Understanding Supernovae with Condor

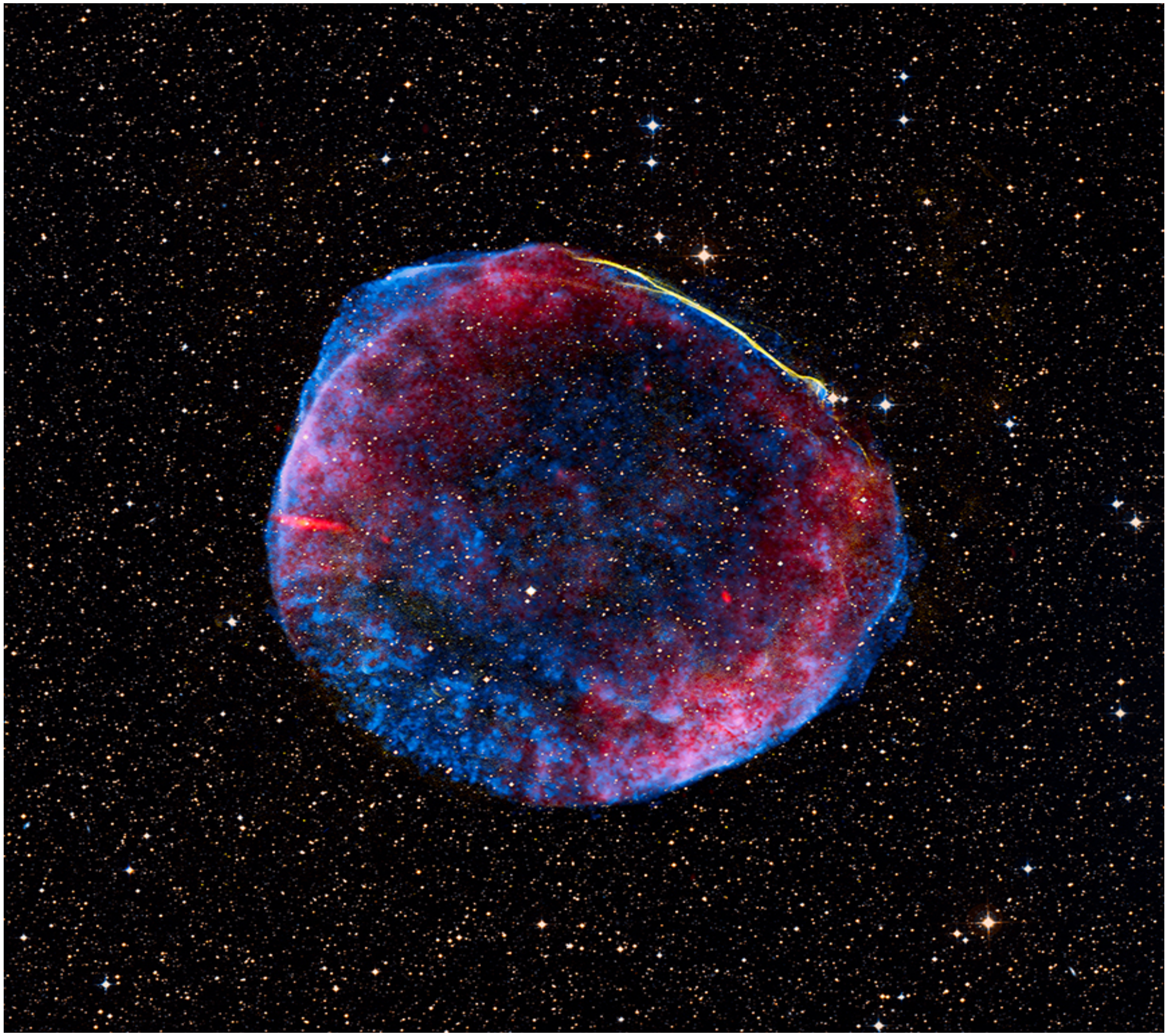
Bang!

Scott Teige

April 13, 2010

INDIANA UNIVERSITY

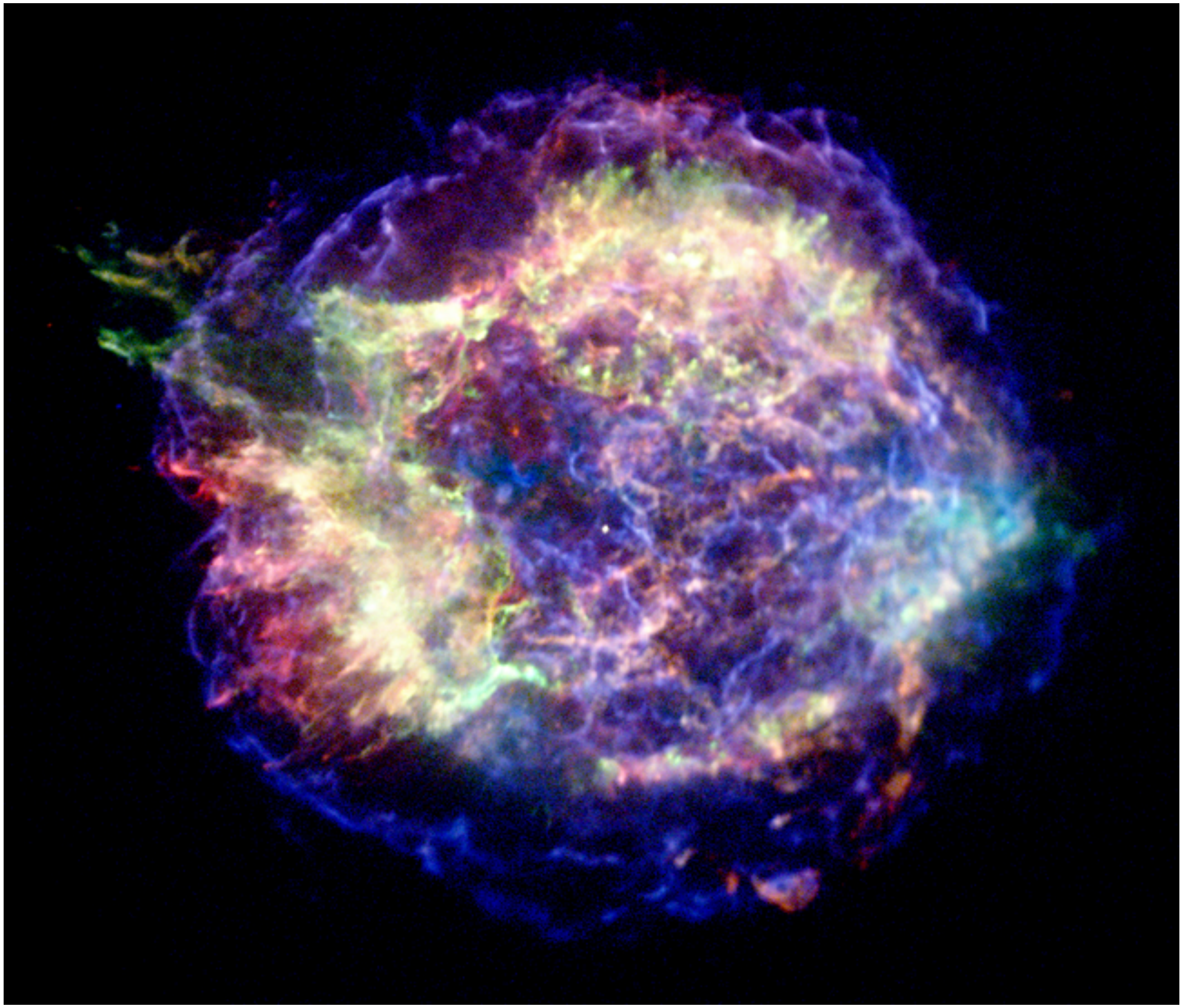
SN 1006
Type I Supernova
May 1, 1006



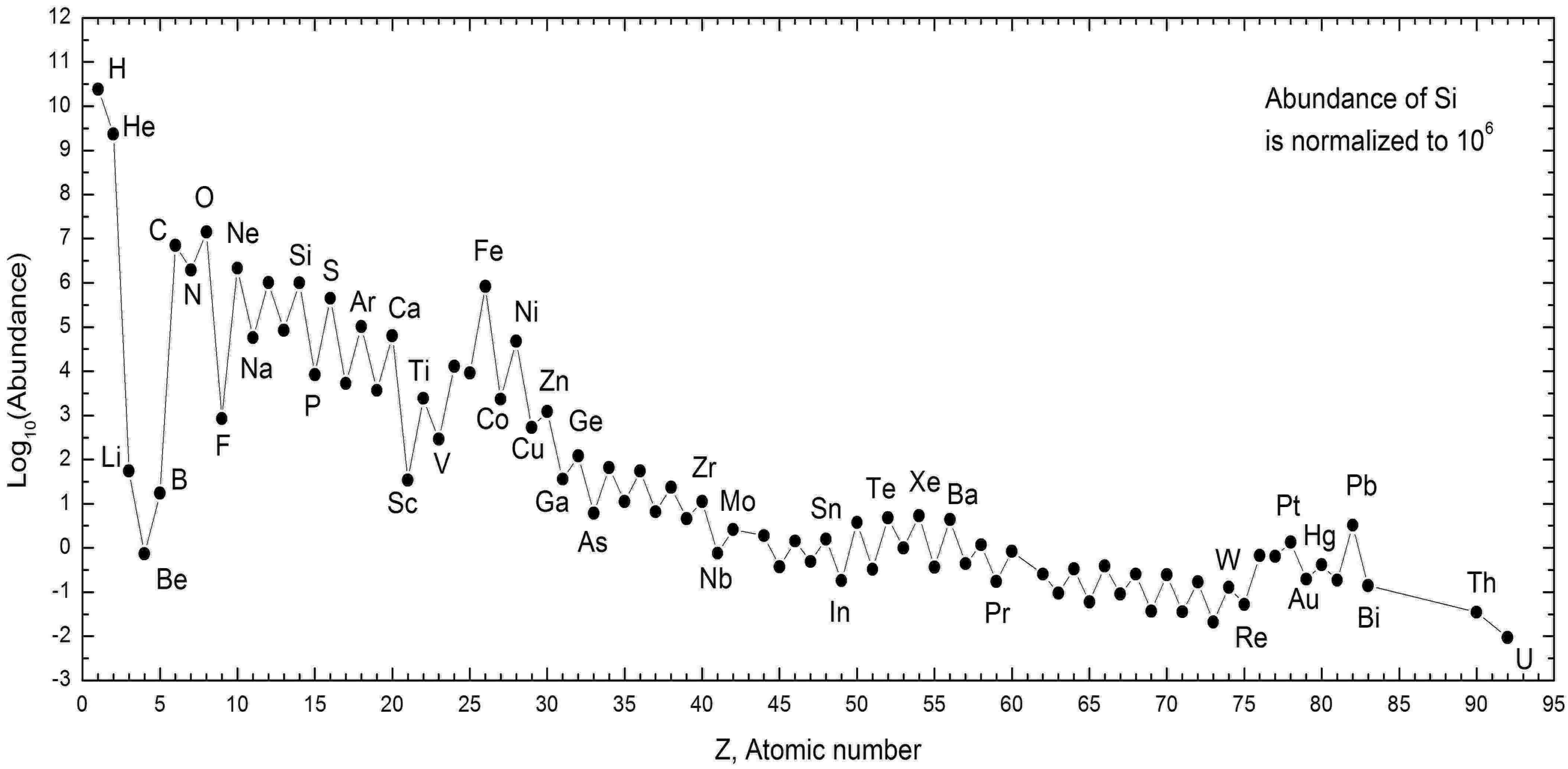
Accretion of matter onto a companion star.

Cassiopeia A,
A type II supernova
November 11, 1572

Collapse of a
massive star



Heavier than *He*, normal star, Heavier than *Fe*, supernova



How does this happen?

- Star becomes too massive to support itself
- Gravity finally wins, compression starts
- Outer layers ignite, inner layers undergo complex nuclear reactions
- Inward shock compresses core
- Outward shock expels heavy elements
- http://www.youtube.com/watch?v=grwpeEyt_98&NR=1

How do you simulate this?

- Start with a model star, choose some mass, composition, density etc.
- Follow the collapse in small time steps calculating the state of the matter at many cells of the simulated star
- Given the temperature, pressure, free energy, etc. at each cell, propagate to the next time step.

So, why is that hard?

- All parts are hard but step 2 is hardest
- Matter under these conditions cannot be observed directly
- Nuclear reactions effect the state of matter, the state of matter effects the reactions
- Next time step gives you a different density, composition, free energy, etc.

The computational problem:

- Calculating the Equation of State at a particular set of values for temperature, density and composition takes from a few seconds to a few days.
- A large range of values is needed to simulate a supernova, for example, densities from 10^{-8} to 1.6 (in some units)
- The calculations require “smooth” behavior for self consistency.

How to solve it:

- Calculate the Equation of State once and for all up front.
- Partition the relevant range of input variables into a grid, sufficiently fine to allow reliable interpolation, coarse enough to finish in one graduate student lifetime (or less)
- EoS exists as a table of values at each value of (T,ρ,η)

Where does Condor fit in?

- Each point is independent of all other points
- There are many points (~2,000 initially), each calculable by a single processor task. (No MPI, OpenMP or any of that)
- The output from a calculation at a point is small (256 bytes with embedded indexing information and the answer)

Our Condor Pool

- Student Technology Center computers
- These are paid for by student fees, the students own them, there is high sensitivity to using them for non-student activities
- There are thousands of them, most in the flock
- Use them in a “cycle scavenging mode”

The constraints:

- Must use a single, non multicore, executable
- Must run on Windows OS
- Cannot require “large” input data or produce “large” amount of output data
- Will be terminated as soon as a user attempts to use the computer running the job (touches the mouse, for example)

How did we do it?

- Code was developed on a Linux workstation
- Mapped parameter space onto integers
- Built the executable with the GCC suite under cygwin
- Packaged it all up and copied it to the condor server head node
- Submitted a single job and…

```
Executable      = dirr_single.exe
Universe        = vanilla
Requirements = ( Arch == "INTEL" && Opsys == "WINNT60" )
should_transfer_files   = YES
when_to_transfer_output = ON_EXIT

transfer_input_files    = cygwin1.dll, dir_high.dat, dir_low.dat, dirr_single.exe.local

notification    = never

Arguments = -j2067
Queue

Arguments = -j10690
Queue

Arguments = -j355
Queue

Arguments = -j3514
Queue
…
```

Our Condor experience

- Ran ~1,700 jobs over the 4'th of July weekend
- It was my first experience building executables under any flavor of Windows
- It was my first experience using Condor antwhere
- It was easy
- Worked like a charm, absolutely painless

What else do we do?

- The constraints are rather severe so our use is rather limited
- Have run a few hundreds of batches of jobs
- Almost all of these are rendering jobs, made a very nice, HiRes movie of a beating heart
- Some animations for local PBS station