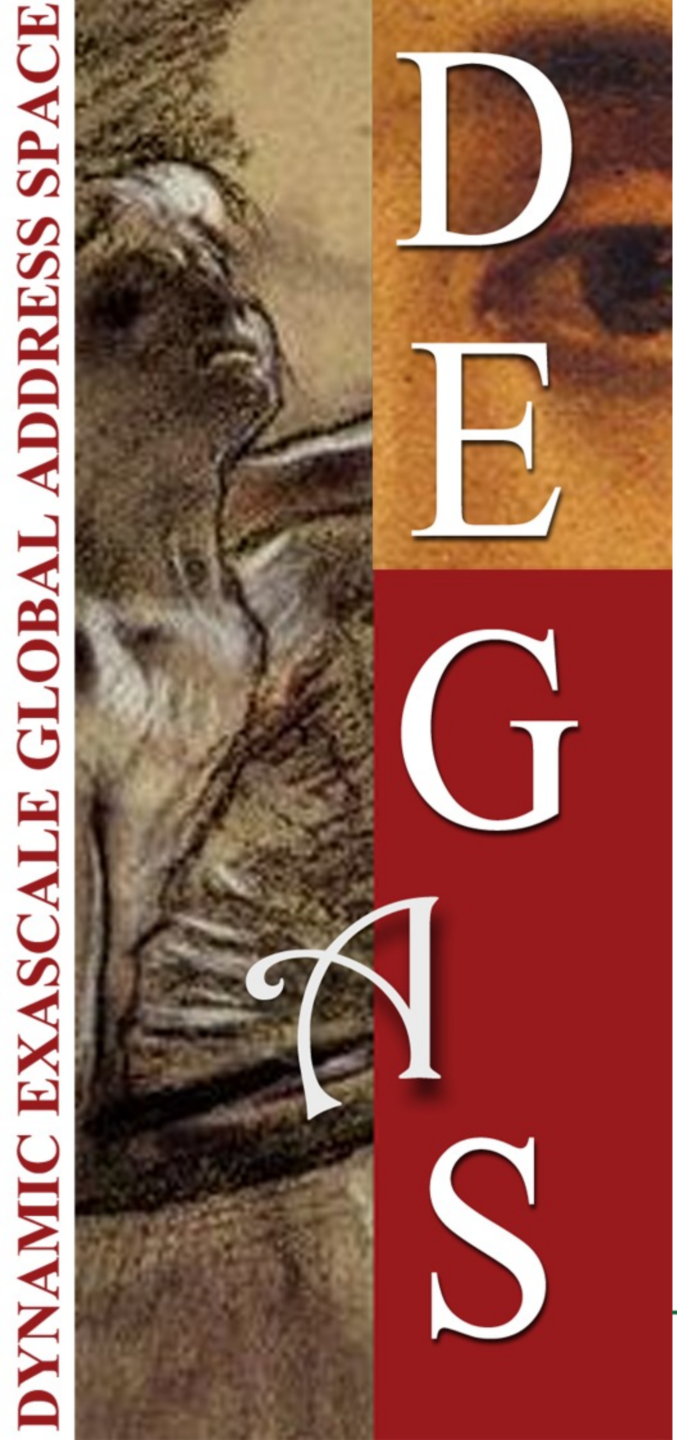
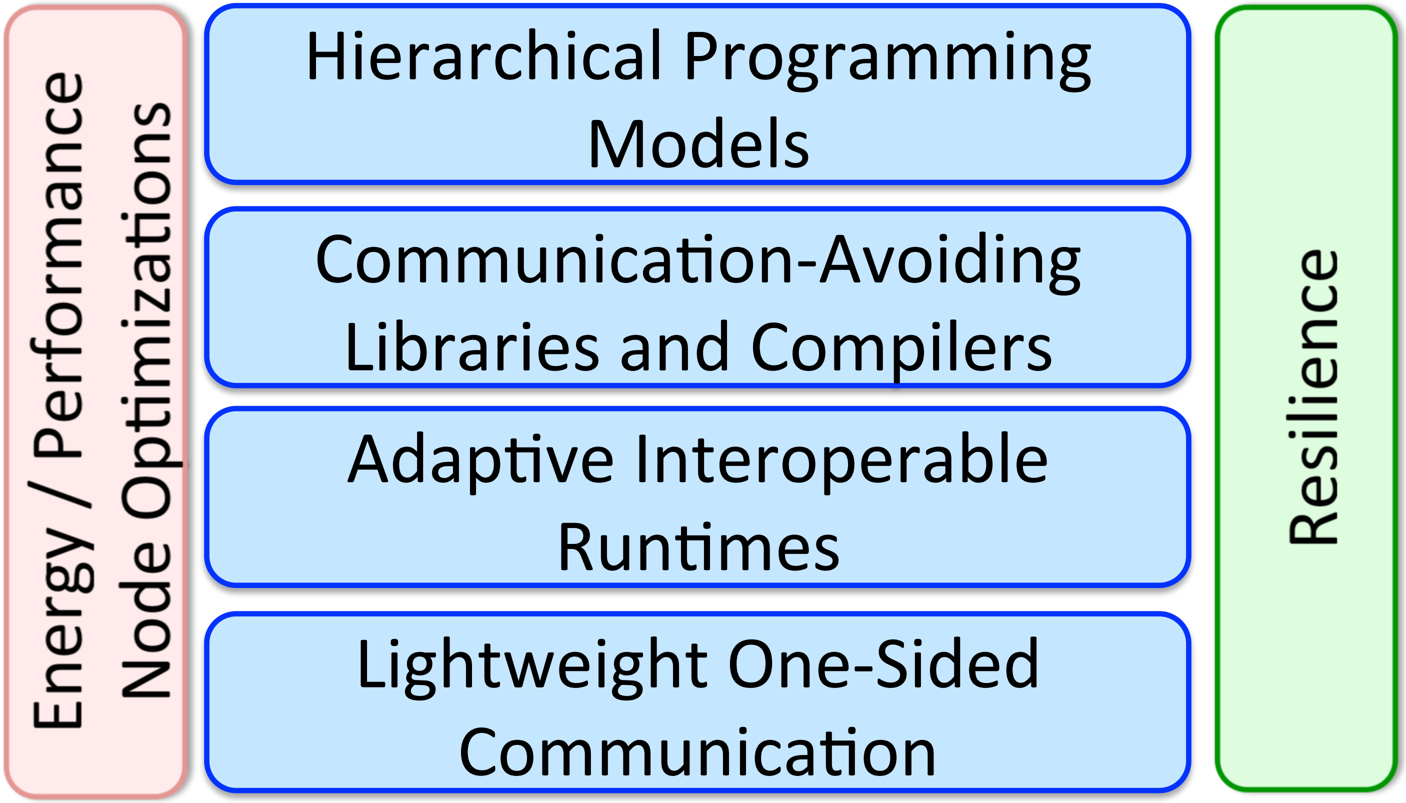


DEGAS

Dynamic Exascale Global Address Space

Katherine Yelick, LBNL PI
Vivek Sarkar & John Mellor-Crummey, Rice
James Demmel, Krste Asanoviç UC Berkeley
Mattan Erez, UT Austin
Dan Quinlan, LLNL
Paul Hargrove, Steven Hofmeyr, Costin Iancu, Khaled Ibrahim, Leonid Oliker, Eric Roman, John Shalf, Erich Strohmaier, Samuel Williams, Yili Zheng, LBNL
+ Several postdocs and students!

DEGAS: Dynamic Exascale Global Address Space



Energy / Performance Node Optimizations

Hierarchical Programming Models

Communication-Avoiding Libraries and Compilers

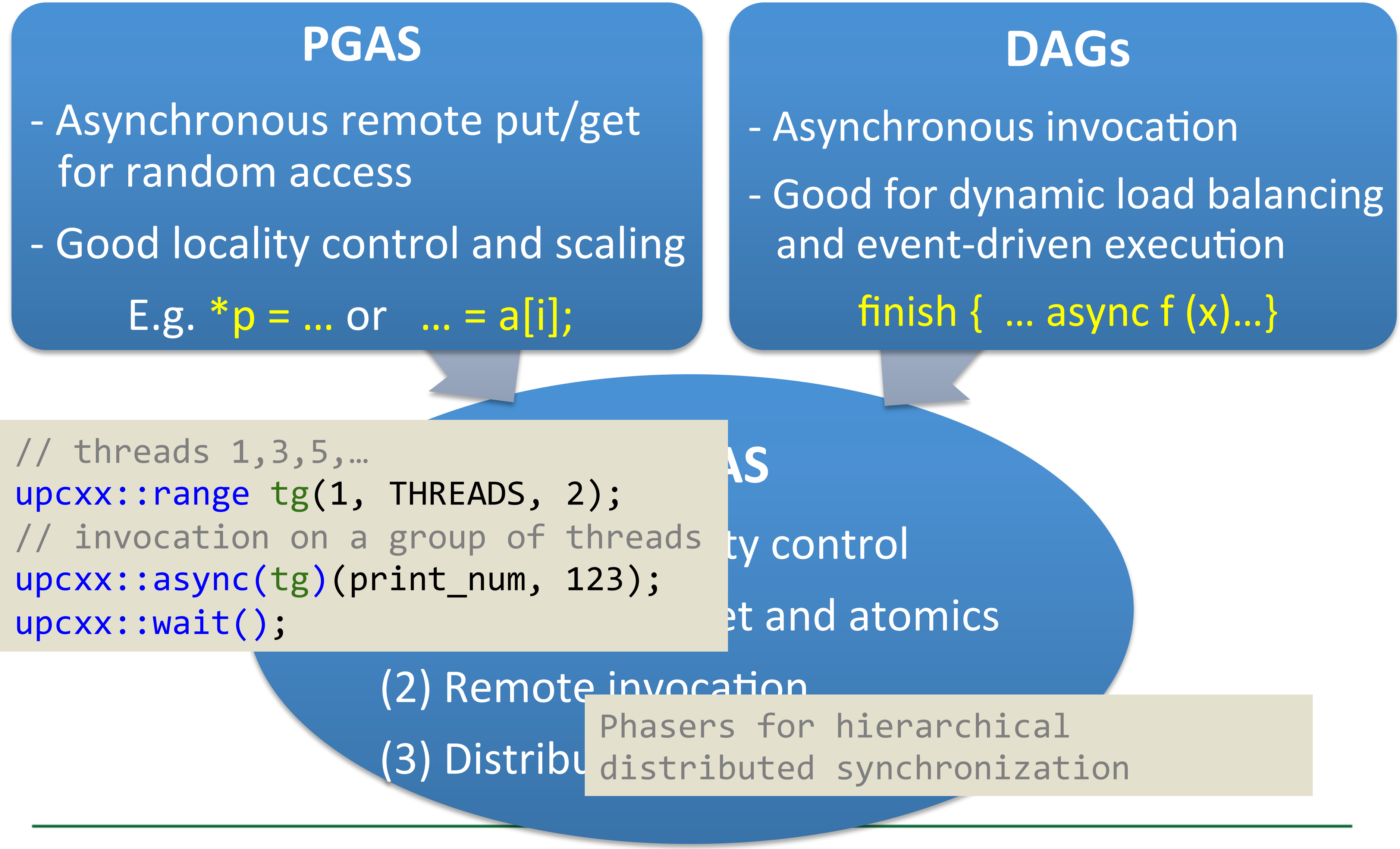
Adaptive Interoperable Runtimes

Lightweight One-Sided Communication

Resilience

Communication-avoiding algorithms generalized to compilers, and communication optimizations in PGAS

Making PGAS more Dynamic; DAG Programming more Locality-Aware



PGAS

- Asynchronous remote put/get for random access
- Good locality control and scaling

E.g. *p = … or … = a[i];

DAGs

- Asynchronous invocation
- Good for dynamic load balancing and event-driven execution

finish { … async f (x)…}

AS

ty control

et and atomics

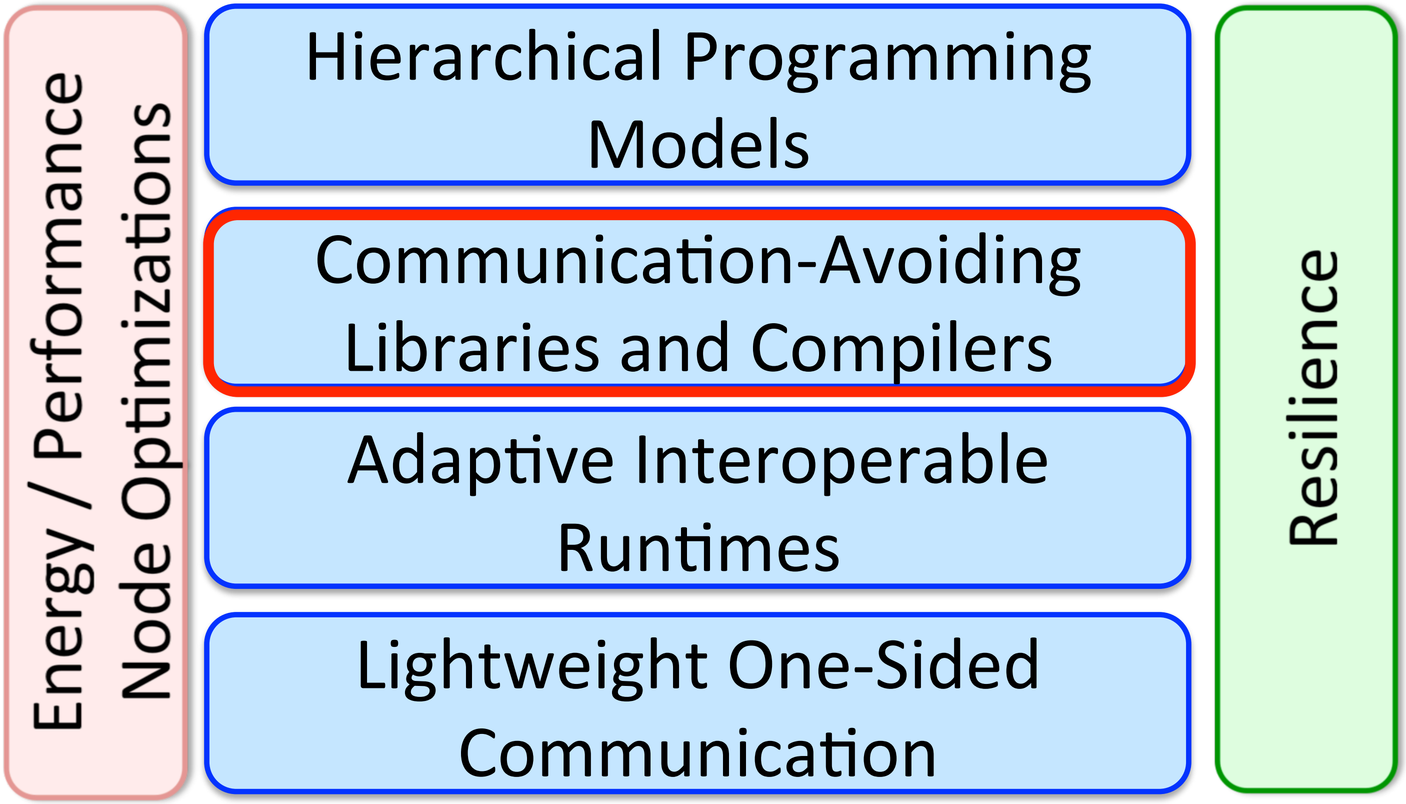
(2) Remote invocation

(3) Distribu

```
// threads 1,3,5,…
upcxx::range tg(1, THREADS, 2);
// invocation on a group of threads
upcxx::async(tg)(print_num, 123);
upcxx::wait();
```

```
Phasers for hierarchical
distributed synchronization
```

DEGAS: Dynamic Exascale Global Address Space



Energy / Performance Node Optimizations

Hierarchical Programming Models

Communication-Avoiding Libraries and Compilers

Adaptive Interoperable Runtimes

Lightweight One-Sided Communication

Resilience

Communication-avoiding algorithms generalized to compilers, and communication optimizations in PGAS

Communication-Avoiding Compilers: Theory to Practice

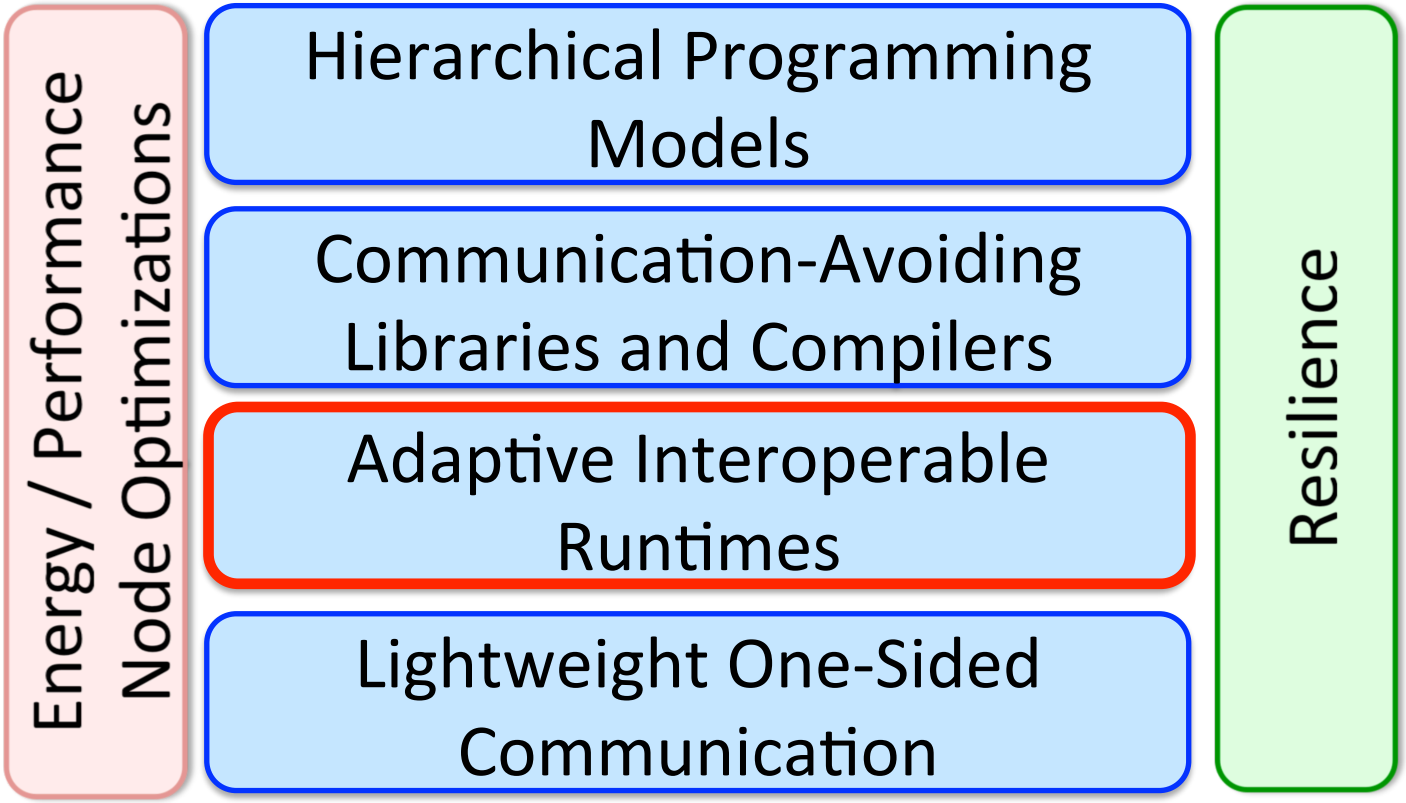
- **Goal: Compilers to generate communication optimal code**
- **Theory**
  - Thm (Christ,Demmel,Knight,Scanlon,Yelick): For any program that "smells like" nested loops, accessing arrays with subscripts that are linear functions of the loop indices

$$\textbf{\#words\_moved} = \Omega\ (\textbf{\#iterations}/M^{e})$$

for some e we can determine

  - Thm (C/D/K/S/Y): Under some assumptions, we can determine the optimal tiles sizes up to constant factors
- **Practice**
  - dHPF compiler at Rice to generate communication-optimal code
  - Series of challenge problems: matmul, n-body, "complex code",…
  - Several hand-analyzed CA algorithms

DEGAS: Dynamic Exascale Global Address Space



Energy / Performance Node Optimizations

Hierarchical Programming Models

Communication-Avoiding Libraries and Compilers

Adaptive Interoperable Runtimes

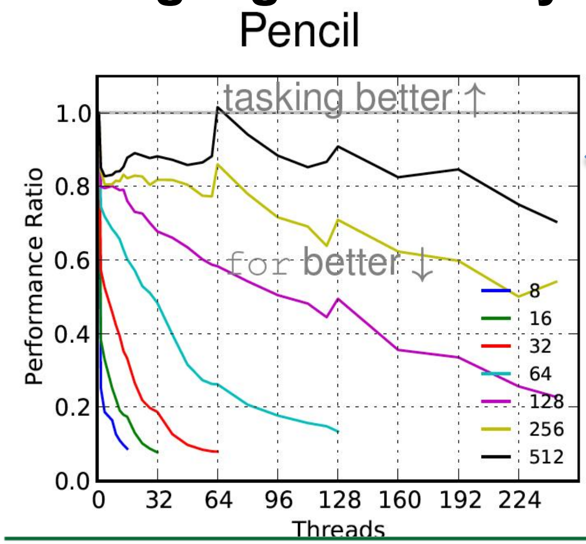
Lightweight One-Sided Communication

Resilience

Communication-avoiding algorithms generalized to compilers, and communication optimizations in PGAS

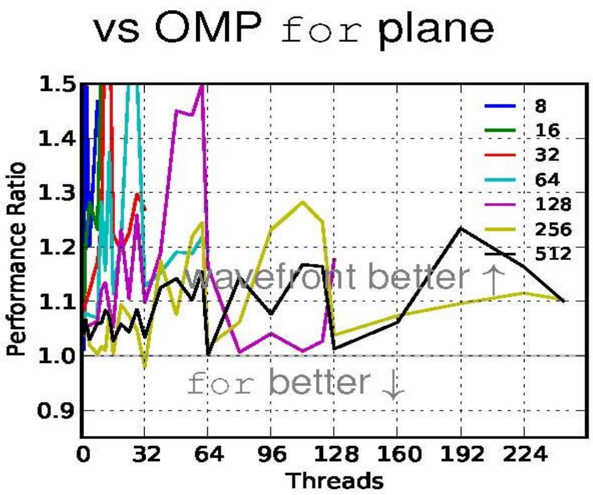
DEGAS Adaptive Interoperable Runtime

- **DEGAS combines global address/name space in tasking**
  - Retains locality control essential for scalability
- **Unifies view of tasking with communication**
  - Not orthogonal: Remote “async” generalized PGAS put/get and creates a remote task invocation (not two-sided)
  - Exploring overlap using tasking vs non-blocking operations
- Integrated UPC and HCLib (Habanero), better than MPI+OpenMP
- **Highlight: Locality aware Ahead-Of-Time task scheduling**

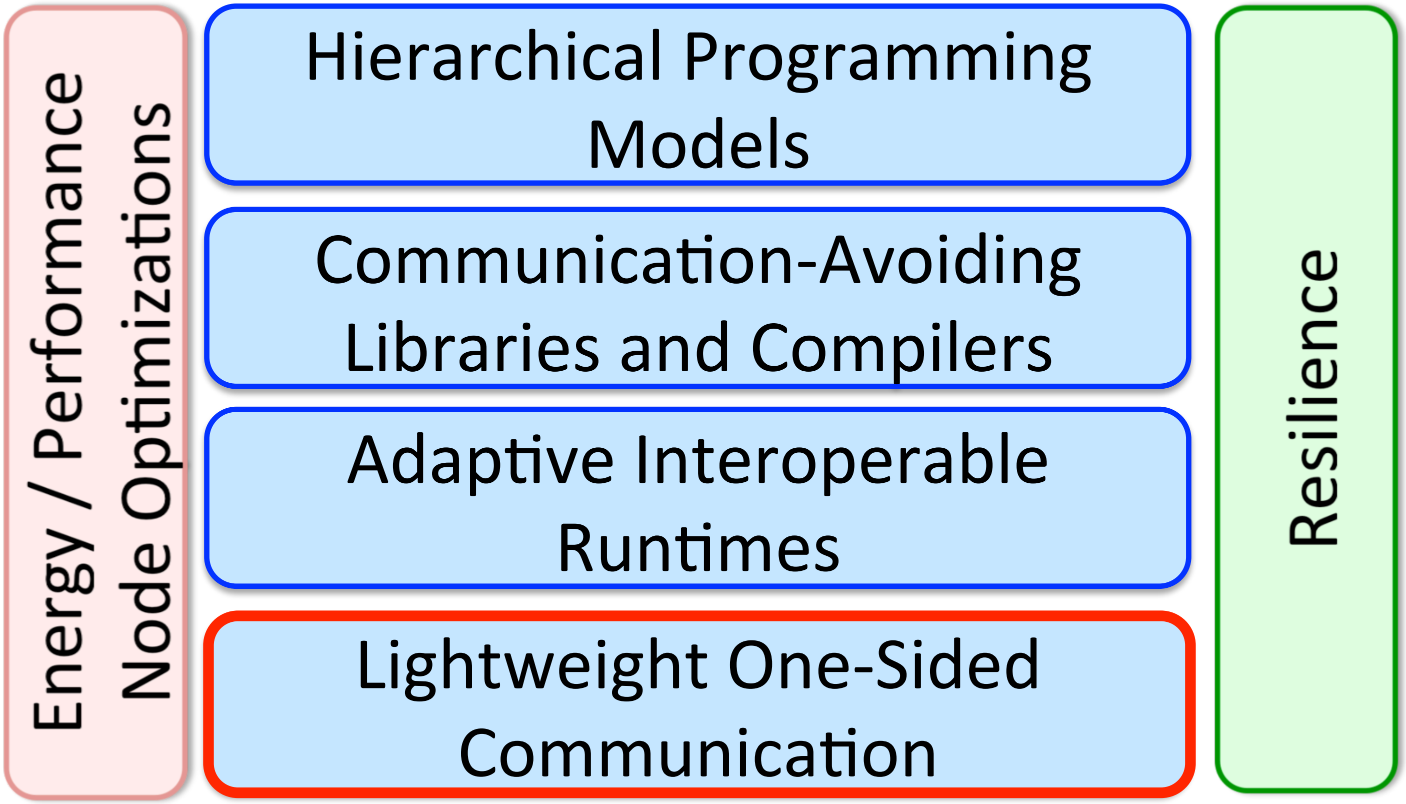

Pencil

OpenMP: tasking up to **5X slower** than for

AOT task **scheduling**:
Tasking **50% faster** than for


vs OMP for plane

DEGAS: Dynamic Exascale Global Address Space



Energy / Performance Node Optimizations

Hierarchical Programming Models

Communication-Avoiding Libraries and Compilers

Adaptive Interoperable Runtimes

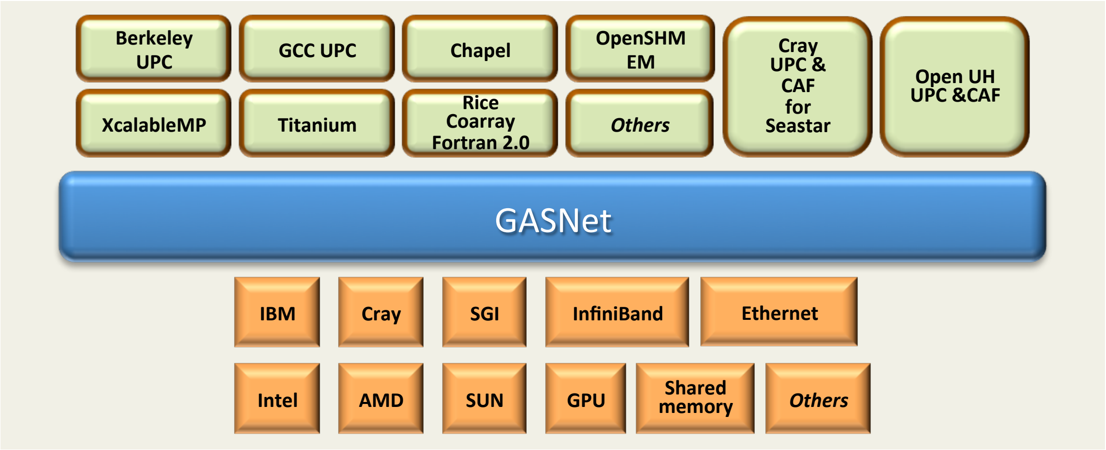
Lightweight One-Sided Communication

Resilience

Communication-avoiding algorithms generalized to compilers, and communication optimizations in PGAS

DEGAS One-Sided Communications R&D

- **Problem**
  - GASNet is ~ubiquitous for PGAS programming
  - Does not address full asynchrony of emerging models and machines



Berkeley UPC

GCC UPC

Chapel

OpenSHM EM

Cray UPC & CAF for Seastar

Open UH UPC &CAF

XcalableMP

Titanium

Rice Coarray Fortran 2.0

Others

GASNet

IBM

Cray

SGI

InfiniBand

Ethernet

Intel

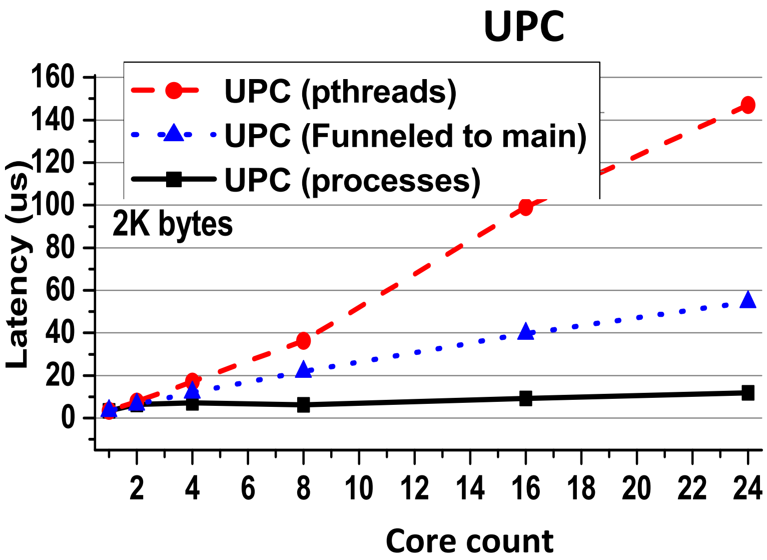
AMD

SUN

GPU

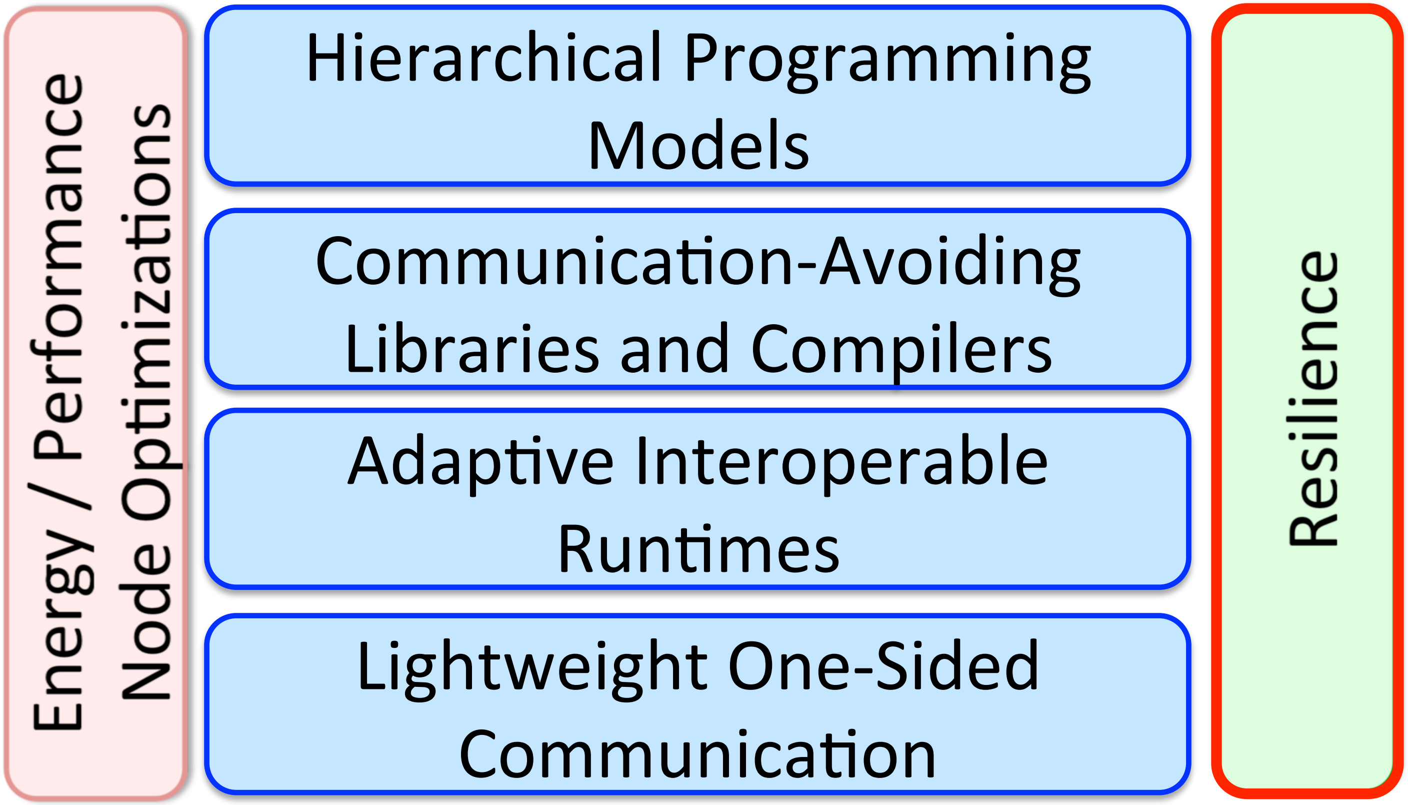
Shared memory

Others



- **GASNet-EX specification is nearly complete after two rounds of review**
- **Several new features (prototyped in GASNet 1.22, released Oct. 2013)**
  - Enables async functions in Habanero-UPC and UPC++
  - Improved performance for both Berkeley UPC and Rice CAF-2.0.
  - Additional improvements in performance and functionality underway

DEGAS: Dynamic Exascale Global Address Space



Energy / Performance Node Optimizations

Hierarchical Programming Models

Communication-Avoiding Libraries and Compilers

Adaptive Interoperable Runtimes

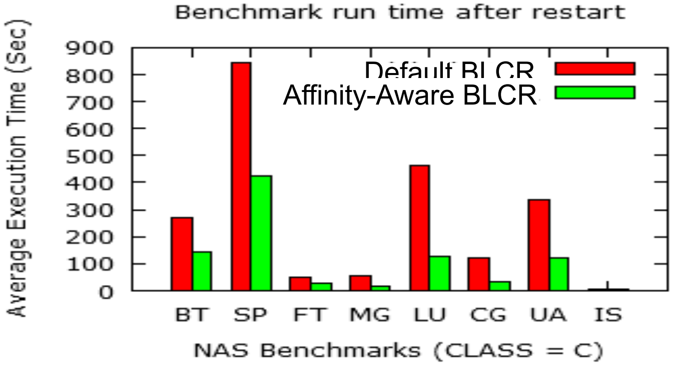
Lightweight One-Sided Communication

Resilience

Communication-avoiding algorithms generalized to compilers, and communication optimizations in PGAS

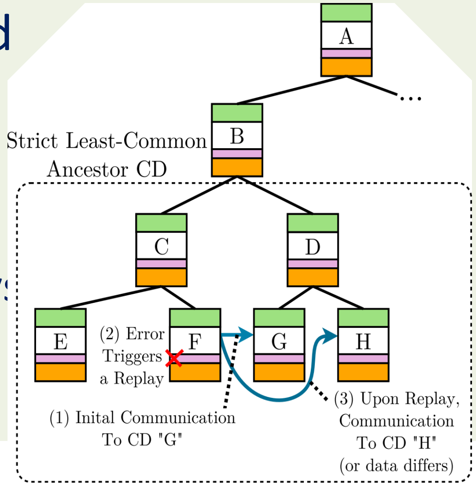
Resilience in a Distributed Exascale GAS

- Resilience strategy: System to Application, GAS-specific
  - Affinity-aware BLCR at NODE-level
  - Consistency coordination at RUNTIME-level
  - Containment domains at APP/LIB-level
- Recent progress
  - GAS-specific CDs (semantics and interfaces, e.g UPC++ API)
  - Affinity-aware BLCR prototyped → **50% speedup!**
  - Consistency coordination designed
- Sane, scalable resilience!



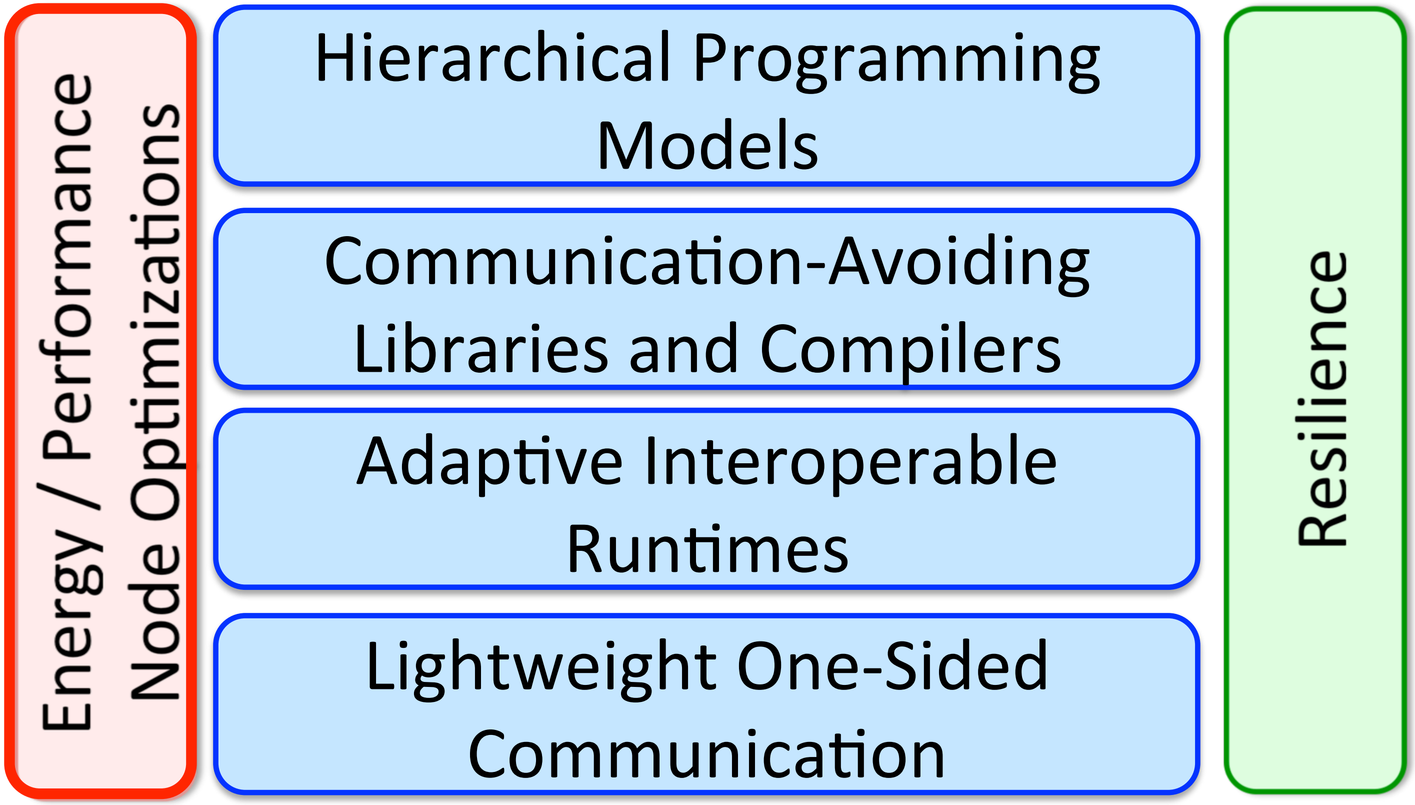
Containment domains GAS semantics

- Strict vs Relaxed
- Relaxed
  - Comm. Logs
  - Dependencies
  - Data exchange v Actual comm.



A
...
Strict Least-Common Ancestor CD
B
C
D
E
F
G
H
(2) Error Triggers a Replay
(1) Inital Communication To CD "G"
(3) Upon Replay, Communication To CD "H" (or data differs)

DEGAS: Dynamic Exascale Global Address Space



Energy / Performance Node Optimizations

Hierarchical Programming Models

Communication-Avoiding Libraries and Compilers

Adaptive Interoperable Runtimes

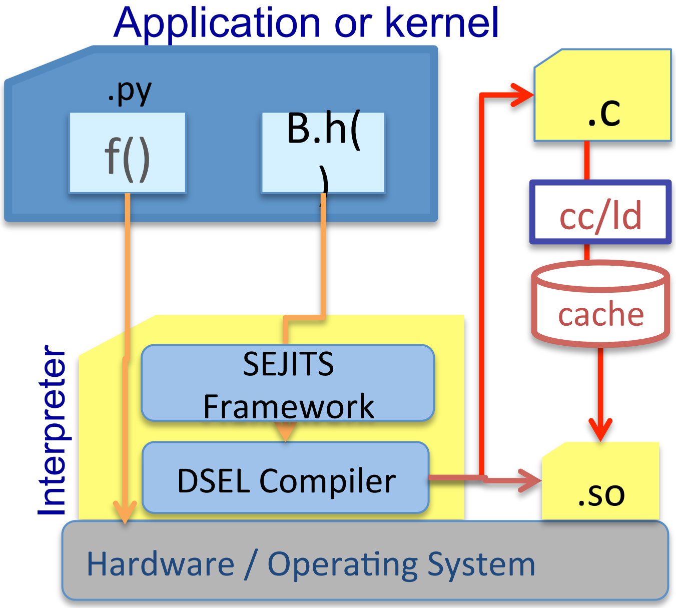
Lightweight One-Sided Communication

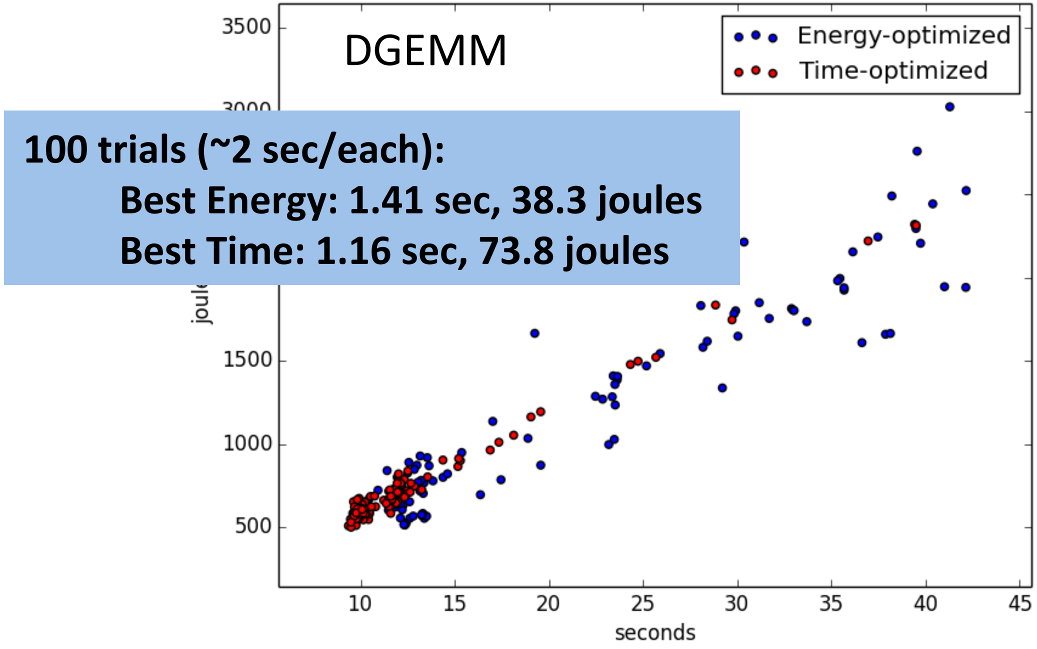
Resilience

Communication-avoiding algorithms generalized to compilers, and communication optimizations in PGAS

Performance and Energy Node Optimizations

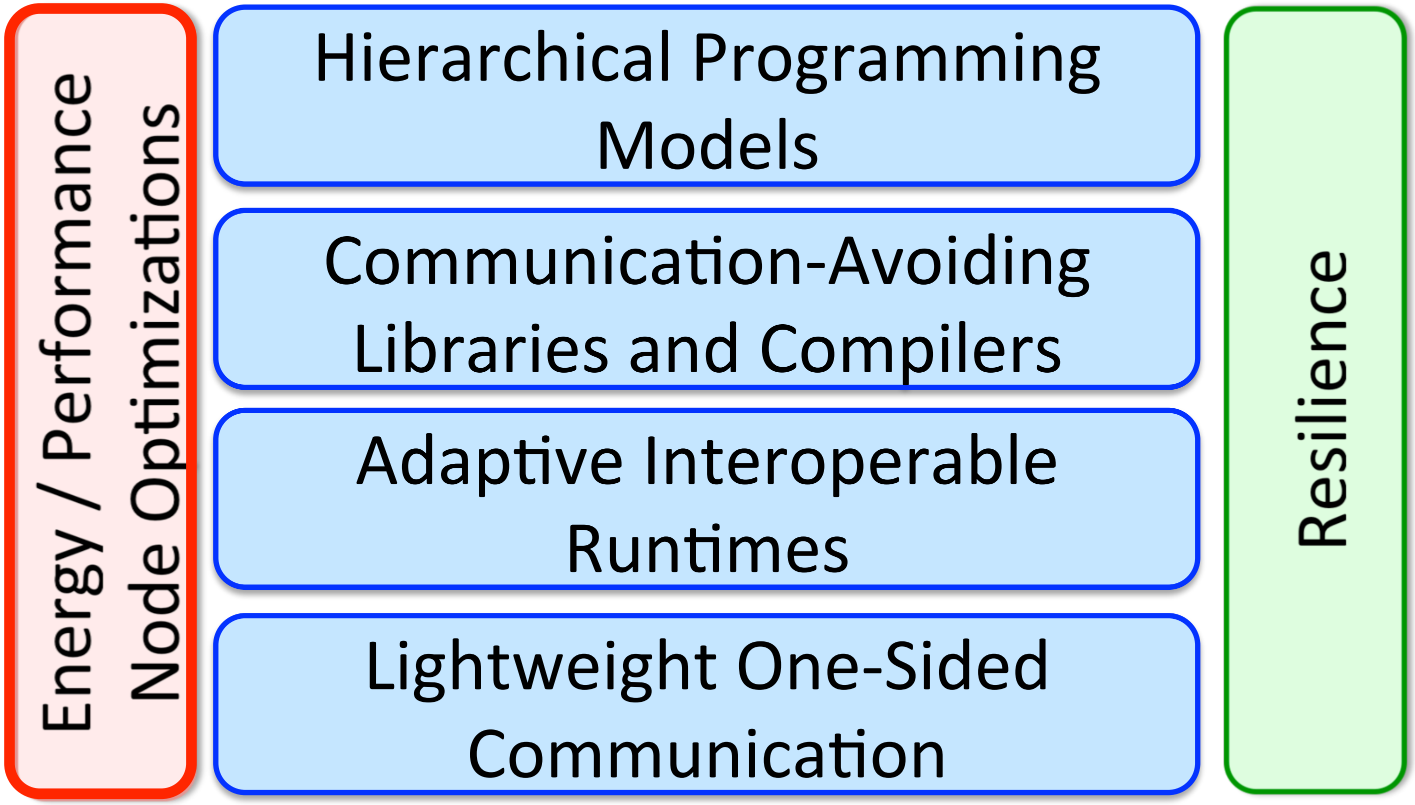
- **Roofline modeling to measure limits**
  - New benchmark (joint with Super) for "roof"
- **Automatic performance tuning to reach limits (uses OpenTuner)**
- **Code generation options: compiler, DSL, annotations,…**
  - DEGAS ctree uses Python introspection on ASTs (joint with ASPIRE)





100 trials (~2 sec/each):
Best Energy: 1.41 sec, 38.3 joules
Best Time: 1.16 sec, 73.8 joules

DEGAS: Dynamic Exascale Global Address Space



Energy / Performance Node Optimizations

Hierarchical Programming Models

Communication-Avoiding Libraries and Compilers

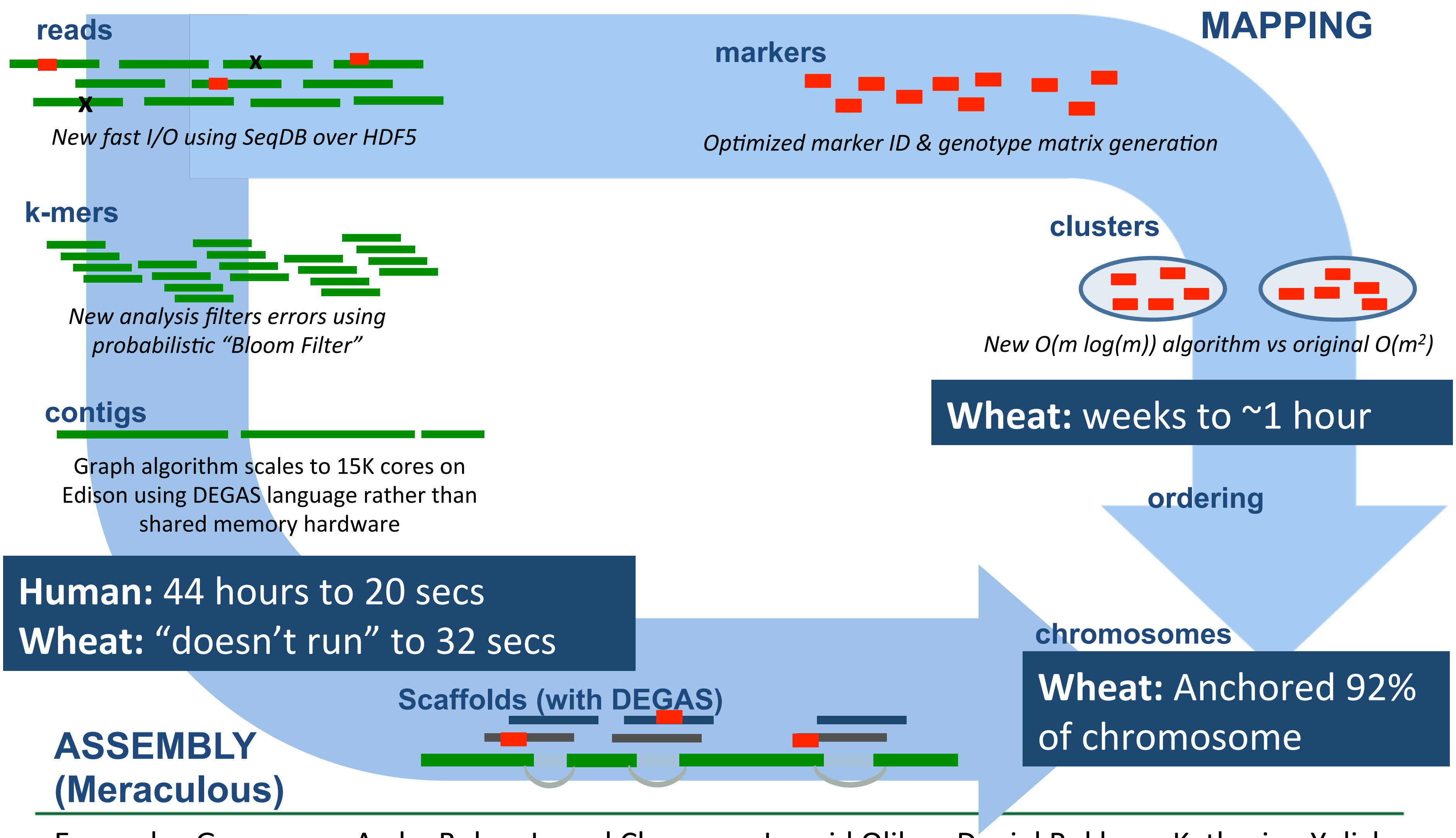
Adaptive Interoperable Runtimes

Lightweight One-Sided Communication

Resilience

Applications that use DEGAS features in non-trivial ways

Algorithms, Programming Models, and Parallelism Help Solve Extreme Data Challenge in Genomics



reads

New fast I/O using SeqDB over HDF5

k-mers

New analysis filters errors using probabilistic "Bloom Filter"

contigs

Graph algorithm scales to 15K cores on Edison using DEGAS language rather than shared memory hardware

Human: 44 hours to 20 secs
Wheat: "doesn't run" to 32 secs

Scaffolds (with DEGAS)

ASSEMBLY (Meraculous)

MAPPING

markers

Optimized marker ID & genotype matrix generation

clusters

New O(m log(m)) algorithm vs original O(m^2)

Wheat: weeks to ~1 hour

ordering

chromosomes

Wheat: Anchored 92% of chromosome

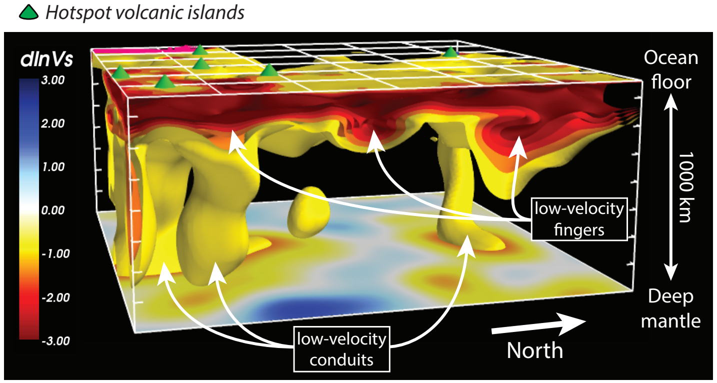
Evangelos Georganas, Aydın Buluc, Jarrod Chapman. Leonid Oliker, Daniel Rokhsar, Katherine Yelick

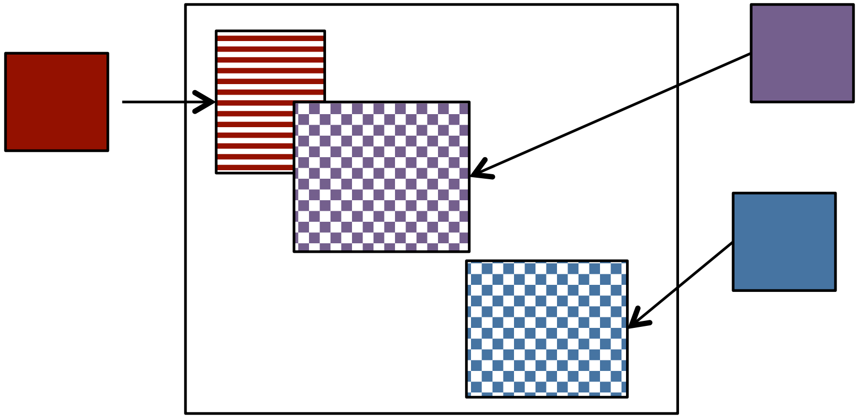
Funding of various pieces from DEGAS, Mantissa, Early Career, LDRD, JGI

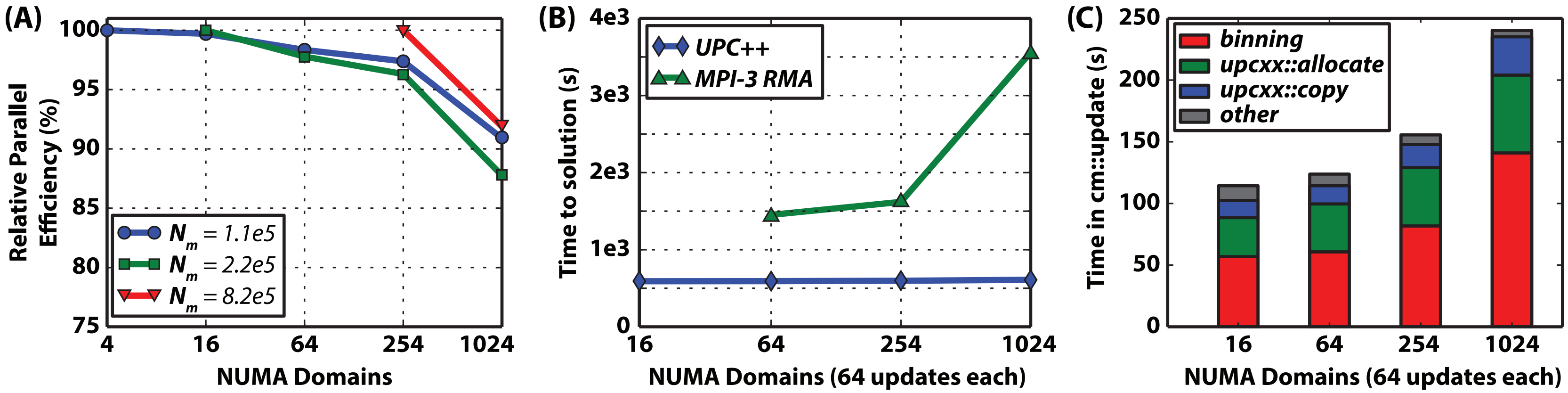
Seismic Waveform Imaging: Data Fusion in UPC++

- Merge measurement data into simulation and evaluate fit
- Matrix is too large for single shared memory; strided writes in global array
- PGAS+Async for previously non-scalable part of MPI / ScaLAPACK, code

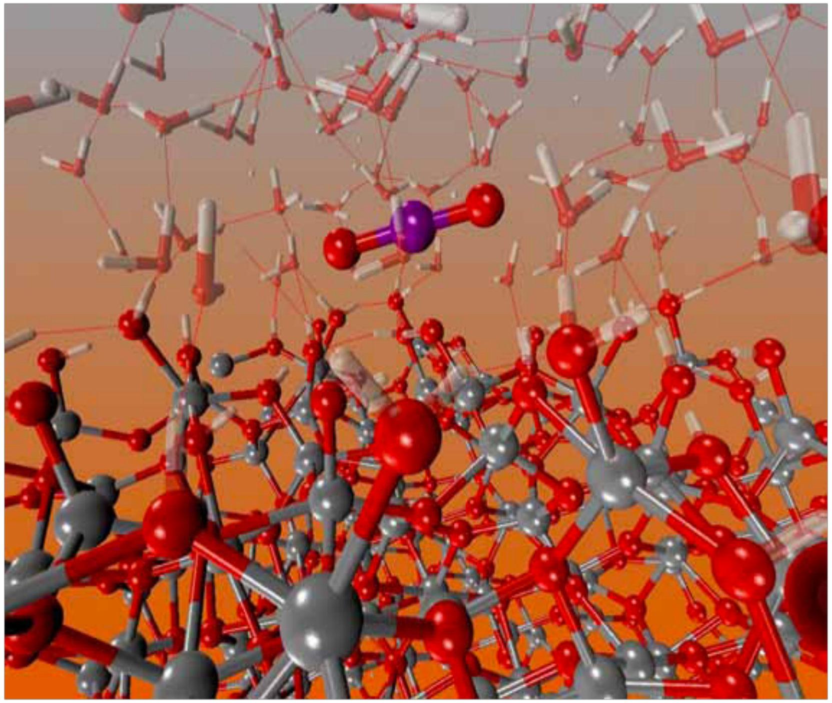
(A) Model SEMum2 (Central Pacific view)







DEGAS in NWChem

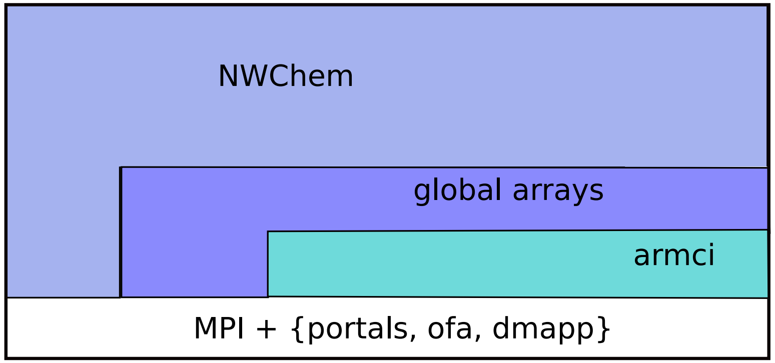


credit:nwchem-sw.org

- **High-performance computational chemistry code**
  - Flagship DOE chemistry software
- **60K downloads world wide**
- **200-250 scientific application publications per year**
- **Over 6M LoC, 25K files**
- **Scales to 100K+ processors**

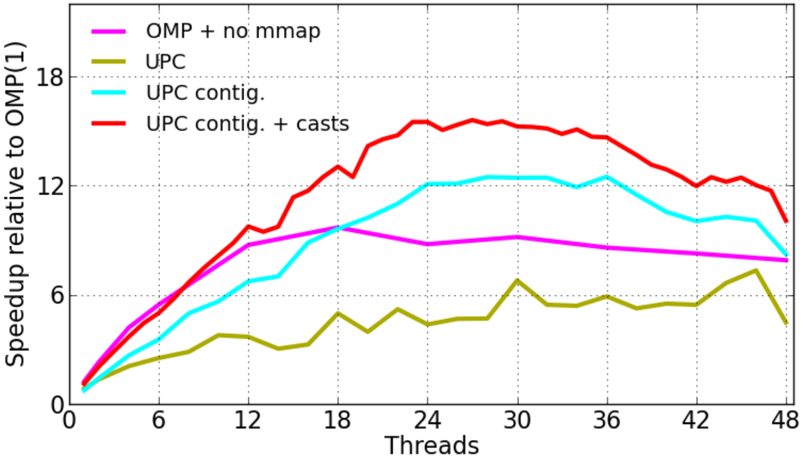
Internal tasking, memory management, and application checkpoint/restart

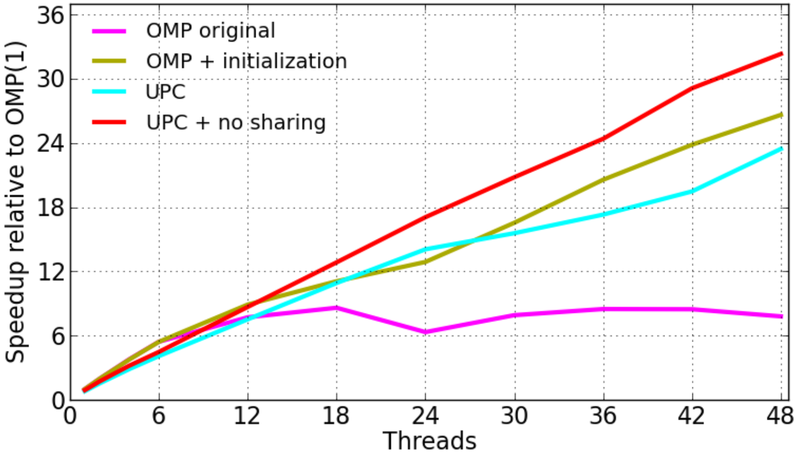
- **DEGAS work on new GA over GASNet (-EX)**
- **DEGAS personnel (Rice, LBNL) on other projects: performance analysis and tuning**



Performance Feedback from Applications

- **Benchmarks and proxy apps:**
  - Smith-Waterman (Habanero-UPC)
  - miniGMG ExaCT (Habanero-UPC, …)
  - Stencil from ExaCT etc. (UPC++)
- **Full (possibly production) applications:**
  - NWChem (GASnet)
  - Contig construction in Meraculous Genome Assembly (UPC)
  - Matrix assembly for using observational data in simulations (UPC++)
- **Libraries abstractions:**
  - Distributed Matrices
  - Multi-dim arrays (AMR)
  - Distributed hash table
  - .5D Array library





DEGAS Vision

Predictable work	Yes	• Static load balance	SPMD
	Semi	• Semi-Static load balance	Graph partition
Regular task graph structure	No	• Dynamic load balance	Task Queue
	Yes	• Data parallel	
Predictable communication	Semi	• Hierarchical data parallel	Phasers
	Yes	• Tree (out-tree)	Asynch
Regular communication	No	• General DAG (or in-tree)	Deadlock free scheduler
	Yes	• Two-sided OK	Send/Receive
Key data structures	No	• One-sided desirable	Put/Get
	Yes	• Neighbor + Collectives	Collectives
Very hierarchical machine?	No	• Any-to-any Collectives	Vertical PGAS
	Yes	• Comm Avoid Compiler	DS Code Gen
Unpredictable machine?	Yes	• Hierarchical parallelism	Hierarchical Ctl
	Yes	• Over-partition work	Annealing sched
Faulty machine?	Yes	• Hierarchical Domains	Contain Doms

Interoperability is built into DEGAS

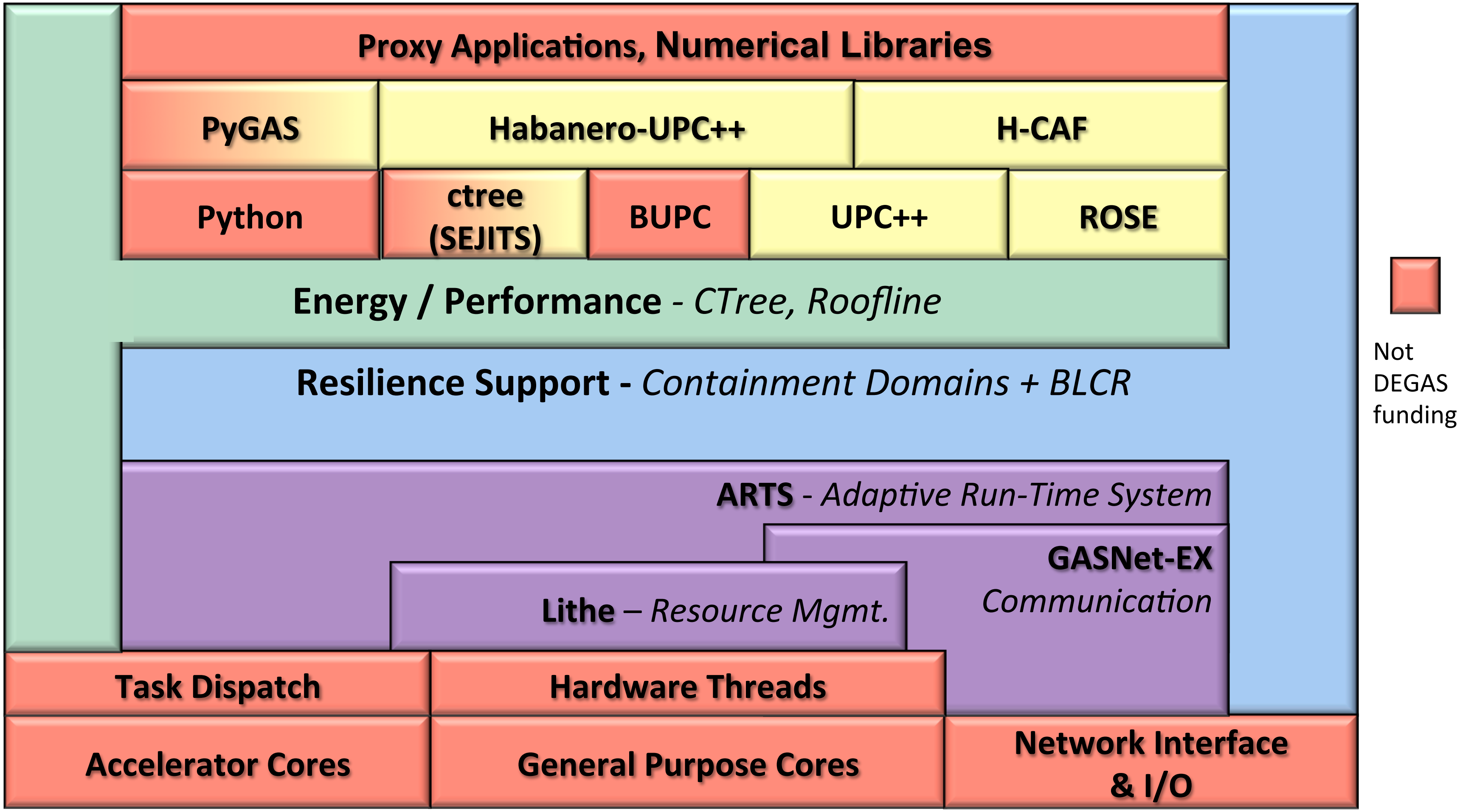
DEGAS: The Rest of the Vision and Status

- Predictable work
- Regular task graph structure
- Predictable communication
- Regular communication
- Key data structures
- Very hierarchical machine?
- Unpredictable machine?
- Faulty machine?

- Multi-Dimensional Grids (arrays)
- Hash Tables
- Cacheable Read-only objects
- .5D Arrays for Comm Avoidance
- Bloom Filters
- Sparse Matrices
- Oct Trees
- Everything else

- SPMD
- Graph partition
- Task Queue
- Phasers
- Asynch
- Deadlock free scheduler
- Send/Receive
- Put/Get
- Collectives
- Vertical PGAS
- DS Code Gen
- Hierarchical Ctl
- Annealing sched
- Contain Doms

DEGAS Software Stack



Comments

- **Dynamic decisions are easiest to implement within a node, but probably most useful between nodes**
- **A "bad" machine can turn easy problems to a hard ones (back edges)**
  - It has to bad enough (unpredictable, faulty) to overcome the locality advantages of a static/semi-static
- **Challenge of designing and selling X-Stack projects today**
  - Most DOE applications get by with static and semi-static load balancing on today's machines; Mini-apps are the worst case for us (too easy)
  - A few have divide and conquer parallelism that encourage dynamic runtimes
  - Some have high compute to communicate ratios tolerate dynamic runtimes

Two reasonable approaches:

- **Provide dynamic communication, scheduling, load balancing, synchronization, data structures as options**
- **Make dynamicism the default and infer locality structure**

Habanero-UPC++ vs. MPI+OpenMP

	Habanero-UPC++	**MPI+OpenMP**
Implementation approach	C++ template, prototype work on LLVM-based code generation on node	MPI – library; OpenMP -- compiler pragmas
Locality management	Data layout, abstraction of machine hierarchy	MPI: Processes + messages OpenMP: affinity control
Support Languages	C and C++, with strong emphasis on modern C++	C, C++, FORTRAN,… (C++ API is the same as the C API)
Internode parallelism	Remote read/write and invocation. Plans for team (mixed parallelism) and load balancing libraries	Message passing, collective operations, Communicators (teams) for hierarchy
Intranode parallelism	Multidimensional arrays, async tasks, work stealing	Fork-join work sharing, parallel for loops

Remark: Interoperability is goal; it is fine to use Habanero-UPC++ plus MPI+OpenMP, e.g., our seismic imaging app.

Highlights of Future Plans

- **Programming Models**
  - Report on arrays; additional irregular data structures
  - Finalize hierarchy abstractions,
- **Communication-avoiding compilers and adaptive runtimes**
  - CA final theory; implementation (dHPF), hierarchy (HCAF, UPC++)
  - Integrate HClib with UPC++
  - Experiment on degrees of dynamicism with various task graph structures
- **GASnet-EX and Resilience**
  - Spec and implementation for emerging architectures
- **Performance and energy optimizations**
  - Complete CTree code gen;
- **Demonstrations and reports**
  - ExaCT Chemistry application (PGAS, .5D,…) from collaborathon
  - Genome contig generation integrated in assembly pipeline

Collaborations with Co-Design Centers

- **ExMatEx:**
  - "Collaborathon" in March 2014 focused on UPC++, DAG-scheduling, Resilience, and communication-avoiding algorithms
    - Follow-up visit by Yelick to LANL and others in the ExMatEx team to discuss a particular problem in CA Sparse MatMul in chemistry
  - Use of Lulesh throughout DEGAS (resilience, languages, runtimes,…)
- **ExaCT:**
  - Shared personnel (Sam Williams);
    - Proxy-App MiniGMG developed by Williams used throughout DEGAS
  - Co-Design/X-Stack postdocs Cy Chan & Didem Unat (Shalf supervised):
    - Participate in all DEGAS meetings, retreats, etc. with special interest on hierarchical data structures and DAG scheduling
- **CESAR:**
  - Planned visit by Andrew Seigel to Berkeley Lab to discuss particular "PGAS" related algorithmic challenge

Collaborations with Other Applications

- **NWChem**
  - NWChem ported to run on GASNet (had been only ARMCI)
  - Performance tuning work ongoing
- **Bioinformatics (with D. Rokhsar, J. Chapman, Aydin Buluc, JGI/LBNL)**
  - "Contig" construction phase of assembly pipeline parallelized
  - Uses PGAS (UPC) rather than shared memory (prior art)
  - Rest of pipeline also being optimized by other projects (LDRD, Buluc's ASCR-Graph, etc.)
- **Seismic modeling (with Barbara Romanowicz, Scott French, UCB)**
  - Full interior earth model as seen by seismic waves for basic science, energy production, carbon sequestration, and policy verification (Comprehensive Nuclear-Test-Ban Treaty).
  - PGAS used in building large distributed matrix from observational and simulation data. Interoperates with MPI and ScaLAPACK.

Collaboration with Other X-Stack Projects

- **Corvette**
  - Shared personnel (Demmel and Iancu) and use of PGAS as target for their analyses
- **X-Tune**
  - Common personnel (Williams, Oliker); closely tied to code generation approach for novel node architectures (X-Tune uses annotated general purpose languages; DEGAS has domain-specific code generators; latter also with ASPIRE DARPA project at UCB)
- **Resilience collaboration plans with GVR, Argo and Hobbes**
  - Through Frank Mueller, Costin Iancu, Steve Hofmeyer, etc.
- **OCR**
  - Common personnel (at Rice) and use of OCR under Habanero
  - Ongoing work to understand relative strengths of approaches