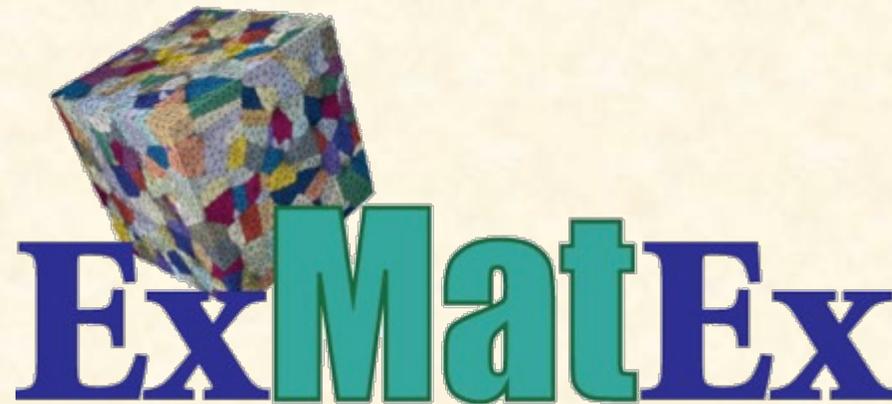


Exploiting Asynchrony for Materials in Extreme Environments: Programming Models and Runtime Requirements



Timothy C. Germann

Los Alamos National Laboratory

The Salishan Conference on High-Speed Computing

April 21-24, 2014

tcg@lanl.gov

Abstract

Within the Exascale Co-design Center for Materials in Extreme Environments (ExMatEx), we have initiated an early and deep collaboration between domain (computational materials) scientists, applied mathematicians, computer scientists, and hardware architects, in order to establish the relationships between algorithms, software stacks, and architectures needed to enable exascale-ready materials science application codes within the next decade. We anticipate that we will be able to exploit hierarchical, heterogeneous architectures to achieve more realistic large-scale simulations with adaptive physics refinement, and are using tractable application scale-bridging proxy application testbeds to assess new approaches to resilience, OS/runtime and execution models, and power management. The current scale-bridging strategies accumulate (or recompute) a distributed response database from fine-scale calculations (tasks), in a top-down rather than bottom-up multiscale approach. I will demonstrate this approach and our initial assessments, using simplified proxies to encapsulate the expected scale-bridging workload and workflow.

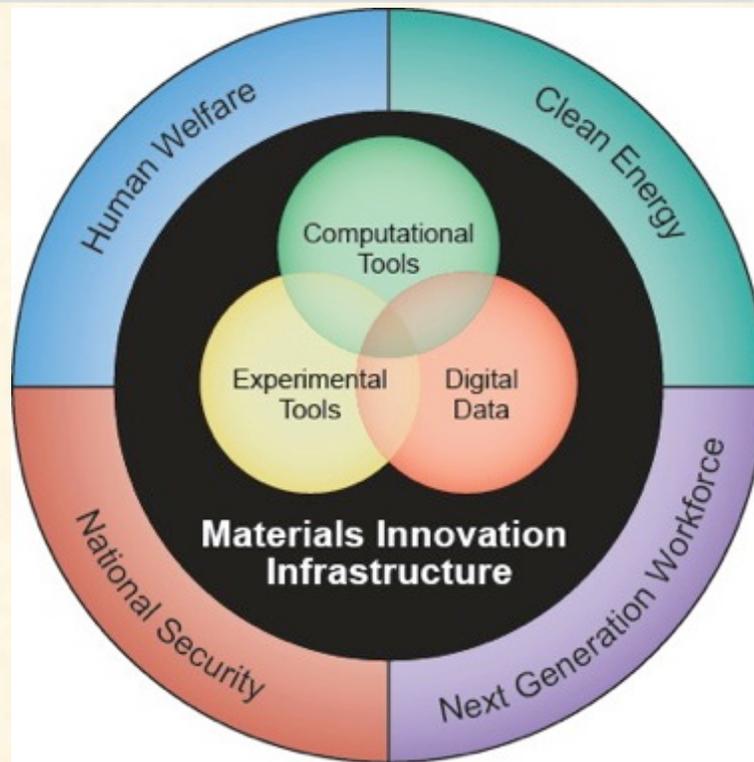
Modeling and simulation is playing an increasing role in materials design and certification

- High-strength, light-weight structural materials are required for products from cars and airplanes to gas, wind, and jet turbine blades

Atoms to airplanes
New structures technologies, developed across Boeing, are helping accelerate product development *By Bill Seil*
Boeing Frontiers (2010)

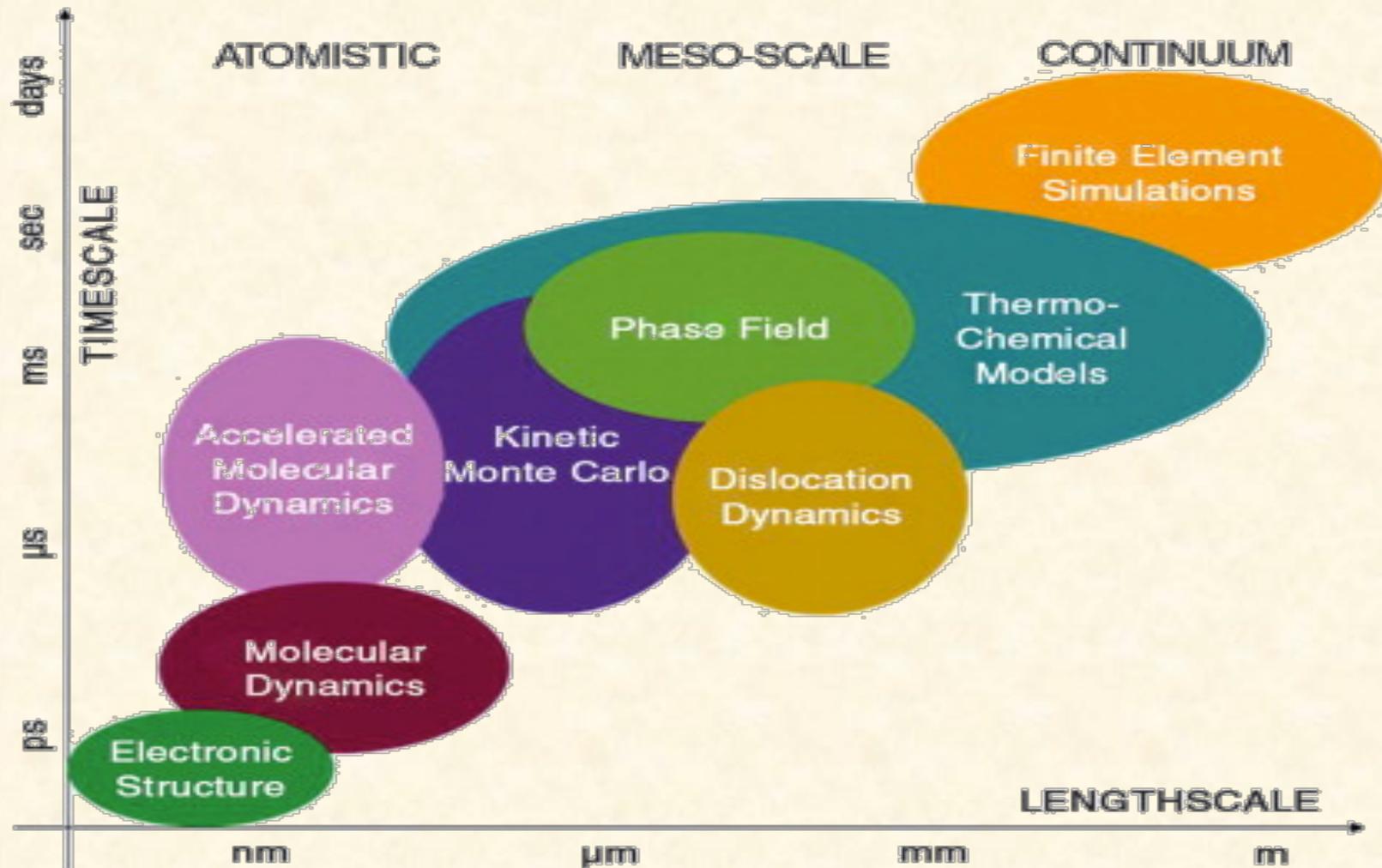


- Materials Genome Initiative



<http://www.whitehouse.gov/MGI>

Computational materials science involves a hierarchy of length and time scales



M. Stan, Materials Today **12**(11), 20 (2009)

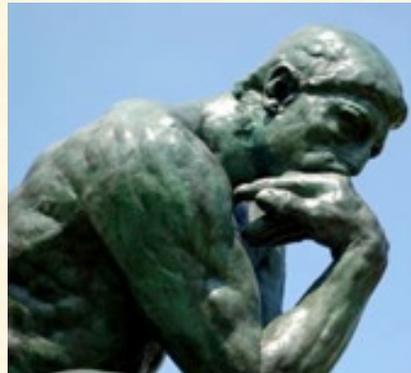
Traditional approach to subscale models: “sequential multiscale”

- Subscale models (e.g. interatomic potentials, equation of state and strength models) are developed from a combination of theory, experiment, and simulation.
 - *The specific combination depends on the developer, and may involve as much art as science.*

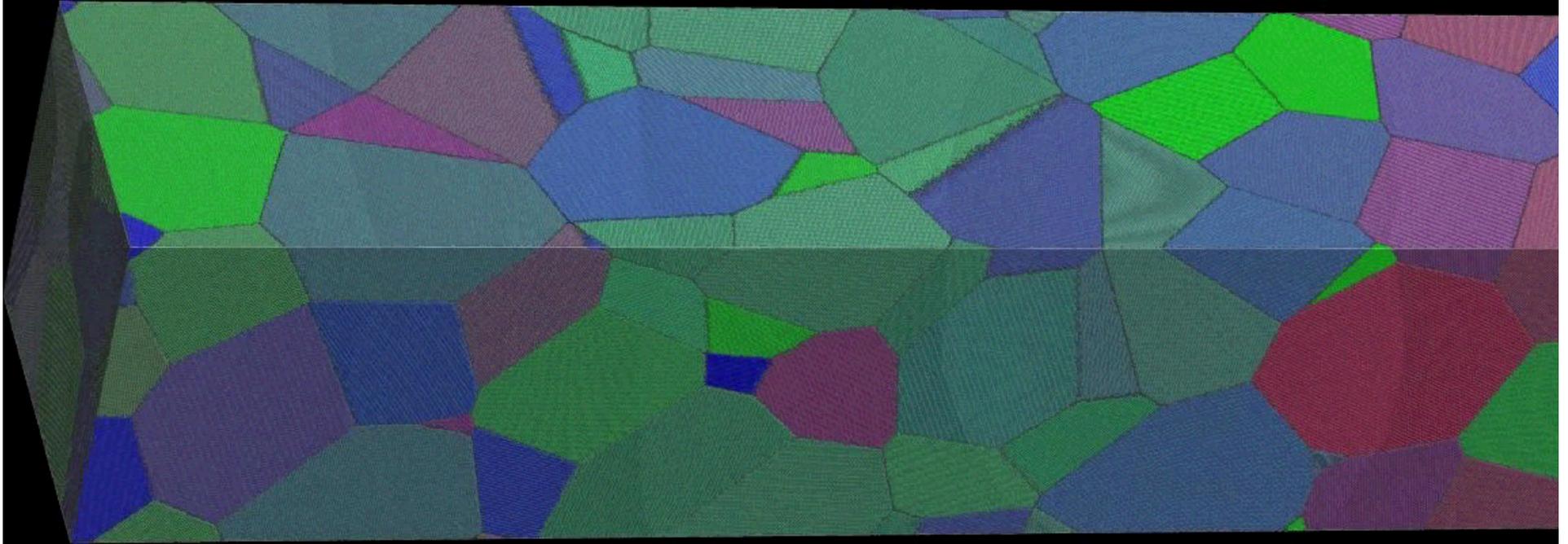
Calculations

Theory

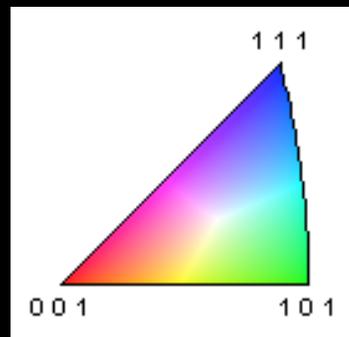
Experiment



Shock-induced plasticity and twinning of nanocrystal Ta



50nm grains
90x90x600 nm
~270 M atoms
 $u_p = 1.2$ km/s
 $P_H = 100$ GPa



R. Ravelo, T.C. Germann, et al,
Phys Rev B **88**, 134101 (2013)

Homepage | U.S. DOE Office of Science (SC)

http://science.energy.gov/

SC Home Organization Jobs Contact

U.S. DEPARTMENT OF **ENERGY** Office of Science

Search SC Website SC Site

Programs Laboratories User Facilities Universities Funding Opportunities Discovery & Innovation

AAAS.ORG | FEEDBACK | HELP | LIBRARIANS

NEWS SCIENCE JOURNALS CAREERS

Special: Communication in Science News Coverage: Science and the Shutoff

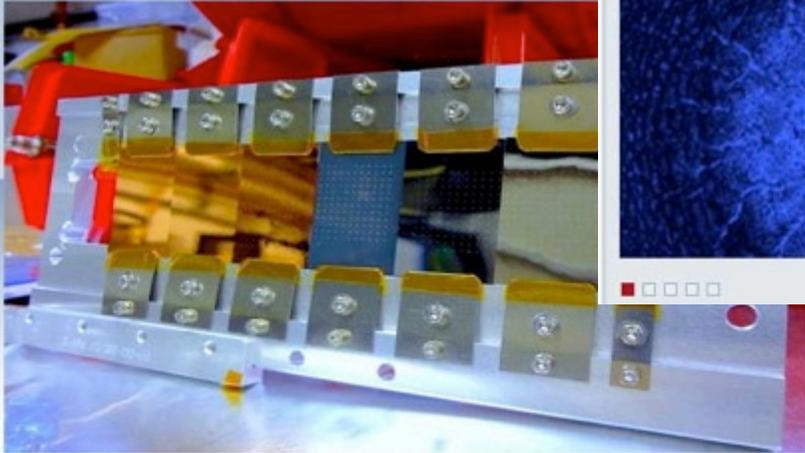
Stress Actually Makes You Stronger ... At Least Some of the Time

Researchers at SLAC test the mettle of metals, with potential benefits for all.

Read More »

Nov 2013

1 of 3



11 OCTOBER 2013
PHYSICS
From Elastic to Plastic

Using ultrafast x-ray diffraction, researchers were able to capture the response to shock in polycrystalline copper as it evolved from elastic to plastic.

PREVIOUS NEXT

Milathianaki et al,
Science 342, 220 (2013)

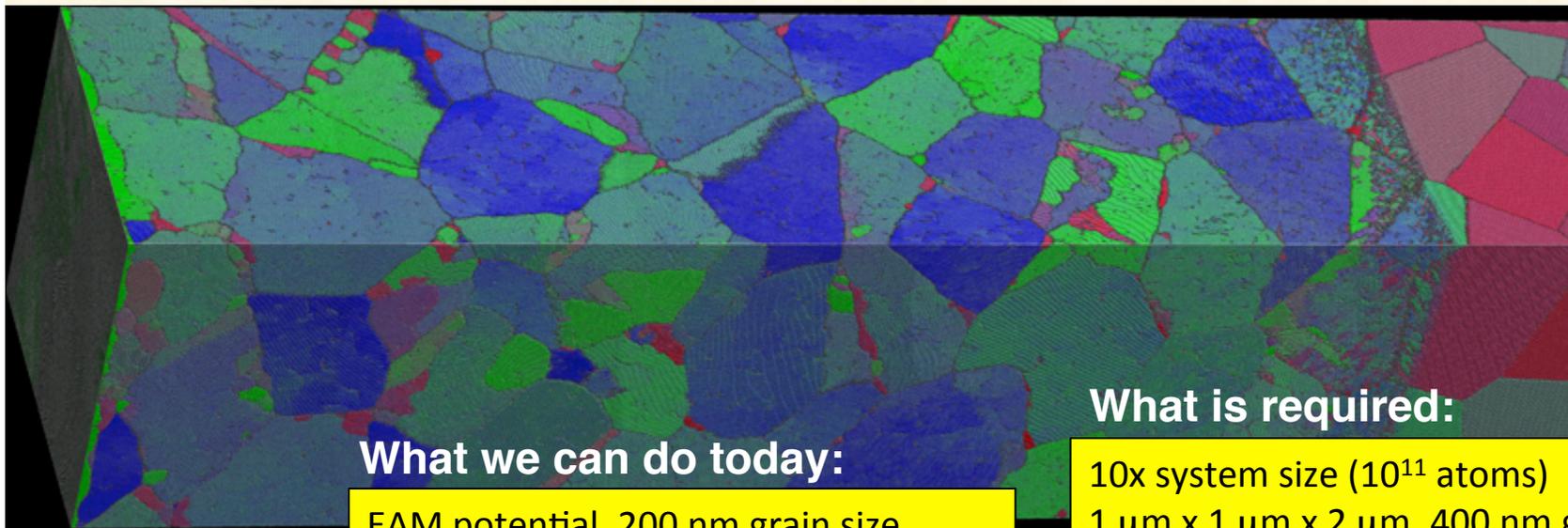
“First, the stress also serves as a **direct test of supercomputer simulations that model how metals behave. The better the data that goes in, the more reliable are the results that come out. That's important in trying to model the exact behavior of metals under stress, say the crash of a car or the impact of a bullet into armor.** And it's especially important for the Office of Science, since several of its labs are home to world-class supercomputers, which researchers are using for everything from simulating the 'subatomic soup' of the early universe to modeling air turbulence and thereby improving airplane performance.

Those better metal models could, in turn, lead to the design of even stronger and more durable materials. And those materials might come in handy for technologies that operate in extreme environments, such as shielding for satellites and space probes. They'll likely be useful in more everyday applications too.”

Exascale use case: competing dislocation, twinning, and/or phase transitions under shock loading

Direct non-equilibrium molecular dynamics simulation matching time and length scales of LCLS experiments:

- *~1-2 μm thick nanocrystalline samples (Cu, Ti, Fe, Ta), ~400 nm grain size*
- *Laser drive: 10-20 ps rise time, 150 ps duration*
- *50 fs duration X-ray “snapshot” interrogation pulses at 10 ps intervals*



NEMD simulation of shocked nc-Ta on Cielito (R. Ravelo, LANL/UTEP)

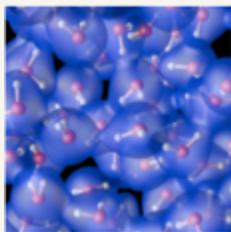
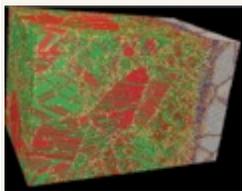
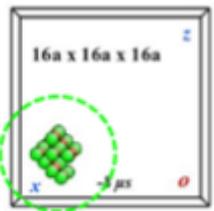
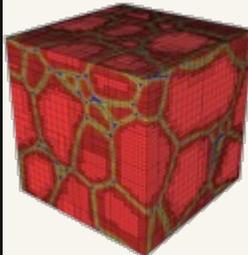
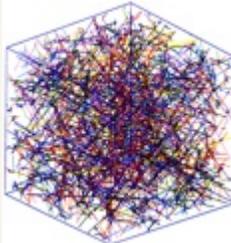
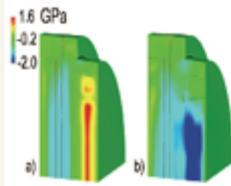
What we can do today:

EAM potential, 200 nm grain size
 10^{10} atoms ($0.5 \mu\text{m} \times 0.5 \mu\text{m} \times 1.5 \mu\text{m}$)
Simulation time: 4 nsec (10^6 steps)
Wall clock: 2 days on Mira ($\frac{1}{2}$ Sequoia)

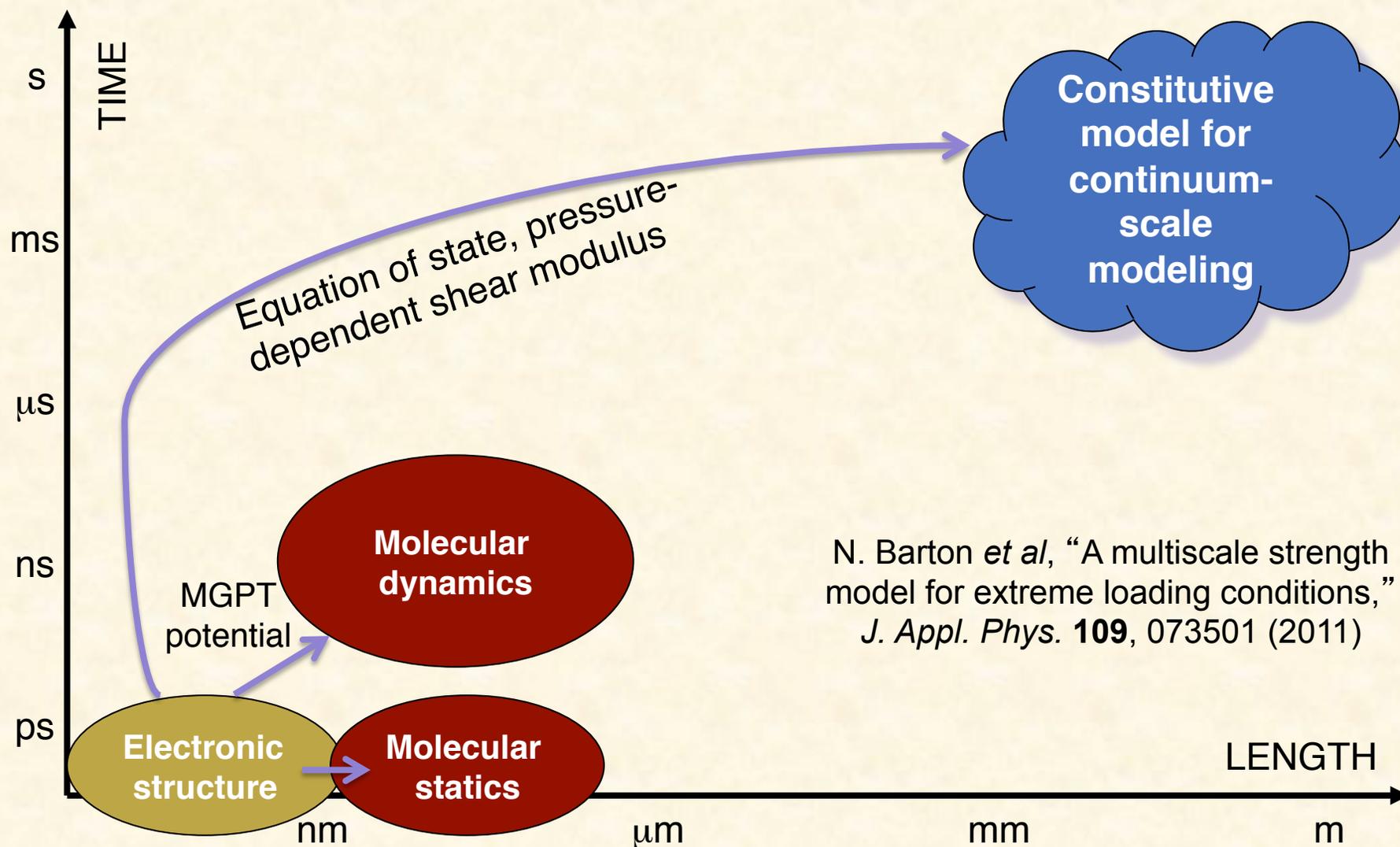
What is required:

10x system size (10^{11} atoms)
 $1 \mu\text{m} \times 1 \mu\text{m} \times 2 \mu\text{m}$, 400 nm grain size
More accurate MGPT potential: 100x
3 weeks on exascale system

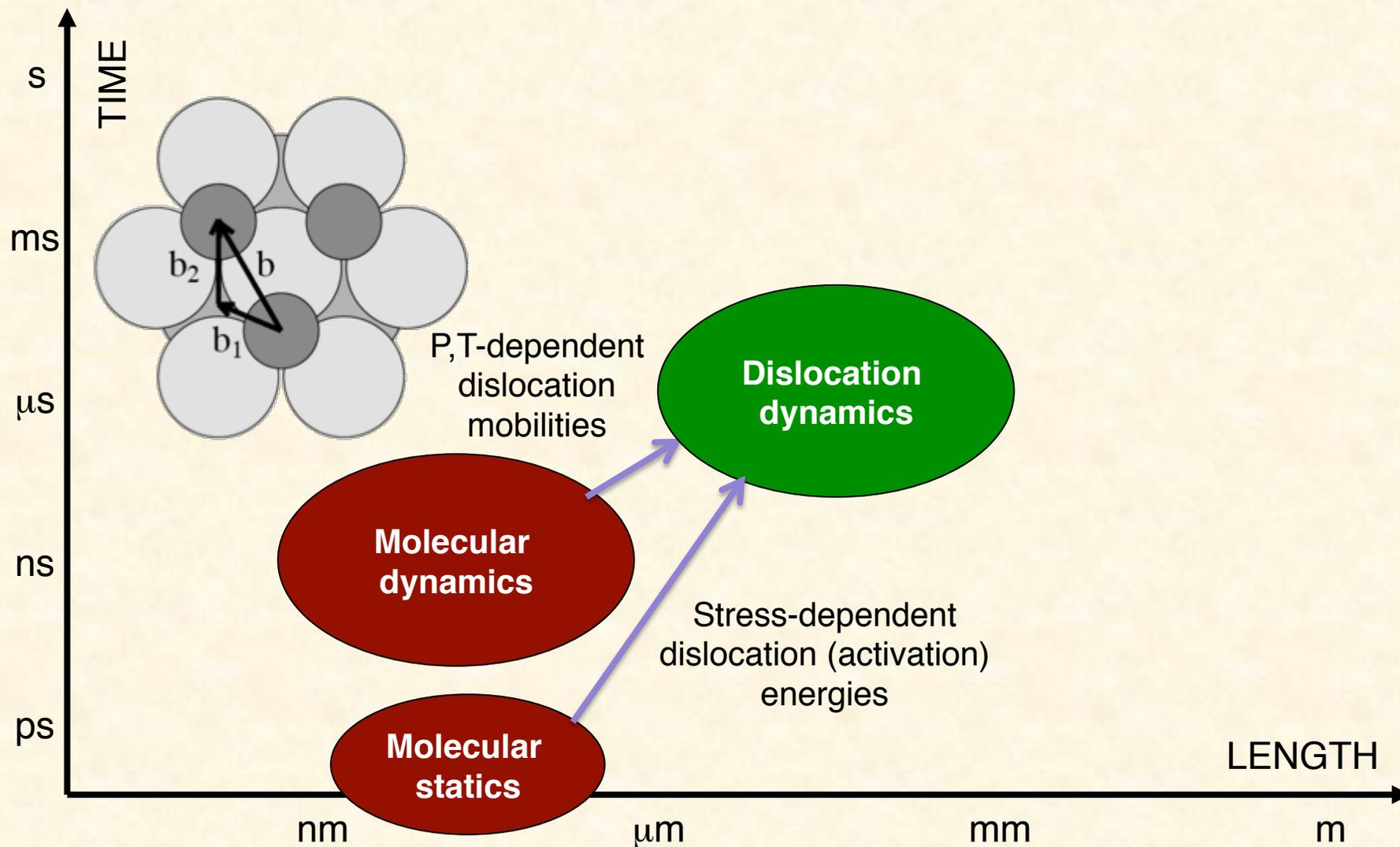
Seven pillars of computational materials science

Ab-initio	MD	Long-time	Phase Field	Dislocation	Crystal	Continuum
Inter-atomic forces, EOS	Defects and interfaces, nucleation	Defects and defect structures	Meso-scale multi-phase evolution	Meso-scale strength	Meso-scale material response	Macro-scale material response
						
Code: Qbox/ LATTE	Code: SPaSM/ ddcMD/CoMD	Code: SEAKMC	Code: AMPE/ CoGL	Code: ParaDis	Code: VP-FFT	Code: ALE3D/ LULESH
Motif: Particles and wavefunctions, plane wave DFT, ScaLAPACK, BLACS, and custom parallel 3D FFTs	Motif: Particles, explicit time integration, neighbor and linked lists, dynamic load balancing, parity error recovery, and <i>in situ</i> visualization	Motif: Particles and defects, explicit time integration, neighbor and linked lists, and <i>in situ</i> visualization	Motif: Regular and adaptive grids, implicit time integration, real-space and spectral methods, complex order parameter	Motif: “segments” Regular mesh, implicit time integration, fast multipole method	Motif: Regular grids, tensor arithmetic, meshless image processing, implicit time integration, 3D FFTs.	Motif: Regular and irregular grids, explicit and implicit time integration.
Prog. Model: MPI + CUBLAS/ CUDA	Prog. Model: MPI + Threads	Prog. Model: MPI + Threads	Prog. Model: MPI	Prog. Model: MPI	Prog. Model: MPI + Threads	Prog. Model: MPI + Threads

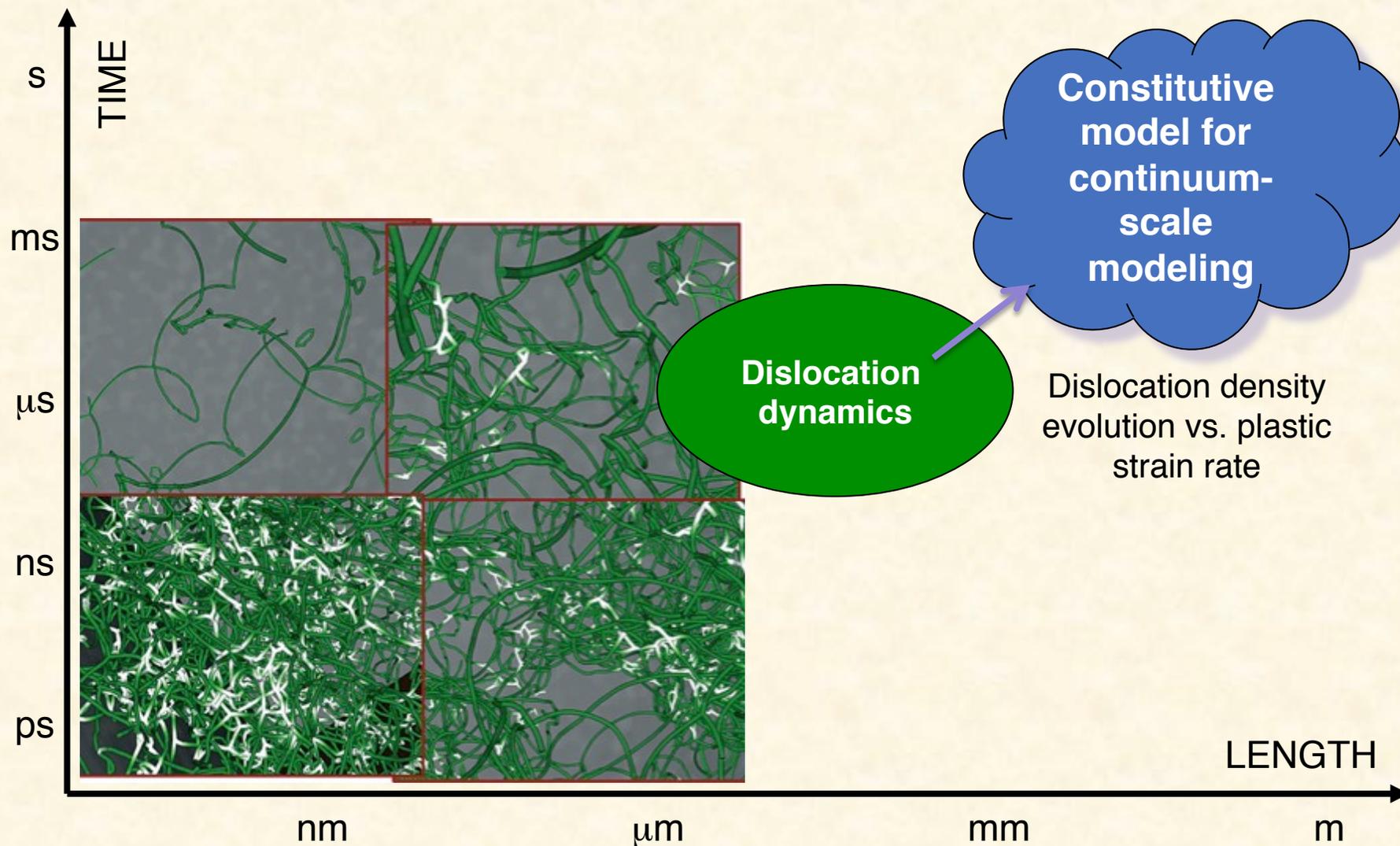
Example: “sequential” multiscale strength model



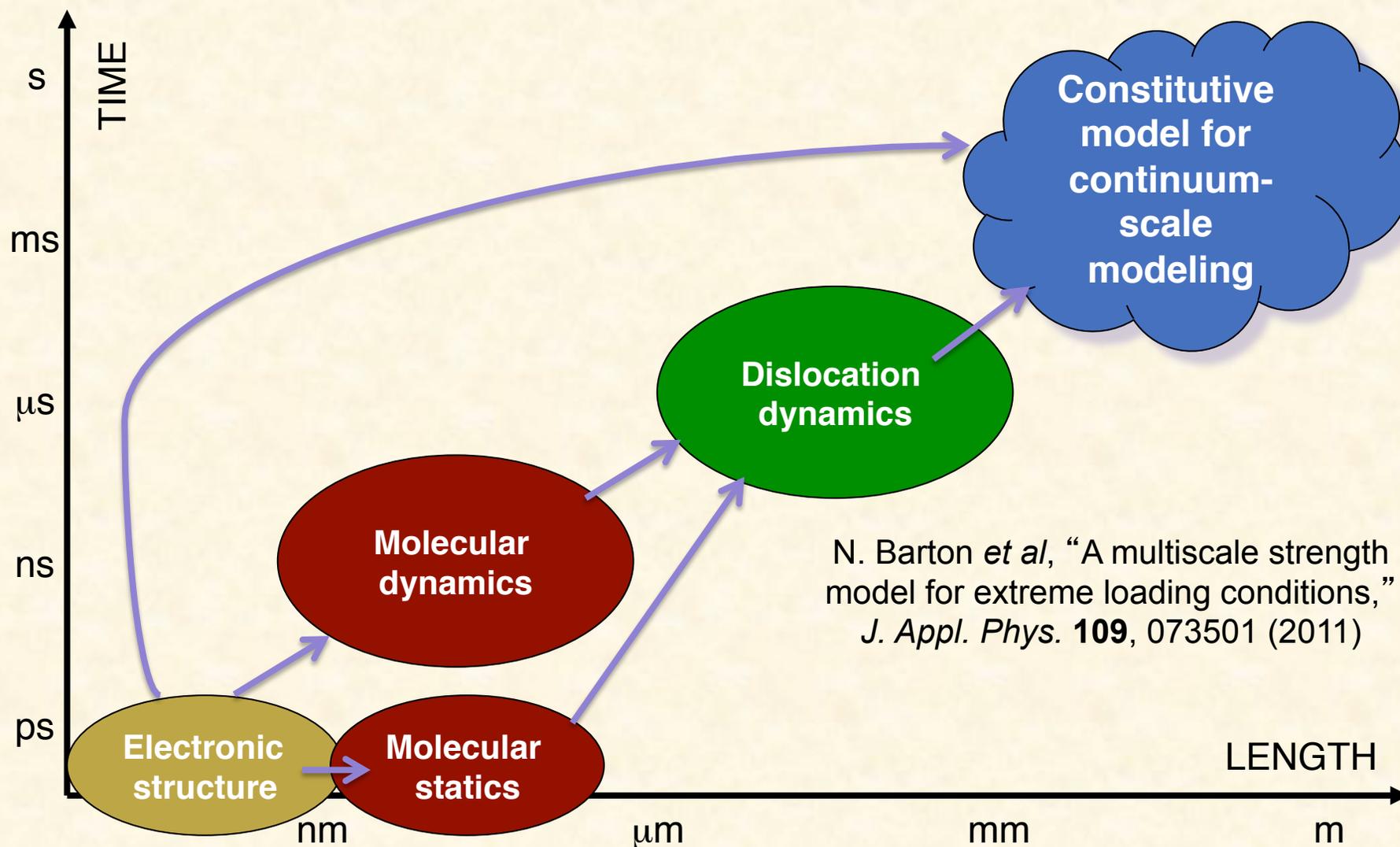
Example: “sequential” multiscale strength model



Example: “sequential” multiscale strength model



Example: “sequential” multiscale strength model

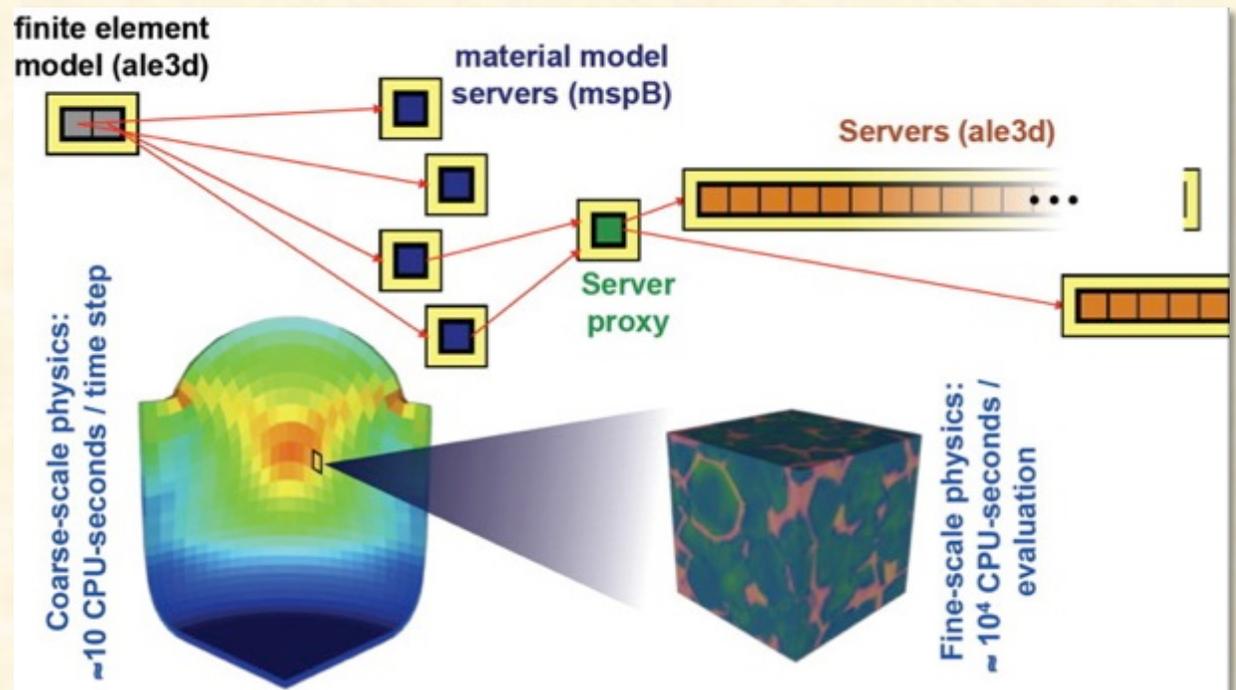


Challenges of a “sequential” multiscale approach

- Information is passed up a hierarchy of coupled length/time scales via a sequence of subscale models and parameters.
- This relies upon understanding how phenomena at shorter length/time scales control the behavior at longer length/time scales.
- Model complexity (and uncertainty) grows with each new physical mechanism.
 - *E.g. adding twinning and/or phase transformations to dislocation-based strength model*
 - *May need to account for coupling/competition between different physical processes*
 - *How does one include path (history) dependence (e.g., what is the strength of a material that has melted and then recrystallized?)*

Adaptive sampling techniques have been demonstrated under the LLNL “Petascale Initiative” LDRD.

- A coarse-scale model (e.g. FEM) calls a lower length-scale model (e.g. polycrystal plasticity) and stores the response obtained for a given microstructure, each time this model is interrogated.
- A microstructure-response database is thus populated.
- The fine-scale workload varies dramatically over the coarse-scale spatial and temporal domain.
- This requires dynamic workload balancing in a task parallel context.



N. R. Barton, J. Knap, A. Arsenlis, R. Becker, R. D. Hornung, and D. R. Jefferson.
Embedded polycrystal plasticity and adaptive sampling. *Int. J. Plast.* **24**, 242-266 (2008)

N. R. Barton *et al.* A call to arms for task parallelism in multi-scale materials modeling. *Int. J. Numer. Meth. Engng* **86**, 744–764 (2011)

Use Case: Shaped-charge jets, breakup and 3D effects (e.g. spinning) require crystal plasticity and anisotropy

What is required:

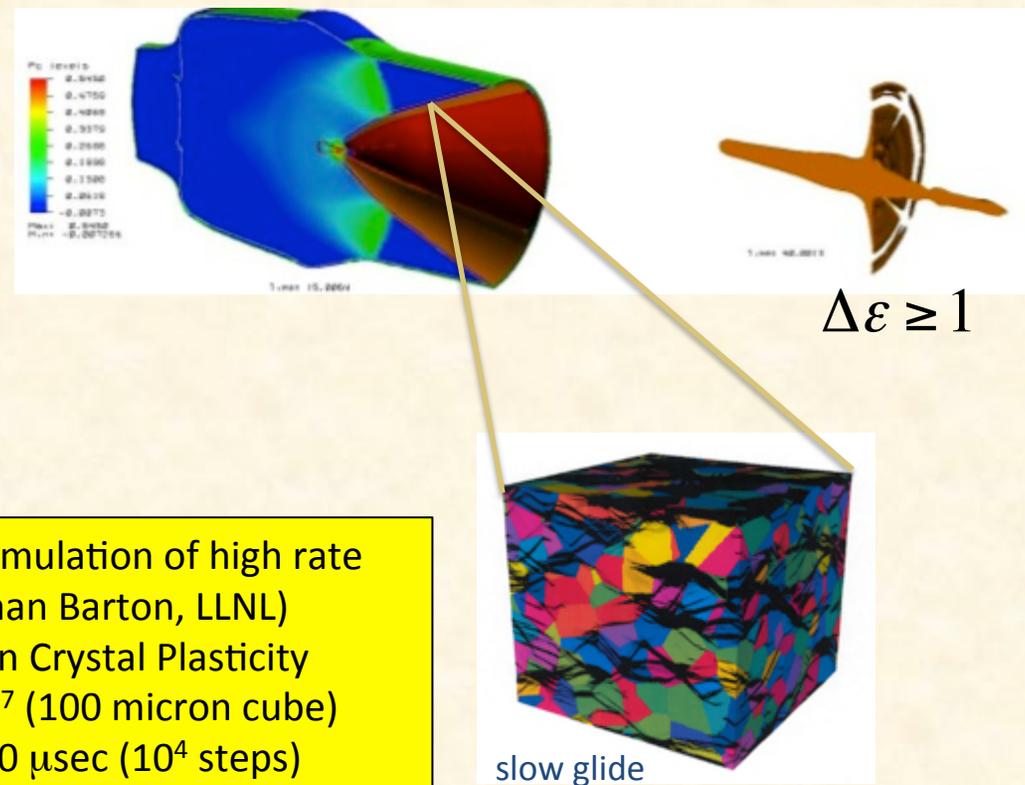
Resolution: 10^{12} zones (10 cm cube)
 Simulation time: 100 μ sec (10^5 steps)
 Strain rate: 10^6 /sec
 Strain: 1-3

Using Small Strain Crystal Plasticity Model:
 $\sim 10^4$ sec (~ 3 h) wall clock on 10^9 cores

Large Strain Crystal Plasticity Model: 10x

Twinning / Scale Bridging Model: 100x

ALE3D simulation of shaped-charge jet
 (Rose McCallen, LLNL)

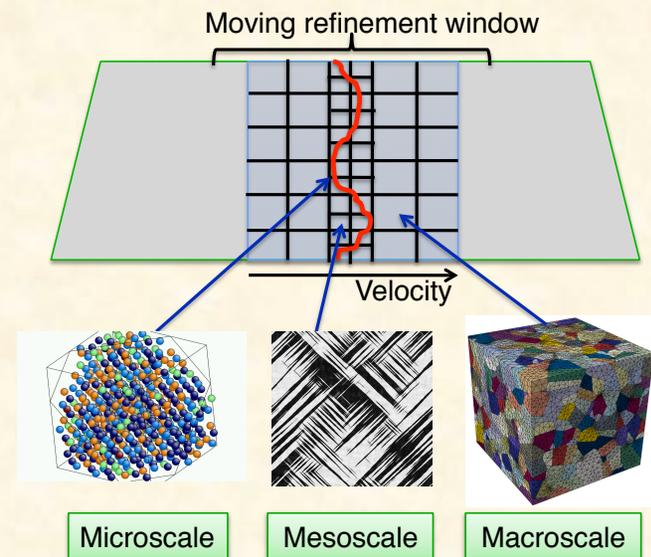


What we can do today:

Crystal plasticity simulation of high rate deformation (Nathan Barton, LLNL)
 Model: Small Strain Crystal Plasticity
 Number Zones: 10^7 (100 micron cube)
 Simulation time: 10 μ sec (10^4 steps)
 Strain rate: 10^6 /sec
 Strain: 0.15
 Wall Clock: 1 day on 1/10 Cielo

Exascale Co-Design Center for Materials in Extreme Environments

- Goal: to establish the relationships between algorithms, software stack, and architectures needed to enable exascale-ready materials science applications
- Two ultimate objectives:
 - *Identifying the requirements for the exascale ecosystem that are necessary to perform computational materials science simulations (both single- and multi-scale).*
 - *Demonstrating and delivering a prototype scale-bridging materials science application based upon adaptive physics refinement.*

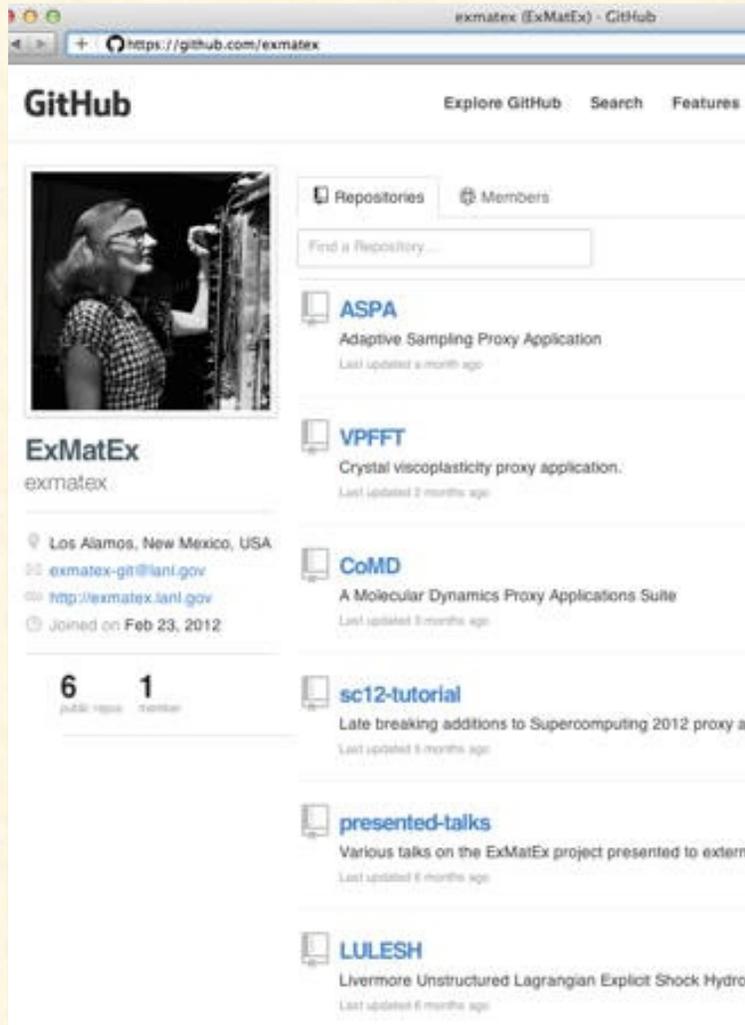


Proxy applications are central to co-design

- Proxy applications are a primary mechanism for collaboration between hardware architects, computer scientists, and domain scientists.
- Proxy apps representing the workflow have been an effective mechanism for:
 - *Identifying language/compiler weaknesses*
 - *Indicating bottlenecks that more complex computational workflows may have (vs. conventional benchmarks)*
 - *Providing tractable application testbeds for new approaches to resilience, OS/runtime/execution models, power management, ...*
 - *Evaluating alternative programming models, e.g. task-based execution models & runtimes*
- Open-source Mantevo suite
 - *Sandia National Laboratories*
+ AWE, LANL, LLNL, NVIDIA



Our focus during the first 18 months was establishing the initial suite of single-scale SPMD proxy apps.



github.com/exmatex

- Single-scale proxies primarily address node-level SPMD issues:
 - *Microscale: CoMD*
 - » Molecular dynamics; particle-based
 - *Mesoscale: VPFFT, CoGL*
 - » Crystal plasticity, phase field; regular Eulerian grids (Fourier- & real-space alternatives)
 - *Macroscale: LULESH*
 - » Shock hydro; unstructured Lagrangian mesh
- CoMD and LULESH are two of the small set (~6) of compact applications that several of the vendor FastForward teams have focused on as part of their projects.
- Several hackathons and deep dives have enhanced this collaboration.

Proxy apps are being used to identify critical features of programming models

The single-scale proxy apps developed in Year 1, primarily CoMD and LULESH, were used as the primary vehicle for the co-design process, notably several “hackathons” with vendor and X-stack partners.

From these activities, and exploration of various node and component-level programming models, several critical features were identified.

Namely, they need to enable the developer to:

- Express control of workflow beyond communicating serial processes
- Express information (e.g. data dependencies) for higher-level dynamic control of workflow
- Express fine grain concurrency
- Express data locality / data layout
- Express asynchrony
- Express heterogeneity and hierarchy

Present benchmarks/apps vs. the apps we need for the future

- Most (all?) current single-physics proxy apps (e.g. CoMD and LULESH) represent current petascale applications and their algorithms.
 - *These will likely represent a substantial piece of the workload on group/department-level petascale racks and capacity platforms*
 - *Although they can/have been implemented in Charm++, HPX, etc, the underlying algorithms weren't re-designed with an asynchronous task-based programming model in mind*
 - *Thus overdecomposition is primarily exploited to provide load balancing – less so for asynchrony (e.g. timestepping, operator splitting)*
 - *Some methods are better suited, e.g. multigrid, Monte Carlo, and various flavors of electronic structure which result in structured or unstructured sparse matrix algebra and graph representations (LANL DFTB-MD LDRD)*

Present benchmarks/apps vs. the apps we need for the future

How should we express future-looking, “born asynchronous task-parallel” algorithms?

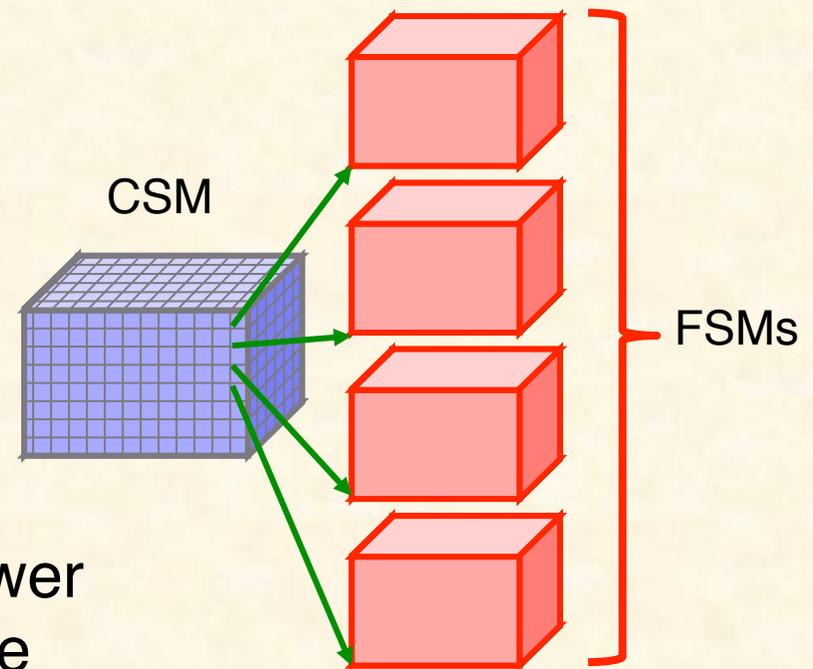
- a) Put all your eggs in a programming model/language/runtime basket, and hope that it survives, or at least the general approach it represents:
 - *MPI + X*
 - *PGAS*
 - *Event-driven task model*
- b) Write it in a domain-specific language, and hope that it’s a sufficiently broad domain to apply to multiple apps (amortizing the effort), while sufficiently narrow (and not at too high a level of abstraction) to enable performance.

Particularly well-suited are scale-bridging approaches, either moment-based acceleration (cf. Dana Knoll, Luis Chacon) or the class of adaptive physics refinement methods that ExMatEx is exploring.

Direct multi-scale embedding requires full utilization of exascale concurrency and locality

Heterogeneous, hierarchical task-based MPMD algorithms:

- Escape the traditional bulk synchronous SPMD paradigm
- Map naturally to anticipated heterogeneous, hierarchical architectures
- Leverage concurrency and heterogeneity at exascale while enabling novel data models, power management, and fault tolerance strategies

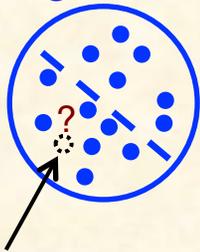


Kriging estimates are based on previously computed fine-scale responses.

Fine-scale responses accumulated in a database are interpolated (with error estimation) via a kriging algorithm.

- = fine scale evaluation
- = linear regression model

Kriging model 1



Sample point near existing model and satisfies tolerance:

- Just interpolate (saves fine-scale evaluation)

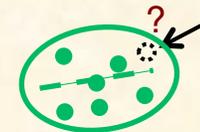
Kriging model 2



Sample point too far from existing models:

- Evaluate fine scale
- Create new model

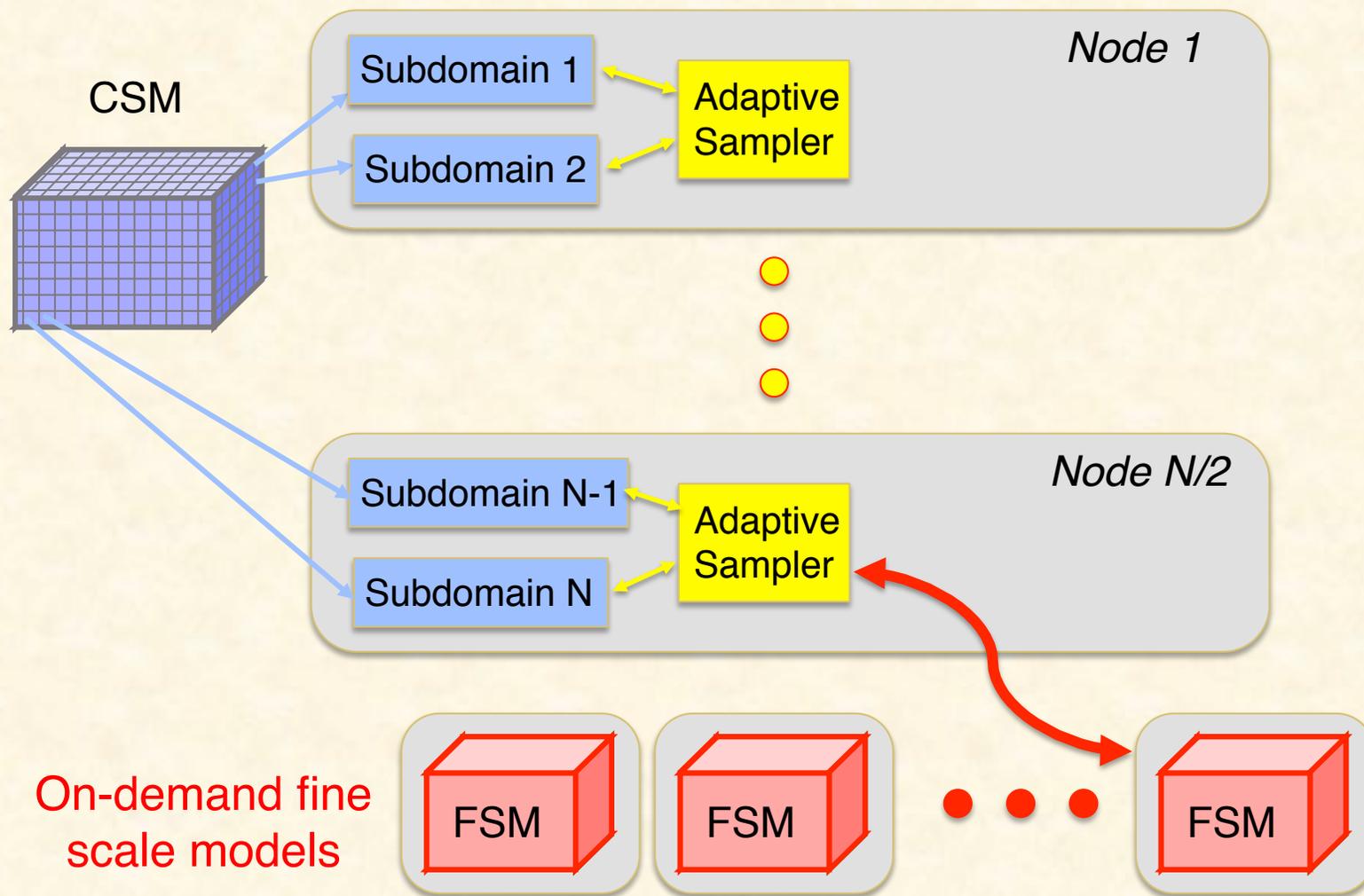
Kriging model 3



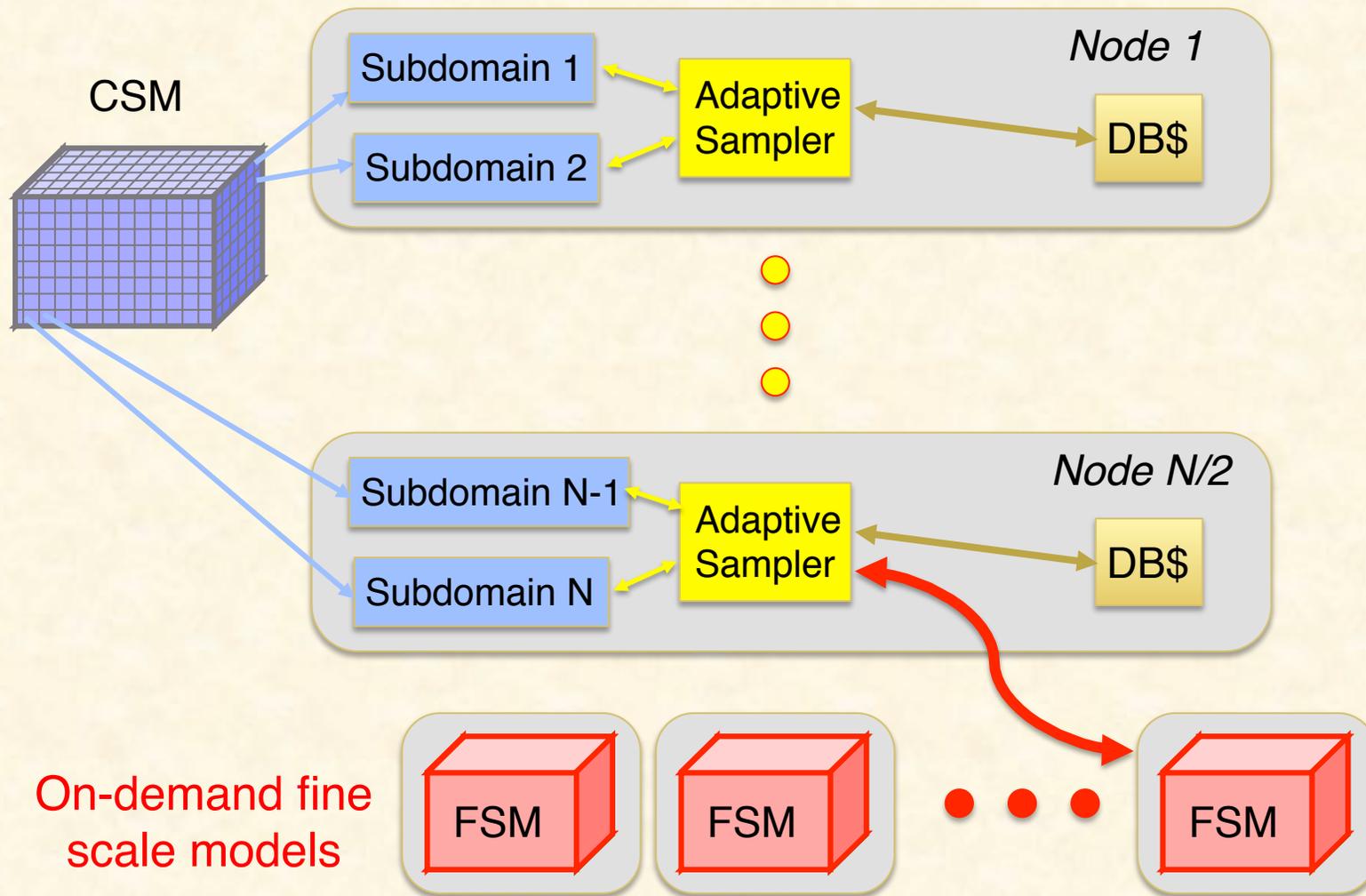
Sample point near existing model, but fails error tolerance:

- Evaluate fine scale
- Add to existing model

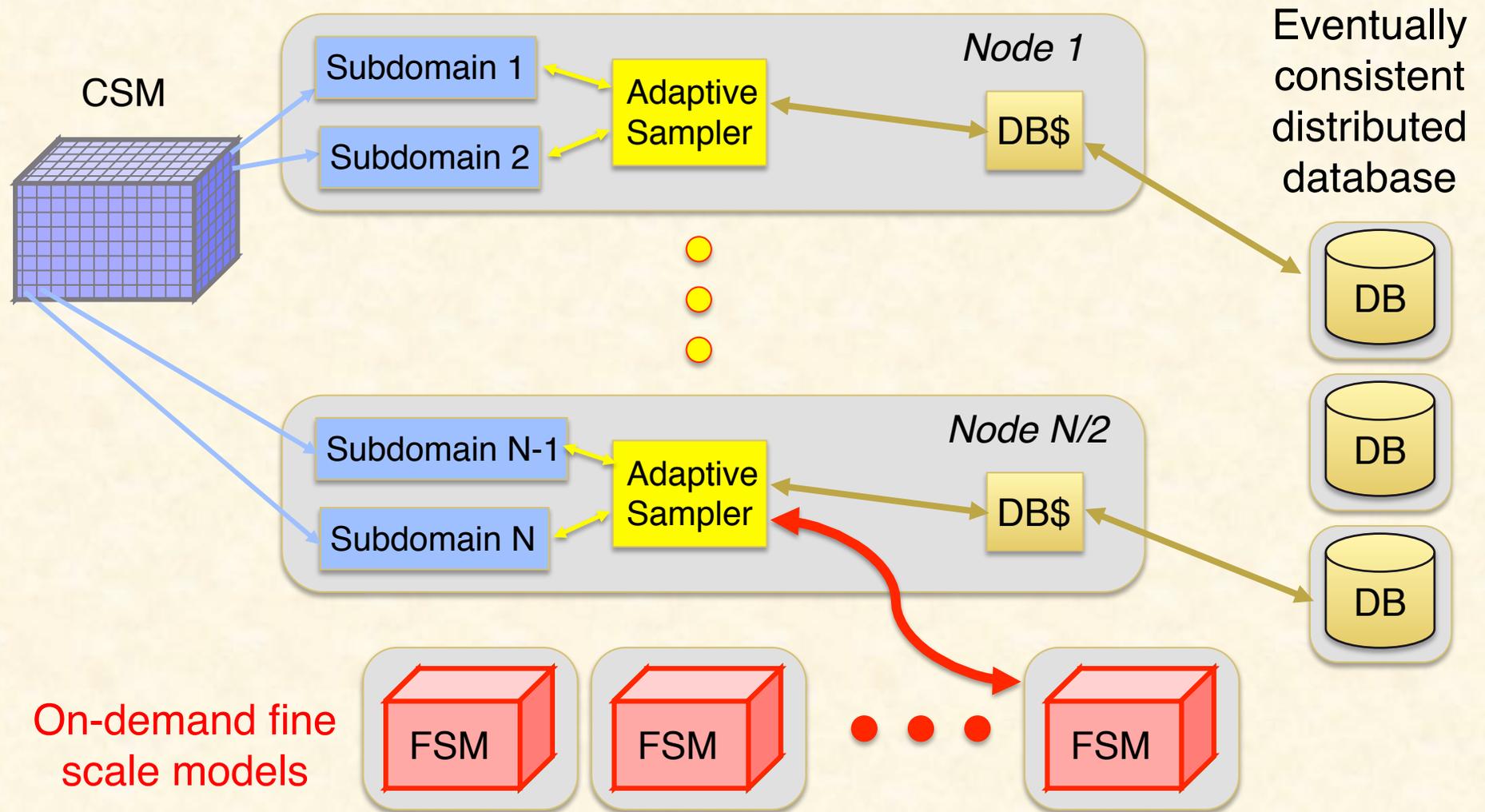
Emerging approach to subscale models: “concurrent multiscale”



Emerging approach to subscale models: “concurrent multiscale”



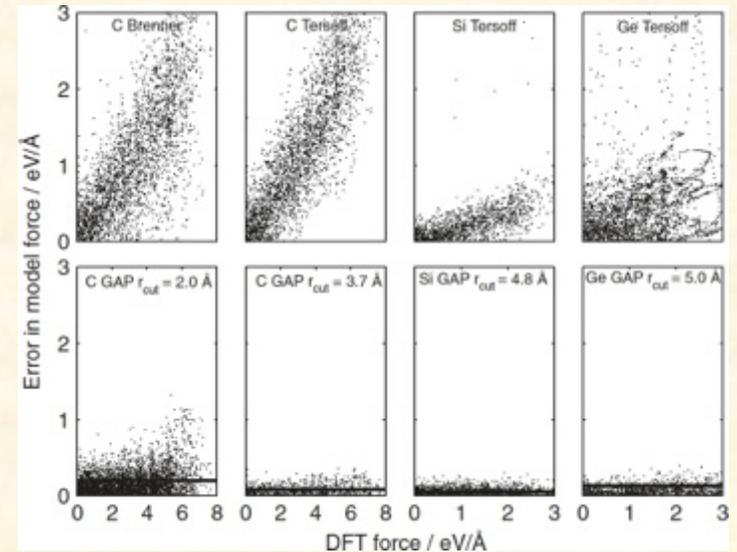
Emerging approach to subscale models: “concurrent multiscale”



Concurrent scale-bridging approaches are being pursued in other materials science contexts

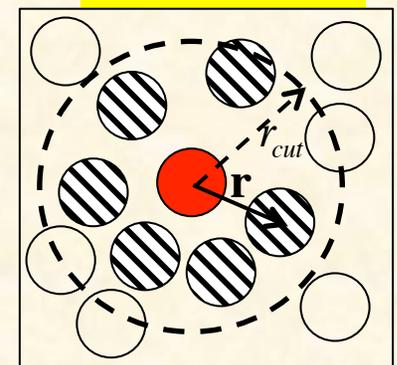
- Directly computing a potential surface from *ab initio* calculations

- *Distinct from ab initio molecular dynamics*
- *GAP: Gaussian approximation potentials*
 - » Bartók et al, PRL **104**, 136403 (2010)
- *SNAP: Spectral neighbor analysis potentials*
 - » Aidan Thompson et al, SNL-NM
- *Configurational database-driven dynamics*
 - » Jones and Shaughnessy, SNL-CA



- On-the-fly kinetic Monte Carlo
 - Henkelman and Jonsson, *J. Chem. Phys.* **115**, 9657 (2001)
- Self-learning kinetic Monte Carlo
 - Trushin et al, *Phys. Rev. B* **72**, 115401 (2005)
- Self-evolving atomistic kinetic Monte Carlo
 - Xu, Osetsky, and Stoller, *Phys. Rev. B* **84**, 132103 (2011)

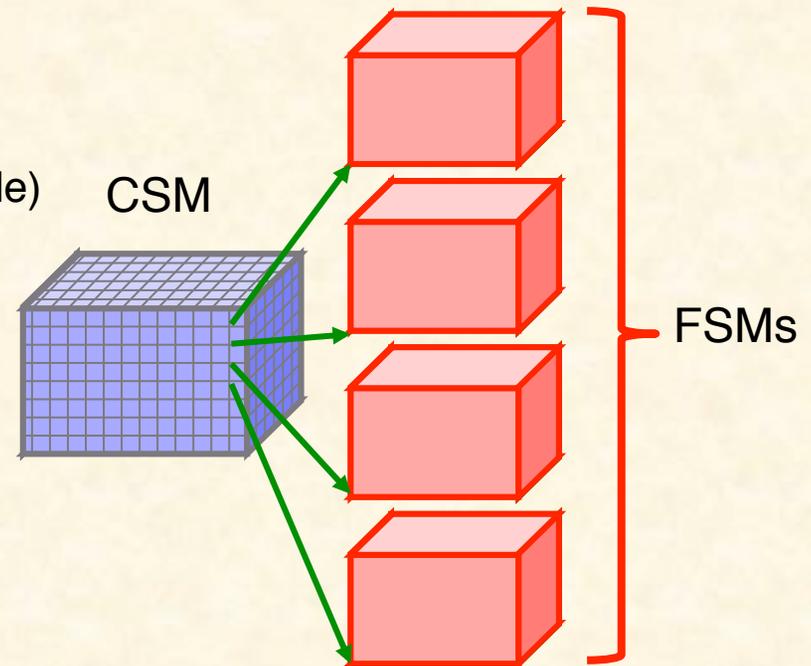
GAP/SNAP



Our work on scale-bridging has followed two complementary paths.

- “Top-down”

- We (Milo Dorr, LLNL) have developed an Adaptive Sampling Proxy App (ASPA) that represents the fine-scale query, database lookup, and kriging interpolation steps.
- LULESH (coarse-scale) and VPFPT (fine-scale) proxies are coupled via ASPA to study the workflow for our target application problems.
 - » “Speeds & feeds”
 - » What are the frequency, number, and duration of fine-scale calculations?
 - » What size and type of data are communicated between scales?

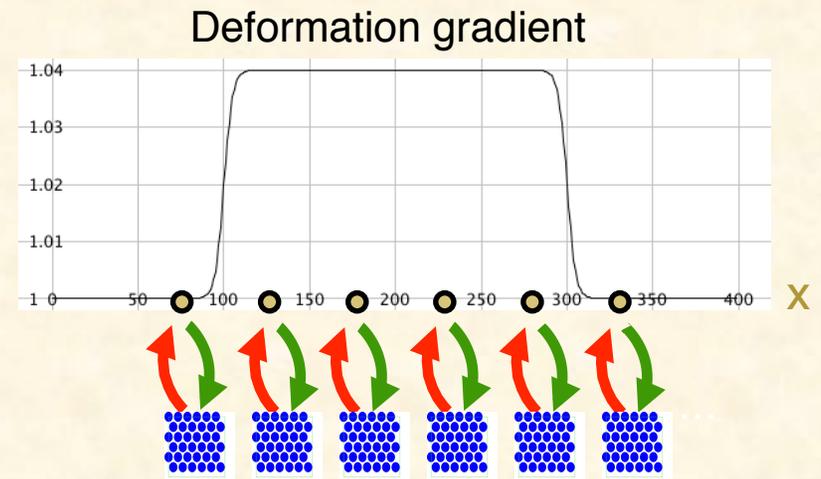


- “Bottom-up”

- We (Kip Barros et al, LANL) have developed a tractable scale-bridging proxy (CoHMM) that represents the basic task-based modeling approach we are targeting.
- It is being used to evaluate task-based OS/runtime requirements.

We are using the Heterogeneous Multiscale Method* as a scale-bridging prototype

- CoHMM presents the basic workflow requirements of a scale-bridging materials application.
- A full fine scale model (FSM, e.g. a crystal plasticity or molecular dynamics model) is run for every zone & time step of coarse scale model (CSM, e.g. an ALE code).
- It is being used to assess *basic* requirements for task-based runtime systems.
 - *The original HMM* is limited by its predictable, uniform workload pattern.*
 - *Adaptive coarsening provides a more dynamic and realistic workload.*

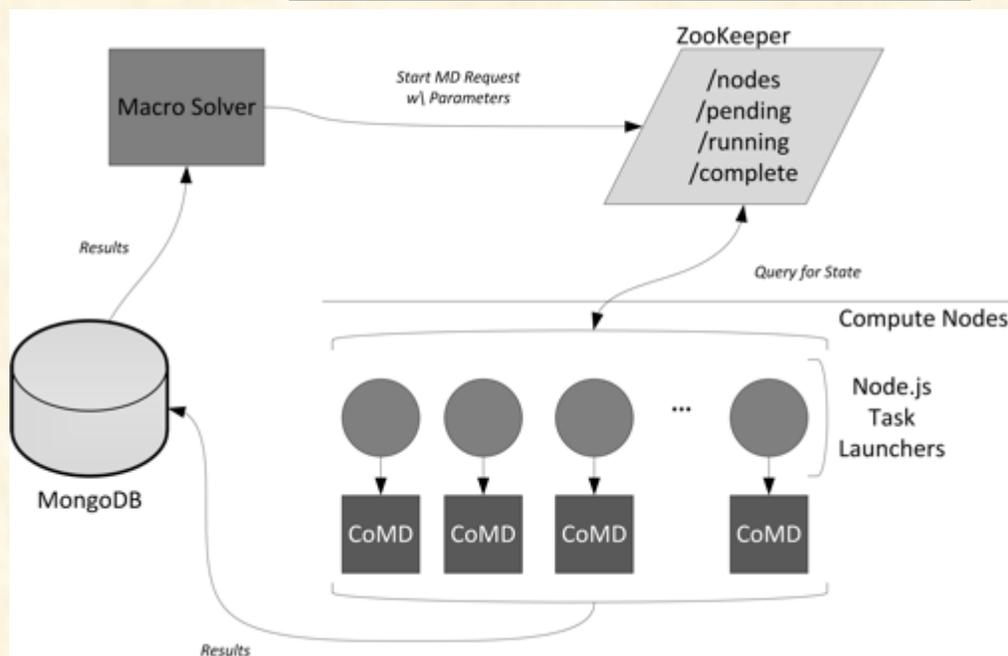


*Xiantao Li and Weinan E, “Multiscale modeling of the dynamics of solids at finite temperature,” *J. Mech. Phys. Solids* **53**, 1650–1685 (2005)

CoHMM: Early “Cloud” implementation

- Leverage technologies usually not seen in scientific simulation apps
- Apache ZooKeeper
 - *Distribute computation to pool of nodes*
- Node.js
 - *Launch CoMD computations*
 - *Stateless, run and exit*
- redis
 - *NoSQL database*
 - *Used to communicate results*
 - » CoMD stores results
 - » 1D HMM code reads results

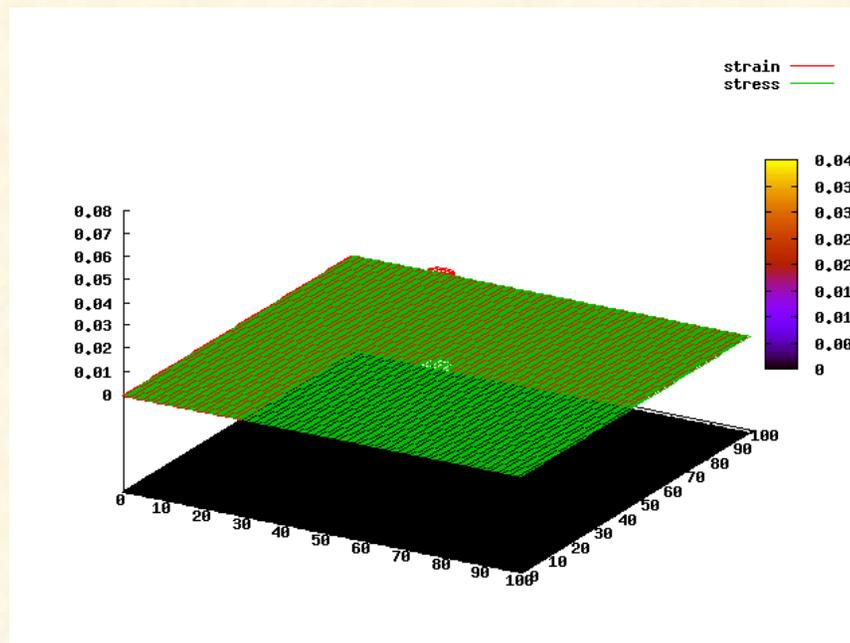
```
mitche11@centos:~/multiscale-demo
File Edit View Search Terminal Help
Task Completed: '/tasks/complete/t-0000000189' / '1,-1,188,/home/mitchell/multiscale-demo/CoMD/CoMD,-s 1.01'
# children = 443, First child = t-0000000188
taskInfo = 1,-1,187,/home/mitchell/multiscale-demo/CoMD/CoMD,-s 1.01
path = /tasks/running/t-0000000188
DEBUG: params var = 1,-1,187,/home/mitchell/multiscale-demo/CoMD/CoMD,-s 1.01
Launching with: >/home/mitchell/multiscale-demo/CoMD/CoMD -s 1.01
Task exited with code: 0
Task Completed: '/tasks/complete/t-0000000086' / '1,-1,54,/home/mitchell/multiscale-demo/CoMD/CoMD,-s 1.01'
# children = 442, First child = t-0000000085
taskInfo = 1,-1,55,/home/mitchell/multiscale-demo/CoMD/CoMD,-s 1.01
path = /tasks/running/t-0000000085
DEBUG: params var = 1,-1,55,/home/mitchell/multiscale-demo/CoMD/CoMD,-s 1.01
Launching with: >/home/mitchell/multiscale-demo/CoMD/CoMD -s 1.01
Task exited with code: 0
Task Completed: '/tasks/complete/t-0000000188' / '1,-1,187,/home/mitchell/multiscale-demo/CoMD/CoMD,-s 1.01'
# children = 441, First child = t-0000000080
taskInfo = 1,-1,60,/home/mitchell/multiscale-demo/CoMD/CoMD,-s 1.01
path = /tasks/running/t-0000000080
DEBUG: params var = 1,-1,60,/home/mitchell/multiscale-demo/CoMD/CoMD,-s 1.01
Launching with: >/home/mitchell/multiscale-demo/CoMD/CoMD -s 1.01
```



Chris Mitchell, LANL

CoHMM proxy: runtime system research

- Two-scale bridging
 - *Macroscale (HMM) with*
 - *Microscale (CoMD)*
- 2012 early efforts
 - *Scala*
 - *Erlang*
 - *“Cloud”*
- Significantly extended in 2013
 - *Co-Design Summer School*
 - *Problem made more mathematically accurate (for publication)*
 - *Evaluate multiple monolithic and service-based runtime systems*
 - *Multiple acceleration strategies added to test runtime features*
 - *Use in-memory database for acceleration and fault tolerance*



Co-design Summer School

- Los Alamos IS&T Co-Design Summer School (AI McPherson)
 - For recruiting and advertising LANL's co-design work
 - Small (6), multi-disciplinary team of students
 - 50/50 mix of US/foreign national citizens
 - Work on co-design problem
 - » 2011 & 2012: CoCoMANS LDRD
 - » 2013: ExMatEx
 - » 2014: ASC
 - Publish results
 - » Open source, reports, talks, posters
 - » Students @ SC, SIAM, nVidia GTC
 - Enhanced CoHMM Proxies
 - Explored multiple runtime approaches
 - » Industry
 - » Academic (including X-Stack)



2013 Summer School: Students

Name	School	Area
Robert Pavel	University of Delaware	CS
Axel Rivera	University of Utah	CS
Venmugil Elango	“The” Ohio State University	CS
Emmanuel Cieren	Laboratoire Bordelais de Recherche en Informatique	HPC
Dominic Roehm	Universität Stuttgart	Physics
Bertrand Rouet-Leduc	École Normale Supérieure / Cambridge	Physics

The 2014 ExMatEx co-design summer school will be held at Lawrence Livermore National Laboratory.

Contact: Jim Belak (belak@llnl.gov)

2013 Los Alamos IS&T Co-design Summer School

- Al McPherson and Tim Germann, co-mentors
- Focused on various task-based programming, execution, and runtime models
- Heterogeneous multiscale method with adaptive refinement and coarsening



Execution models / runtime systems which were evaluated:

- Charm++
- **CnC**
- DART
- Habanero
- HPX
- Pathos
- **Scioto**
- **Spark**
- SWARM(+GA)
- Swift/T

2013 Los Alamos IS&T Co-design Summer School

- Al McPherson and Tim Germann, co-mentors
- Focused on various task-based programming, execution, and runtime models
- Heterogeneous multiscale method with adaptive refinement and coarsening



For actual details, see Axel Rivera (U. Utah) at the Wed. evening Student Poster Session

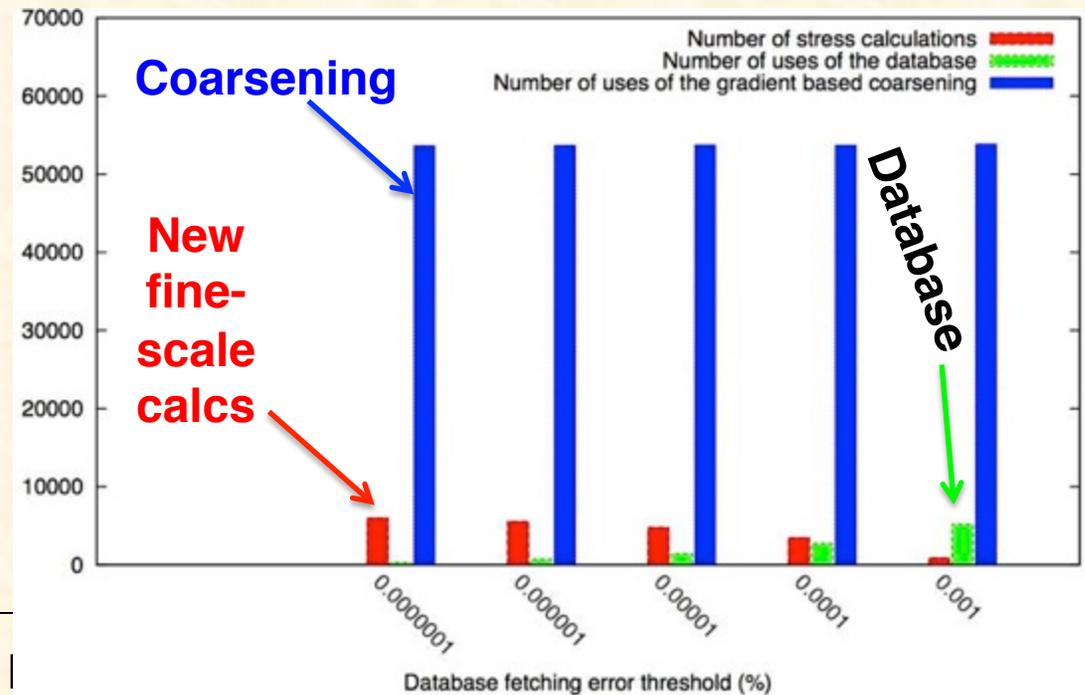
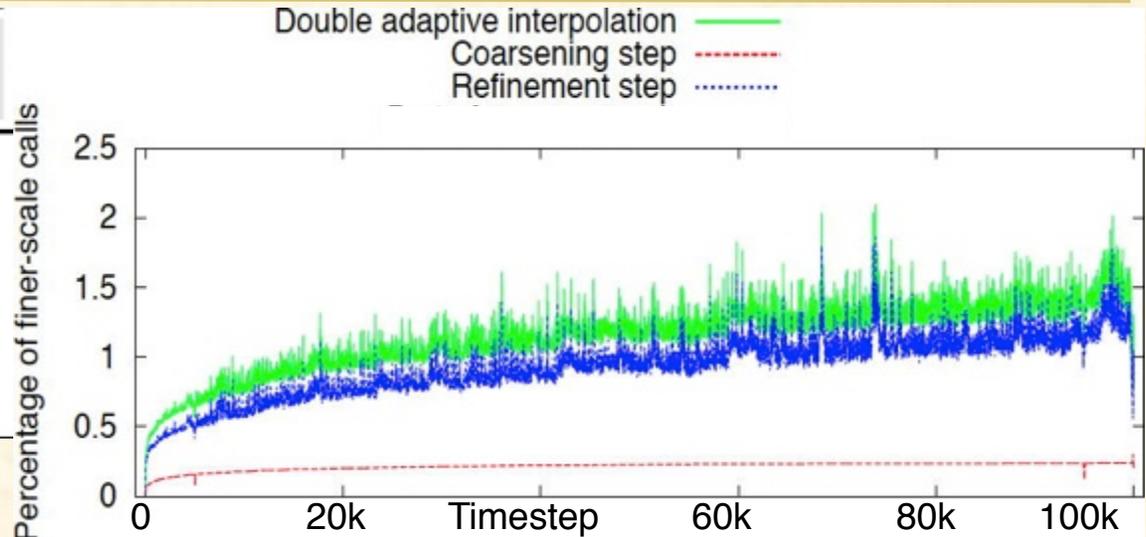
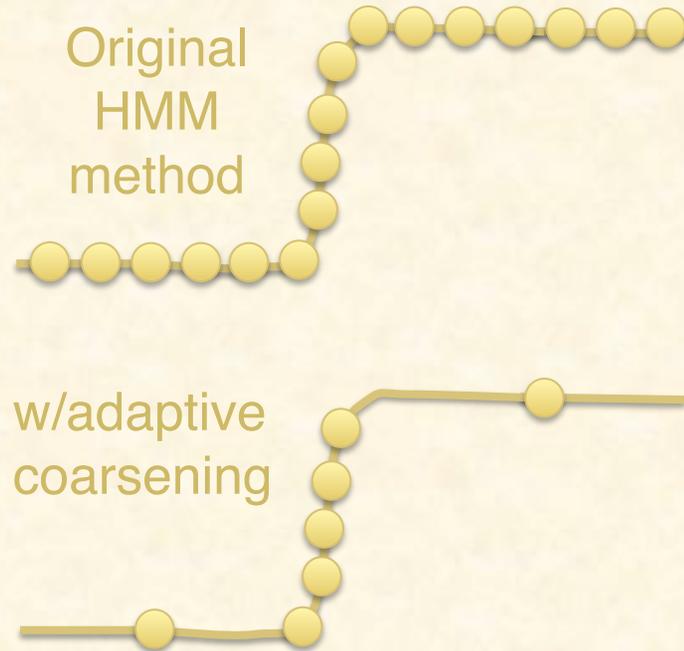
Adaptive coarsening of the macro-scale mesh



Spatial adaptive sampling in multiscale simulation

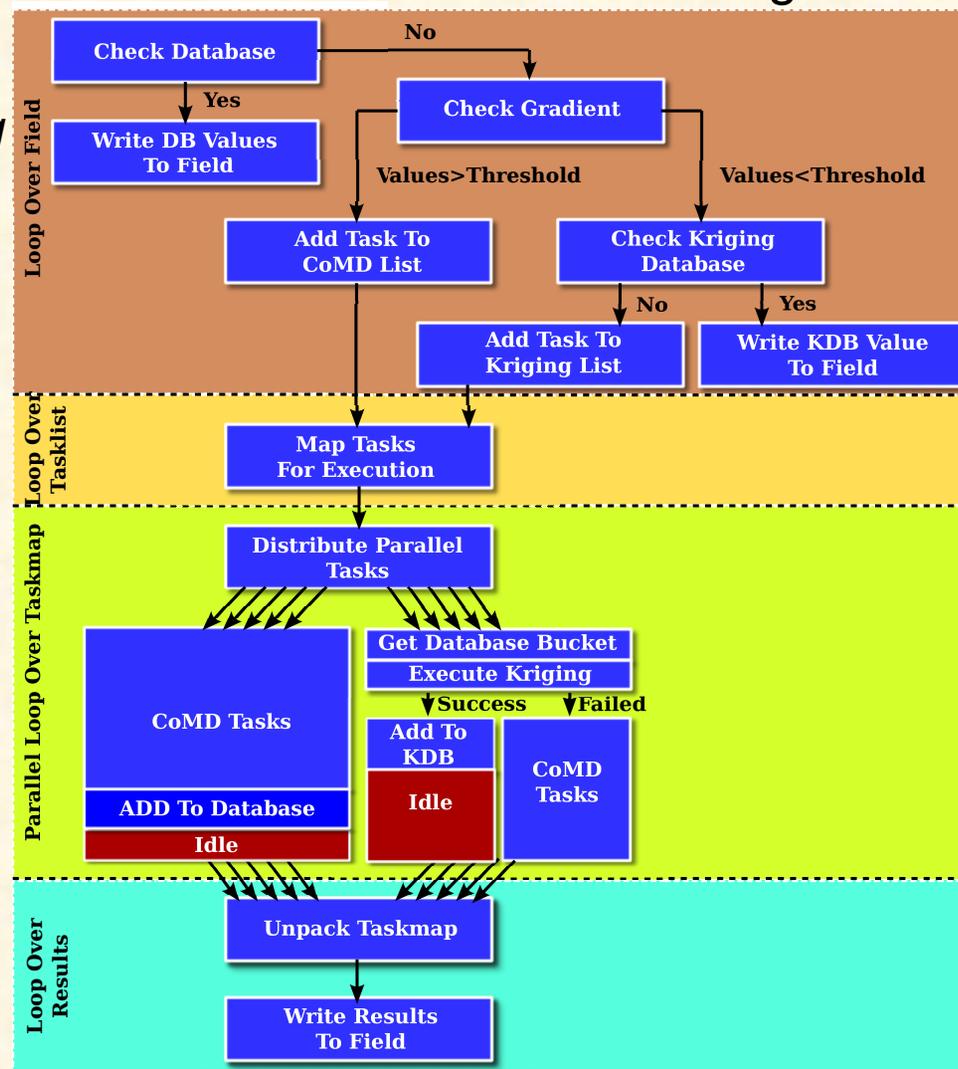
Bertrand Rouet-Leduc^{a,c}, Kipton Barros^{a,d}, Emmanuel Cieren^{a,d}, Venmugil Elango^{b,h},
 Christoph Junghans^a, Turab Lookman^a, Jamaludin Mohd-Yusof^b, Robert S. Pavel^{b,e},
 Axel Y. Rivera^{b,f}, Dominic Roehm^{a,g}, Allen L. McPherson^b, Timothy C. Germann^a

^a Theoretical Division, Los Alamos National Laboratory, Los Alamos, NM 87545, USA
^b Computer and Computational Sciences Division, Los Alamos National Laboratory, Los Alamos, NM 87545, USA
^c Department of Materials Science and Metallurgy, University of Cambridge, Cambridge CB3 0PS, UK
^d CEA, DAM, DIF, F-91257 Ardenne, France
^e Department of Electrical and Computer Engineering, University of Delaware, Newark, DE 19716, USA
^f School of Computing, University of Utah, Salt Lake City, UT 84112, USA
^g Institute for Computational Physics, Universitat Stuttgart, 70569 Stuttgart, Germany
^h Department of Computer Science and Engineering, The Ohio State University, Columbus, OH 43210, USA

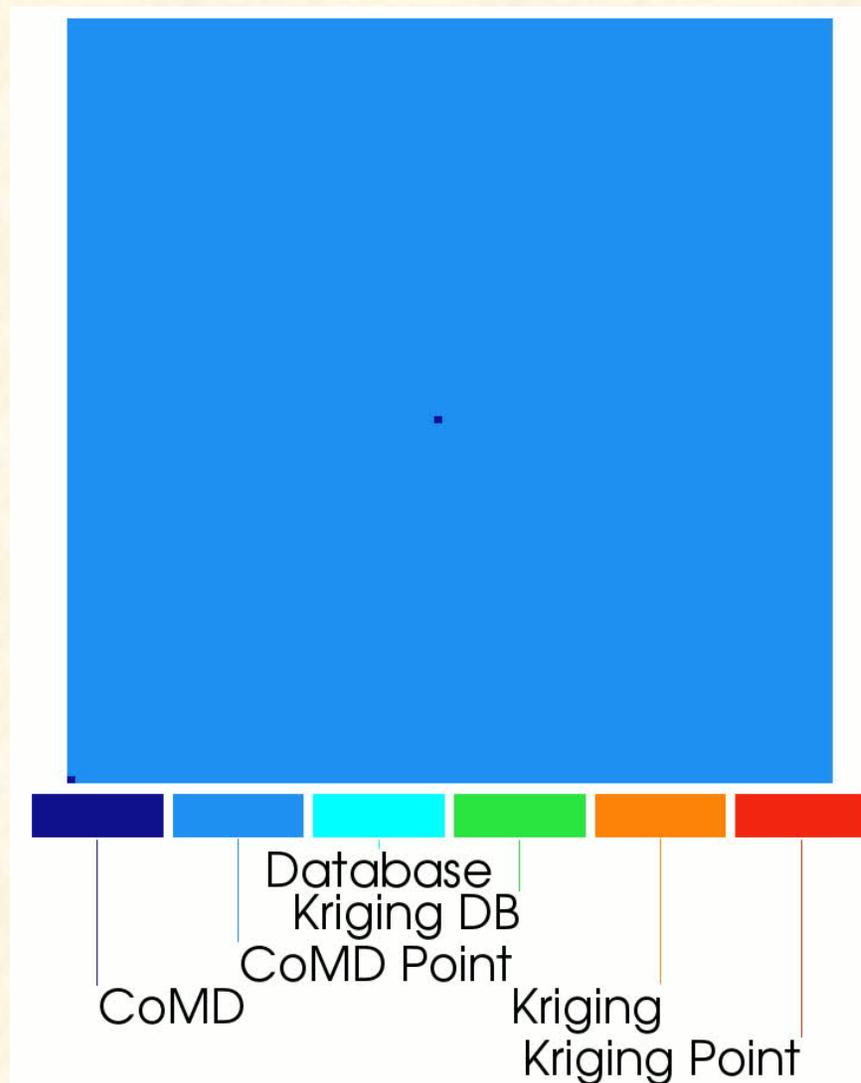
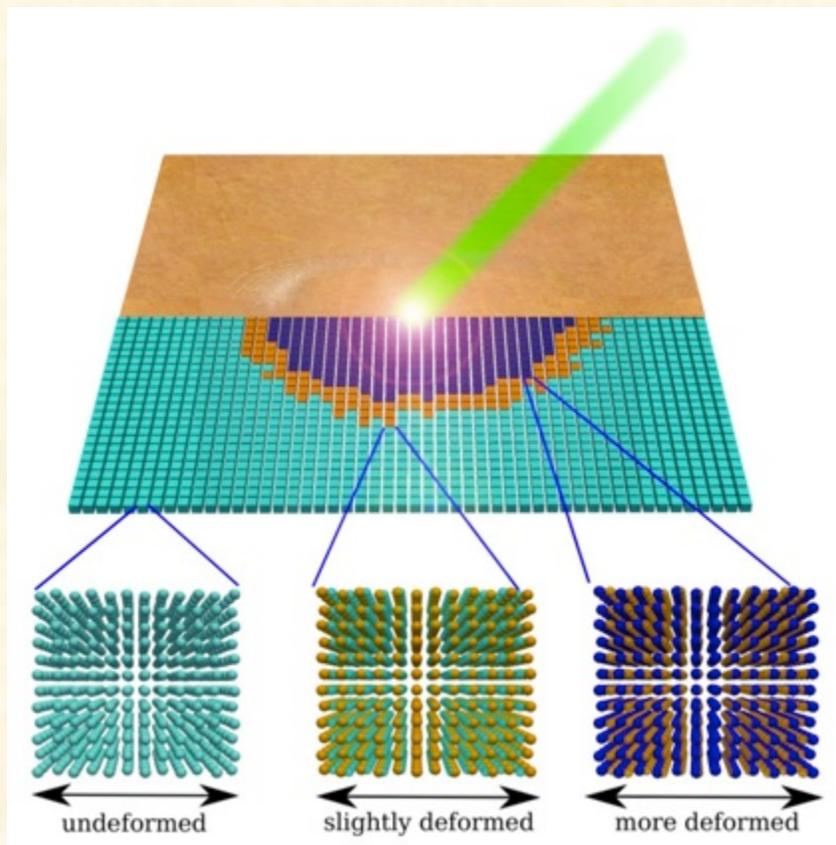


CoHMM acceleration strategies

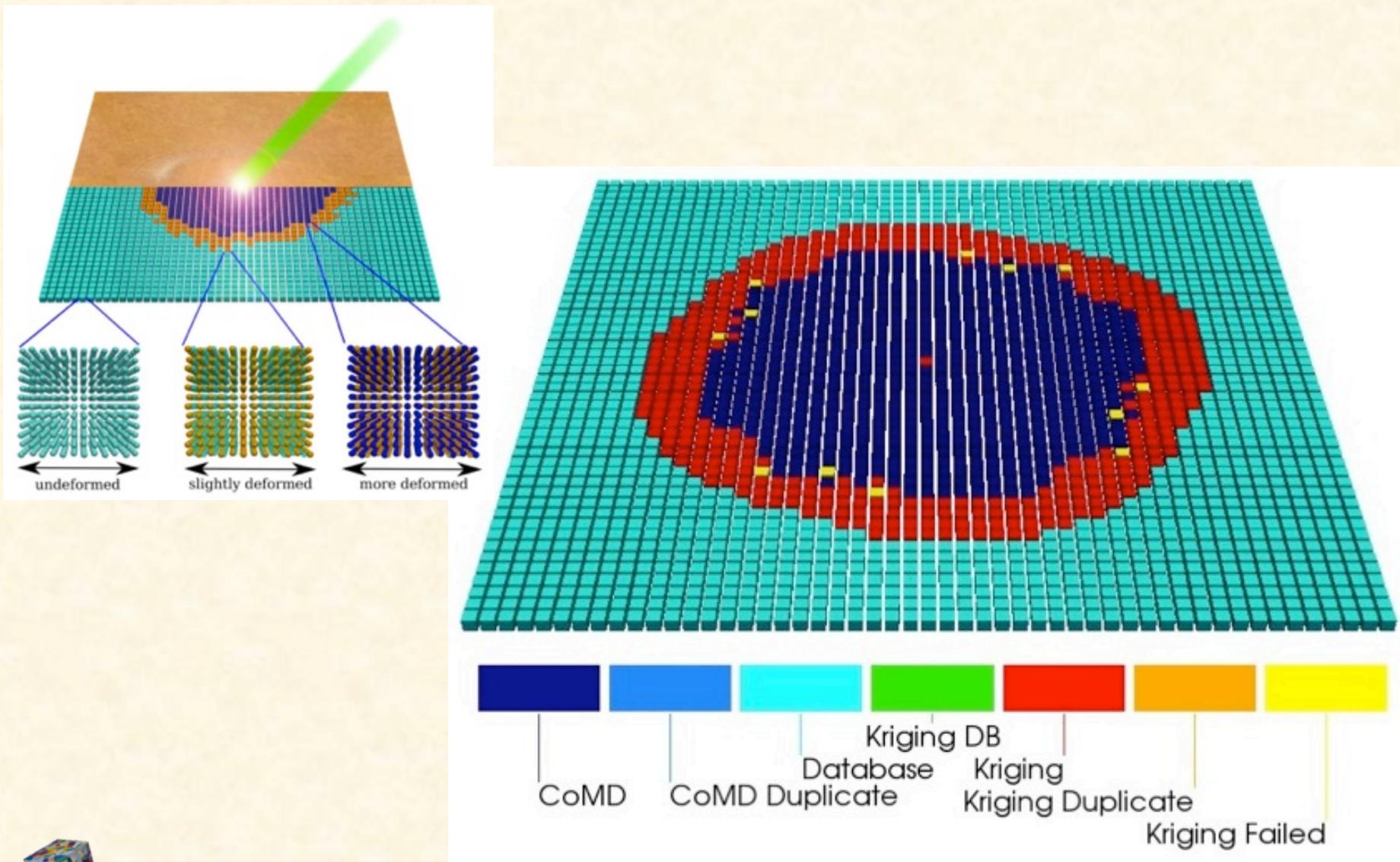
- Runtime functionality enables a combination of acceleration strategies
 - *Dynamic task launch*
 - *In-memory databases for caching*
- Gradient-based CoMD task launch
- CoMD task launch short circuit
 - *Schedule all CoMD tasks per timestep*
 - *Check parameters; launch those that differ*
- CoMD database
 - *Cache previously computed CoMD results*
- Kriging task launch short circuit
- Kriging database



CoHMM: visualization of accelerations



CoHMM: visualization of accelerations



CoHMM: summer school implementation

- The key CS idea is to use open source software for fundamental services: scheduling, messaging, caching
- Acceleration with adaptive task scheduling only where needed
- Acceleration by caching previously computed results
- Fault tolerance by caching particle positions for scheduling a restart at the node level—computation runs through a failure
- Scheduling
 - *Spark, Mesos (more “cloudy”)*
 - *CnC, Erlang, etc. (“steal” a service from a “monolithic” language)*
- Messaging
 - *MPI (single component)*
 - *ZeroMQ, RabbitMQ, NSQ*
- Caching
 - *NoSQL databases*
 - » *MongoDB, Cassandra, redis, RAMCloud*
 - *Potentially most useful*
 - » *Fault tolerance*
 - » *Material properties, EOS (service!), etc.*
 - » *Avoid redundant computation*
 - » *Communication*

CoHMM: Summer School Analysis

- CoMD 1.1 performance
- HMM (macrosolver performance, parallelization)
- Runtime and language overhead
 - *Spawning*
 - *Load balancing*
 - *Communication*
- Database overhead and performance
- Solution accuracy
 - *Gradient-based and Kriging tradeoffs*
- Fairly expensive (~seconds) CoMD task, so overheads are low for:
 - *Scheduling of CoMD runs*
 - *Read/write performance of in-memory database*
- Service provided by runtime systems made many, varied implementations possible in short amount of time (10 weeks)
- Must evaluate runtime system in context of current and near-term machines
 - *Will not have permissions to drastically change operations*
 - *Must run within “user space” and static scheduler*

We used the CoHMM proxy app to perform an initial evaluation of runtime system requirements for our scale-bridging workload.

System	Dimension	Adaptive	Database	Fault Tolerant	Status
HPX	Bugs and lack of documentation. Triage it away.				Eval. only
Scioto	1D, 2D	AMR, Kriging	redis	No	OK
Pathos	1D	Yes	Not tested	Process	OK
Intel CnC	2D	No	No	No	OK
Charm++	Synthetic benchmarks only. Evaluate load-balance.				Eval. only
Spark	1D, 2D	AMR, Kriging	redis	CoMD atom	OK
Mesos	Evaluated favorably. Installation issues.				Eval. only
Swift	1D	No	No	Process	CoMD 1.0
Erlang	1D	No	No	Process	CoMD 1.0
Scala	1D	No	No	No	Simple MD
“Cloud”	1D	No	multiple	Process	CoMD 1.1

Traditional HPC Software Stack

- Current HPC software stack
 - *FORTRAN, C, C++, “X” (and libraries)*
 - *MPI*
 - *Static scheduler*
- Application writer does everything
 - *Load balancing, fault tolerance, dynamic communication patterns, dynamic task scheduling, data migration, code migration*
- More powerful tools can help
- Multiple ways to implement these application capabilities
 - *“Monolithic” languages*
 - *System services*
- Insulate developers and users with APIs and abstractions
- Polyglot approach—no single uber HPC programming model/language

Evolving the HPC Software Stack

- Recall that developer shouldn't be required to “do everything”
- An ExMatEx software stack—system **services** provide support
 - *Node-level work still focused on “X”*
- Leverage “web” and “cloud” software services
- Many of today's successful startups use diverse software stacks
 - *Build an application*
 - *Scale to 100's of millions of users*
 - *Sell your self to Facebook for...*
 - » \$1B: Instagram
 - » \$19B: WhatsApp (450M users on Erlang with 10 engineers!)
- Identify gaps and shortcomings in these technologies
 - *Are there areas where engineering dollars can enable adoption?*
- Must carefully manage granularity to absorb overheads

ExMatEx Contact Information

- <http://exmatex.org>
 - *Project web site*
 - *“Research Areas” → “Runtime Systems” for info related to this talk*
 - » Publications
 - § CoHMM and Summer School (1 accepted, 1 in submission, 1 in preparation)
 - *“News” announcing publication status and proxy release*
- <https://github.com/exmatex>
 - *Project open source site*
 - *CoMD 1.1*
- exmatex@lanl.gov
 - *Project mailing list*
 - exmatex-leads@lanl.gov *if you don't want to spam entire list*

