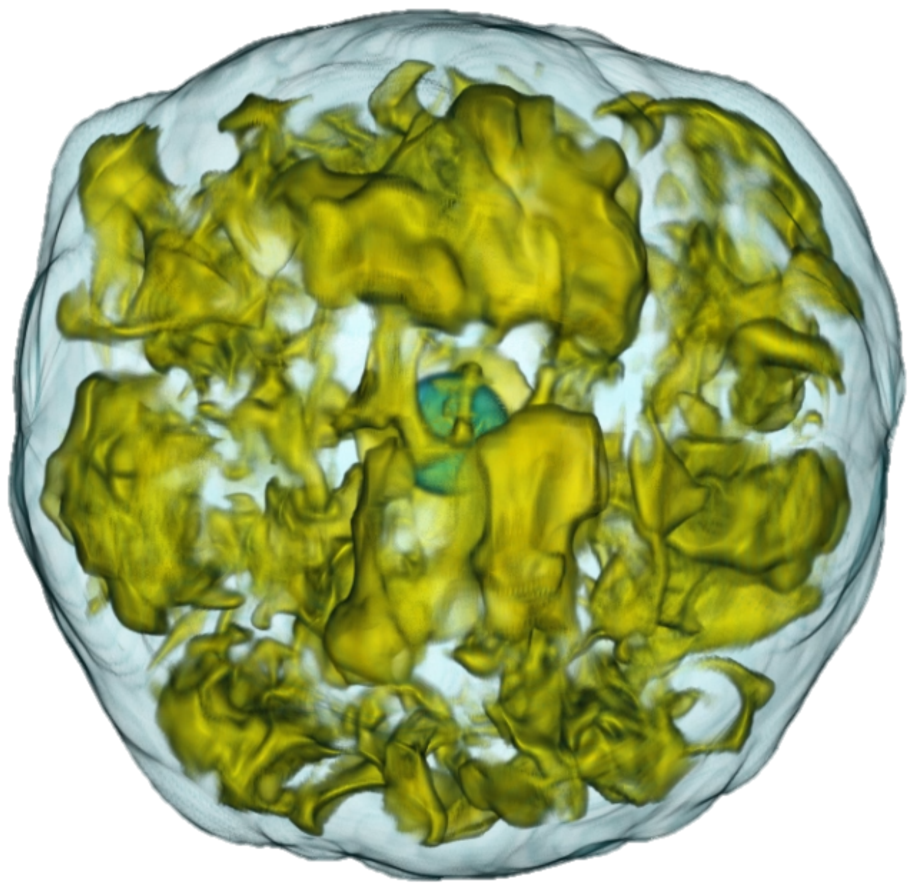
Mapping Multi-Physics Problems to Today's Architectures



ORNL is managed by UT-Battelle
for the US Department of Energy

Bronson Messer

Scientific Computing
& Theoretical
Astrophysics Groups

Oak Ridge National
Laboratory

&

Department of
Physics & Astronomy

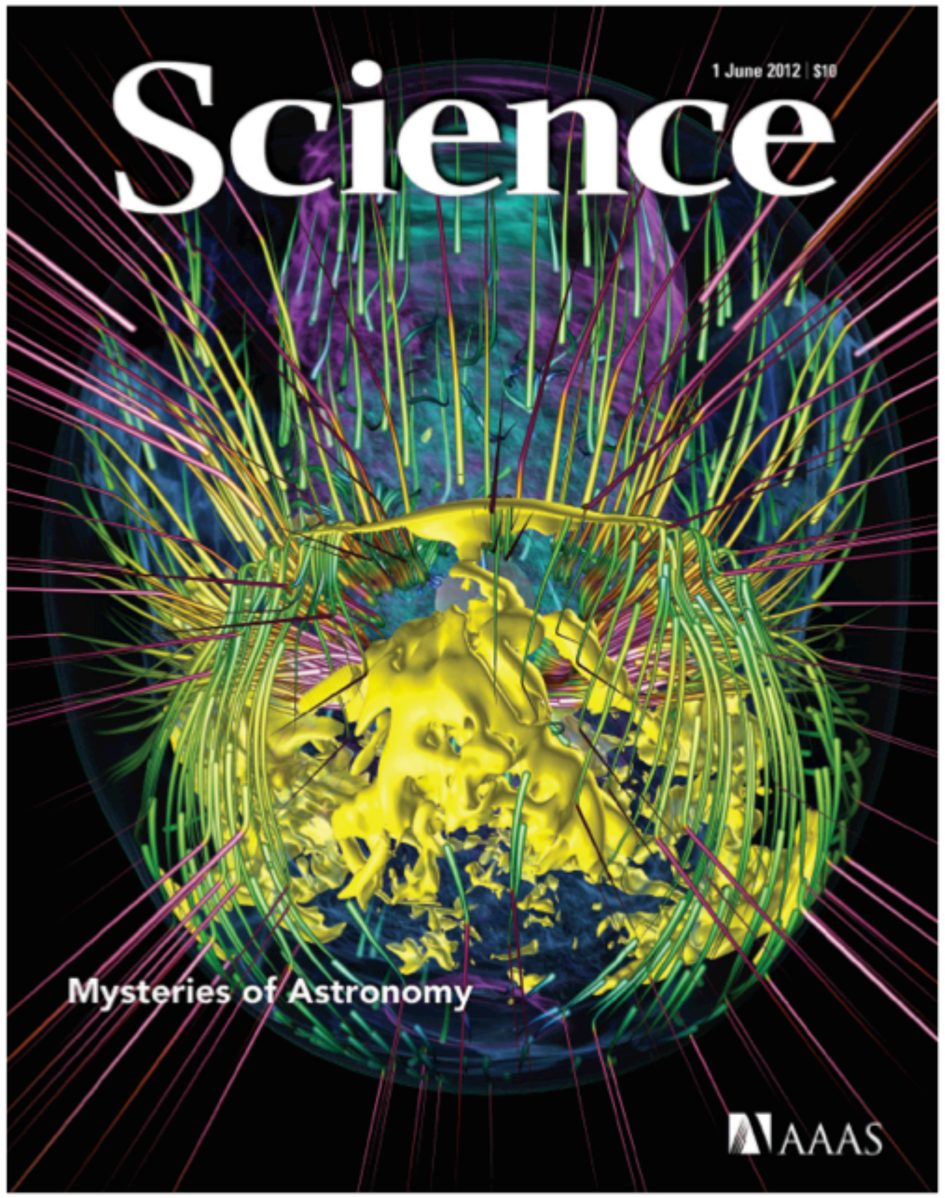
University of
Tennessee

Strange things are afoot...

- We’ve done some parallel programming so far this week.
- Mostly confined to domain decomposition, implemented with MPI.
- This is 20th-century computing.

Nuclear astrophysics INCITE allocations from 2010 - present

- Average number of cpu-hours/yr ~152 M
- In aggregate, just less than 10% of the total available each year from 2010 - 2014
  - Allocations at NERSC are also above-average in size
  - PRAC is dominated by astrophysics, including considerable allocations for stellar astrophysics
- This excellent record is due to
  - the formulation of large, important problems
  - demonstrated ability to efficiently exploit the largest computational platforms
- Will this trend continue to the exascale? Can we continue to solve big problems efficiently?



OAK RIDGE National Laboratory | OAK RIDGE LEADERSHIP COMPUTING FACILITY

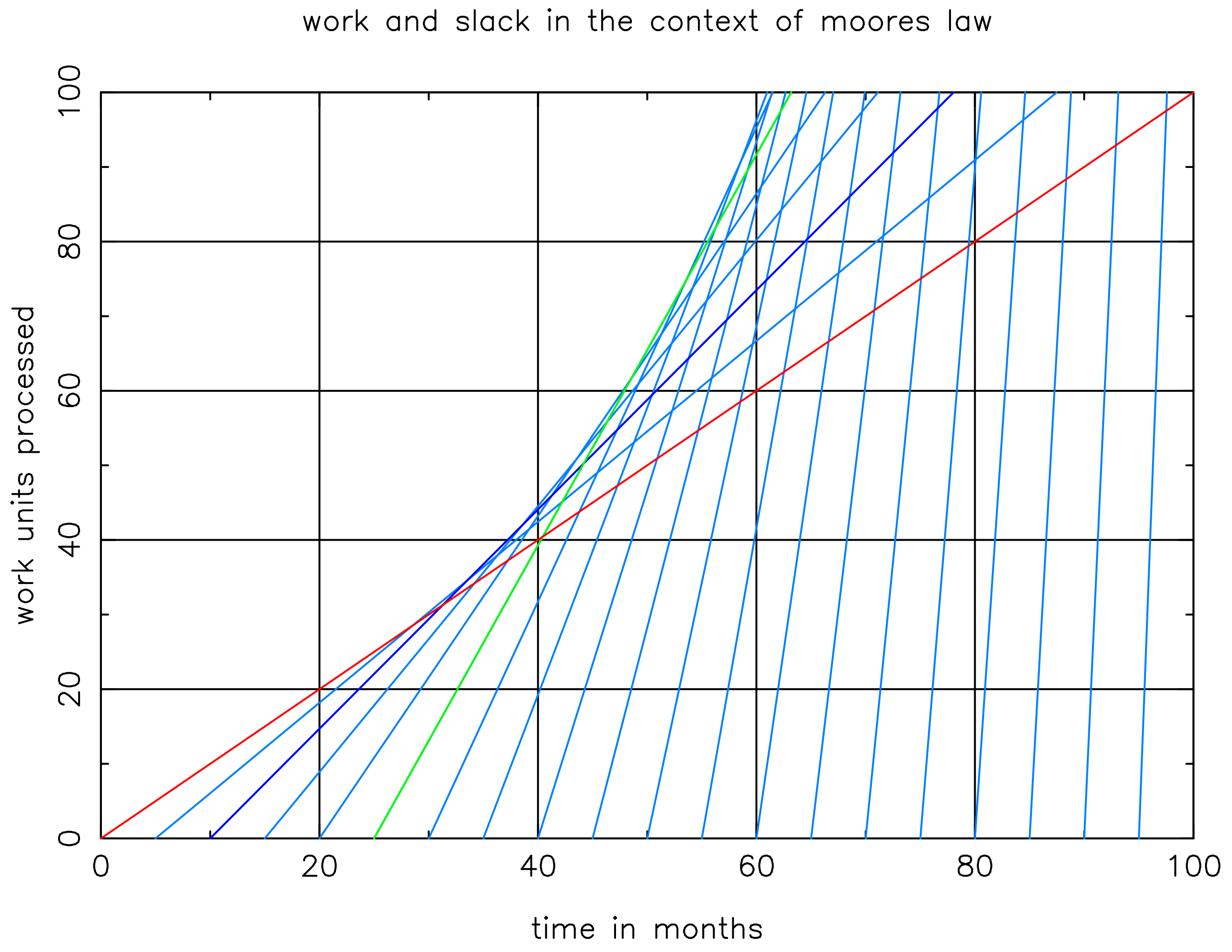
The Effects of Moore's Law and Slacking [1] on Large Computations

Chris Gottbrath, Jeremy Bailin, Casey Meakin, Todd Thompson,
J.J. Charfman
Steward Observatory, University of Arizona

[1]This paper took 2 days to write

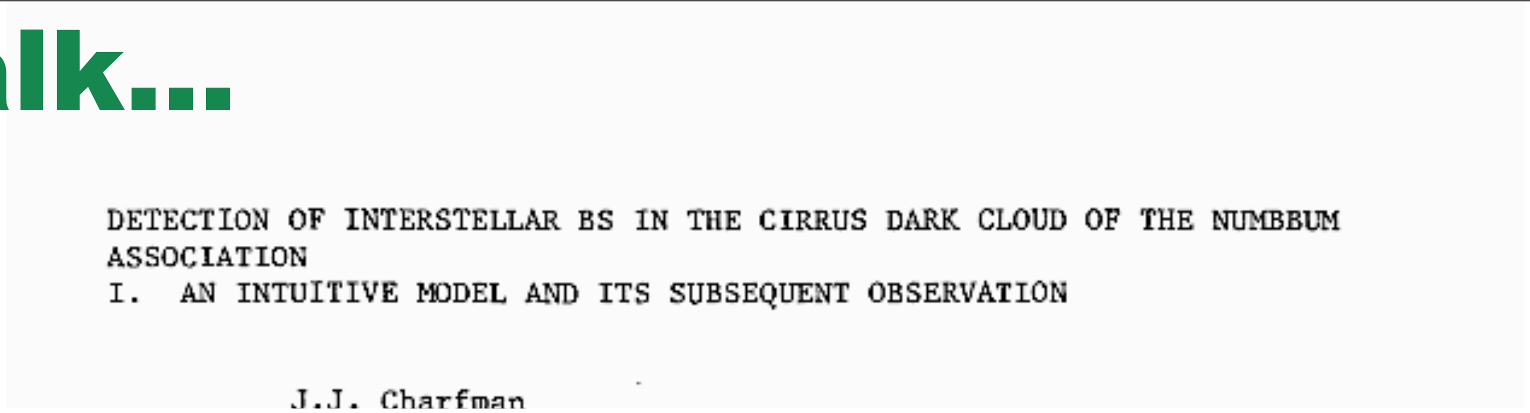
Abstract

We show that, in the context of Moore's Law, overall productivity can be increased for large enough computations by 'slacking' or waiting for some period of time before purchasing a computer and beginning the calculation.



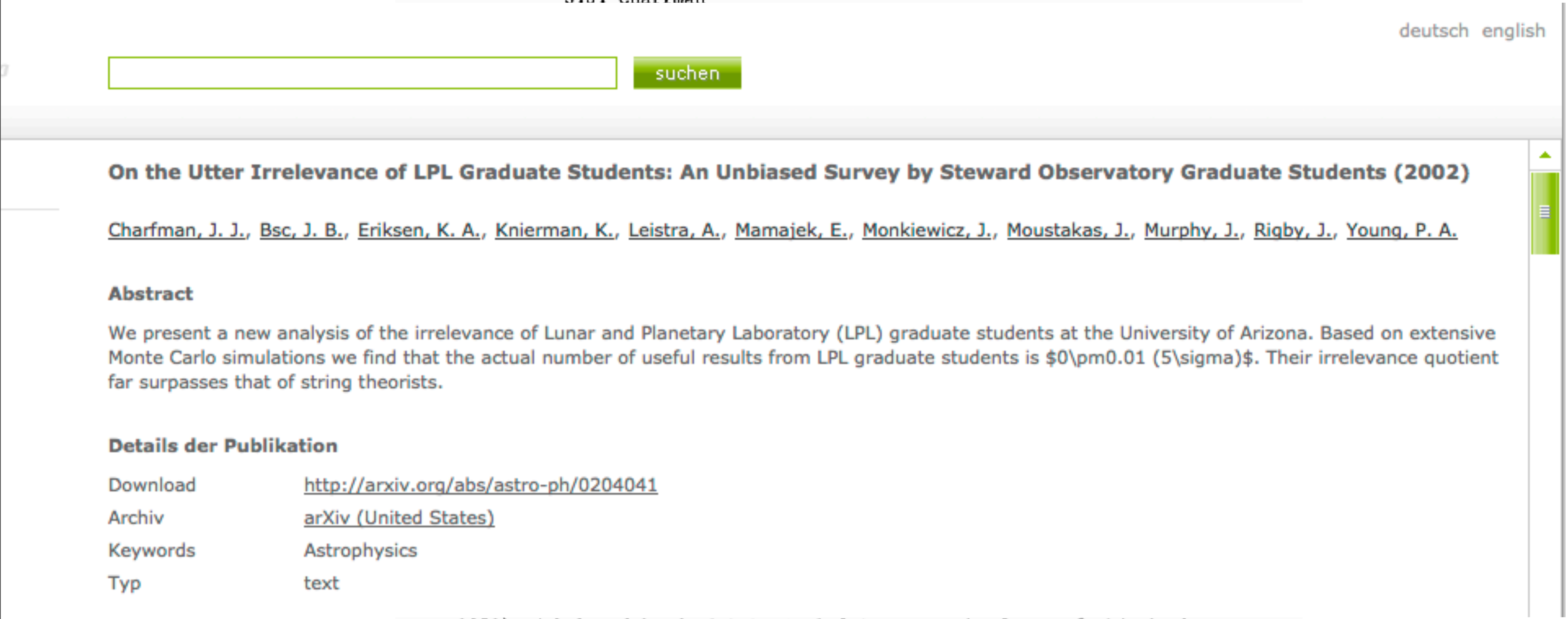
astro-ph/9912202

Google-stalk...



DETECTION OF INTERSTELLAR BS IN THE CIRRUS DARK CLOUD OF THE NUMBBUM ASSOCIATION
I. AN INTUITIVE MODEL AND ITS SUBSEQUENT OBSERVATION

J.J. Charfman



On the Utter Irrelevance of LPL Graduate Students: An Unbiased Survey by Steward Observatory Graduate Students (2002)

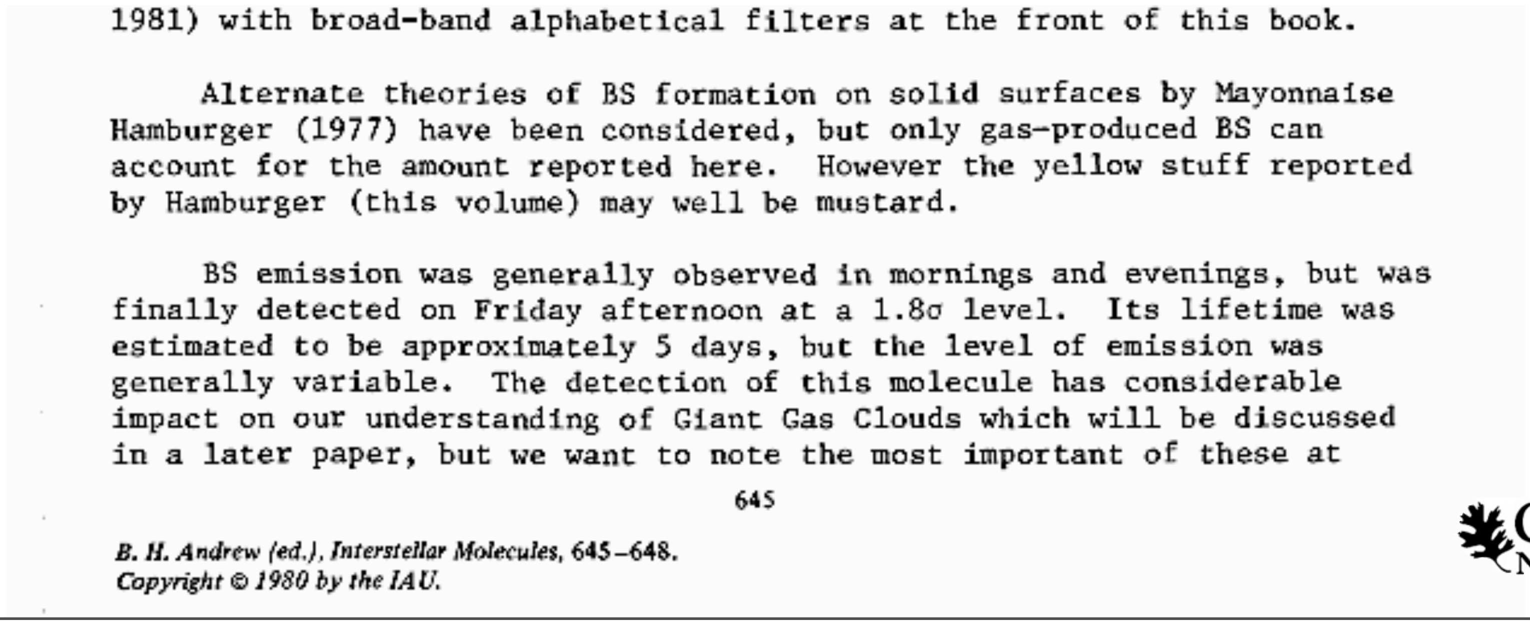
Charfman, J. J., Bsc, J. B., Eriksen, K. A., Knierman, K., Leistra, A., Mamajek, E., Monkiewicz, J., Moustakas, J., Murphy, J., Rigby, J., Young, P. A.

Abstract

We present a new analysis of the irrelevance of Lunar and Planetary Laboratory (LPL) graduate students at the University of Arizona. Based on extensive Monte Carlo simulations we find that the actual number of useful results from LPL graduate students is \$0\pm0.01 (5\sigma)\$. Their irrelevance quotient far surpasses that of string theorists.

Details der Publikation

Download	http://arxiv.org/abs/astro-ph/0204041
Archiv	arXiv (United States)
Keywords	Astrophysics
Typ	text



1981) with broad-band alphabetical filters at the front of this book.

Alternate theories of BS formation on solid surfaces by Mayonnaise Hamburger (1977) have been considered, but only gas-produced BS can account for the amount reported here. However the yellow stuff reported by Hamburger (this volume) may well be mustard.

BS emission was generally observed in mornings and evenings, but was finally detected on Friday afternoon at a 1.8σ level. Its lifetime was estimated to be approximately 5 days, but the level of emission was generally variable. The detection of this molecule has considerable impact on our understanding of Giant Gas Clouds which will be discussed in a later paper, but we want to note the most important of these at

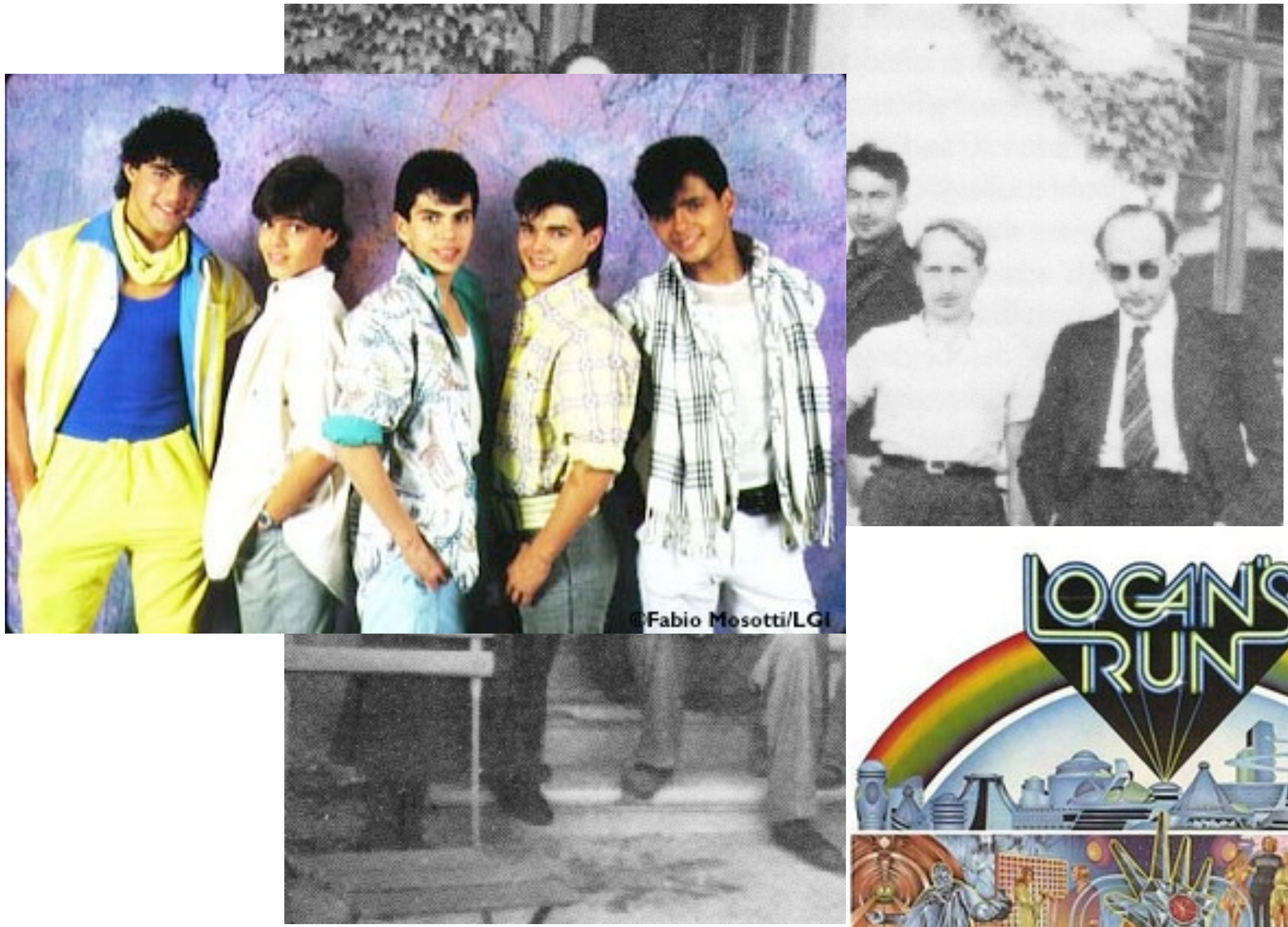
645

B. H. Andrew (ed.), Interstellar Molecules, 645–648.
Copyright © 1980 by the IAU.

J. J. Charfman - The Snarky Nicolas Bourbaki of Arizona’s Steward Observatory

Ø

☡

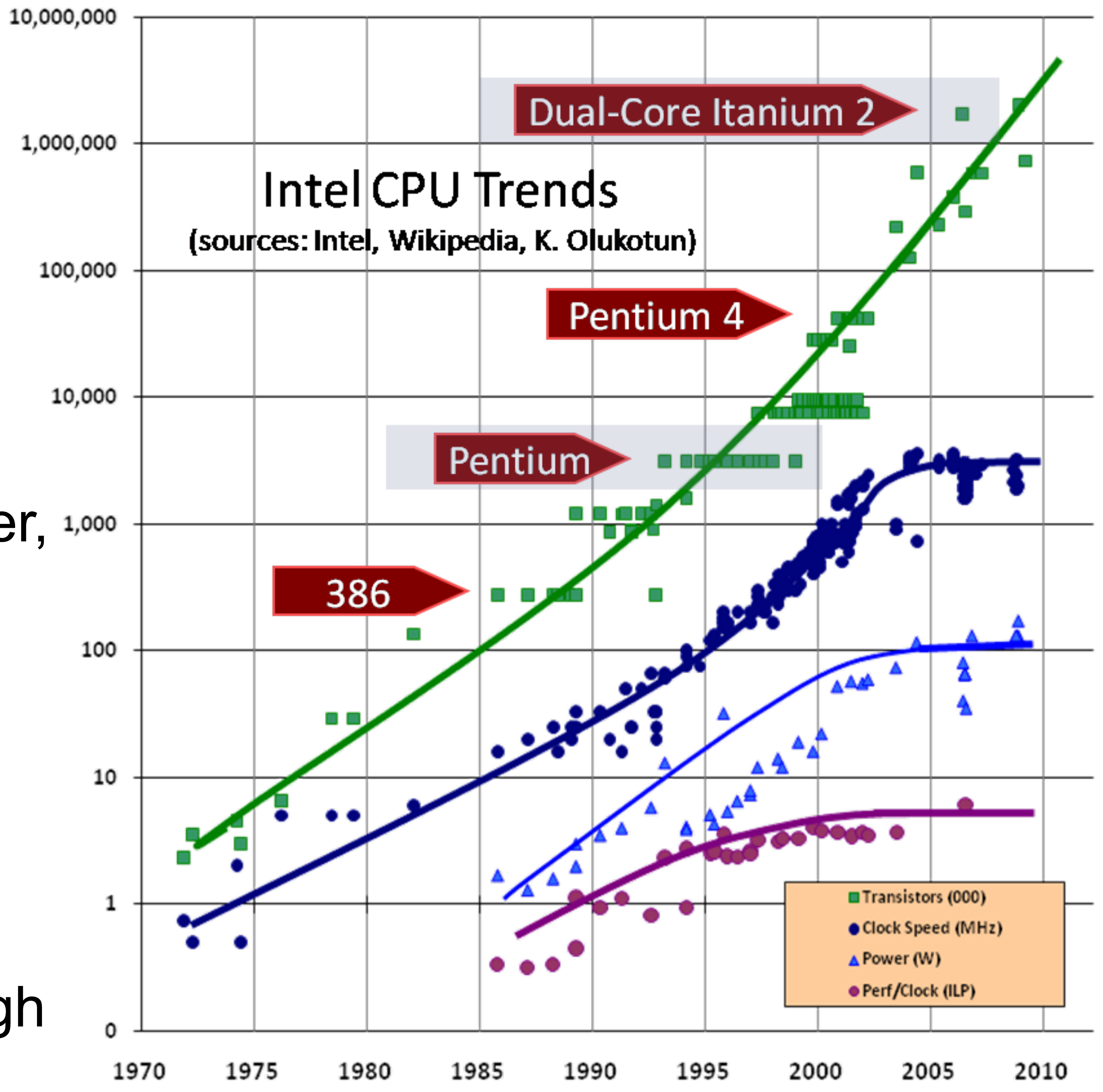


dangerous bends, indeed,
are ahead...

OAK RIDGE National Laboratory | OAK RIDGE LEADERSHIP COMPUTING FACILITY

Architectural Trends – No more free lunch

- CPU clock rates quit increasing in 2003
- $P = CV^2f$
  Power consumed is proportional to the frequency and to the square of the voltage
- Voltage can't go any lower, so frequency can't go higher without increasing power
- Power is capped by heat dissipation and $$$
- Performance increases have been coming through increased parallelism



Herb Sutter: Dr. Dobb's Journal:
http://www.gotw.ca/publications/concurrency-ddj.htm

OAK RIDGE National Laboratory | OAK RIDGE LEADERSHIP COMPUTING FACILITY

Why are machines slowing down?

7 MegaWatts

- 1 MW-hour = 3.6x10^9 joules
- 1 gram TNT = 4184 joules (NIST)
- 1 jaguar-hour ~ 7 MW-hours ~ 2.5x10^{10} joules ~ 6 tons TNT
  - 1 supernova simulation ~ 0.6 jaguar-months ~ 2.6 kton
  - Trinity ~ 19 kton

But, you should care even if you don't use this much electricity...

What Burning Hot Laptops Can Do to Your Legs

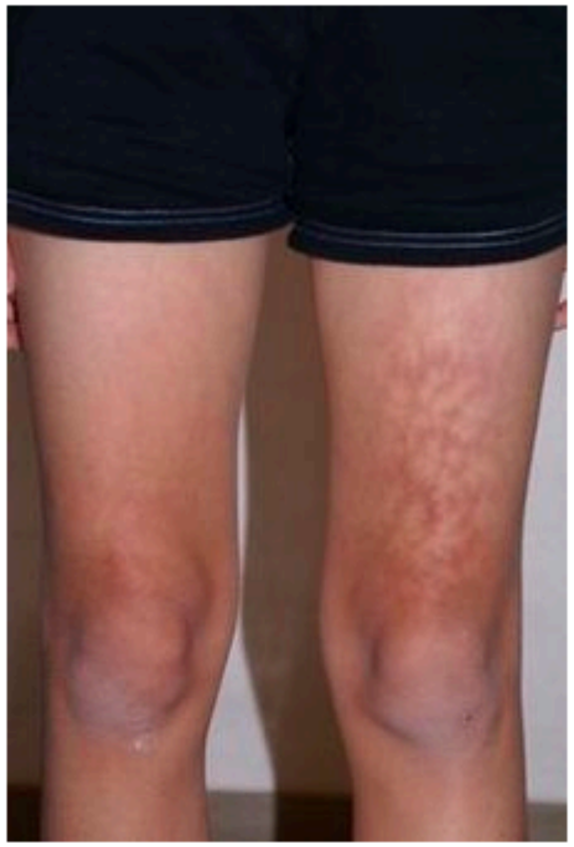
5:10 AM - October 6, 2010 - By Jane McEntegart -
Source : Tom's Guide US

Like 57 Send Twitter 17 +1 0 StumbleUpon 1 Share 18

Hot laptops can lead to permanent skin discoloration from 'Toasted Skin Syndrome.'

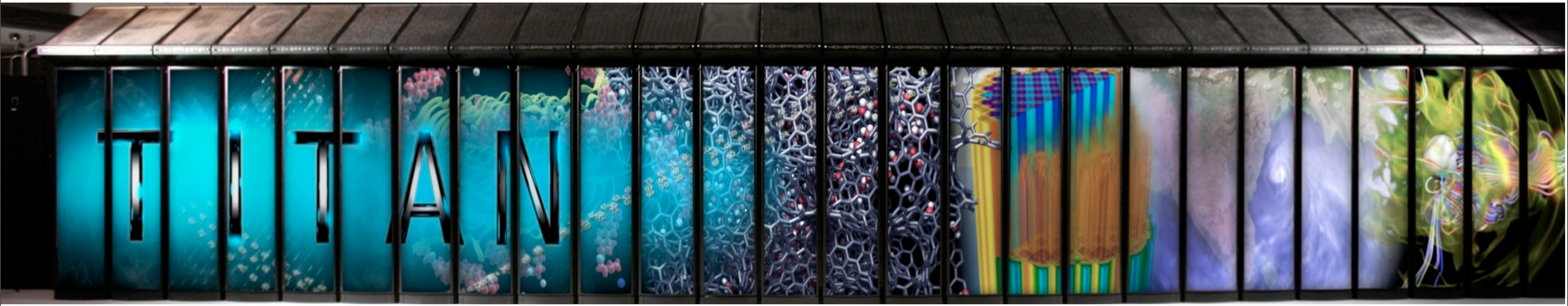
Most laptop owners are familiar with what happens when they use their computers on their laps. Things can get pretty warm and it doesn't take long for it to get uncomfortable. While most of us would move to the coffee table or stick a cushion under the laptop when things start to get sweaty, it seems not everyone takes the same measures and it's resulting in laptop-induced injuries.

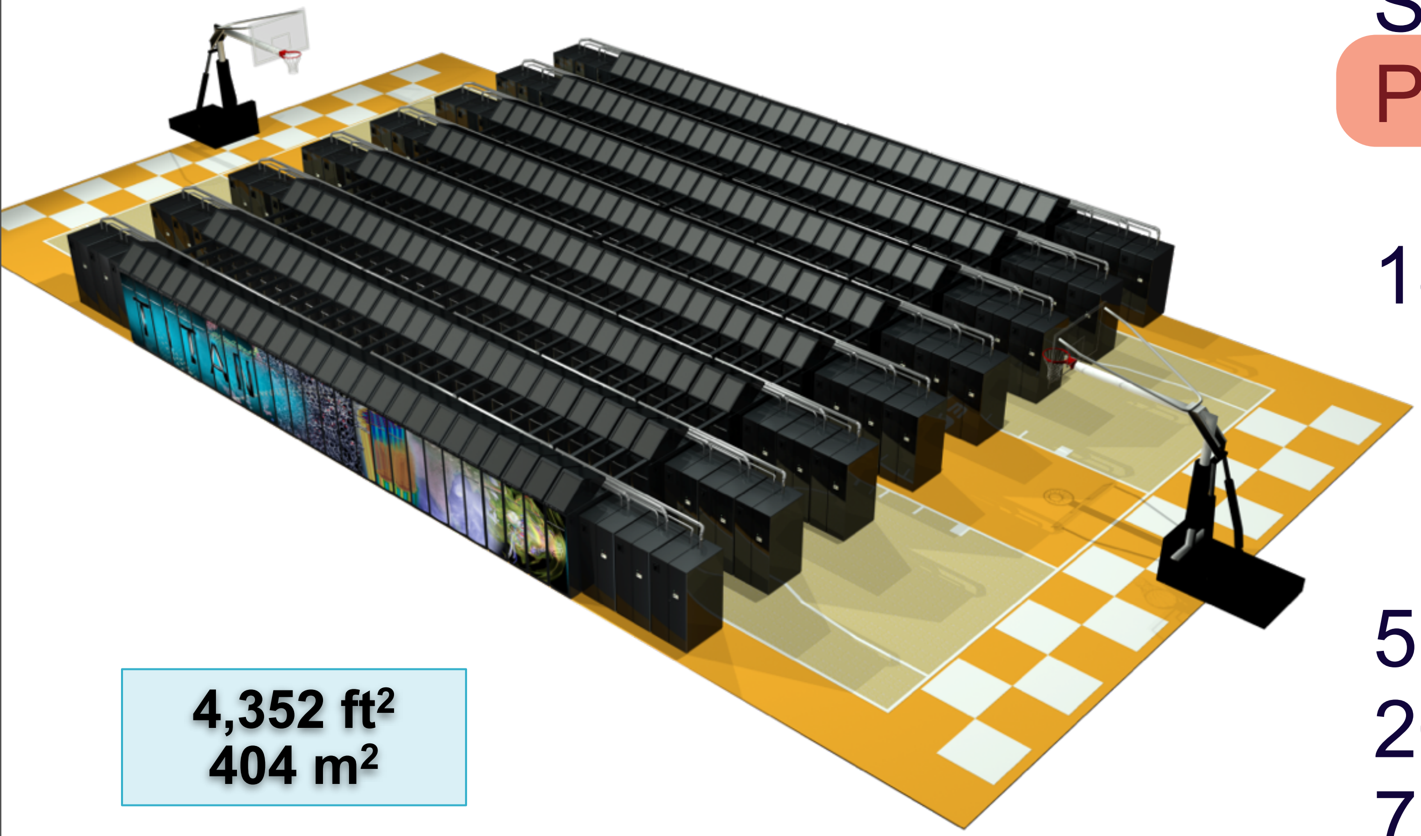
The journal Pediatrics this week warned of the dangers of resting a hot laptop on your legs for extended periods of time and highlighted a couple of cases where it had resulted in erythema abigne, a skin condition the journal describes as "a reticular, pigmented, sometimes telangiectatic dermatosisthat is caused by prolonged exposure to a heat or infrared source." Laptop-induced rythema abigne is now being referred to as "Toasted Skin Syndrome."



Zoom

ORNL's "Titan" Hybrid System: Cray XK7 with AMD Opteron and NVIDIA Tesla processors





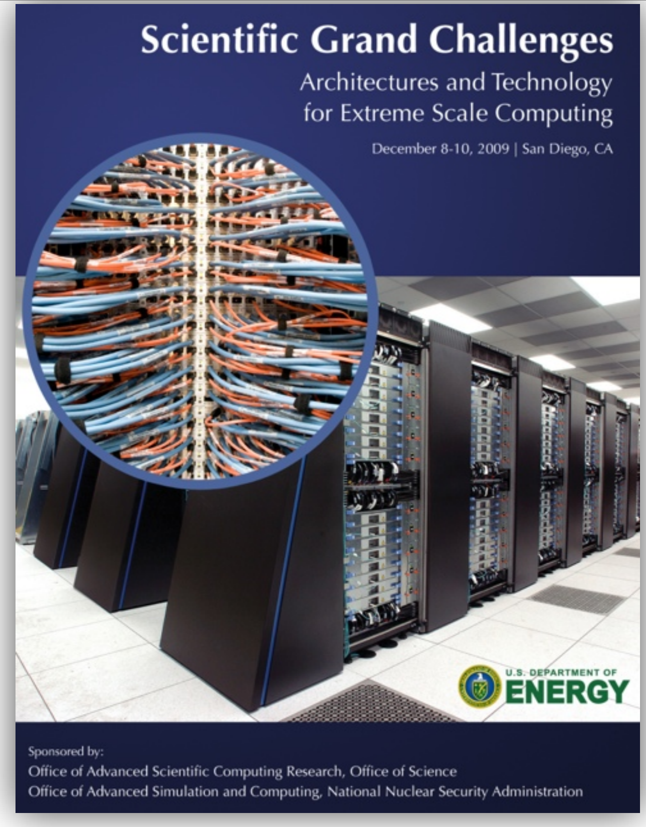
4,352 ft^2
404 m^2

SYSTEM SPECIFICATIONS:

- Peak performance of 27.1 PF >10x
  - 24.5 GPU + 2.6 CPU
- 18,688 Compute Nodes each with:
  - 16-Core AMD Opteron CPU
  - NVIDIA Tesla "K20x" GPU
  - 32 + 6 GB memory
- 512 Service and I/O nodes
- 200 Cabinets
- 710 TB total system memory
- Cray Gemini 3D Torus Interconnect
- 8.8 MW peak power ~2x

OAK RIDGE National Laboratory | OAK RIDGE LEADERSHIP COMPUTING FACILITY

What will the exascale look like?


Scientific Grand Challenges
Architectures and Technology for Extreme Scale Computing
December 8-10, 2009 | San Diego, CA
U.S. DEPARTMENT OF ENERGY
Sponsored by:
Office of Advanced Scientific Computing Research, Office of Science
Office of Advanced Simulation and Computing, National Nuclear Security Administration

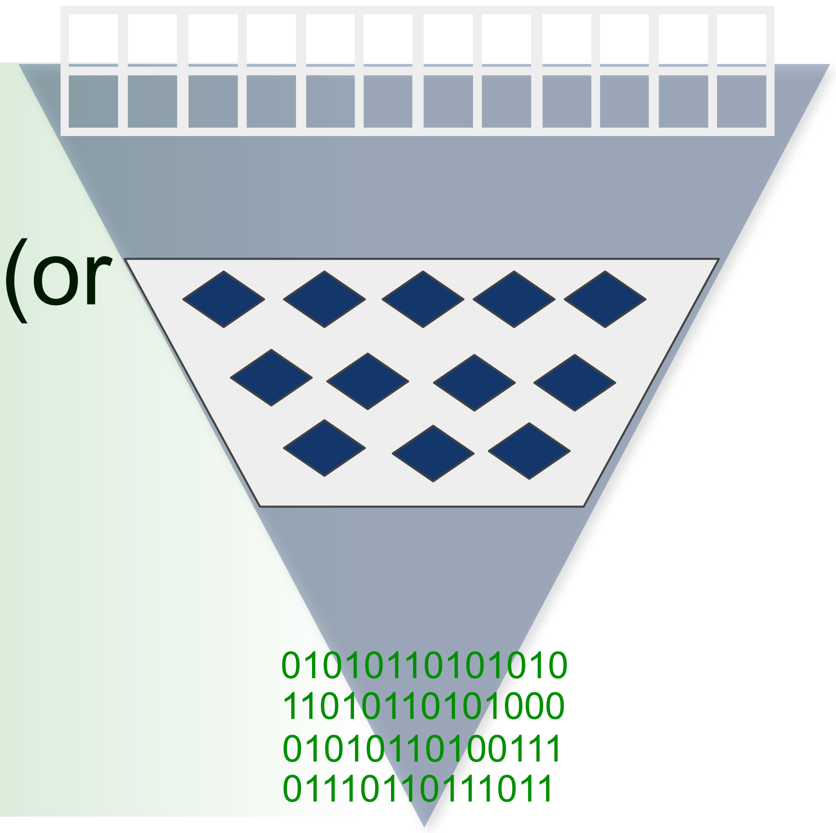
- "Node architectures are expected to change dramatically in the next decade, becoming more hierarchical and heterogeneous."
- ". . . computer companies are dramatically increasing on-chip parallelism to improve performance. The traditional doubling of clock speeds every 18 to 24 months is being replaced by a doubling of cores or other parallelism mechanisms."
- "Systems will consist of one hundred thousand to one million nodes and perhaps as many as a billion cores."

Architectures and Technology for Extreme Scale Computing, Workshop Report, 2009;
http://www.er.doe.gov/ascr/ProgramDocuments/Docs/Arch-TechGrandChallengesReport.pdf

OAK RIDGE National Laboratory | OAK RIDGE LEADERSHIP COMPUTING FACILITY

Hierarchical Parallelism

- MPI parallelism between nodes (or PGAS)
- On-node, SMP-like parallelism via threads (or subcommunicators, or…)
- Vector parallelism
  - – SSE/AVX/etc on CPUs
  - – GPU threaded parallelism



- **Exposure of unrealized parallelism is essential to exploit all near-future architectures.**
- **Uncovering unrealized parallelism and improving data locality improves the performance of even CPU-only code.**
- **Experience with vanguard codes at OLCF suggests 1-2 person-years is required to "port" extant codes to GPU platforms.**
  - **Likely less if begun today, due to better tools/compilers**

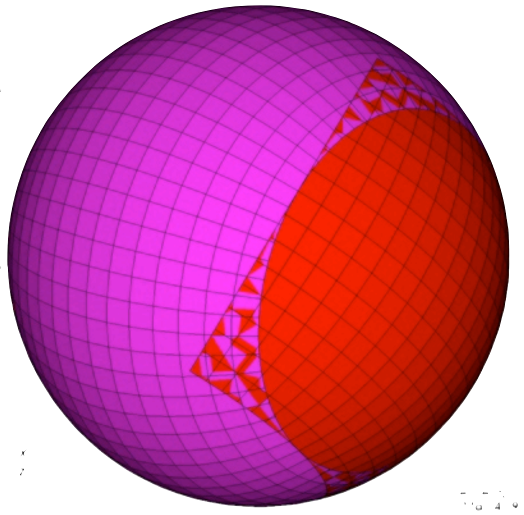
Oak Ridge National Laboratory | Oak Ridge Leadership Computing Facility

So how should we program for these new systems?

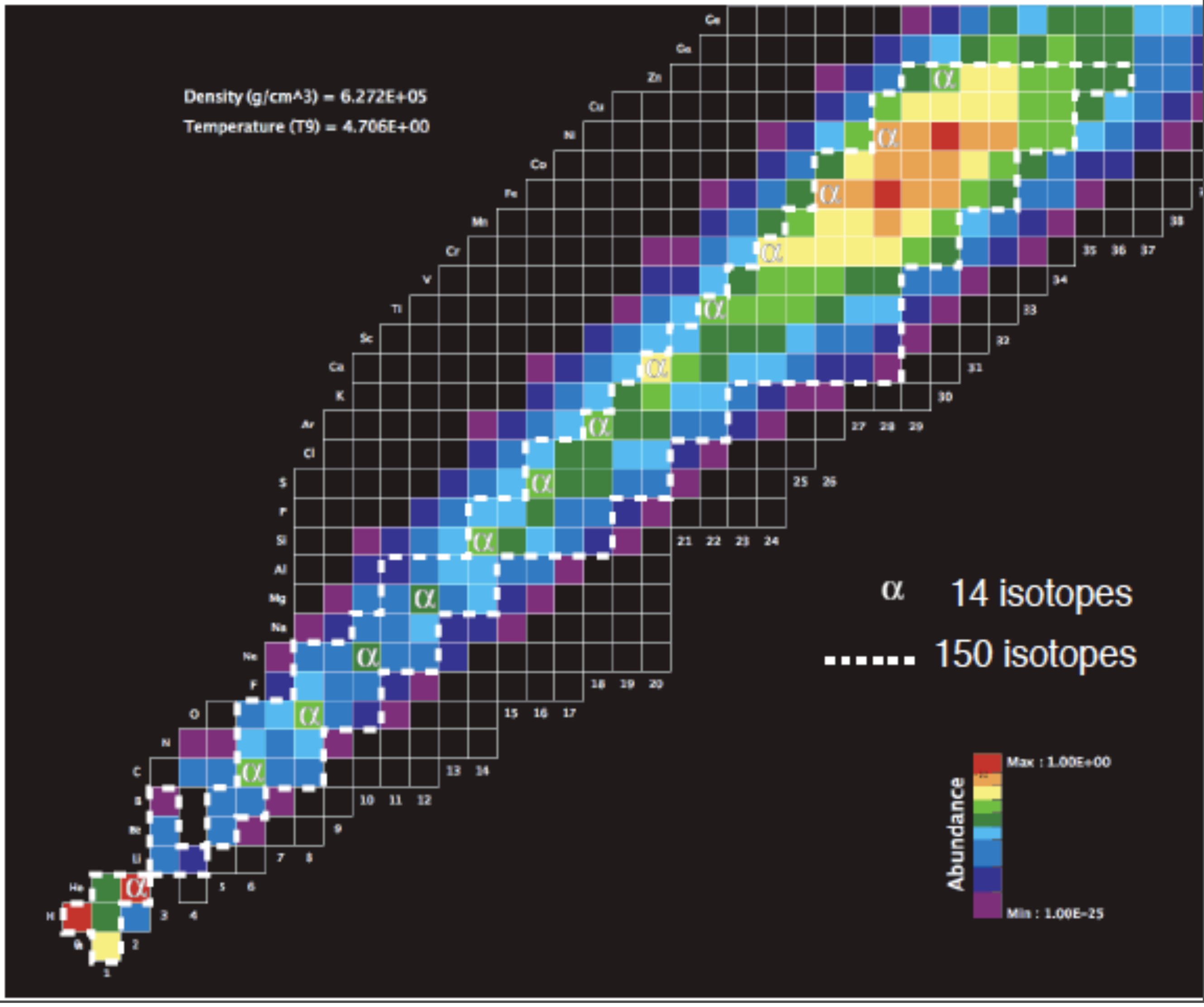
- What to do – Good Threading (OpenMP)
  - Must do high level threading
  - Thread must access close shared memory rather than distant shared memory
  - Load Balancing
- What to do – Good Vectorization
  - Vectorization advantage allows for introducing overhead to vectorize
  - Vectorization doesn’t rely on having HW that is described as “vector”

Good news! Stellar astrophysics tends to have a lot of unrealized parallelism at present

- Simulation codes for stellar evolution and explosions
  - Exemplars of “multiphysics application codes”
  - Typically many degrees-of-freedom per spatial grid point
    - radiation transport
    - nuclear burning
  - Spatial domains typically parallelized via domain decomposition



- Related, computationally-intensive topics will, perhaps, have to work harder to identify additional parallelism outside of large stellar simulations, but plenty of opportunity exists.
- high-density physics
- nuclear structure and reactions



OpenACC
DIRECTIVES FOR ACCELERATORS

- A common directive programming model for today’s GPUs
  - Announced at SC11 conference
  - Offers portability between compilers
    - Drawn up by: NVIDIA, Cray, PGI, CAPS
    - Multiple compilers offer portability, debugging,
      - permanence
  - Works for Fortran, C, C++
    - Standard available at www.OpenACC-standard.org
    - Initially implementations targeted at NVIDIA GPUs
- Current version: 1.0 (November 2011)
- Compiler support:
  - Cray CCE: complete 1.0
  - PGI Accelerator: complete 1.0 + extensions
  - CAPS: complete 1.0 + extensions

The OpenACC™ API
QUICK REFERENCE GUIDE

The OpenACC Application Program Interface describes a collection of compiler directives to specify loops and regions of code in standard C, C++ and Fortran to be offloaded from a host CPU to an attached accelerator, providing portability across operating systems, host CPUs and accelerators.

Most OpenACC directives apply to the immediately following structured block or loop; a structured block is a single statement or a compound statement (C or C++) or a sequence of statements (Fortran) with a single entry point at the top and a single exit at the bottom.

CAPS
CRAY THE SUPERCOMPUTER COMPANY
NVIDIA.
PGI

Version 1.0, November 2011

All four companies intend to work within OpenMP organization to merge OpenACC and create a common specification that extends OpenMP to support accelerators.

© 2011 OpenACC-standard.org all rights reserved.

OAK RIDGE National Laboratory | OAK RIDGE LEADERSHIP COMPUTING FACILITY

Why use OpenACC Directives?

- Productivity
  - –Higher level programming model
  - –*a la* OpenMP
- Portability
  - –ignore directives, portable to the host
  - –portable across different accelerators
  - –*performance portability*
- Performance feedback

OpenCL

```
                        0, workSize[i] * sizeof(float) * WA, 0, NULL, NULL);

        // create OpenCL buffer on device that will be initiatlize from the host memory on first use
        // on device
        d_B[i] = clCreateBuffer(cxGPUContext, CL_MEM_READ_ONLY | CL_MEM_COPY_HOST_PTR,
                                mem_size_B, h_B_data, NULL);

        // Output buffer
        d_C[i] = clCreateBuffer(cxGPUContext, CL_MEM_WRITE_ONLY,  workSize[i] * WC * sizeof(float), NULL,NULL);

        // set the args values
        clSetKernelArg(multiplicationKernel[i], 0, sizeof(cl_mem), (void *) &d_C[i]);
        clSetKernelArg(multiplicationKernel[i], 1, sizeof(cl_mem), (void *) &d_A[i]);
        clSetKernelArg(multiplicationKernel[i], 2, sizeof(cl_mem), (void *) &d_B[i]);
        clSetKernelArg(multiplicationKernel[i], 3, sizeof(float) * BLOCK_SIZE *BLOCK_SIZE, 0 );
        clSetKernelArg(multiplicationKernel[i], 4, sizeof(float) * BLOCK_SIZE *BLOCK_SIZE, 0 );

        if(i+1 < ciDeviceCount)
            workOffset[i + 1] = workOffset[i] + workSize[i];
    }
    // Execute Multiplication on all GPUs in parallel
    size_t localWorkSize[] = {BLOCK_SIZE, BLOCK_SIZE};
    size_t globalWorkSize[] = {shrRoundUp(BLOCK_SIZE, WC), shrRoundUp(BLOCK_SIZE, workSize[0])};
    // Launch kernels on devices
        for(unsigned int i = 0; i < ciDeviceCount; i++)
        {
            // Multiplication - non-blocking execution
            globalWorkSize[1] = shrRoundUp(BLOCK_SIZE, workSize[i]);
            clEnqueueNDRangeKernel(commandQueue[i], multiplicationKernel[i], 2, 0, globalWorkSize, localWorkSize,
                                   0, NULL, &GPUExecution[i]);
        }
        for(unsigned int i = 0; i < ciDeviceCount; i++)
    {
            clFinish(commandQueue[i]);
        }
    for(unsigned int i = 0; i < ciDeviceCount; i++)
    {
 // Non-blocking copy of result from device to host
        clEnqueueReadBuffer(commandQueue[i], d_C[i], CL_FALSE, 0, WC * sizeof(float) * workSize[i],
                            h_C + workOffset[i] * WC, 0, NULL, &GPUDone[i]);
    }
 // CPU sync with GPU
    clWaitForEvents(ciDeviceCount, GPUDone);

 // Release mem and event objects
    for(unsigned int i = 0; i < ciDeviceCount; i++)
    {
        clReleaseMemObject(d_A[i]);
        clReleaseMemObject(d_C[i]);
        clReleaseMemObject(d_B[i]);
        clReleaseEvent(GPUExecution[i]);
        clReleaseEvent(GPUDone[i]);
    }
}
__kernel void
matrixMul( __global float* C, __global float* A, __global float* B,
           __local float* As, __local float* Bs)
{
    int bx = get_group_id(0), tx = get_local_id(0);
    int by = get_group_id(1), ty = get_local_id(1);
    int aEnd   = WA * BLOCK_SIZE * by + WA - 1;

    float Csub = 0.0f;

    for (int a = WA*BLOCK_SIZE*by , b = BLOCK_SIZE * bx;
         a <= aEnd; a += BLOCK_SIZE, b += BLOCK_SIZE*WB) {
        As[tx + ty * BLOCK_SIZE] = A[a + WA * ty + tx];
        Bs[tx + ty * BLOCK_SIZE] = B[b + WB * ty + tx];
        barrier(CLK_LOCAL_MEM_FENCE);
        for (int k = 0; k < BLOCK_SIZE; ++k)
            Csub += As[k + ty * BLOCK_SIZE]*Bs[tx + k * BLOCK_SIZE] ;
        barrier(CLK_LOCAL_MEM_FENCE);
    }
    C[get_global_id(1) * get_global_size(0) + get_global_id(0)] = Csub;

}
```

CUDA C

```
void
__global__ void matrixMul( float* C, float* A, float* B, int wA, int wB)
{
    int bx = blockIdx.x;
    int by = blockIdx.y;
    int tx = threadIdx.x;
    int ty = threadIdx.y;
    int aBegin = wA * BLOCK_SIZE * by;
    int aEnd   = aBegin + wA - 1;
    int aStep  = BLOCK_SIZE;
    int bBegin = BLOCK_SIZE * bx;
    int bStep  = BLOCK_SIZE * wB;
    float Csub = 0;
    for (int a = aBegin, b = bBegin;
         a <= aEnd;
         a += aStep, b += bStep) {
        __shared__ float As[BLOCK_SIZE][BLOCK_SIZE];
        __shared__ float Bs[BLOCK_SIZE][BLOCK_SIZE];
        AS(ty, tx) = A[a + wA * ty + tx];
        BS(ty, tx) = B[b + wB * ty + tx];
        __syncthreads();
        for (int k = 0; k < BLOCK_SIZE; ++k)
            Csub += AS(ty, k) * BS(k, tx);
        __syncthreads();
    }
    int c = wB * BLOCK_SIZE * by + BLOCK_SIZE * bx;
    C[c + wB * ty + tx] = Csub;
}

void
domatmul( float* C, float* A, float* B, unsigned int hA, unsigned int wA , unsigned wB )
{
    unsigned int size_A = WA * HA;
    unsigned int mem_size_A = sizeof(float) * size_A;
    unsigned int size_B = WB * HB;
    unsigned int mem_size_B = sizeof(float) * size_B;
    unsigned int size_C = WC * HC;
    unsigned int mem_size_C = sizeof(float) * size_C;
    float *d_A, *d_B, *d_C;

    cudaMalloc((void**) &d_A, mem_size_A);
    cudaMalloc((void**) &d_B, mem_size_B);
    cudaMalloc((void**) &d_C, mem_size_C);
    cudaMemcpy(d_A, h_A, mem_size_A, cudaMemcpyHostToDevice);
    cudaMemcpy(d_B, h_B, mem_size_B, cudaMemcpyHostToDevice);

    dim3 threads(BLOCK_SIZE, BLOCK_SIZE);
    dim3 grid(WC / threads.x, HC / threads.y);

    matrixMul<<< grid, threads >>>(d_C, d_A, d_B, WA, WB);

    cudaMemcpy(h_C, d_C, mem_size_C, cudaMemcpyDeviceToHost);

    cudaFree(d_A);
    cudaFree(d_B);
    cudaFree(d_C);
}
```

OpenAcc

```
void
computeMM0_saxpy(float C[][WB], float A[][WA], float B[][WB],
                 int hA, int wA, int wB)
{
#pragma acc region
{
#pragma acc for parallel vector(16) unroll(4)
    for (int i = 0; i < hA; ++i) {
        for (int j = 0; j < wB; ++j) {
            C[i][j] = 0.0 ;
        }
        for (int k = 0; k < wA; ++k) {
            for (int j = 0; j < wB; ++j) {
                C[i][j] = C[i][j]+A[i][k]*B[k][j];
            }
        }
    }
}
}
```

OAK RIDGE National Laboratory | OAK RIDGE LEADERSHIP COMPUTING FACILITY

Risk factors

- Will there be machines to run my OpenACC code on?
  - Now? Lots of Nvidia GPU accelerated systems
    - Cray XK7s: CSCS tödi, HLRS hermit, ORNL titan...
    - Lots of other GPU machines in Top100 (OpenACC is multi-vendor)
  - Future? OpenACC can be targeted at other accelerators
    - PGI and CAPS already target Intel Xeon Phi, AMD GPUs
  - Plus you can always run on CPUs using same codebase
- Will OpenACC continue?
  - Support? Cray and PGI (at least) are committed to support OpenACC
    - Lots of big customer pressure to continue to run OpenACC
  - Develop? OpenACC v2.0 standard out
  - Lots of new partners joined committee at end of last year
- Will OpenACC be superseded by something else?
  - Auto-accelerating compilers? If only!
    - Never really managed it for threading real HPC applications on the CPU
    - Data locality adds to the challenge
  - OpenMP accelerator directives? OpenACC work not wasted
    - Very similar programming model; can transition easily
    - Cray (co-chair), PGI very active in OpenMP accelerator subcommittee

OAK RIDGE National Laboratory | OAK RIDGE LEADERSHIP COMPUTING FACILITY

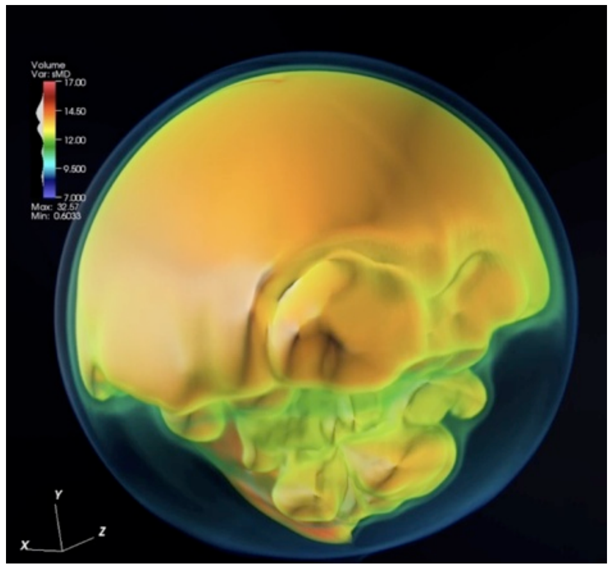
Posited Exascale Specs

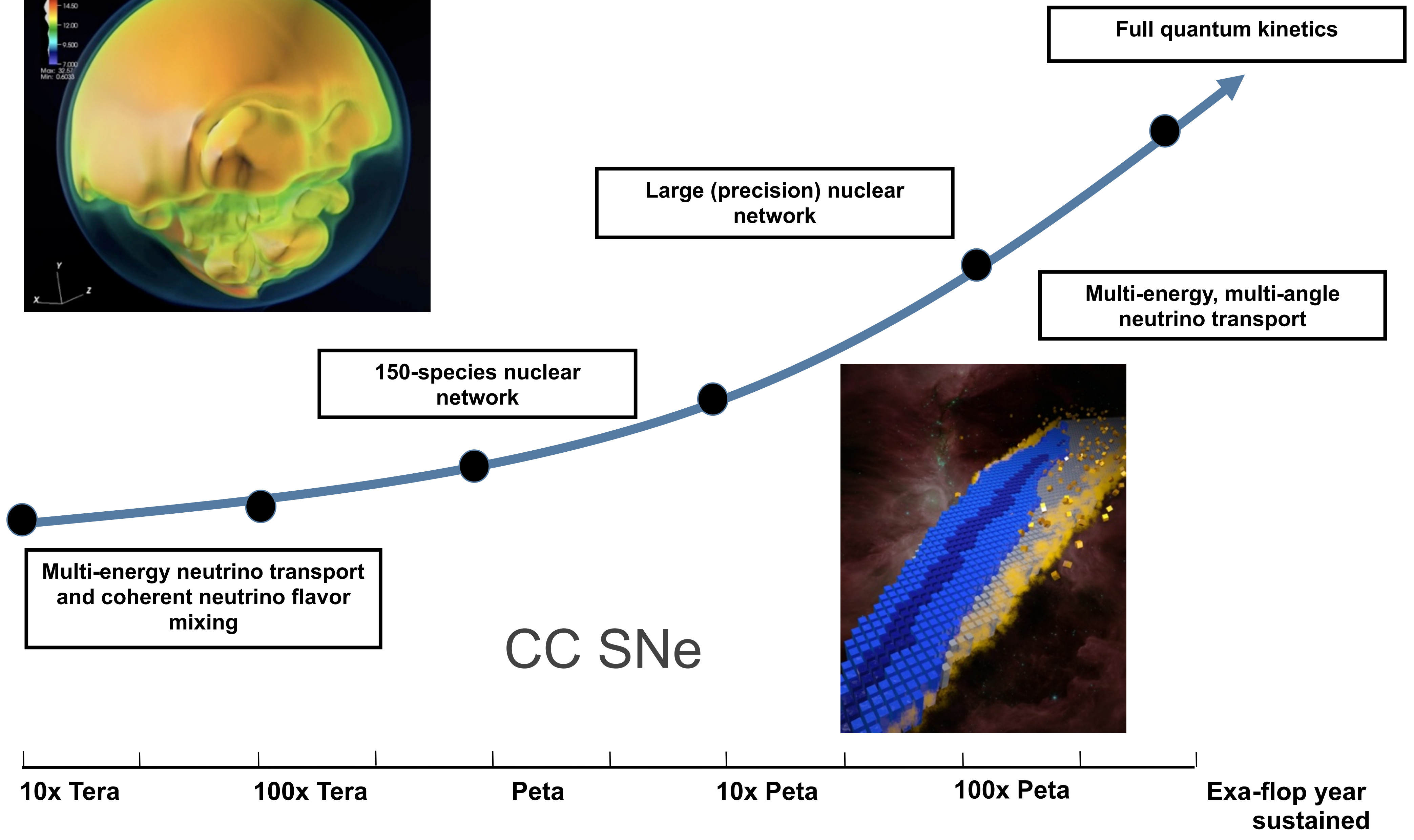
		2018		2023?	
System attributes	**2010**	**“20~~1~~5”**		**“20~~1~~8”**	
System peak	2 PF	200 PF/s		1 Exaflop/s	
Power	6 MW	15 MW		20 MW	
System memory	0.3 PB	5 PB		32–64 PB	
Node performance	125 GF	0.5 TF	7 TF	1 TF	10 TF
Node memory BW	25 GB/s	0.1 TB/s	1 TB/s	0.4 TB/s	4 TB/s
Node concurrency	12	O(100)	O(1,000)	O(1,000)	O(10,000)
System size (nodes)	18,700	50,000	5,000	1,000,000	100,000
Total node interconnect BW	1.5 GB/s	150 GB/s	1 TB/s	250 GB/s	2 TB/s
MTTI	day	O(1 day)		O(1 day)	

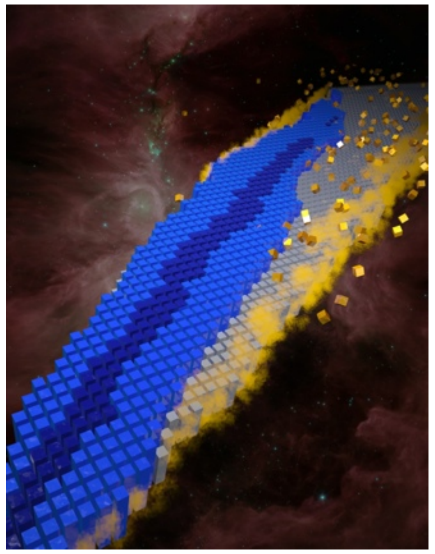
OAK RIDGE National Laboratory | OAK RIDGE LEADERSHIP COMPUTING FACILITY

Example goals/highlights from NP Exascale Workshop report (2009)

http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Np_report.pdf

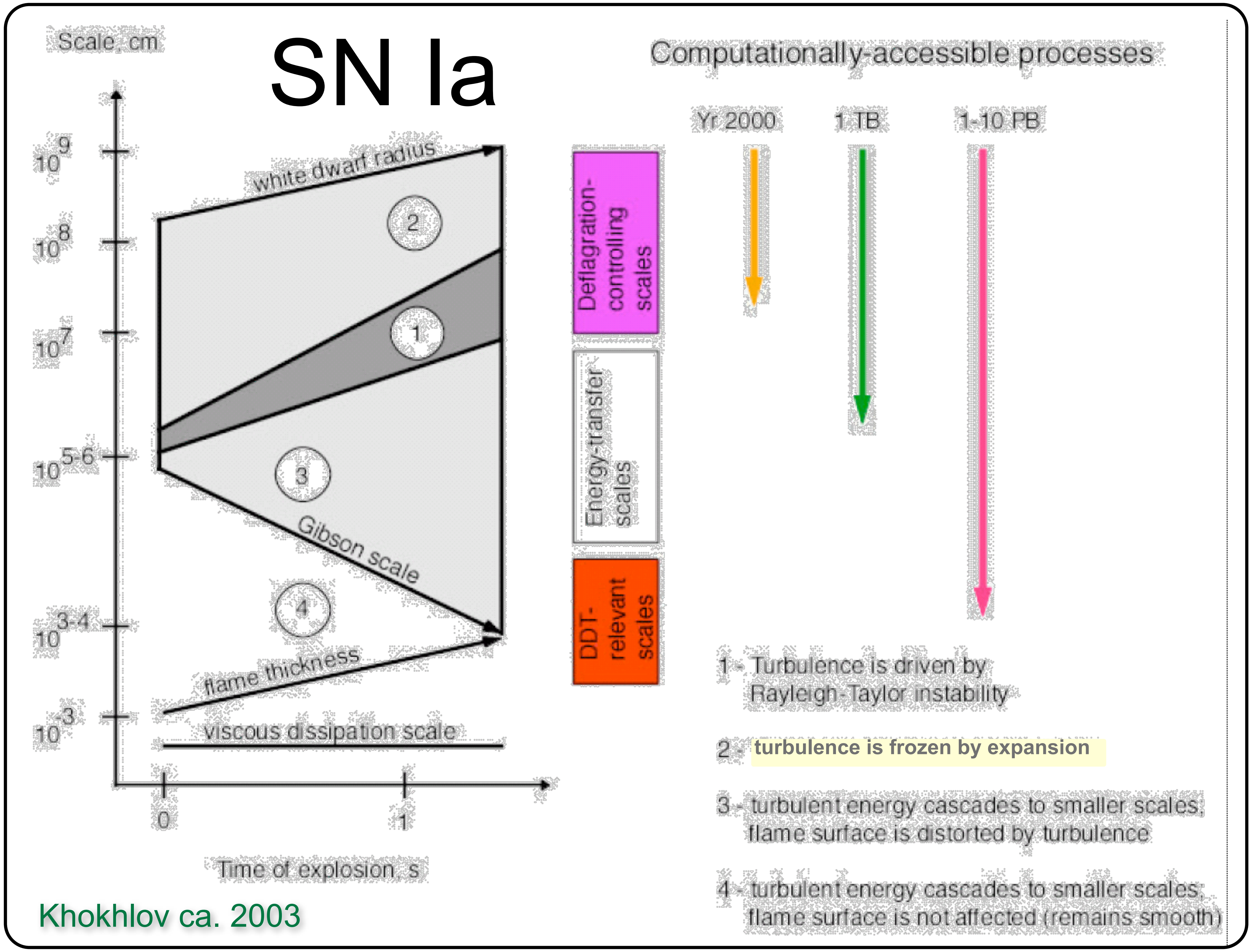



Full quantum kinetics
Large (precision) nuclear network
Multi-energy, multi-angle neutrino transport
150-species nuclear network
Multi-energy neutrino transport and coherent neutrino flavor mixing
CC SNe
10x Tera
100x Tera
Peta
10x Peta
100x Peta
Exa-flop year sustained



All of these goals are attainable, but will require new algorithms and implementations to bridge the gap to the posited architectures.

Achieving high spatial (or phase-space, etc.) resolution will be very difficult.



1 - Turbulence is driven by Rayleigh-Taylor instability

2 - turbulence is frozen by expansion

3 - turbulent energy cascades to smaller scales; flame surface is distorted by turbulence

4 - turbulent energy cascades to smaller scales; flame surface is not affected (remains smooth)

Total memory on the entire exascale system will be O(10 PB)

Locally bulk-synchronous programming model is not a viable path for maximum performance on these new platforms

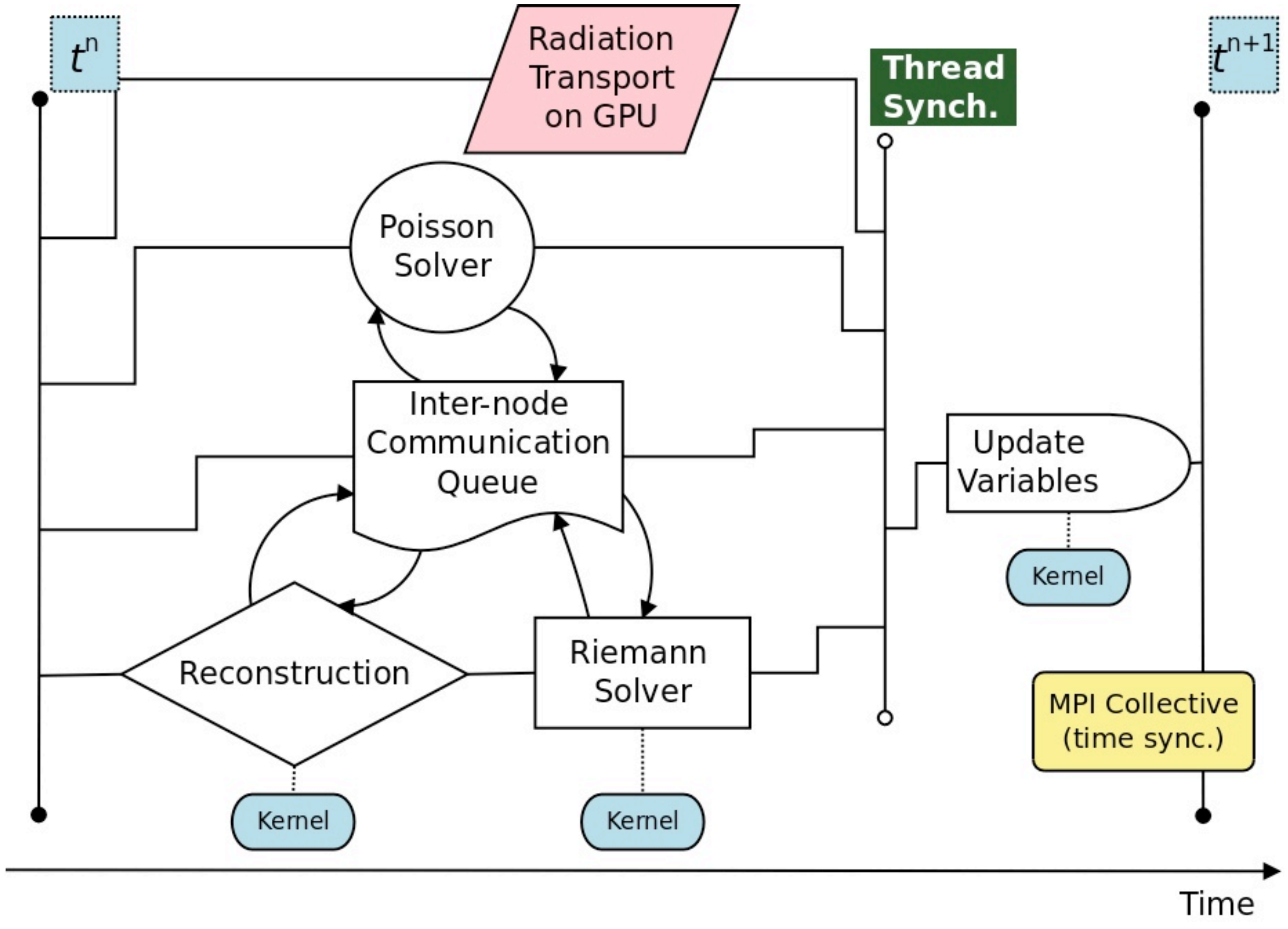
- FLOP/s are cheap and moving data is expensive.
- Even perfect knowledge of resource capabilities at every moment and perfect load balancers will not rescue billion-thread SPMD implementations of PDE simulations.
- Cost of rebalancing frequently is too large, but the Amdahl penalty of failing to rebalance is fatal.
- To take full advantage of asynchronous algorithms, we need to develop greater expressiveness in scientific programming.
  - Create separate threads for logically separate tasks, whose priority is a function of algorithmic state, not unlike the way a time-sharing OS works.
  - Join priority threads in a directed acyclic graph (DAG), a task graph showing the flow of input dependencies; fill idleness with noncritical work or steal work.

Comments taken directly from keynote address by David Keyes at EU-US HPC Summer School, June 2012

OAK RIDGE National Laboratory | OAK RIDGE LEADERSHIP COMPUTING FACILITY

Asynchronous execution models via task scheduling

- Examples exist already in other domains
  - MAGMA (linear algebra)
  - MADNESS (DFT)
  - Uintah (terrestrial combustion)
- Operator-split physics modules become "tasks" associated with execution threads



OAK RIDGE National Laboratory | OAK RIDGE LEADERSHIP COMPUTING FACILITY

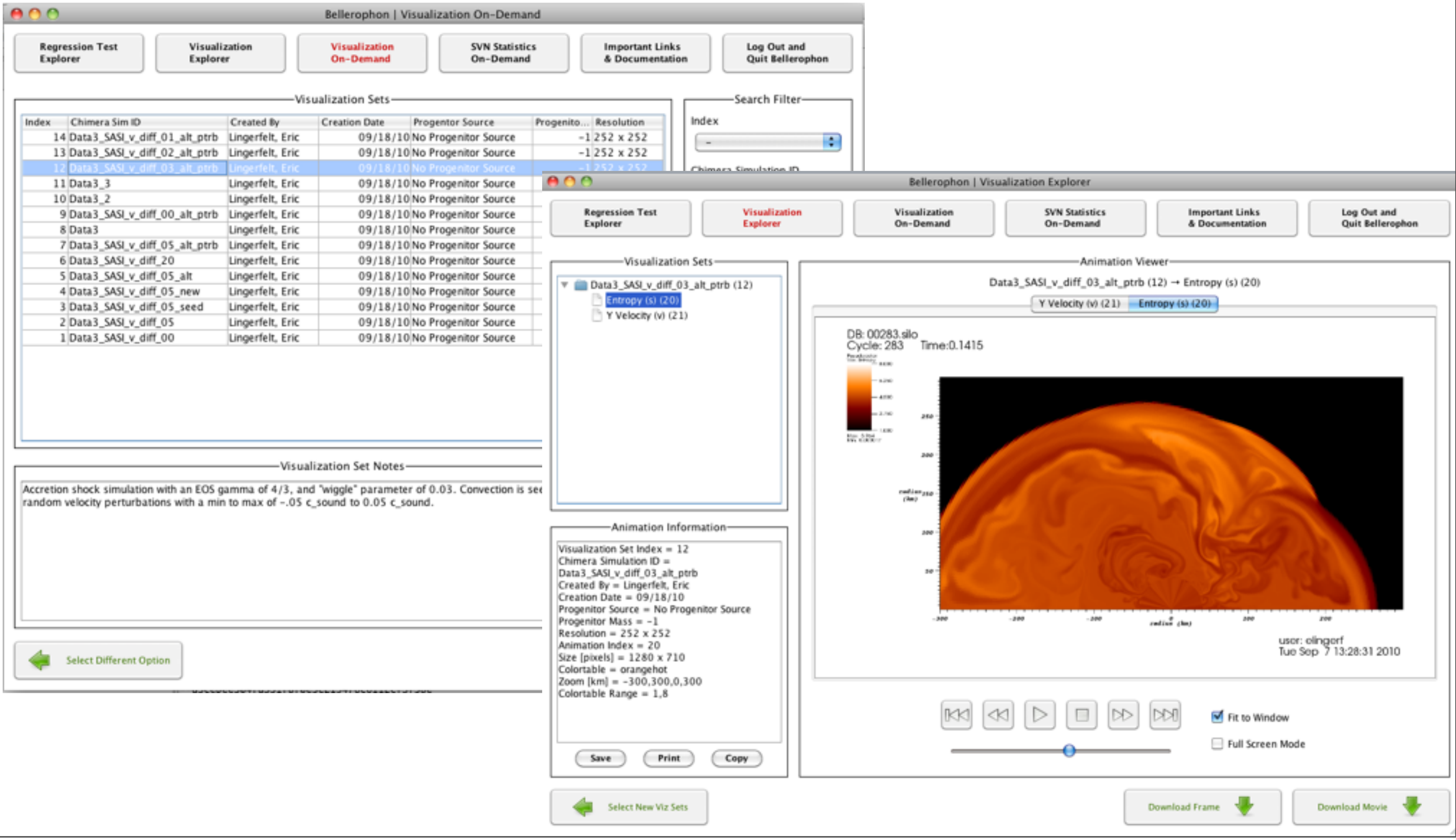
Will the exascale (or before) machine be primarily a "strong-scaling" platform?

- Memory constraints provide a hard ceiling for spatial resolution and number of unknowns.
  - bytes/FLOP goes down by an order of magnitude
- Simulations will be certainly be larger, but likely not as large as one would expect if scaling with FLOPs is assumed.
  - no more than ~10x the number of MPI ranks?
  - this connotes no more than factors of ~2 in resolution in each dimension for 3D
- OK: considerable understanding can be realized by fully exploring parameter space.
  - progenitor mass, rotation, metallicity
  - transport approximations, additional physics

OAK RIDGE National Laboratory | OAK RIDGE LEADERSHIP COMPUTING FACILITY

Simulation, code, and data management become even harder

- Revision control, regression testing, viz, workflow...



Index	Chimera Sim ID	Created By	Creation Date	Progentor Source	Progenito...	Resolution
14	Data3_SASI_v_diff_01_alt_ptrb	Lingerfelt, Eric	09/18/10	No Progenitor Source	-1	252 x 252
13	Data3_SASI_v_diff_02_alt_ptrb	Lingerfelt, Eric	09/18/10	No Progenitor Source	-1	252 x 252
12	Data3_SASI_v_diff_03_alt_ptrb	Lingerfelt, Eric	09/18/10	No Progenitor Source	-1	252 x 252
11	Data3_3	Lingerfelt, Eric	09/18/10	No Progenitor Source		
10	Data3_2	Lingerfelt, Eric	09/18/10	No Progenitor Source		
9	Data3_SASI_v_diff_00_alt_ptrb	Lingerfelt, Eric	09/18/10	No Progenitor Source		
8	Data3	Lingerfelt, Eric	09/18/10	No Progenitor Source		
7	Data3_SASI_v_diff_05_alt_ptrb	Lingerfelt, Eric	09/18/10	No Progenitor Source		
6	Data3_SASI_v_diff_20	Lingerfelt, Eric	09/18/10	No Progenitor Source		
5	Data3_SASI_v_diff_05_alt	Lingerfelt, Eric	09/18/10	No Progenitor Source		
4	Data3_SASI_v_diff_05_new	Lingerfelt, Eric	09/18/10	No Progenitor Source		
3	Data3_SASI_v_diff_05_seed	Lingerfelt, Eric	09/18/10	No Progenitor Source		
2	Data3_SASI_v_diff_05	Lingerfelt, Eric	09/18/10	No Progenitor Source		
1	Data3_SASI_v_diff_00	Lingerfelt, Eric	09/18/10	No Progenitor Source		

Visualization Set Notes

Accretion shock simulation with an EOS gamma of 4/3, and "wiggle" parameter of 0.03. Convection is see
random velocity perturbations with a min to max of -.05 c_sound to 0.05 c_sound.

Visualization Sets

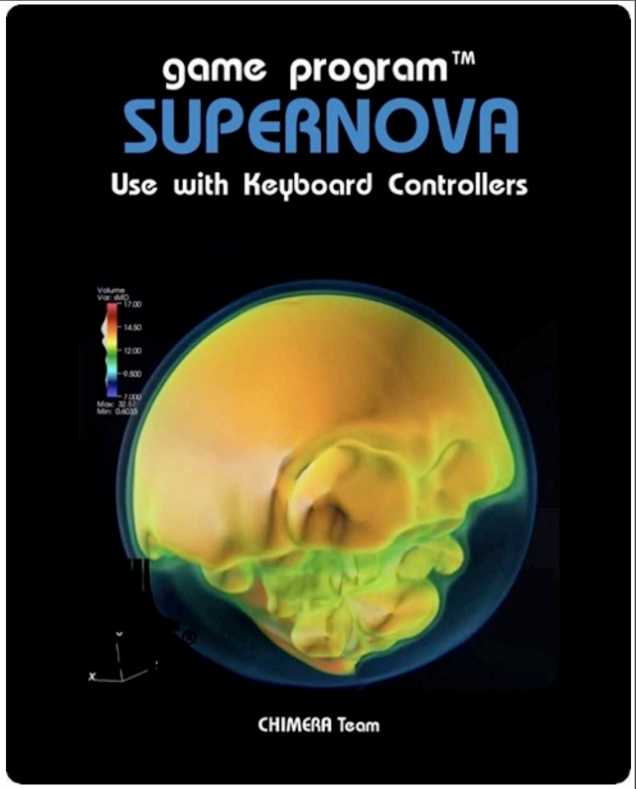
- Data3_SASI_v_diff_03_alt_ptrb (12)
  - Entropy (s) (20)
  - Y Velocity (v) (21)

Data3_SASI_v_diff_03_alt_ptrb (12) → Entropy (s) (20)

Animation Information

```
Visualization Set Index = 12
Chimera Simulation ID =
Data3_SASI_v_diff_03_alt_ptrb
Created By = Lingerfelt, Eric
Creation Date = 09/18/10
Progenitor Source = No Progenitor Source
Progenitor Mass = -1
Resolution = 252 x 252
Animation Index = 20
Size [pixels] = 1280 x 710
Colortable = orangehot
Zoom [km] = -300,300,0,300
Colortable Range = 1,8
```

Summary



- The future is now! Computers are not getting faster from the perspective of a nuclear astrophysicist. They are only getting "wider." This is true on the world's largest supercomputers and on your laptop!!!!

- The Xeon(Phi)/GPU/BG\Q choice is no choice at all. They are all versions of a single narrative.

- Stellar astrophysics is rife with unrealized parallelism, but architectural details and memory (i.e. cost, power) constraints will present considerable challenges.

- Bulk-synchronous execution is a terrible way to try to exploit near-future architectures. A new programming model will require considerably more effort than a simple multi/many-core port.

- Managing large simulations is something we can barely do now, but how about managing 1000's, 10's of thousands, or 100's of thousands of simulations? We should not expect to rely on solutions to be thrown over the fence from developers in other communities.