Climate Computing: The State of Play

ECMWF HPC Workshop
Keynote talk

V. Balaji
with contributions from many

NOAA/GFDL and Princeton University

29 October 2014

Outline

Outline

Evolution of Algorithms for the Ocean Free Surface

V. Balaji
SGI/GFDL

ECMWF TeraComputing Workshop
19 November 1998

1

Teracomputing workshop conclusion: explicit methods parallelize

Conclusions

Low order models benefit from being formulated in terms of a series of balance assumptions that reduce the number of prognostic equations. In the limit, atmospheric and oceanic dynamics could in principle be formulated in terms of a single prognostic variable, the *potential vorticity*, and a balance model that allows us to recover the mass and momentum fields from it.

As we move to higher resolutions, it becomes less easy to justify balance models, and models tend to solve more independent prognostic equations.

Happily, these are also the algorithms that lend themselves best to parallel formulation.

In short...

"Teracomputing" workshop conclusion

Nature does not vectorize, it parallelizes!

22

"Realizing Teracomputing": high-level expressions of parallelism

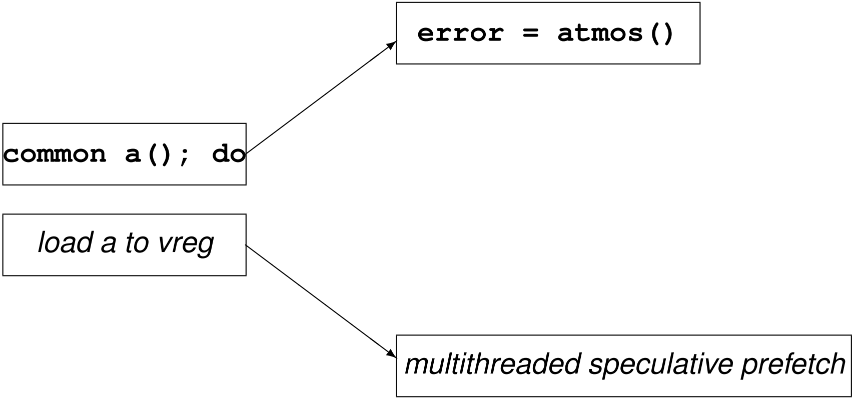
Parallel numerical kernels

$$\frac{\eta^{n+1} - \eta^{n}}{\Delta t} = -H(\nabla \cdot \mathbf{u})^{n} \quad (1)$$

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^{n}}{\Delta t} = -g(\nabla \eta)^{n+1} + f\mathbf{k} \times \left(\frac{\mathbf{u}^{n+1} + \mathbf{u}^{n}}{2}\right) + \mathbf{F} \quad (2)$$

```
program shallow_water
  type(scalar2D) :: eta(0:1)
  type(hvector2D) :: utmp, u, forcing
  integer tau=0, taup1=1
...
  f2 = 1./(1.+dt*dt*f*f)
  do l = 1,nt
     eta(taup1) = eta(tau) - (dt*h)*div(u)
     utmp = u - (dt*g)*grad(eta(taup1)) + (dt*f)*kcross(u) + dt*forcing
     u = f2*( utmp + (dt*f)*kcross(utmp) )
     tau   = 1 - tau
     taup1 = 1 - taup1
  end do
end program shallow_water
```

The jaws of code complexity



The complete memory model

All the above mechanisms of sharing data can be combined into a single uniform memory model, where an object has two states **`READ`** and **`WRITE`**. There are three types of access operations, **`request`**, **`require`**, and **`release`**.

Access	State	MPI	shmem	MPI-2	Threads
Request	READ	irecv	status=WAIT	post	WRLOCK?
Require	READ	wait	wait(status=OK) unbuffer put(put=OK)	wait	wait(!WRLOCK) lock RDLOCK
Release	READ				unlock RDLOCK
Request	WRITE	isend	wait(put=OK) put(buffer) put=WAIT	start put	RDLOCK?
Require	WRITE	wait	fence put(status=OK)	complete	wait(!RDLOCK) lock WRLOCK
Release	WRITE				unlock WRLOCK

MPP, ShmemPP: Parallel Processing for Sceptics

V. Balaji
SGI/GFDL

29 July 1998

Commodity clusters: Beowulf

20 Years of Beowulf: Workshop to Honor Thomas Sterling's 65th Birthday

- About
- Call For Papers
- Registration
- Program Committee
- Venue
- Hotel
- Directions
- Agenda
- Contact

About the Workshop

This workshop will mark the 20th anniversary of the introduction of commodity (AKA Beowulf) clusters, an architectural approach to creating parallel computers using mostly or entirely commodity components and open source system software. The initial target of the Beowulf cluster project was inexpensive, small to moderate parallel computing platforms; the Beowulf approach was extremely successful and adopted worldwide by teams ranging from high-school students to senior scientists. The Beowulf approach is now the basis of most of the world's most powerful computers as well.

The workshop will also celebrate the 65th birthday of Thomas Sterling, who has made major contributions over his career (so far), including playing a key role in conceiving and implementing commodity cluster computing (aka Beowulf), HPC architecture, run time systems, and exascale systems.

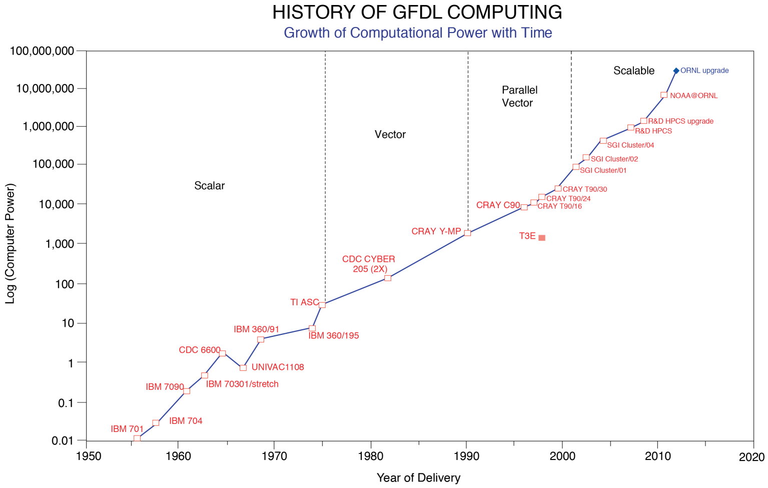
Dates

October 13-14, 2014

sighpc

`http://crest.iu.edu/beowulf14/`

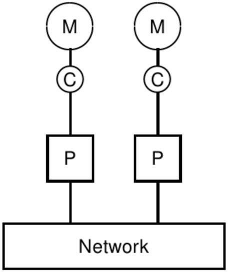
History of GFDL Computing

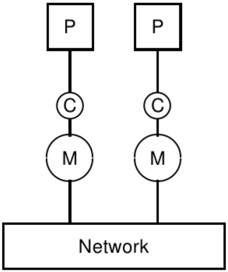


Courtesy Brian Gross, NOAA/GFDL.

The commodity cluster era

- Speedups from Moore's Law: transistor density doubles every 18 months.
- Dennard scaling: power density is constant as fab shrinks.
- Moore's Law and Dennard scaling have both reached end of life!
- Networks and I/O still ripe for improvement.





Outline

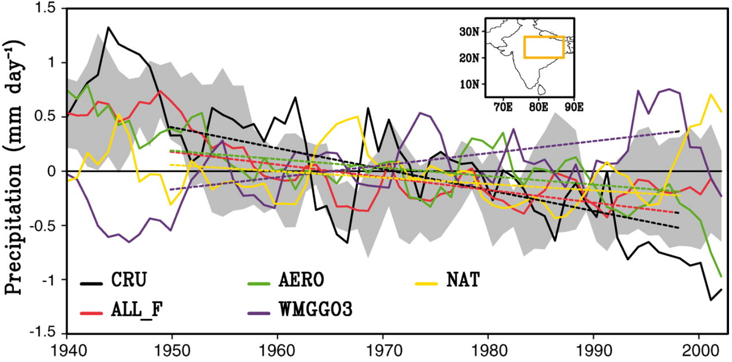
Climate modeling, a computational profile

- Intrinsic variability at all timescales from minutes to millennia; all space scales from microbes to megacontinents.
- physics components have predictable data dependencies associated with grids;
- algorithms generally possess weak scalability;
- Adding processes and components improves scientific understanding;
- ... this complexity implies lots of diagnostic I/O;
- New physics and higher process fidelity at higher resolution;
- thus, coupled multi-scale multi-physics modeling;
- Ensemble methods to sample uncertainty (ICEs, PPEs, MMEs...)

In sum, climate modeling requires long-term integrations of weakly-scaling, I/O and memory-bound models of enormous complexity.

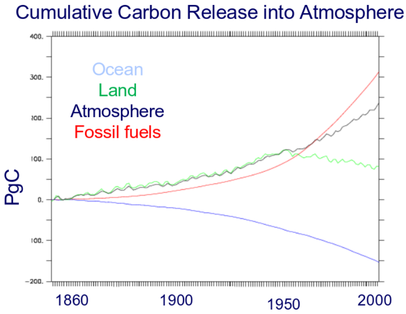
Outline

Aerosol indirect effects weaken South Asian monsoon



Cloud-aerosol feedbacks induce a weakening of the Indian monsoon (Figure courtesy Bollasina et al., **Science** 2011).

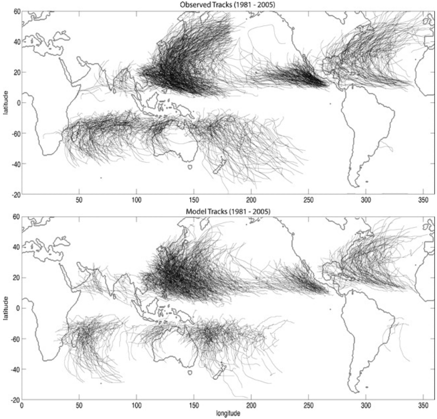
Carbon sources and sinks



- Land carbon fluxes dominant before 1960; then trend changes sign.
- Fossil fuels dominant contemporary source.
- Ocean uptake scales with pCO_2.

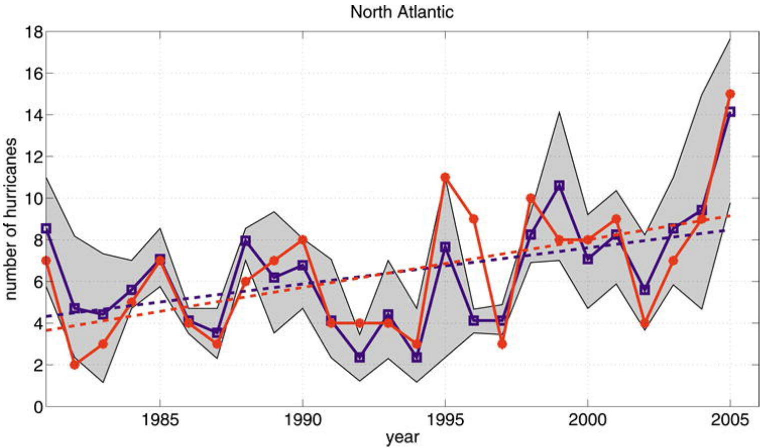
Figure courtesy Ron Stouffer, NOAA/GFDL; pre-publication.

Hurricane statistics from global high-resolution atmosphere models



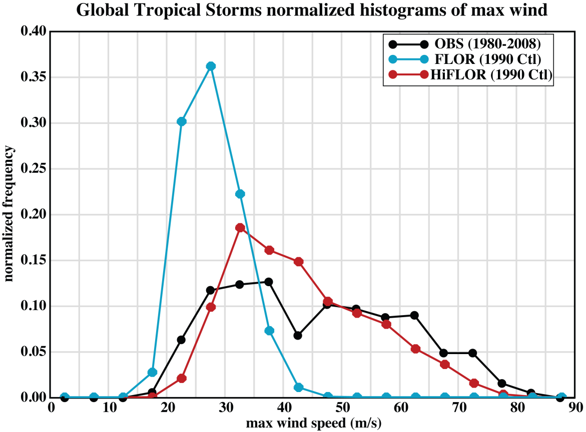
Observed and modeled hurricane tracks from 1981-2005 in a global 50 km (C180) atmospheric model forced by observed SSTs. (Figure 3 from Zhao and Held 2009).

Interannual variability of hurricane frequency



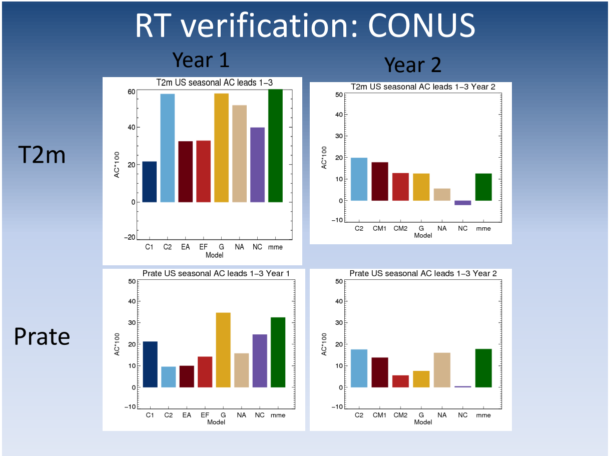
Interannual variability of W. Atlantic hurricane number from 1981-2005 in the C180 runs. (Figure 7 from Zhao and Held 2009).

"TC-permitting" models get better with resolution



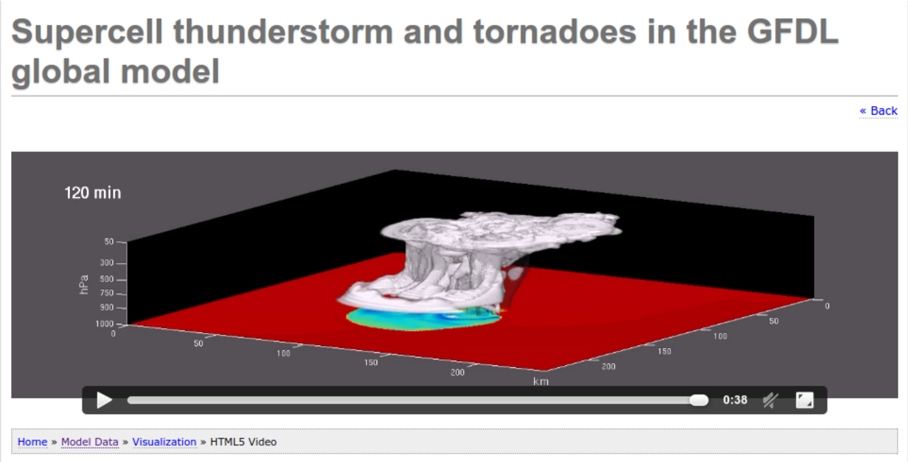
Intensity distribution improves with resolution. Figure courtesy Gabe Vecchi.

"TC-permitting" model FLOR is now used in the NMME



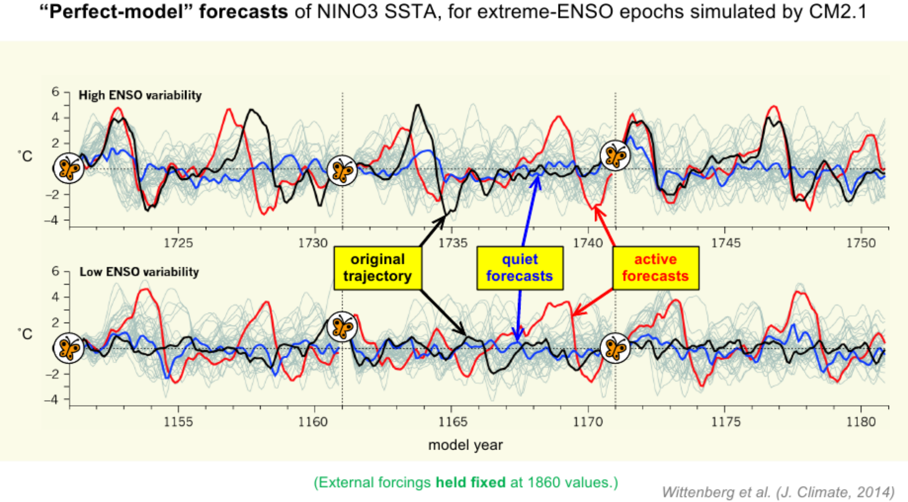
Seasonal forecasting product used in NMME and SPECS. Figure courtesy Gabe Vecchi.

Towards global cloud-resolving models



Variable-resolution grid in the FV3 model, courtesy S-J Lin.

ENSO modulation: is it decadally predictable?



Effects of the proverbial "flap of a butterfly's wing"...

Outline

The hardware jungle

Upcoming hardware roadmap looks daunting! GPUs, MICs, DSPs, and many other TLAs…

- Intel straight line: IvyBridge/SandyBridge, Haswell/Broadwell: "traditional" systems with threading and vectors.
- Intel knight's move: Knights Corner, Knights Landing: MICs, thread/vector again, wider in thread space.
- Hosted dual-socket systems with GPUs: SIMD co-processors.
- BG/Q: CPU only with hardware threads, thread and vector instructions. No followon planned.
- ARM-based systems coming. (e.g with DSPs).
- FPGAs? some inroads in finance.
- Specialized processors: Anton for molecular dynamics, GRAPE for astrophysics.

The software zoo

Exascale using nanosecond clocks implies billion-way concurrency!
It is unlikely that we will program codes with $10^6 - 10^9$ MPI ranks: it will be MPI+X. Solve for X ...

- CUDA and CUDA-Fortran: proprietary for NVIDIA GPUs. Invasive and pervasive.
- OpenCL: proposed standard, not much penetration.
- ACC from Portland Group, now a new standard OpenACC.
- Potential OpenMP/OpenACC merging...?
- PGAS languages: Co-Array Fortran, UPC, a host of proprietary languages.

GFDL between jungle and zoo

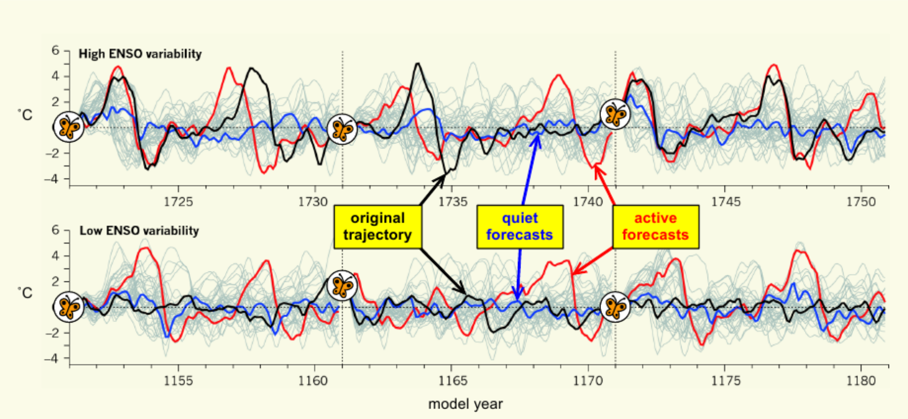
GFDL is taking a conservative approach:

- it looks like it will be a mix of MPI, threads, and vectors.
- Developing a three-level abstraction for parallelism: components, domains, blocks. Kernels work on blocks and must have vectorizing inner loops.
- Recommendation: sit tight, make sure MPI+OpenMP works well, write vector-friendly loops, reduce memory footprint, offload I/O.
- Other concerns:
  - Irreproducible computation
  - Tools for analyzing performance.
  - Debugging at scale.

Recent experience on Titan, Stampede and Mira reaffirm this approach.

ENSO modulation: is it decadally predictable?

“Perfect-model” forecasts of NINO3 SSTA, for extreme-ENSO epochs simulated by CM2.1

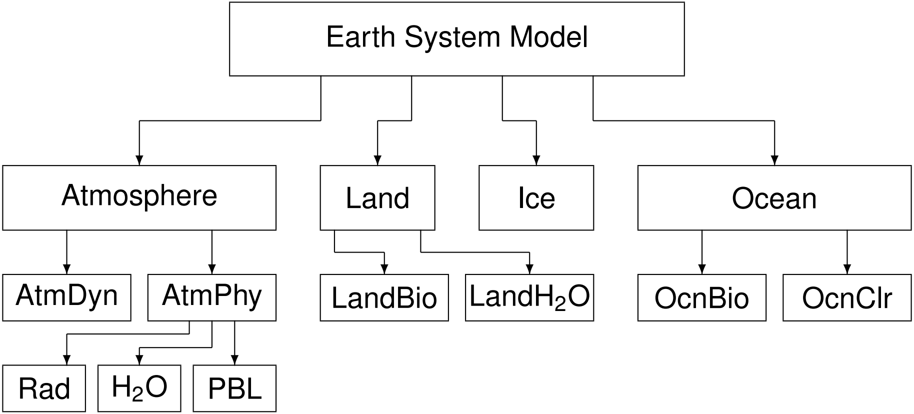


(External forcings **held fixed** at 1860 values.)

Wittenberg et al. (J. Climate, 2014)

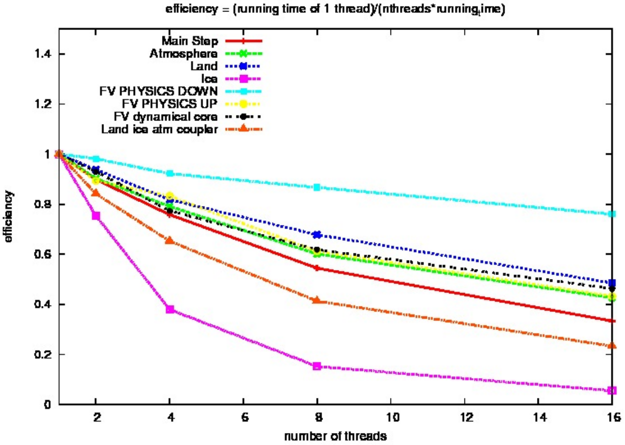
Effects of the proverbial “flap of a butterfly’s wing”...

Earth System Model Architecture



Complexity implies many different instruction sequences; no hotspots.

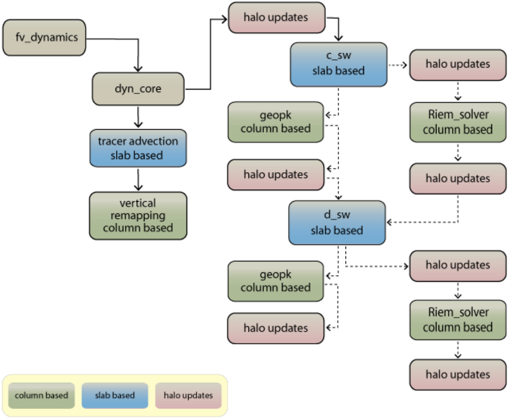
Most of FMS is now threaded/vectorized



CM4 on up to 16 threads on gaea. (Figure courtesy Zhi Liang, 16 Sep 2014)

Vectors (AVX) need to be used with care.

Analysis of dycore architecture for GPU/MIC



Study of code for MPI, threads, vectors. (Chris Kerr, Zhi, Kareem Sorathia (NASA), Duane Rosenberg (ORNL), Eric Dolven (Cray)...)

Blocking the dycore for GPU/MIC

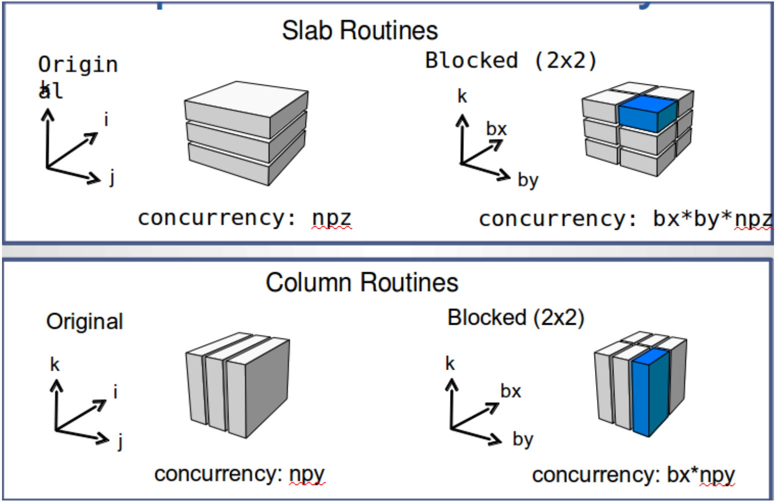


Figure courtesy Kareem Sorathia (NASA). Inner loops on **`i`** are retained for vectorization.

Performance summary: Xeon-SNB vs Xeon-Phi

Phi "speedup" over SNB:

- Overall: 0.73
- Communication: 0.34
- All Computation: 0.86
- Top 4: 0.996

Coding issues:

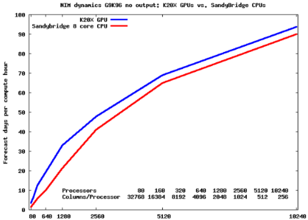
- Vector performance very hard to achieve, even with padding halos for alignment.
- Loop unrolling/stripmining/etc needs to be done by hand.
- Better performance analysis tools needed.

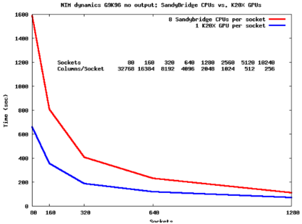
Courtesy Kareem Sorathia, NASA.

Results from NIM icosahedral dycore: SNB vs GPU

NIM Dynamics: GPU versus Intel-SB

- Single source code optimized for CPU, MIC & GPU
  - OpenMP directives for CPU & MIC
  - OpenACC, F2C-ACC for NVIDIA GPU
- 15 KM model, 96 levels, single-precision
  - Strong scaling: 80 - 10240 GPUs
  - GPU 2-3x faster than CPU socket for 8192 columns





Courtesy Mark Govett, NOAA/ESRL.

OpenACC

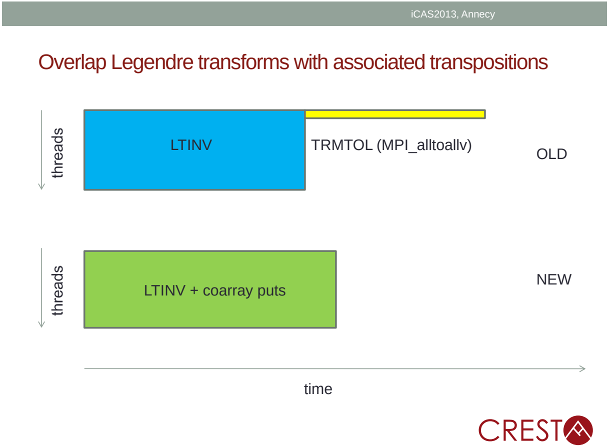
```
!$acc parallel num_gangs(ihe-ips+1) vector_length(64)
!$acc loop gang
    do ipn=ips,ihe
!$acc loop vector
      do k=1,nvl
        flxhi(k) = vnorm(k,edg,ipn)*dp_edg(k,edg,ipn)
```

Can merge gang and vector on same axis:

```
do k = kts,kte
!$acc loop gang vector
     do i = its,ite
       za(i,k) = 0.5*(zq(i,k)+zq(i,k+1))
```

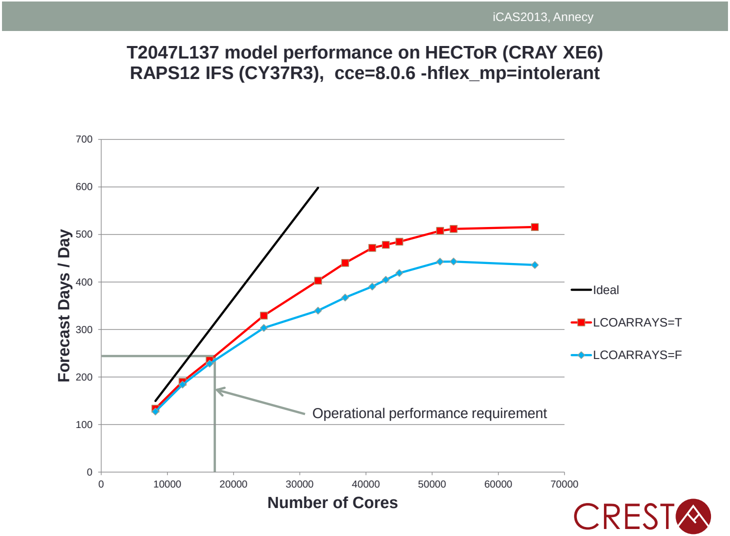
Courtesy Mark Govett, NOAA/ESRL.

ECMWF uses PGAS (Co-Array Fortran)



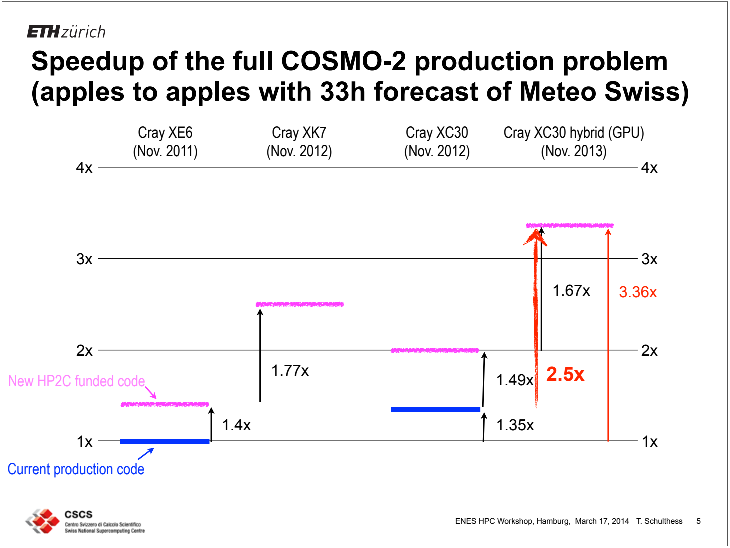
Co-array assignments become one-sided puts from within threaded regions.
Courtesy George Mozdzynski, ECMWF.

CAF results using Cray compiler CCE

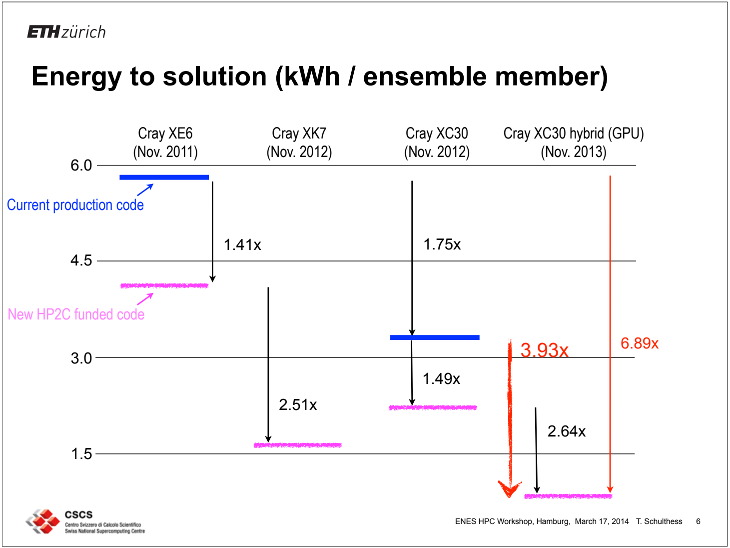


Courtesy George Mozdzynski, ECMWF.

COSMO: NWP production code using GPUs



COSMO: energy to solution



Summary of results in the jungle and zoo

- Billion-way concurrency still a daunting challenge for everyone: no magic bullets anywhere to be found. ECMWF's PGAS approach is interesting, and there is at least one production GPU model.
- GPU/MIC based systems show nominal $\sim$10 increase in flops/socket, but actual performance about 1-2X (thus percent of peak drops from $\sim$10% to $\sim$1%)
- Software investment paid back in power savings (Schulthess).
- More computational intensity needs to be found: to fit 10^{18} op/s within a 1 MW power budget
  - an operation should be 1 pJ: data movement is $\sim$10 pJ to main memory; $\sim$100 pJ on network!
- DARPA: commodity improvements will slow to a trickle within 10 years: go back to specialized computing?
- DOE: double investment in exascale.

GFDL between jungle and zoo

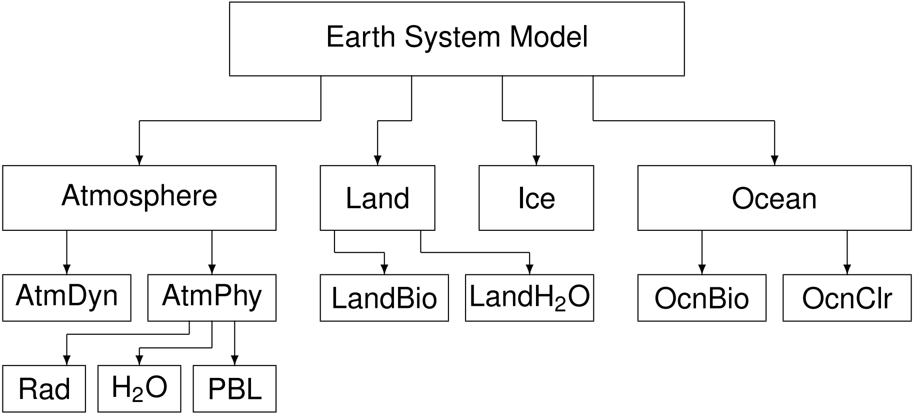
GFDL is taking a conservative approach:

- it looks like it will be a mix of MPI, threads, and vectors.
- Developing a three-level abstraction for parallelism: components, domains, blocks. Kernels work on blocks and must have vectorizing inner loops.
- Recommendation: sit tight, make sure MPI+OpenMP works well, write vector-friendly loops, reduce memory footprint, offload I/O.
- Other concerns:
  - Irreproducible computation
  - Tools for analyzing performance.
  - Debugging at scale.

Recent experience on Titan, Stampede and Mira reaffirm this approach.

Outline

Earth System Model Architecture

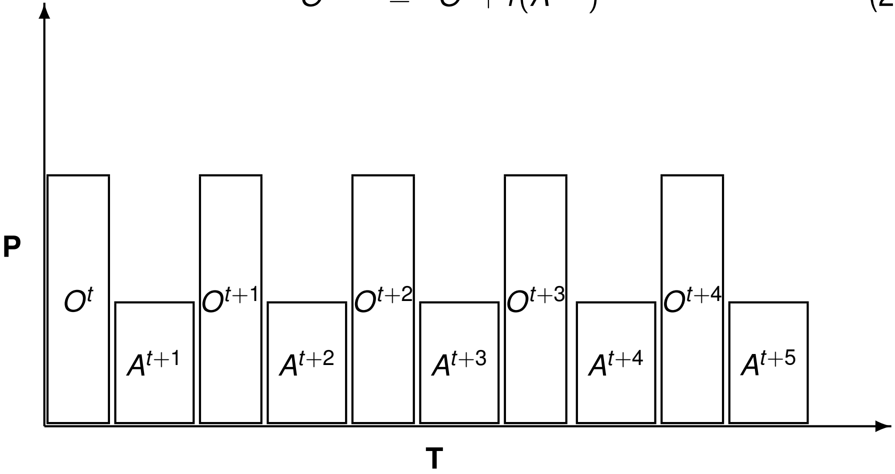


Extending component parallelism to $\mathcal{O}(10)$ requires a different physical architecture!

Serial coupling

Uses a forward-backward timestep for coupling.

$$A^{t+1} = A^t + f(O^t) \quad (1)$$
$$O^{t+1} = O^t + f(A^{t+1}) \quad (2)$$

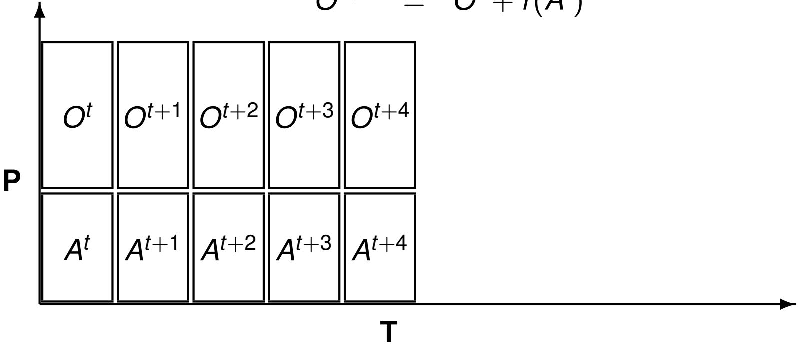


Concurrent coupling

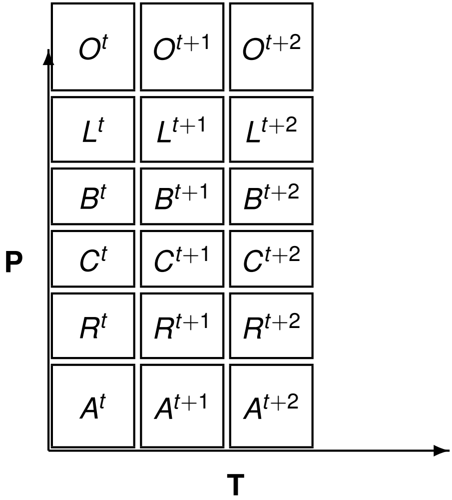
This uses a forward-only timestep for coupling. While formally this is unconditionally unstable, the system is strongly damped*. Answers vary with respect to serial coupling, as the ocean is now forced by atmospheric state from Δt ago.

$$A^{t+1} = A^t + f(O^t) \qquad (3)$$

$$O^{t+1} = O^t + f(A^t) \qquad (4)$$

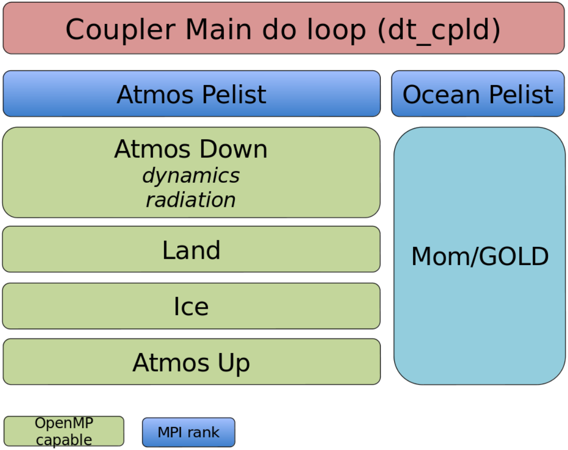


Massively concurrent coupling



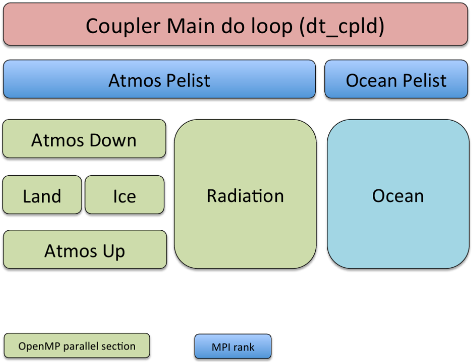
Components such as radiation, PBL, ocean biogeochemistry, each could run with its own grid, timestep, decomposition, even hardware. Coupler mediates state exchange.

Traditional coupling sequence



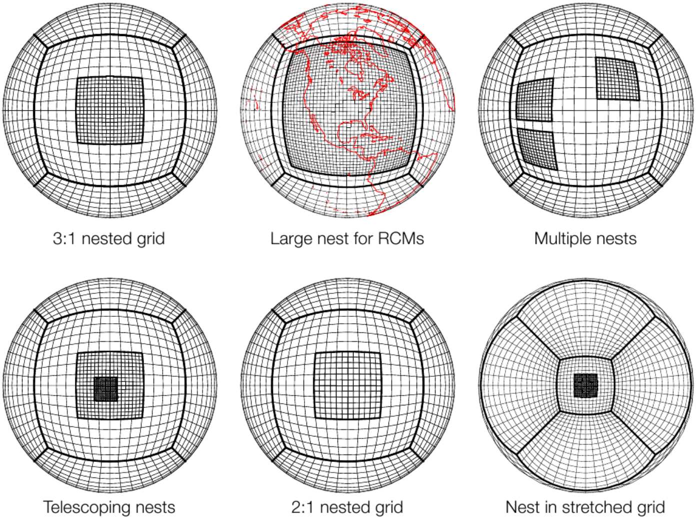
Radiation timestep much longer than physics timestep.
(Figure courtesy Rusty Benson, NOAA/GFDL).

Concurrent radiation coupling sequence



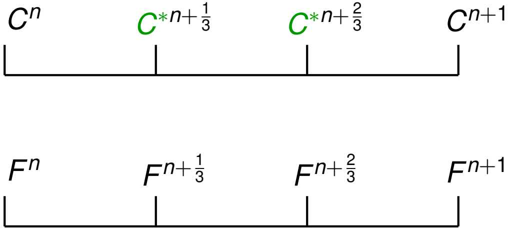
Physics and radiation share memory. Radiation executes on physics timestep from lagged state. Threads can be dynamically reassigned between components. This model has completed AMIP runs and further analysis is underway.
(Figure courtesy Rusty Benson, NOAA/GFDL).

Nested grids



Concurrent two-way nesting

Typical nesting protocols force serialization between fine and coarse grid timestepping, since the C^* are estimated by interpolating between C^n and C^{n+1}.



We enable concurrency by instead estimating the C^* by extrapolation from C^{n-1} and C^n, with an overhead of less than 10%. (See Harris and Lin 2012 for details.)

Outline

Multi-model ensembles for climate projection

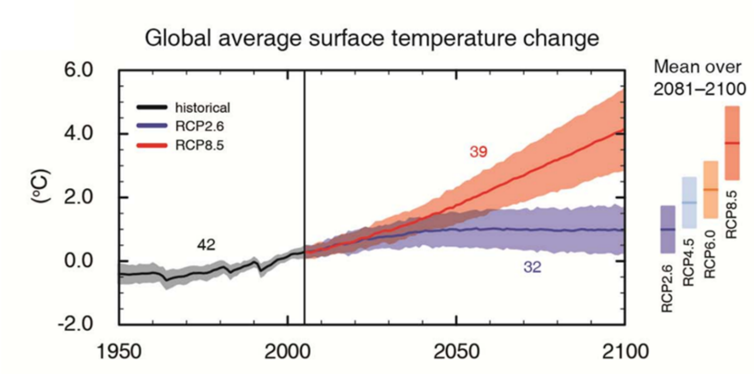
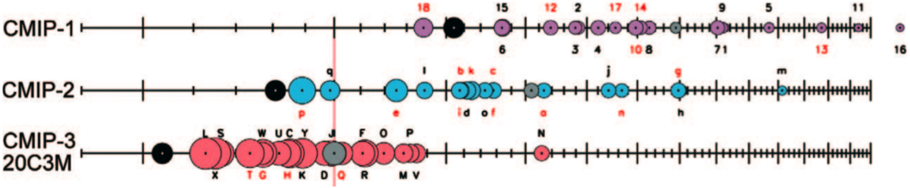


Figure SPM.7 from the IPCC AR5 Report. 20th century warming cannot be explained without greenhouse gas forcings.

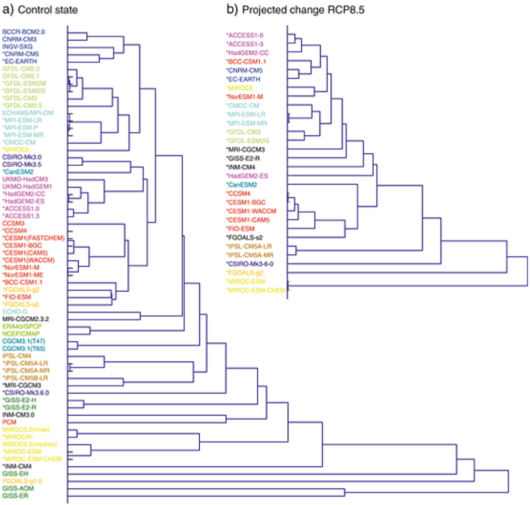
Multi-model ensembles to overcome “structural uncertainty”



Reichler and Kim (2008), Fig. 1: compare models’ ability to simulate 20th century climate, over 3 generations of models.

- Models are getting better over time.
- The ensemble average is better than any individual model.
- Improvements in understanding percolate quickly across the community.

Genealogy of climate models



There is a close link between “genetic distance” and “phenotypic distance” across climate models (Fig. 1 from Knutti et al, GRL, 2013).

NRC Report on “Advancing Climate Modeling”

The 2012 NRC Report “A National Strategy for Advancing Climate Modeling” (Google for URL...) made several recommendations:

- Structural uncertainty: key issue to be addressed with common modeling experiments: maintain model diversity while using common infrastructure to narrow the points of difference.
- Global data infrastructure as critical infrastructure for climate science: data interoperability, common software requirements.
- “Nurture” at least one unified weather-climate effort: NWP methods to address climate model biases; climate runs to address drift and conservation in weather models.
- Forum to promote shared infrastructure: identify key scientific challenges, design common experiments, set standards for data interoperability and shared software.

Real model performance: some considerations

- Productions runs may be configured for capability (minimizing time to solution or SYPD) or capacity (minimizing allocation or CHSY).
- Computing resources can be applied to resolution or complexity: what is a good measure of model complexity?
- ESM architecture governs component concurrency: need to measure load balance and coupler cost.
- Codes are memory-bound: locate bloat (memory copies by user or compiler).
- Models configured for scientific analysis bear a significant I/O load (can interfere with optimization of computational kernels). Data intensity (GB/CH) is a useful measure for designing system architecture.
- Actual SYPD tells you if you need to devote resources to system and workflow issues rather than optimizing code.

Analysis of several GFDL models

- Measure overall computation cost for capability (Speed) or capacity (Throughput) configurations.
- Measure complexity as number of prognostic variables in the model. (There may be better measures based on cluster coefficients, etc.)
- Measure coupler cost and load imbalance separately.
- Measure memory bloat as actual memory (resident set size) compared to ideal memory (number of variables $\times$ data domain size).
- Measure I/O load by rerunning model with diagnostics off. (input files and restart files are considered an unavoidable cost and aren't counted here.)
- Measure actual SYPD for a complete run (from when you typed run to when the last history file was archived).

Land and Ice components are ignored in this analysis.

Analysis of GFDL models: results

Model	Resolution	Cmplx.	SYPD	CHSY	Coupler	Load Imb.	I/O	MBloat	ASYPD
CM2.6 S	A0.5L32 O0.1L50	18	2.2	212,465	5.71%	20%		12%	1.6
CM2.6 T	A0.5L32 O0.1L50	18	1.1	177,793	1.29%	60%	24%	12%	0.4
CM2.5 T	A0.5L32 O0.25L50	18	10.9	14,327	17%	0%			6.1
FLOR T	A0.5L32 O1L50	18	17.9	5,844	0%	57%	5.1%	31%	12.8
CM3 T	A2L48 O1L50	124	7.7	2.974	0.5%	41%	14.76%	3%	4.9
ESM2G S	A2L24 O1L50	63	36.5	279	8.91%	1%		34%	25.2
ESM2G T	A2L24 O1L50	63	26.4	235	2.63%	22%		34%	11.4

- More details are available (layout on MPI/thread, aggregate I/O per CH or SD, platform, optimization, cost per component...)
- Is this a basis for a cross-model comparison of performance (CPMIP, anyone?) for a common understanding of the roadblocks to performance?

Preliminary cross-model comparisons

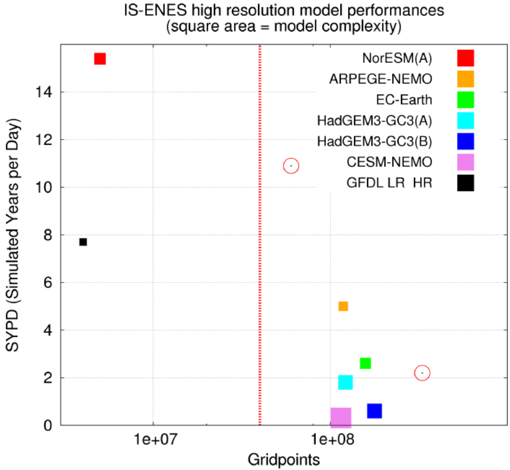


Figure courtesy Eric Maisonnave, Joachim Biercamp, Giovanni Aloisio and others on the ISENES2 team.

Preliminary cross-model comparisons: layout

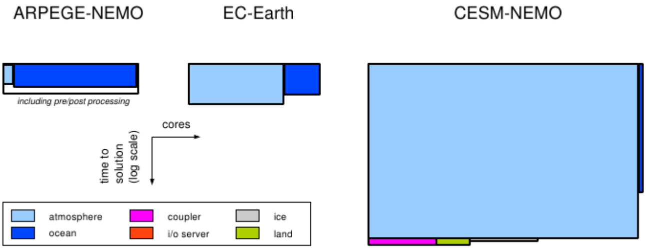
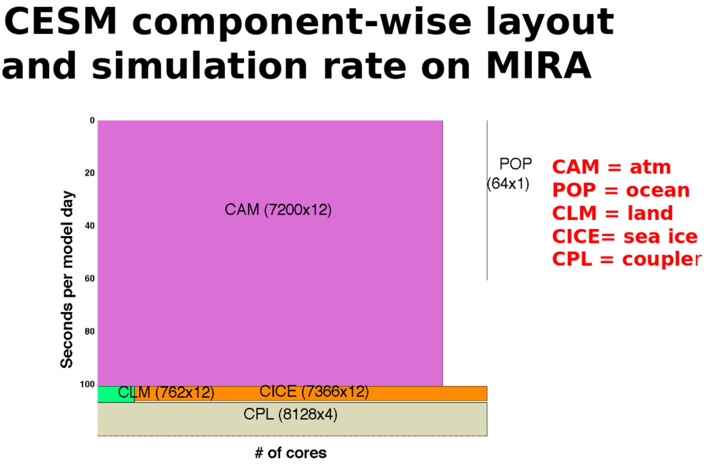


Figure 2: Parallelism and execution time (inverse of SYPD) for three of the participating ESMs

Figure courtesy Eric Maisonnave, Joachim Biercamp, Giovanni Aloisio and others on the ISENES2 team.

CESM on Mira



Courtesy John Dennis and Rich Loft, NCAR. 0.25°atmosphere, 1°ocean on 32k cores of Mira at ∼2 SYPD.

Conclusions

- The commodity computing era has taken us from the von Neumann model to the "sea of functional units" (Kathy Yelick's phrase). Not easy to understand, predict or program performance.
- The "free lunch" decade encouraged us to indulge in very abstract programming, and now they've come to take away your plates.
- The "component" abstraction still may let us extract some benefits out of the machines of this era:
  - sharing of the wide thread space.
  - distribute components among heterogeneous hardware?
- Can we approach models as experimental biological systems? (single organism or "cell line" not exactly reproducible; only the ensemble is.)
- Radically new computing paradigms – neuromorphic, biological, quantum – are several acquisition cycles away.
- The NWP and climate communities are all in the same boat: greater cooperation is advisable (NRC Report 2012).