between aggressive applications and austere architectures**
  - **full employment program for computational scientists and engineers**
  - **see, e.g., recent postdoc announcements from**
    - **Berkeley (8) for Cori Project (Cray & Intel MIC)**
    - **Oak Ridge (8) for CORAL Project (IBM & NVIDIA NVLink)**
    - **IBM (10) for Data-Centric Systems initiative**

  for porting applications to emerging hybrid architectures

Required software

Model-related

- Geometric modelers
- Meshers
- Discretizers
- Partitioners
- Solvers / integrators
- Adaptivity systems
- Random no. generators
- Subgridscale physics
- Uncertainty quantification
- Dynamic load balancing
- Graphs and combinatorial algs.
- Compression

Development-related

- Configuration systems
- Source-to-source translators
- Compilers
- Simulators
- Messaging systems
- Debuggers
- Profilers

High-end computers come with little of this stuff. Most has to be contributed by the user community

Production-related

- Dynamic resource management
- Dynamic performance optimization
- Authenticators
- I/O systems
- Visualization systems
- Workflow controllers
- Frameworks
- Data miners
- Fault monitoring, reporting, and recovery

Optimal hierarchical algorithms

- **At large scale, one must start with algorithms with optimal asymptotic scaling, $O(N \log^p N)$**
- **Some optimal hierarchical algorithms**
  - **Fast Fourier Transform (1960's)**
  - **Multigrid (1970's)**
  - **Fast Multipole (1980's)**
  - **Sparse Grids (1990's)**
  - **H matrices (2000's)**

"With great computational power comes great algorithmic responsibility." – Longfei Gao

Recap of algorithmic agenda

- **New formulations with**
  - **greater arithmetic intensity (flops per byte moved into and out of registers and upper cache)**
    - **including assured accuracy with (adaptively) less floating-point precision**
  - **reduced synchronization and communication**
    - **less frequent *and/or* less global**
  - **greater SIMD-style thread concurrency for accelerators**
  - **algorithmic resilience to various types of faults**
- **Quantification of trades between limited resources**
- ***Plus* all of the exciting analytical agendas that exascale is meant to exploit**
  - **"post-forward" problems: optimization, data assimilation, parameter inversion, uncertainty quantification, etc.**

Algorithmic bottlenecks in sci & eng

- **Dominant consumers in applications that tie up major supercomputer centers are:**
  - **Linear algebra on dense symmetric/Hermitian matrices**
    - **generalized eigenproblems (Schroedinger) in chemistry/materials**
    - **reduced Hessians in optimization**
    - **covariance matrices in statistics**
  - **Poisson solves**
    - **highest order operator in many PDEs in fluid and solid mechanics, E&M, DFT, MD, etc.**
- **These are two of the major thrusts of the ECRC at KAUST**

A technical completion of this talk

- **See my lectures in the Argonne Training Program for Extreme Scale Computing (ATPESC 2013, 2014, 2015, …)**
  - **http://extremecomputingtraining.anl.gov/files/2015/08/Keyes.Algorithmic.pdf**
- **Youtube video of one-hour version also available at the ATPESC site**

New algorithmic infrastructure

Sample algorithmic "points of light" that accomplish one or more of these agendas

- DAG-based data flow for dense symmetric linear algebra
- GPU implementations of dense symmetric linear algebra
- Multicore implementations of sparse linear algebra
- Fast Multipole for Poisson solves
- Algebraic Fast Multipole for variable coefficient problems
- Nonlinear preconditioning for Newton's method
- New programming paradigms for PDE codes


جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology
Merci beaucoup
شكرا
david.keyes@kaust.edu.sa