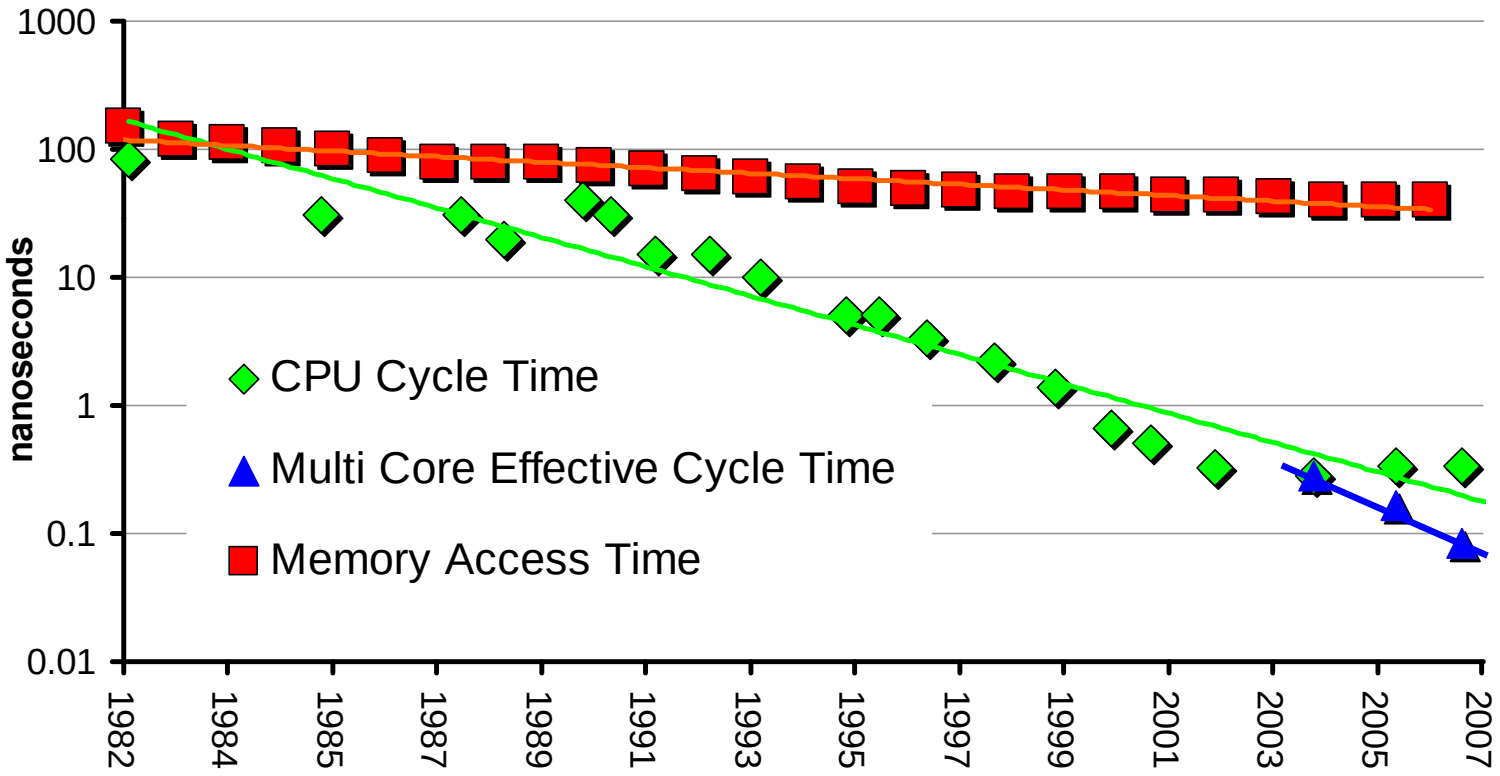
Red Shift


1000
100
10
1
0.1
0.01
nanoseconds
CPU Cycle Time
Multi Core Effective Cycle Time
Memory Access Time
1982
1984
1985
1987
1989
1991
1993
1995
1997
1999
2001
2003
2005
2007

Because of Red Shift

- **Today's Petascale systems typically run at about 10% efficiency on full-system calculations in the following sense; processors spend most of their time just waiting for data to arrive from the local memory hierarchy or from other processors.**
- **A large number of techniques attempt to improve this low efficiency at various levels of the hardware/software stack.**
- **To list just a few:**

Techniques for dealing with Red Shift

- **At the hardware level, caches**
- **Again in hardware, prefetch engines**
- **Runtime systems may (depending on the system) attempt to move or copy memory pages from non-local to local memory in a distributed cc-NUMA environment, thus after repeated remote accesses they could optimize the best “horizontal” data layout.**
- **Compilers may try to structure data accesses for maximum locality as for example via cache-blocking or loop-fusion transformations.**
- **Programming languages may provide means for programmers to express locality that in turn (at least in theory) can be exploited by the compiler or runtime**
- **Threads, a paradigm that may be supported in hardware to tolerate latency of data motion.**

YET TODAY THESE TECHNIQUES ARE ALL JUST POINT-SOLUTIONS THAT DO NOT INTEROPERATE AND MAY EVEN FIGHT WITH EACH OTHER IN AN ATTEMPT TO IMPROVE EFFICIENCY OF DATA MOTION

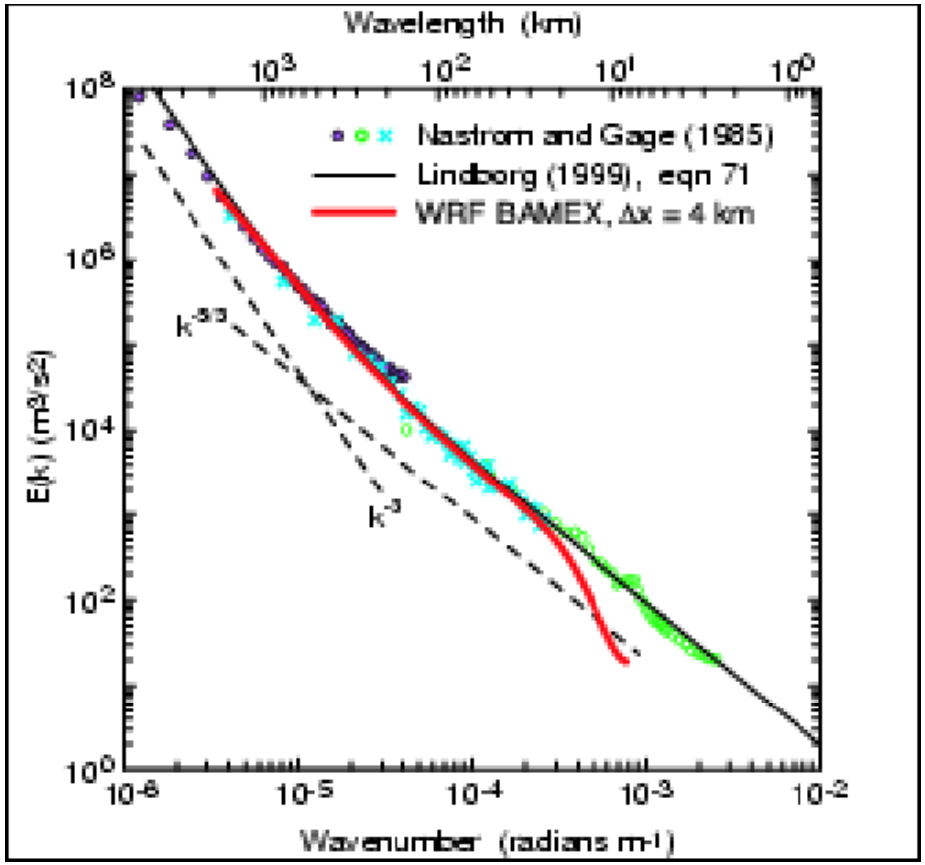
Rest of talk

- **Some heroic calculations and hoops you have to jump through**
  - WRF Nature
  - SpecFEM3D
    - Performance modeling
- **A brainstorm idea: a whole system approach to improving global data motion?**

WRF: Description of Science

Hypothesis: enhanced mesoscale predictability with increased resolution can now be addressed

- **Kinetic energy spectrum of the atmosphere has a slope transition from k^{-3} to $k^{-5/3}$ (e.g. Lindborg, 1999)**
- **Increased computational power enabling finer resolution forecasts into the $k^{-5/3}$ regime**
- **Improve understanding of scale interactions:**
  - for example, wave-turbulence interactions
  - improve predictability and subscale parameterizations

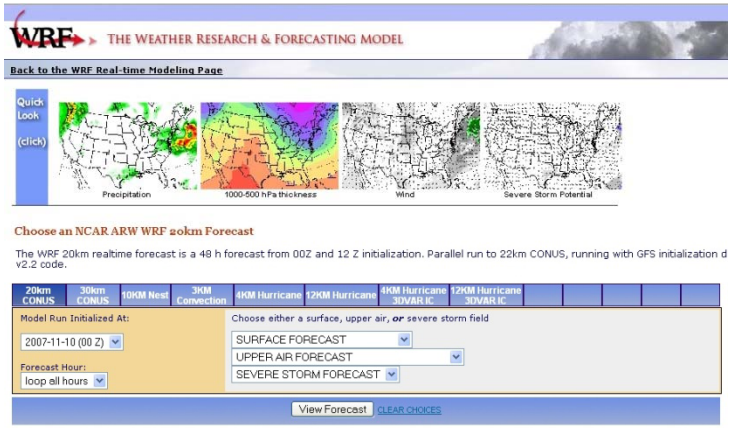


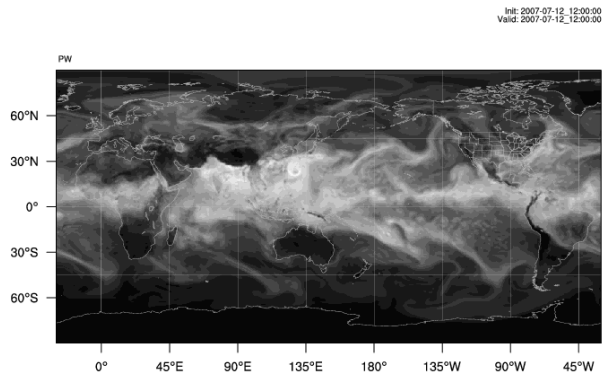
Skamarock, W. S., 2004: Evaluating Mesoscale NWP Models Using Kinetic Energy Spectra. Mon. Wea. Rev., 132, 3019--3032.

WRF Overview

- **Large collaborative effort to develop next-generation community model with direct path to operations**
  - Limited area, high-resolution
  - Structured (Cartesian) with mesh-refinement (nesting)
  - High-order explicit dynamics
  - Software designed for HPC
  - 3000+ registered users
- **Applications**
  - Numerical Weather Prediction
  - Atmospheric Research
  - Coupled modeling systems
  - Air quality research/prediction
  - High resolution regional climate
  - Global high-resolution WRF

http://www.wrf-model.org





5 day global WRF forecast at 20km horizontal resolution. running at 4x real time 128 processors of IBM Power5+ (blueice.ucar.edu)

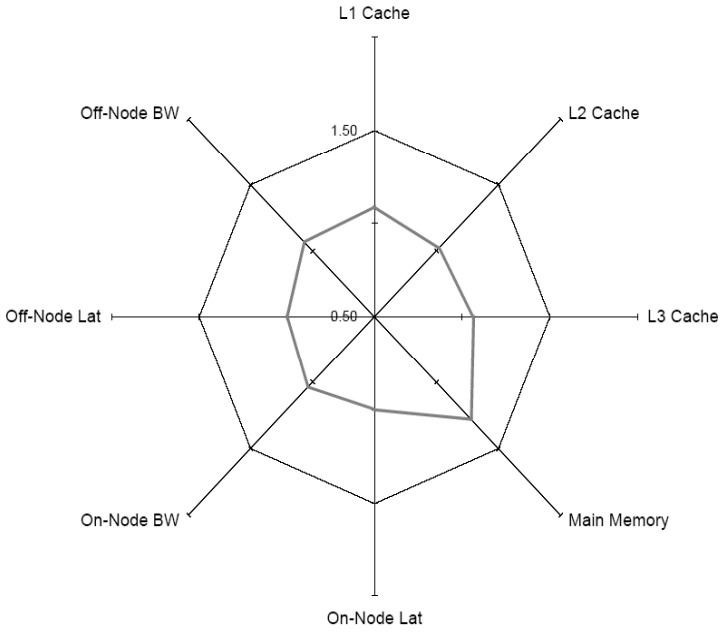
Nature Run: Methodology

- **Configuration and Domain**
  - Idealized (no terrain) **hemispheric** domain
    - 4486 x 4486 x 100 (2 billion cells)
    - 5KM horizontal resolution, 6 second time step
    - Polar projection
  - Mostly adiabatic (dry) processes
  - Forced with Held-Suarez climate benchmark
  - 90-day spin-up from rest at coarse resolution (75km)

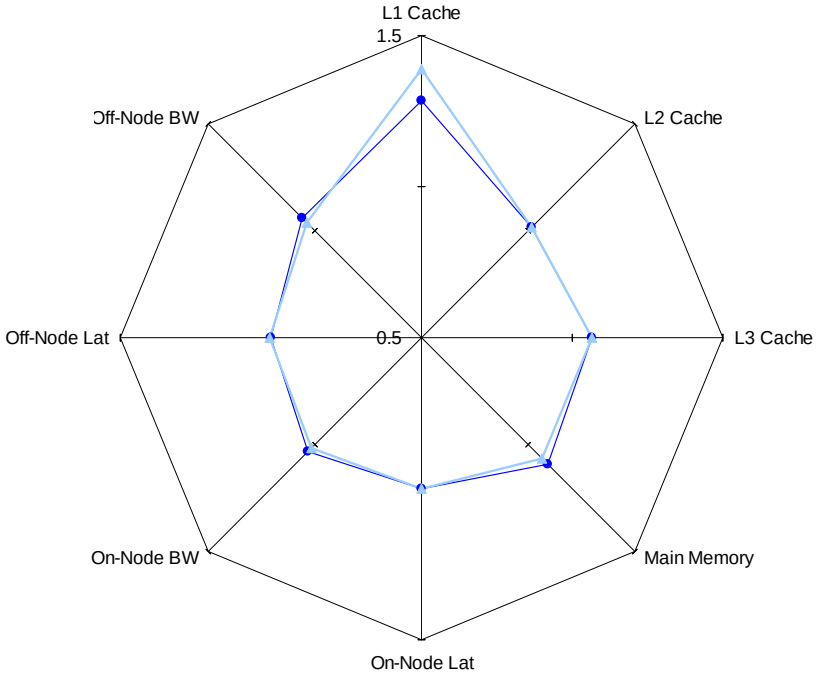
Tuning challenges

- **Data decomposition (boundary conditions)**
- **I/O (parallel I/O required)**
- **Threads thrash each other**
- **Cache volumes wasted**
- **Load imbalance**
- **Lather-rinse-repeat**

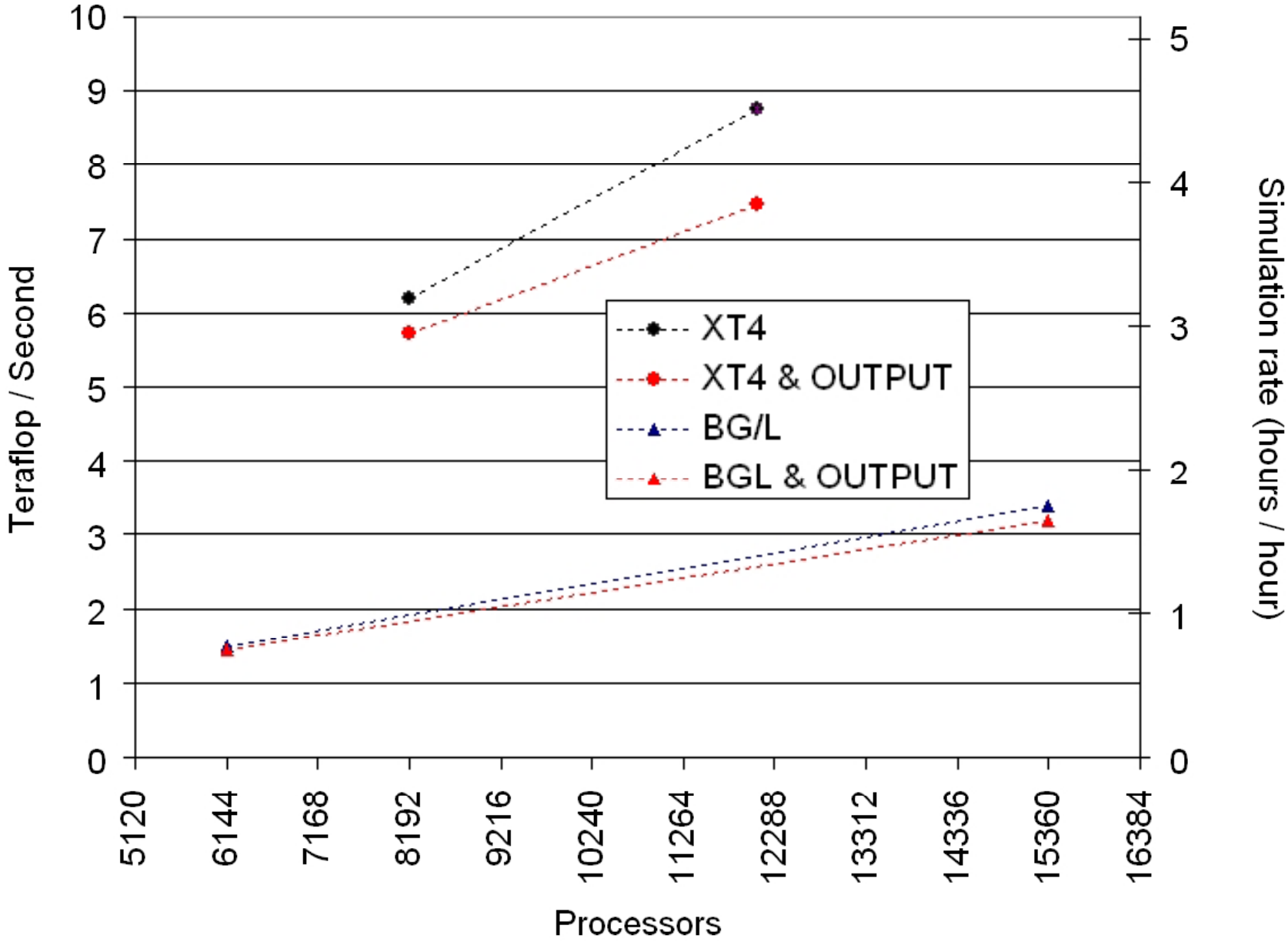
A performance model of WRF



tuning



Effective Floating Point Rate

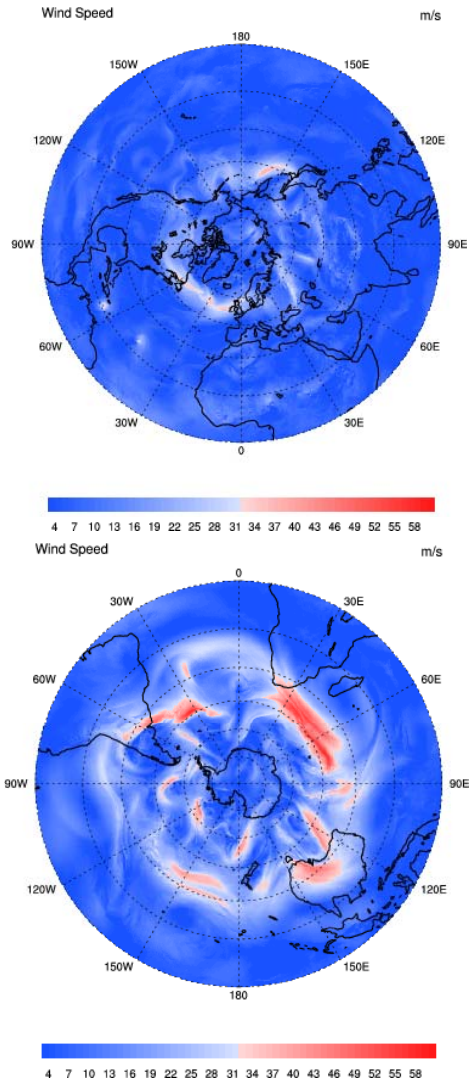


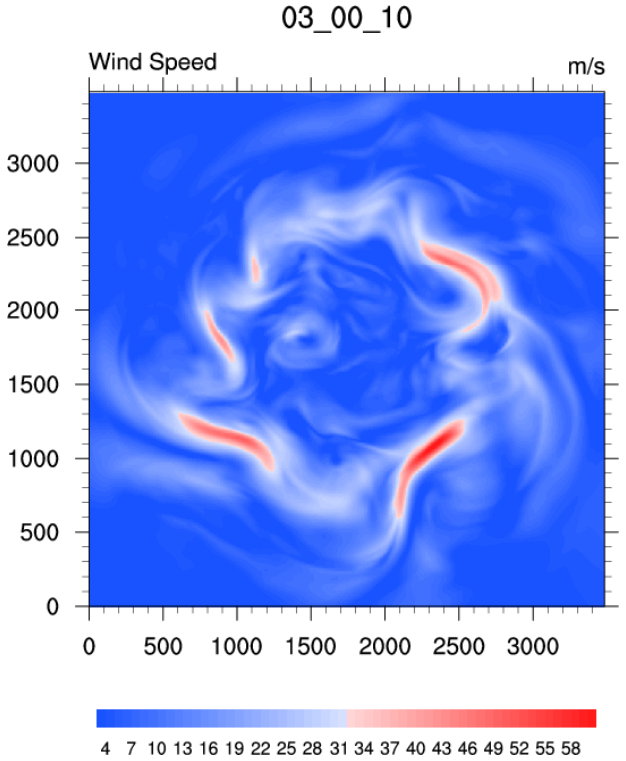
Initial simulation results

N.H.

Real Data Forecast
20km Global WRF
July 22, 2007

S.H.

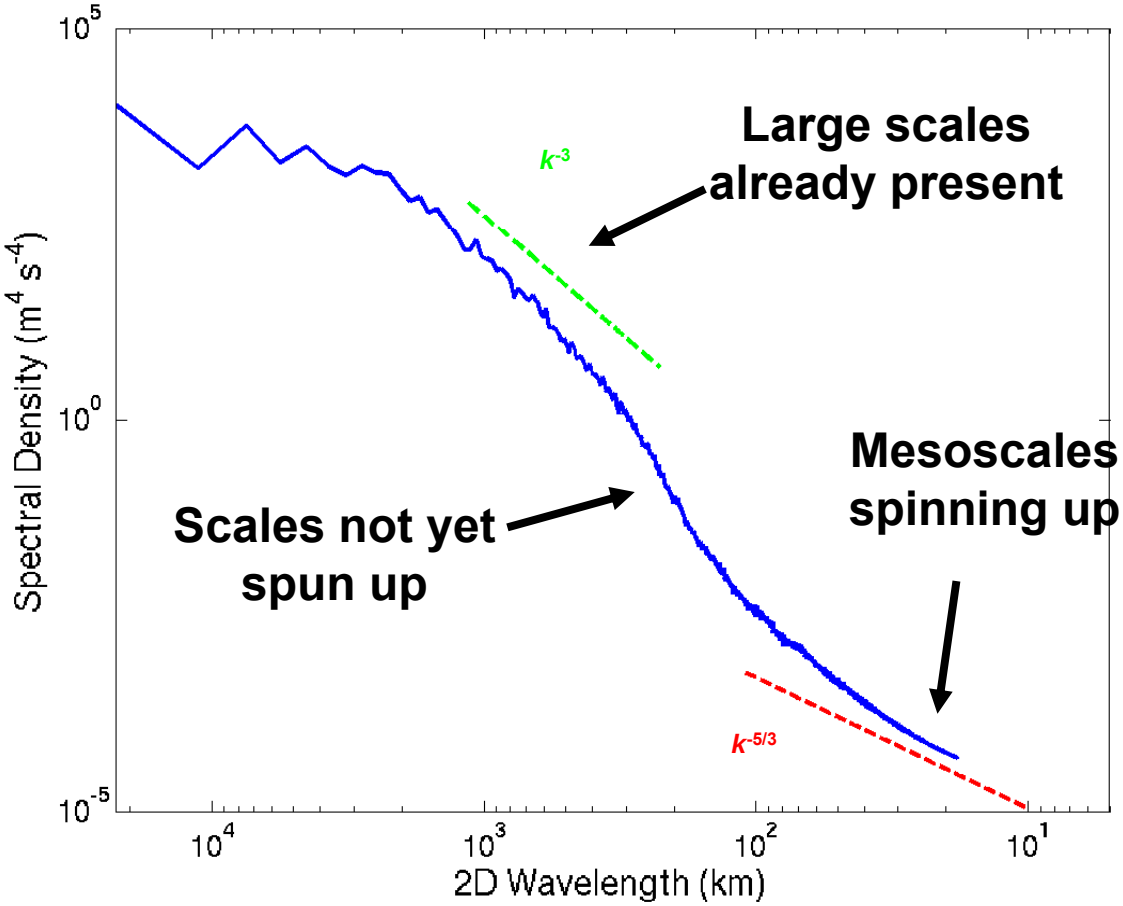




WRF Nature Run
5km (idealized)
Capturing large scale structure already (Rossby Waves)
Small scale features spinning up (next slide)

Kinetic Energy Spectrum

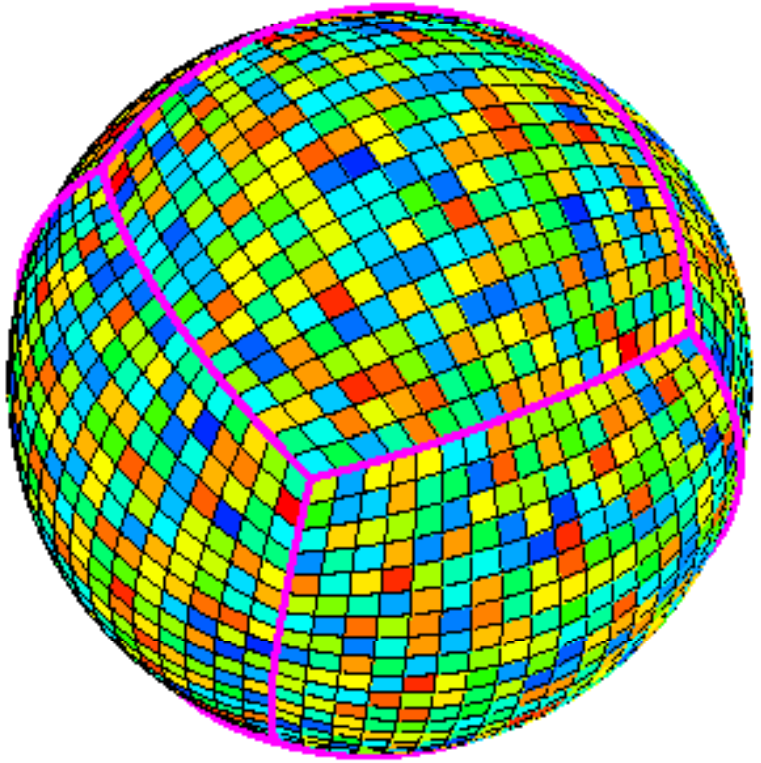
At 3:30 h into the simulation, the mesoscales are still spinning up and filling in the spectrum. Large scales were previously spun up on a coarser grid



High-Frequency Simulations of Global Seismic Wave Propagation

- A seismology challenge: model the propagation of waves near 1 hz (1 sec period), the highest frequency signals that can propagate clear across the Earth.
- These waves help reveal the 3D structure of the Earth's “enigmatic” core and can be compared to seismographic recordings.
- We reached 1.84 sec. using 32K cpus of Ranger (a world record) and plan to reach 1 hz using 62K on Ranger

- **The Gordon Bell Finals Team**: Laura Carrington, Dimitri Komatitsch, Michael Laurenzano, Mustafa Tikir, David Michéa, Nicolas Le Goff, Allan Snavely, Jeroen Tromp

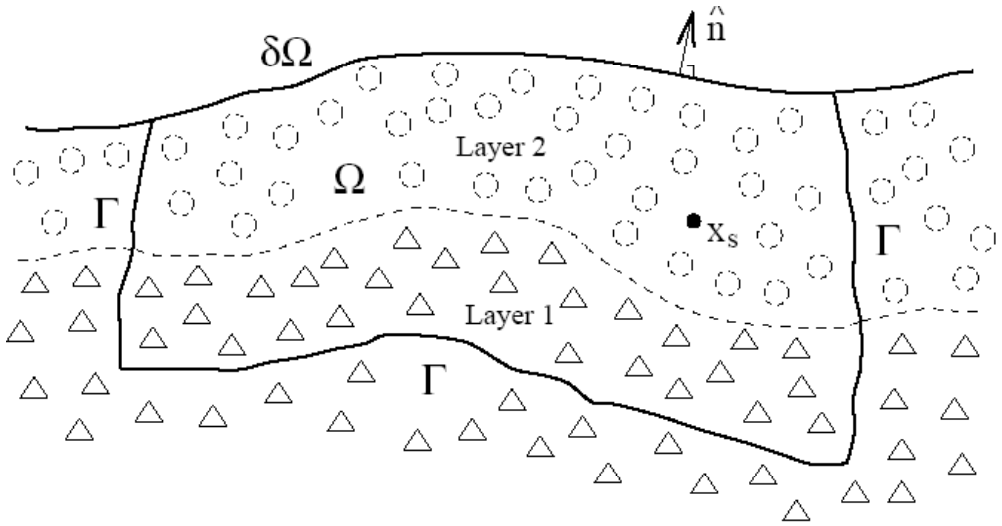


The cubed-sphere mapping of the globe represents a mesh of 6 x 182 = 1944 slices.

Why do it?

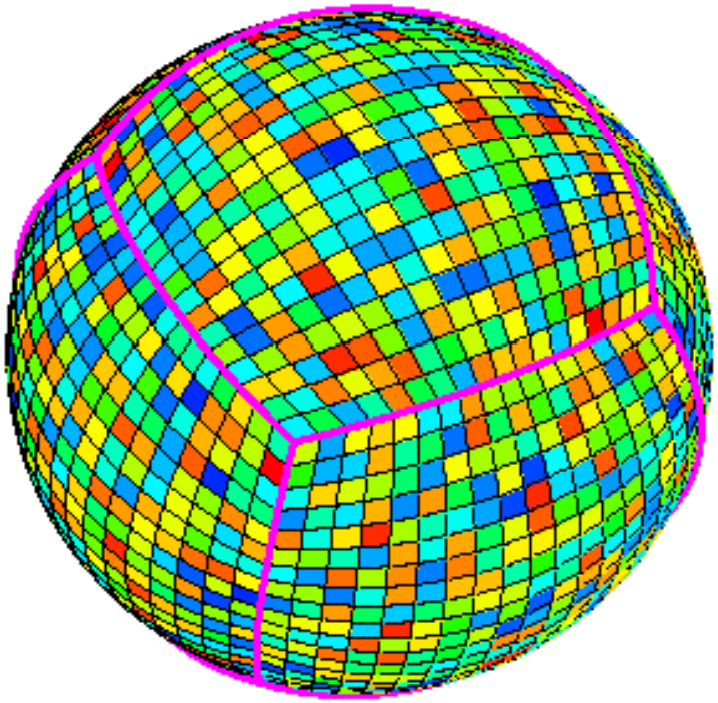
- **These waves at periods of 1 to 2 seconds, generated when large earthquakes (typically of magnitude 6.5 or above) occur in the Earth, help reveal the detailed 3D structure of the Earth's deep interior, in particular near the core-mantle boundary (CMB), the inner core boundary (ICB), and in the enigmatic inner core composed of solid iron. The CMB region is highly heterogeneous with evidence for ultra-low velocity zones, anisotropy, small-scale topography, and a recently discovered post-perovskite phase transition.**

A Spectral Element Method (SEM)



Finite Earth model with volume Ω and free surface ∂Ω.
An artificial absorbing boundary Γ is introduced if the physical model is for a "regional" model

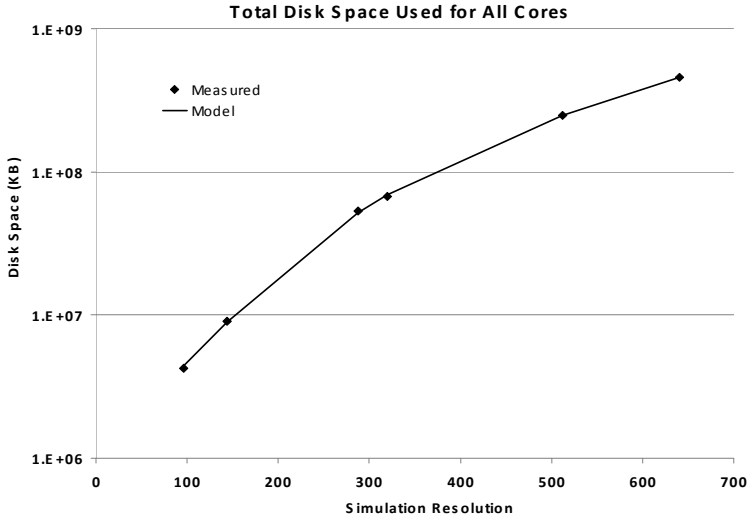
Cubed sphere



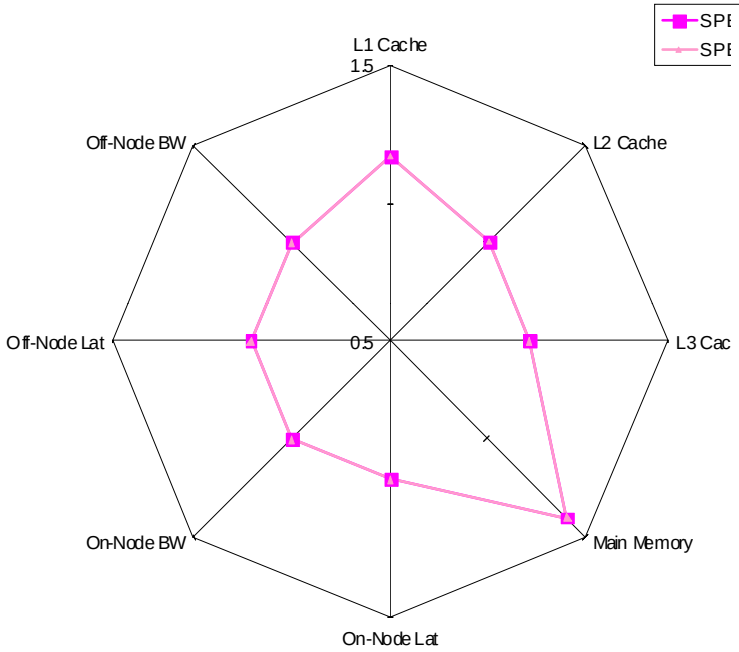
Split the globe into 6 chunks, each of which is further subdivided into n^2 mesh slices for a total of 6 x n^2 slices,
The work for the mesher code is distributed to a parallel system by distributing the slices

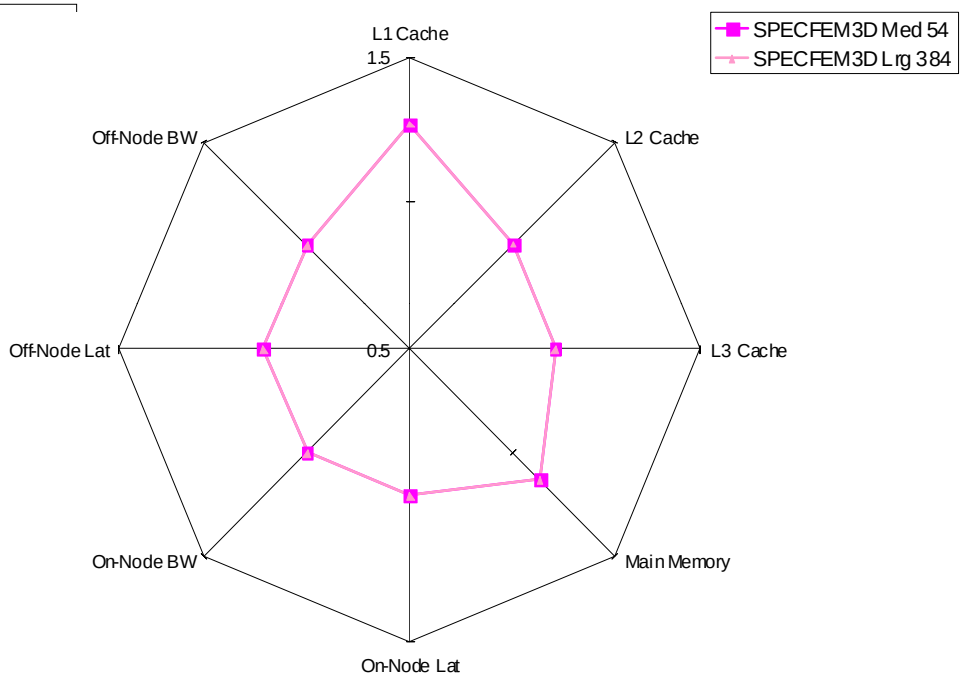
Model guided sanity checking

- **Performance model predicted that to reach 2 seconds 14 TB of data would have to be transferred between the mesher and the solver; at 1 second, over 108 TB**
- **So the two were merged**



Model guided tuning





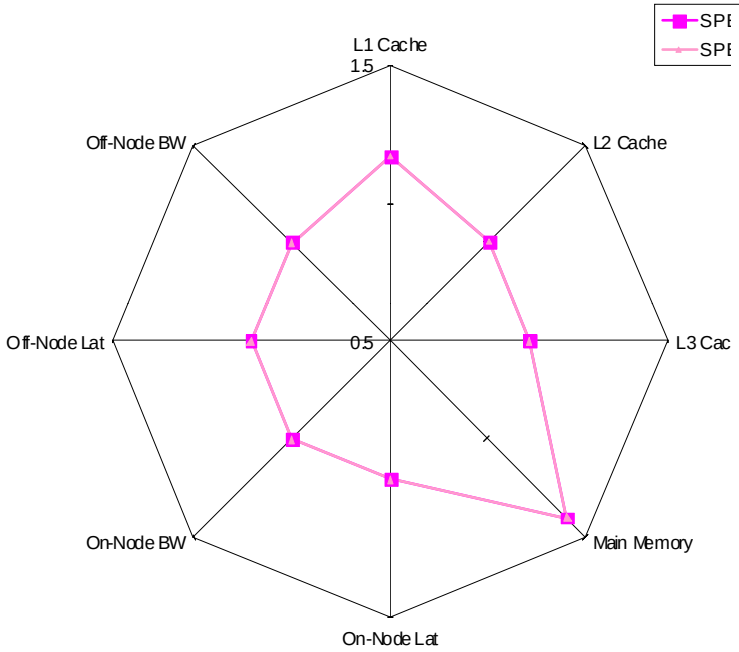
Pre-tune

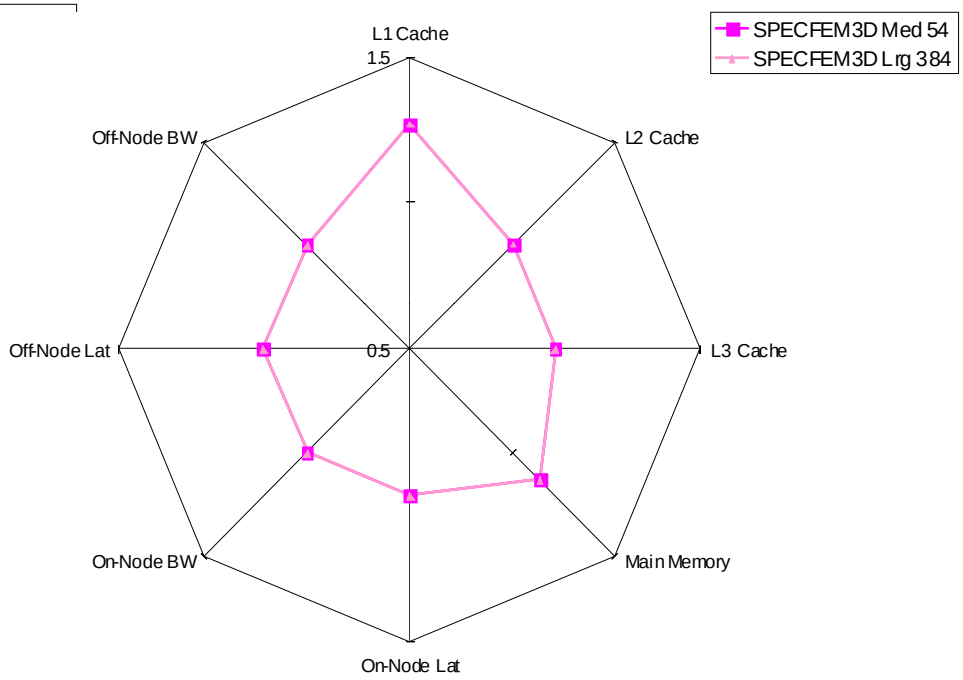
Post tune

Improving locality

- **To increase spatial and temporal locality** for the global access of the points that are common to several elements, the order in which we access the elements can then be optimized. The goal is to find an order that minimizes the memory strides for the global arrays.
- **We used the classical reverse Cuthill-McKee algorithm,** which consists of renumbering the vertices of a graph to reduce the bandwidth of its adjacency matrix**.**

Model guided tuning





Pre-tune

Post tune

Results

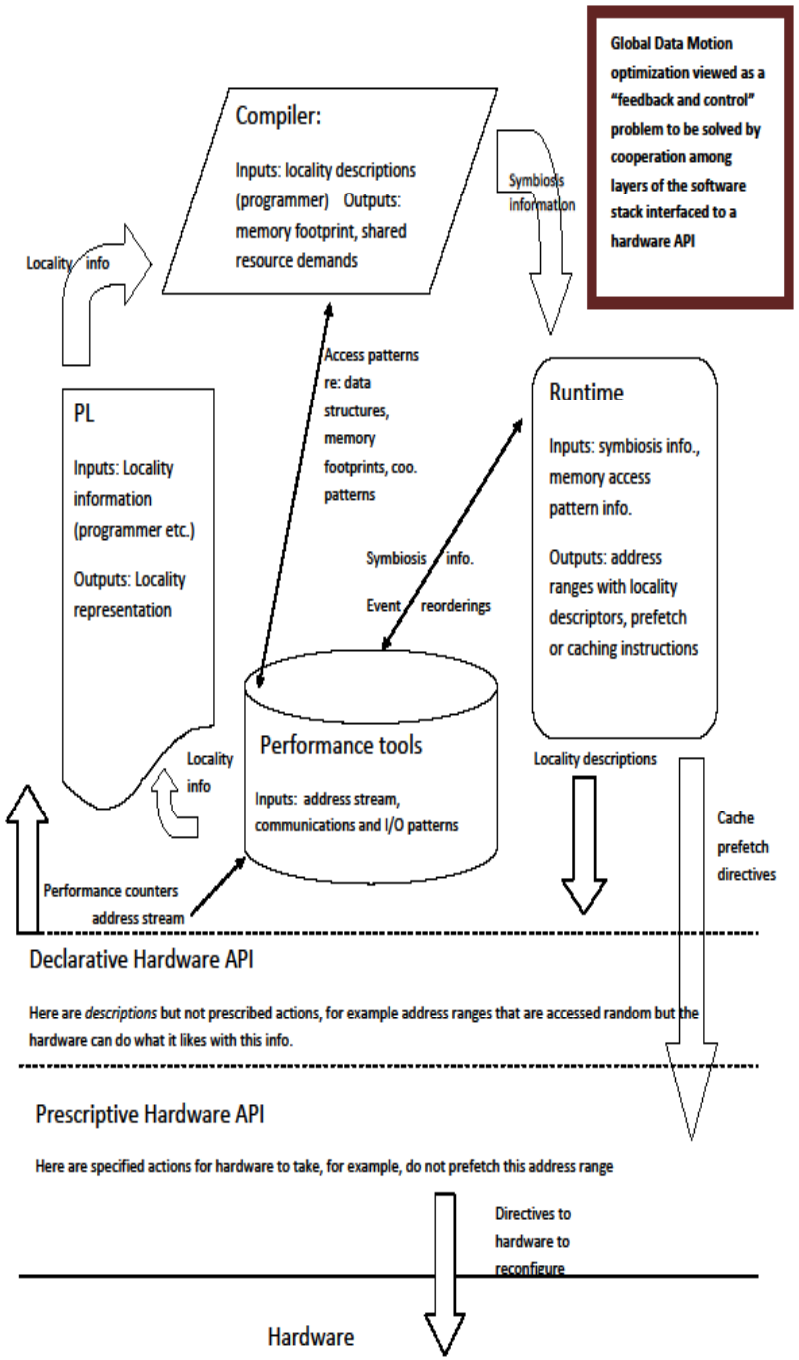
- **Simulation of an earthquake in Argentina** was run successively on 9,600 cores (12.1 Tflops sustained), 12,696 cores (16.0 Tflops sustained), and then 17,496 cores of NICS's **Kraken** system. The 17K core run sustained 22.4 Tflops and had a seismic period length of **2.52 seconds**; temporarily a new resolution record.
- On the **Jaguar** system at ORNL we simulated the same event and achieved a seismic period length of 1.94 seconds and a sustained **35.7 Tflops** (our current flops record) using 29K cores.
- On the **Ranger** system at TACC the same event achieved a seismic period length **1.84 seconds** (our current resolution record) with sustained 28.7 Tflops using 32K cores**.**

Why is tuning such a challenge?

- **Partly it is intellectually inherently hard but also:**
  - Caches normally have a fixed line size. This means they implicitly fetch say 8 or 16 contiguous elements of memory at a time.
  - A hardware prefetcher may monitor the address stream and try to guess which data will be accessed next, then fetch it. Frequently it guesses
  - A prime example of the runtime system moving pages is embodied in the SGI ALTIX system. Very quirky.

More fighting systems components that don't talk to each other

- Compilers block loops for cache, or choose to fuse (or not) contiguous loops based on hard-wired cache size parameters and don't tell you what they did
- The most common HPC programming paradigm of today, C or FORTRAN + MPI, does not provide explicit means for programmers to express memory-hierarchy locality (one cannot express for example whether a data structure should or should not be cached)
- Threads cause fully as many efficiency problems as they solve on today's machines.


Global Data Motion optimization viewed as a "feedback and control" problem to be solved by cooperation among layers of the software stack interfaced to a hardware API
Compiler:
Inputs: locality descriptions (programmer) Outputs: memory footprint, shared resource demands
Symbiosis information
Locality info
Access patterns re: data structures, memory footprints, coo. patterns
PL
Inputs: Locality information (programmer etc.)
Outputs: Locality representation
Runtime
Inputs: symbiosis info., memory access pattern info.
Outputs: address ranges with locality descriptors, prefetch or caching instructions
Symbiosis info.
Event reorderings
Performance tools
Inputs: address stream, communications and I/O patterns
Locality info
Locality descriptions
Cache prefetch directives
Performance counters address stream
Declarative Hardware API
Here are descriptions but not prescribed actions, for example address ranges that are accessed random but the hardware can do what it likes with this info.
Prescriptive Hardware API
Here are specified actions for hardware to take, for example, do not prefetch this address range
Directives to hardware to reconfigure
Hardware